Adaptively Secure Proxy Re-encryption

Georg Fucshbauer[1], Chethan Kamath[2], Karen Klein[2], and Krzysztof Pietrzak[2]

[1] Inria and ENS Paris
georg.fuchsbauer@ens.fr
[2] IST Austria
{ckamath@ist,karen.klein,pietrzak}@ist.ac.at

Abstract. A proxy re-encryption (PRE) scheme is a public-key encryption scheme that allows the holder of a key pk to derive a re-encryption key for any other key pk'. This re-encryption key lets anyone transform ciphertexts under pk into ciphertexts under pk' without knowing the underlying message, while transformations from pk' to pk should not be possible (unidirectional). Security is defined in a multi-user setting against an adversary that gets the users' public keys and can ask for re-encryption keys and can corrupt users by requesting their secret keys. Any ciphertext that the adversary cannot trivially decrypt given the obtained secret and re-encryption keys should be secure.
All existing security proofs for PRE only show *selective* security, where the adversary must first declare the users it wants to corrupt. This can be lifted to more meaningful *adaptive* security by guessing the set of corrupted users among the n users, which loses a factor exponential in n, rendering the result meaningless already for moderate n.
Jafargholi et al. (CRYPTO'17) proposed a framework that in some cases allows to give adaptive security proofs for schemes which were previously only known to be selectively secure, while avoiding the exponential loss that results from guessing the adaptive choices made by an adversary. We apply their framework to PREs that satisfy some natural additional properties. Concretely, we give a more fine-grained reduction for several unidirectional PREs, proving adaptive security at a much smaller loss. The loss depends on the graph of users whose edges represent the re-encryption keys queried by the adversary. For trees and chains the loss is quasi-polynomial in the size and for general graphs it is exponential in their depth and indegree (instead of their size as for previous reductions). Fortunately, trees and low-depth graphs cover many, if not most, interesting applications.
Our results apply e.g. to the bilinear-map based PRE schemes by Ateniese et al. (NDSS'05 and CT-RSA'09), Gentry's FHE-based scheme (STOC'09) and the LWE-based scheme by Chandran et al. (PKC'14).

1 Introduction

A proxy re-encryption (PRE) scheme is a public-key encryption scheme with an additional functionality: Alice and Bob, who have key pairs $(\mathtt{pk}_A, \mathtt{sk}_A)$ and $(\mathtt{pk}_B, \mathtt{sk}_B)$, respectively, can generate a re-encryption key (re-key, for short)

$\mathtt{rk}_{A,B}$ that allows its holder, say Peggy, to act as a proxy; that is, she can transform ciphertexts under $\mathtt{pk}_A$ to ciphertexts under $\mathtt{pk}_B$ without having to know the underlying message. A trivial way to accomplish this would be for Alice to hand her secret key $\mathtt{sk}_A$ to Peggy, who can then decrypt ciphertexts under $\mathtt{pk}_A$, encrypt them under $\mathtt{pk}_B$ and send them to Bob. Alice's secret key acts thus as the re-key and de- and encryption are used for re-encryption. However, this approach requires Alice t reveal her secret key to Peggy and therefore place complete trust on her. The more interesting cases are when the parties are mutually distrustful.

Bidirectional vs. unidirectional. In the above setting, if the re-key $\mathtt{rk}_{A,B}$ allows Peggy to also transform ciphertexts under $\mathtt{pk}_B$ to $\mathtt{pk}_A$, the PRE scheme is called "bidirectional". For such schemes the re-key is necessarily a function of both $\mathtt{sk}_A$ and $\mathtt{sk}_B$. In this paper we are interested in the more interesting case of "unidirectional" PRE schemes where the re-key $\mathtt{rk}_{A,B}$ can only transform ciphertexts from $\mathtt{pk}_A$ to $\mathtt{pk}_B$, and not vice-versa, and ciphertexts under $\mathtt{pk}_B$ remain secure even given $\mathtt{sk}_A$ and $\mathtt{rk}_{A,B}$. (Henceforth we will always assume PREs to be unidirectional.) As opposed to bidirectional PREs, the re-key generation algorithm in a unidirectional PRE takes as input "source" keys $(\mathtt{pk}_A, \mathtt{sk}_A)$ and only the "target" public key $\mathtt{pk}_B$.

Single hop vs. multiple hops. Suppose a third user, Charlie, holding keys $(\mathtt{pk}_C, \mathtt{sk}_C)$, enters the picture and suppose Peggy obtains the re-key $\mathtt{rk}_{B,C}$ that allows her to transform ciphertexts under Bob's public key to ciphertexts under Charlie's public key. Peggy can, by definition, transform a ciphertext c_A under $\mathtt{pk}_A$ to a ciphertext c_B under $\mathtt{pk}_B$ using her re-key $\mathtt{rk}_{A,B}$. If it allows Peggy to transform ciphertext c_B, which has already been re-encrypted once, to a ciphertext c_C under $\mathtt{pk}_C$ using the re-key $\mathtt{rk}_{B,C}$ then we say that the PRE scheme allows two "hops". In a similar manner, one can consider multiple hops of re-encryptions. Such a scheme is termed "multi-hop" as opposed to a "single-hop" scheme (which does not allow re-encryptions of already re-encrypted ciphertexts).

1.1 Modelling Security

The basic notion of security for unidirectional PREs is that of indistinguishability under chosen-plaintext attack (CPA). There are n users and, at the beginning of the game, the adversary gets their public keys $\mathtt{pk}_1, \ldots, \mathtt{pk}_n$ from the challenger. In the first phase, the adversary can corrupt users of its choice by requesting their secret keys; in the second phase, it can obtain re-keys $\mathtt{rk}_{i,j}$ and re-encryptions for ciphertexts of its choice. The scheme is CPA-secure if it is infeasible for the adversary to distinguish encryptions of two messages under a key that the adversary has not corrupted either directly or indirectly (through a re-key or re-encryption query to a corrupted user).

Just as in standard public-key encryption, the above security definition can be strengthened to chosen-ciphertext attack (CCA) by allowing the adversary

access to a decryption oracle which, on input a ciphertext and a public key $\mathtt{pk}_i$ returns the decryption of the ciphertext under $\mathtt{sk}_i$. The conditions to ensure non-triviality have to be altered accordingly.

We note that both definitions are *selective* in nature: the adversary must choose the set of players it corrupts before issuing any queries.

1.2 Prior Work

Bidirectional PREs were introduced as "atomic proxy cryptography" by Blaze, Bleumer and Strauss [BBS98], who constructed a multi-hop scheme under the decisional Diffie-Hellman assumption. Unidirectional PREs were introduced later by Ateniese et al. [AFGH05]. Their main motivation was to limit the amount of trust placed on the proxy, as required bye their application to access control for distributed storage. Since the notion of security for unidirectional PRE is different from a bidirectional PRE, they also reformulated the notion of CPA (for the single-hop setting). Assuming hardness of certain problems on bilinear groups, they constructed CPA-secure schemes that are single-hop and unidirectional.

The definition of CCA security for single-hop bidirectional schemes is due to Canetti and Hohenberger [CH07] and is more involved than previous definitions, mainly because the adversary is allowed adaptive corruption. They gave a scheme satisfying their notion under the standard decisional bilinear Diffie-Hellman assumption. The definition of CCA security in the unidirectional setting is due to Libert and Vergnaud [LV08], who instantiate it under a slightly non-standard assumption on bilinear groups.

The earlier constructions of multi-hop, unidirectional schemes were based on program obfuscation [HRsV07, CCV12]. In his seminal paper, Gentry [Gen09] gave a generic construction of PREs from fully homomorphic encryption. The first construction (with succinct ciphertexts) based on a standard assumption is due to Chandran et al. [CCL+14]: their scheme is CPA-secure assuming decisional learning with errors. Phong et al. [PWA+16] followed up with a construction that, in addition, enjoys a security property called "key-privacy". The only construction of a CCA-secure multi-hop, unidirectional scheme is due to Fan and Liu [FL17]. In their paper, they also rigorously defined the security models (CPA and CCA) for the multi-hop setting.

Cohen [Coh17] has recently argued that CPA security might be too weak for some applications and introduced *indistinguishability against honest-reencryption attack* (HRA), a notion that lies between CPA and CCA. He also showed that if a PRE satisfies a property called "source-hiding", which several existing CPA-secure schemes do, then HRA security reduces to CPA security.

1.3 Our Contribution

Our starting point is the observation that, unlike bidirectional PREs, the security definitions for unidirectional PREs (that is, CPA, HRA and CCA) are all *selective* in nature: the adversary must choose the set of parties it corrupts before issuing any queries. A more meaningful notion would be *adaptive* security,

where the adversary is allowed to corrupt users at any time during the game. However, modelling this turns out to be as tricky as in the bidirectional setting. In this paper, we lift the definitions for CPA and HRA to the adaptive setting.

1.3.1 First Contribution: Modelling Adaptive Corruption. The main problem that arises when we allow the adversary to adaptively corrupt users is that we must ensure that the adversary cannot trivially win the security game. For bidirectional PREs this was handled in [CH07] by defining a relation that keeps track of the dependency between the re-keys and re-encryptions that were issued during the game. Our approach is similar in spirit: the security game maintains a "recoding graph" that has n nodes, and whose edges are derived from the re-keys and re-encryptions issued to the adversary The exact definitions of the recoding graph for adaptive CPA and for adaptive HRA differ slightly, but in both cases it is defined so that no corrupt key is reachable from the challenge key. That is, the adversary is forbidden from making *any* re-key or re-encryption queries to a corrupt user that is reachable from the challenge key. The recoding graph now allows to ensure non-triviality of the adversary's actions by checking a few basic graph properties.

1.3.2 Second Contribution: The Reduction. Proving adaptive security can be reduced to showing selective security by initially guessing the set of users that will be corrupted. However, this reduction loses an exponential factor in n, rendering the reduction meaningless already for moderate n. As our main contribution, we give a more fine-grained reduction from adaptive to selective security which in many practical settings and for several existing schemes (or minor variants) implies adaptive security at much smaller (quasi-polynomial, or even polynomial) loss. More precisely, the loss in our reduction depends on the structure of the recoding graph: for trees and chains we get a quasi-polynomial $n^{O(\log n)}$ loss, whereas for general graphs the loss is exponential in their depth. Fortunately, trees, chains, and low-depth graphs cover many, if not most, interesting applications.

Security assumptions. A key step in our search for a tighter reduction was the identification of the basic security assumptions on a PRE that we required in our arguments. For the case of CPA, it turned out to be ciphertext indistinguishability and *weak* key-privacy, both fairly standard security requirements already explored in some of the previous works.

As the name suggests, a PRE is ciphertext-indistinguishable (or, for short, indistinguishable) if the underlying encryption is. Since the syntax of the encryption algorithm for a PRE is slightly different from that of a standard public-key encryption, the definition of indistinguishability has to be slightly changed. To be precise, the encryption algorithm for a PRE takes also a "level" as input, and we require that the ciphertexts are indistinguishable *on all levels*. It is not hard, therefore, to see that any selectively CPA-secure PRE has to trivially satisfy indistinguishability.

The notion of key-privacy was introduced in a strong form in [ABH09]. We require the PRE to satisfy a much weaker property, namely that a re-key $\mathtt{rk}_{A,B}$ looks pseudorandom given just the source and target public keys $\mathtt{pk}_A$ and $\mathtt{pk}_B$. Existing PRE schemes that satisfy the stronger key privacy as defined in [ABH09] are therefore candidates for our reduction.

To apply our reduction to HRA-secure PRE, we need a third assumption to hold: source-hiding. This is the same property that allowed Cohen [Coh17] to lift a CPA-secure PRE to a HRA-secure one. Informally, a PRE is source-hiding if ciphertexts that result from re-encryptions are distributed close to fresh encryptions (at the corresponding level).

For PRE satisfying these assumptions, we show that the framework of Jafargholi et al. [JKK+17] can be applied. This framework has been used to show adaptive security of a variety of cryptographic protocols (e.g., secret sharing, garbled circuits etc.) while avoiding an exponential loss that typically results from the guessing step when going from selective to adaptive security. We describe their framework in more detail below.

The JKKKPW framework. A standard way to prove adaptive security is to first define a "selective" variant that requires the adversary to commit to some of its choices (e.g., whom to corrupt, or on what input to be challenged at the end) at the beginning of the game. Let $\mathcal{W}$ denote the set of all possible choices.

Consider a selective security notion defined as two games H^0 and H^1 being indistinguishable. A security proof often uses a hybrid argument: one defines a sequence of hybrid games $(\mathsf{H}_0, \ldots, \mathsf{H}_\tau)$ where the first and last games correspond to the original selective games (i.e., $\mathsf{H}^0 = \mathsf{H}_0$ and $\mathsf{H}^1 = \mathsf{H}_\tau$). One then proves that any two consecutive hybrids (H_t and H_{t+1}) are ϵ-indistinguishable. As indistinguishability satisfies the triangle inequality, the extreme games H_0 and H_τ are $(\epsilon \cdot \tau)$-indistinguishable.

Now to prove security against an adaptive adversary (who will not reveal its choices at the beginning), one defines a new reduction that just guesses the adversary's future choices at random from the set $\mathcal{W}$ and then follows the selective reduction. Conditioned on the guess being correct, this reduction has the same success probability as the selective one.

Unfortunately, the overall loss in security of this second step is as large as the size of $\mathcal{W}$, which is typically exponential (e.g., exponential in the number of parties that can be corrupted). Thus, if the selective reduction implied ϵ-indistinguishability (based on some underlying assumption), the adaptive reduction will only imply $(\epsilon \cdot |\mathcal{W}|)$-indistinguishability, which in most cases will be meaningless.

The key observation of Jafargholi et al. [JKK+17] was that in many selective reductions as above, only a highly compressed version $h(w)$ of the information $w \in \mathcal{W}$ that the adversary commits to is actually used in the simulation of intermediate hybrids. Jafargholi et al. called these "partially selective" hybrids, as opposed to the original hybrids, which are "fully selective". They show [JKK+17] that the security loss in such cases is only exponential in the length of $h(w)$ (its

the longest value for any two consecutive hybrids), and not exponential in the length of the entire w.

In all the instances to which the JKKKPW framework has been applied the simulation of the security game depends on some underlying graph (e.g., the access structure in secret sharing or the Boolean circuit in case of garbling) and the hybrid games involve incremental changes to the simulation *depending* on the structure of this graph. Jafargholi et al. managed to decouple the particulars of the simulation from the design of the hybrids by using a pebbling game on the graph (the graph must thus be directed and acyclic). To be more precise, they associated the simulation of a hybrid (H_t) to a pebbling configuration ($\mathcal{P}_t$), and therefore the incremental changes in the simulation to the pebbling sequence $(\mathcal{P}_0, \ldots, \mathcal{P}_\tau)$. In particular, if a vertex carries a pebble then the part of simulation of the hybrid that is dependent on the vertex is carried out in a different manner (e.g., in garbling using Yao's scheme the ciphertexts in the garbled table for a gate are all bogus). The rules of the simulation is what then determines the pebbling rules, i.e., when exactly a pebble can be placed on or removed from a vertex. The extreme hybrids correspond to the initial and final pebbling configurations, and the immediate goal is to show that two hybrids that differ by a pebble are indistinguishable to an adversary. Indistinguishability of the original games then follows by transitivity of indistinguishability.

In the fully selective games of the above examples, the adversary commits to the whole graph; but, as explained above, knowledge of the vertices that are pebbled suffices to simulate the intermediate hybrids. Therefore, in the partially selective game the adversary "commits" to some pebbling configuration. Since we have established a correspondence between the simulation and a pebbling configuration, the task of designing a better sequence of hybrids has been reduced to finding a better pebbling sequence. In particular, the fewer pebbles are on the graph at any particular time, the more concisely we can describe this configuration, and thus the smaller the incurred security loss.

Designing the hybrids. The graph that underlies the simulation in adaptive CPA and HRA is precisely the recoding graph. (Strictly speaking, it suffices to consider the subgraph that is reachable from the challenge vertex, which we will call the "challenge graph".) The presence (or not) of a pebble on a vertex dictates how the re-encryption and re-key queries outgoing from that vertex are simulated. Therefore in the fully selective games, the adversary commits to the recoding graph (which is different from the original selective game in which the adversary committed to the set of corrupt users), whereas in the partially selective games it "commits" just to a pebbling configuration.

Let us first consider adaptive CPA: the edges of the recoding graph correspond to the re-key and re-encryption queries made by the adversary during the game. For simplicity, assume that the recoding graph has a single source vertex i^* that is also the vertex the adversary wants to be challenged on. Once it has made all the queries, the adversary receives its challenge, which is the encryption of either m_0^* or m_1^* under $\mathtt{pk}_{i^*}$; let CPA^0 and CPA^1 denote the respective games. In case there are no outgoing edges from i^*, indistinguishability of CPA^0 and

CPA^1 follows from ciphertext indistinguishability (the first assumption): The reduction embeds the challenge public key (of the indistinguishability game) as the i^*-th key, relays (m_0^*, m_1^*) to its challenger and forwards the challenge ciphertext it receives to the adversary. As there are no outgoing re-keys from i^*, the simulation does not require the secret key $\mathtt{sk}_{i^*}$.

In case i^* does have outgoing edges, the idea is to use a sequence of hybrids to reach a game where knowledge of $\mathtt{sk}_{i^*}$ is not required for simulation, just like above. To argue indistinguishability of hybrids, we use weak key-privacy, which guarantees that a re-key looks pseudorandom given the source and target public keys. Weak key-privacy allows the simulator to fake the outgoing edges from a vertex, after which the secret key for this vertex is not required for simulation anymore. GF: [maybe don't mention this in submission: (We note this is reminiscent of the approach used in [JKK+17] to show adaptive security for generalised selective decryption.) However, the simulator cannot fake edges right away: it has to fake all children of a vertex first, before it can rely on weak key-privacy. Consequently, the pebbling must obey the following rule: in a move, a pebble can be placed on or removed from a vertex only if all its children carry pebbles.

To be precise, in game H_t^b, for each pebbled vertex in $\mathcal{P}_t$ all queried re-keys outgoing from that vertex are faked. Observe that as the secret key corresponding to a vertex is used only for the generation of the re-keys outgoing from that vertex, the simulation of a hybrid can be carried out *without* knowledge of the secret key corresponding to the pebbled vertices. Thus, a pebbling sequence describes a sequence of hybrids. GF: have we mentioned that the goal of the pebbling game is to pebble the challenge vertex?

Main result. Our main result bounds the security loss for arbitrary recoding graphs in terms of their space and time complexity, where a graph is said to have space complexity σ and time complexity τ if there exists a valid pebbling strategy for that graph that uses at most σ pebbles and requires at most τ moves. More generally, a class of graphs has space complexity σ and time complexity τ if this is the case for every graph in that class.

Theorem 1 (Vanilla Theorem 5 and Theorem 6). *Let $\mathcal{G}(n)$ denote a family of graphs on n vertices with space-complexity σ and time-complexity τ. Then a PRE that is ciphertext-indistinguishable and weakly key-private for computationally bounded adversaries is also adaptively CPA-secure against computationally bounded adversaries for recoding graphs in $\mathcal{G}$ with a loss in security of $\approx \tau \cdot n^\sigma$. If the PRE is also statistically source-hiding then it is also adaptively HRA-secure.*

In many applications, the underlying recoding graph is has a very particular structure like trees (or even paths) and low-depth graphs, which cover many interesting applications. For paths, or fixed-arity trees, our reduction only loses a quasi-polynomial factor. For low-depth graphs, the loss is exponential only in the depth (and thus polynomial for fixed depth-graphs). Below, we mention two such applications.

Table 1: PRE schemes we prove adaptively CPA and HRA secure (see §5 for the definitions of the assumptions).

Scheme	Setting	Assumption(s)	Hops
Construction 2 [AFGH05]	Bilinear maps	eDBDH and XDH	Single
Construction 4 [ABH09]	Bilinear maps	eDBDH and DLin	Single
Construction 5 [Gen09]	–	FHE	Multiple
Construction 7 [CCL+14]	Lattices	LWE	Multiple

1. In *key rotation* for encrypted cloud storage, a client has its data encrypted on a server, and occasionally wants to re-encrypt it (say, to restore security after key leakage). As the client does not trust the server, it will not want to hand it the decryption key. When using PRE, the client can simply send a re-key to the server, which enables it to locally re-encrypt all ciphertexts to the new key. In this application the recoding graph is simply a chain.
2. Another common application is *forwarding of encrypted email* without involving the receiver, say, for delegation during vacation or for filtering spam emails. In most cases the underlying delegation structure will be captured by simple graphs. For example, if delegation only happens to subordinates, the depth of the recoding graph is bounded by the depth of the hierarchy of the organisation.

1.3.3 Third Contribution: Adaptively-Secure PREs. Finally, we show that the aforementioned three properties are satisfied by several existing constructions or by minor variants thereof, and thus Theorem 1 can be applied to them. An overview of these schemes is given in Table 1. We consider the most interesting corollary to our results the adaptive security of the LWE-based scheme by Chandran et al. [CCL+14]:

Theorem 2 (Vanilla Theorem 11). *The quasi-polynomially secure decisional LWE problem implies multi-hop, unidirectional adaptively CPA/HRA-secure PRE for chains or complete binary trees.*

2 Formal Definitions

Notation. We use $[a, b]$ to denote $\{a, a+1, \ldots, b\}$ and $[b]$ as a shorthand for $[1, b]$. We will only consider logarithms to the base 2 (i.e., $\log := \log_2$). For two sets $\mathcal{X}, \mathcal{Y}$ we write $\mathcal{X} \Delta \mathcal{Y}$ for the symmetric difference. We write $x \leftarrow \mathcal{X}$ for sampling an element x uniformly at random from the set $\mathcal{X}$; analogously, $x_1, \ldots, x_n \leftarrow \mathcal{X}$ denotes sampling $x_1, \ldots, x_n$ independently and uniformly at random from the

set $\mathcal{X}$. To indicate sampling according to a distribution X on $\mathcal{X}$, we write $x \leftarrow X$. By $[X]$ we denote the support of X, i.e., the values with positive probability. For two distributions X, Y, $\Delta(X, Y)$ denotes their statistical distance. We write $X \equiv Y$ if X has the same input/output distribution as Y. Two distributions $X = \{X_\kappa\}_{\kappa\in\mathbb{N}}$ and $\{Y_\kappa\}_{\kappa\in\mathbb{N}}$ are (s, ϵ)-indistinguishable, denoted $X \approx_{(s,\epsilon)} Y$, if for every adversary A of size at most s

$$|\mathbb{P}[\mathsf{A}(X) = 1] - \mathbb{P}[\mathsf{A}(Y) = 1]| \leq \epsilon.$$

Throughout the paper, we will repeatedly use the following lemma concerning the transitivity of the indistinguishability relation $\approx$:

Lemma 1. *Let X, Y, Z be distributions on a set $\mathcal{X}$. If $X \approx_{(s_1,\epsilon_1)} Y$ and $Y \approx_{(s_2,\epsilon_2)} Z$, then $X \approx_{(\min(s_1,s_2),\epsilon_1+\epsilon_2)} Z$.*

For indistinguishability-based security games, we use $\langle \mathsf{G}, \mathsf{A}\rangle$ to denote the bit output by the challenger G at the end of its interaction with the adversary A. We say that two games G^0 and G^1 are (s, ϵ)-indistinguishable, denoted $\mathsf{G}^0 \approx_{(s,\epsilon)} \mathsf{G}^1$, if for every adversary A of size at most s

$$|\mathbb{P}[\langle \mathsf{G}^0, \mathsf{A}\rangle = 1] - \mathbb{P}[\langle \mathsf{G}^1, \mathsf{A}\rangle = 1]| \leq \epsilon.$$

For an algorithm A, we use s_A to denote its size; in a similar manner, for a set $\mathcal{X}$, we use $s_\mathcal{X}$ to denote the complexity of sampling from $\mathcal{X}$ uniformly at random.

Notation for graphs. We let $G = (\mathcal{V}, \mathcal{E})$ denote a directed graph with vertices $\mathcal{V}$ (usually $\mathcal{V} = [n]$ for some $n \in \mathbb{N}$) and edges $\mathcal{E} \subseteq \mathcal{V}^2$. The indegree (resp., outdegree) of a vertex is defined as the number of edges coming in to (resp., going out of) that vertex. The indegree (resp., outdegree) of the graph is the maximum indegree (resp., outdegree) over all the vertices. A vertex with indegree (resp., outdegree) zero is called a source (resp., sink). A vertex i is *connected* to another vertex j (or alternatively j is reachable from i) if there is a directed path from i to j in G.

Miscellaneous notation.

1. κ: the security parameter
2. λ: the number of supported levels of re-encryption supported
3. q: the order of a group or a field
4. b: the bit used to select the challenge
5. b': adversary's guess
6. n: number of keys/size of the graph
7. δ/d: degree/depth of a graph

2.1 Proxy Reencryption: Formal Definitions

Definition 1 (Multi-hop, unidirectional PRE). *A multi-hop, unidirectional PRE scheme for a message space $\mathcal{M}$ consists of the six-tuple of algorithms $(\mathsf{S}, \mathsf{K}, \mathsf{RK}, \mathsf{E}, \mathsf{D}, \mathsf{RE})$, which are explained below.*

$\mathsf{S}(1^\kappa, 1^\lambda) \to \mathtt{pp}$**:** *On input the security parameter κ and the maximum level λ (both in unary) supported by the scheme,* **setup** *outputs the public parameters* $\mathtt{pp}$*. We assume that* $\mathtt{pp}$ *is implicit in other function calls.*

$\mathsf{K}(\mathtt{pp}) \to (\mathtt{pk}, \mathtt{sk})$**: Key generation** *returns a public key* $\mathtt{pk}$ *and the corresponding secret key* $\mathtt{sk}$*.*

$\mathsf{RK}((\mathtt{pk}_i, \mathtt{sk}_i), \mathtt{pk}_j) \to \mathtt{rk}_{i,j}$**:** *On input a source key pair $(\mathtt{pk}_i, \mathtt{sk}_i)$ and a target public key $\mathtt{pk}_j$,* **re-key generation** *generates a unidirectional re-encryption key (rekey, for short) $\mathtt{rk}_{i,j}$.*

$\mathsf{E}(\mathtt{pk}, (m, \ell)) \to (c, \ell)$**: Encryption** *takes as input the public key* $\mathtt{pk}$*, a message m and a level $\ell \in [\lambda]$, and outputs a level-ℓ ciphertext (c, ℓ).*

$\mathsf{D}(\mathtt{sk}, (c, \ell)) \to m$ *: On input a ciphertext (c, ℓ) and the secret key* $\mathtt{sk}$*,* **decryption** *outputs a message m, or the symbol $\perp$ (if the ciphertext is invalid).*

$\mathsf{RE}(\mathtt{rk}_{i,j}, \mathtt{pk}_i, \mathtt{pk}_j, (c_i, \ell)) \to (c_j, \ell + 1)$**: Reencryption** *takes a re-key $\mathtt{rk}_{i,j}$, a source public key $\mathtt{pk}_i$, a target public key $\mathtt{pk}_j$ and a level-ℓ ciphertext c_i under $\mathtt{pk}_i$ and transforms it to a level-$(\ell+1)$ ciphertext c_j under $\mathtt{pk}_j$. Only ciphertexts belonging to levels $\ell \in [\lambda - 1]$ can be re-encrypted. In constructions where arguments $\mathtt{pk}_i$ and/or $\mathtt{pk}_j$ are optional, we simply drop them.*

Definition 1 differs slightly from the definition of multi-hop unidirectional PRE in [FL17]. Here, the re-keys are level-agnostic: the same re-key can be used to re-encrypt a ciphertext belonging to any level. In [FL17], however, a re-key associated to a level *cannot* be used to re-encrypt a ciphertext from a different level. We require the PRE to satisfy the following two correctness properties.

Definition 2 (Correctness [ABH09]). *A proxy re-encryption scheme (as in Definition 1) is correct w.r.t. the message space $\mathcal{M}$ if the following two properties hold:*

1. *Correctness of encryption:* $\forall \kappa, \lambda \in \mathbb{N}\ \forall\, \mathtt{pp} \in [\mathsf{S}(1^\kappa, 1^\lambda)]\ \forall\, (\mathtt{pk}, \mathtt{sk}) \in [\mathsf{K}(\mathtt{pp})]\ \forall\, (m, \ell) \in \mathcal{M} \times [\lambda]$*:*

$$\mathbb{P}[\mathsf{D}\big(\mathtt{sk}, \mathsf{E}(\mathtt{pk}, (m, \ell))\big) \neq m] = \mathsf{negl}(\kappa, \lambda),$$

   where the probability is over the random coins of E*.*

2. *Correctness of re-encryption:* $\forall\, \kappa, \lambda \in \mathbb{N}\ \forall\, \mathtt{pp} \in [\mathsf{S}(1^\kappa, 1^\lambda)]\ \forall\, (\mathtt{pk}_i, \mathtt{sk}_i)$, $(\mathtt{pk}_j, \mathtt{sk}_j) \in [\mathsf{K}(\mathtt{pp})]\ \forall\, \mathtt{rk}_{i,j} \in [\mathsf{RK}((\mathtt{pk}_i, \mathtt{sk}_i), \mathtt{pk}_j)]\ \forall\, (m, \ell) \in \mathcal{M} \times [\lambda - 1]$*:*

$$\mathbb{P}[\mathsf{D}\big(\mathtt{sk}_j, \mathsf{RE}(\mathtt{rk}_{i,j}, \mathtt{pk}_i, \mathtt{pk}_j, \mathsf{E}(\mathtt{pk}_i, (m, \ell)))\big) \neq m] = \mathsf{negl}(\kappa, \lambda),$$

   where the probability is over the random coins of E *and* RE*.*

Challenger $\mathsf{sCPA}^b(1^\kappa, 1^\lambda, n)$

1: Set $\mathcal{C} = \emptyset$ ▷ Stores the corrupt public keys
2: $\mathsf{pp} \leftarrow \mathsf{PRE.S}(1^\kappa, 1^\lambda)$, $(\mathtt{pk}_1, \mathtt{sk}_1), \ldots, (\mathtt{pk}_n, \mathtt{sk}_n) \leftarrow \mathsf{PRE.K}(\mathsf{pp})$ ▷ Generate keys
3: $\forall i, j \in [n], i \neq j : \mathtt{rk}_{i,j} \leftarrow \mathsf{PRE.RK}((\mathtt{pk}_i, \mathtt{sk}_i), \mathtt{pk}_j)$ ▷ Generate re-keys
4: $\mathsf{state} \leftarrow \mathsf{A}_1^{(\mathtt{corrupt}, \cdot)}(\mathsf{pp})$ ▷ Phase 1
5: $b' \leftarrow \mathsf{A}_2^{(\mathtt{rekey}, \cdot, \cdot), (\mathtt{reencrypt}, \cdot, \cdot, \cdot), (\mathtt{challenge}, \cdot, \cdot, \cdot)}(\mathtt{pk}_1, \ldots, \mathtt{pk}_n, \mathsf{state})$ ▷ Phase 2
6: **return** b'

Oracle $(\mathtt{corrupt}, i)$

1: Add i to $\mathcal{C}$
2: **return** $\mathtt{sk}_i$

Oracle $(\mathtt{rekey}, i, j)$

1: **if** $i \notin \mathcal{C}$ and $j \in \mathcal{C}$ **then** HALT **end if** ▷ $\mathtt{abort}_2$
2: **return** $\mathtt{rk}_{i,j}$

Oracle $(\mathtt{reencrypt}, i, j, (c_i, \ell))$

1: **if** $i \notin \mathcal{C}$ and $j \in \mathcal{C}$ **then** HALT **end if** ▷ $\mathtt{abort}_3$
2: **return** $(c_j, \ell + 1) \leftarrow \mathsf{PRE.RE}(\mathtt{rk}_{i,j}, \mathtt{pk}_i, \mathtt{pk}_j, (c_i, \ell))$

Oracle $(\mathtt{challenge}, i^*, (m_0^*, m_1^*), \ell^*)$ ▷ Single access

1: **if** $i^* \in \mathcal{C}$ **then** HALT **end if** ▷ $\mathtt{abort}_1$
2: **return** $(c_{i^*}, \ell^*) \leftarrow \mathsf{PRE.E}(\mathtt{pk}_{i^*}, (m_b^*, \ell^*))$

Game 1: sPRE-CPA

2.2 Modelling Security

2.2.1 Selective Corruption. The selective security of a multi-hop, unidirectional PRE scheme against a chosen-plaintext attack is modelled using the security game given in Game 1.[3] It is an extension of the security model for single-hop PRE from [ABH09] to the multi-hop setting.[4] The limiting feature of the model is that the adversary has to fix, beforehand in Phase 1 (see Game 1), the honest and corrupt public keys. Its goal is to distinguish an encryption of m_0 from an encryption of m_1 (for m_0, m_1 of its choice) under a key of its choice. The game aborts if the adversary does one of the following:

- query the challenge oracle on a corrupt public key ($\mathtt{abort}_1$);
- request a re-key from an honest key to a corrupt key ($\mathtt{abort}_2$); or
- query a re-encryption from an honest to a corrupt key ($\mathtt{abort}_3$).

Definition 3 (sPRE-CPA-security). *A PRE scheme is (s, ϵ)-selectively secure against chosen-plaintext attack if* $\mathsf{sCPA}^0 \approx_{(s,\epsilon)} \mathsf{sCPA}^1$*, where* sCPA^b *is defined in Game 1.*

[3] The formulation here is slightly different from the original one in [ABH09]. In [ABH09], the adversary has access to two oracles in Phase 1, one for generating honest keys (i.e., the adversary gets just the public key) and the other for generating corrupted keys (i.e., the adversary gets both public and secret key). In Game 1 the adversary is first given all the public keys and can then, in Phase 1, choose the keys it wants to corrupt.

[4] [FL17] formalised security differently; we stick to the definition from [ABH09].

Challenger $\mathsf{sHRA}^b(1^\kappa, 1^\lambda, n)$

1: Set $\mathcal{C}, \mathcal{E} = \emptyset$ ▷ Stores corrupt keys and issued re-keys and re-encryptions
2: Set $C = 0$ ▷ Counts ciphertexts generated
3: Set $\mathcal{L}, \mathcal{L}^* = \emptyset$ ▷ Stores honest ciphertexts and which derived from challenge
4: $\mathsf{pp} \leftarrow \mathsf{PRE.S}(1^\kappa, 1^\lambda)$, $(\mathtt{pk}_1, \mathtt{sk}_1), \ldots, (\mathtt{pk}_n, \mathtt{sk}_n) \leftarrow \mathsf{PRE.K}(\mathsf{pp})$ ▷ Generate keys
5: $\forall i, j \in [n], i \neq j : \mathtt{rk}_{i,j} \leftarrow \mathsf{PRE.RK}((\mathtt{pk}_i, \mathtt{sk}_i), \mathtt{pk}_j)$ ▷ Generate re-keys
6: $\mathsf{state} \leftarrow \mathsf{A}_1^{(\mathtt{corrupt}, \cdot)}(\mathsf{pp})$ ▷ Phase 1
7: $b' \leftarrow \mathsf{A}_2^{(\mathtt{encrypt}, \cdot, \cdot), (\mathtt{rekey}, \cdot, \cdot), (\mathtt{reencrypt}, \cdot, \cdot, \cdot), (\mathtt{challenge}, \cdot, \cdot, \cdot)}(\mathtt{pk}_1, \ldots, \mathtt{pk}_n, \mathsf{state})$ ▷ Phase 2
8: **return** b'

Oracles `corrupt` and `rekey` are defined like in Game 1.

Oracle $(\mathtt{encrypt}, i, (m, \ell))$

1: $(c, \ell) \leftarrow \mathsf{PRE.E}(\mathtt{pk}_i, (m, \ell))$
2: Increment C and add $(C, i, m, (c, \ell))$ to $\mathcal{L}$
3: **return** (c, ℓ)

Oracle $(\mathtt{reencrypt}, i, j, k)$

1: Retrieve $(k, i, m, (c_i, \ell))$ from $\mathcal{L}$ and increment C
2: $(c_j, \ell + 1) \leftarrow \mathsf{PRE.RE}(\mathtt{rk}_{i,j}, \mathtt{pk}_i, \mathtt{pk}_j, (c_i, \ell))$
3: **if** $k \in \mathcal{L}^*$ **then** ▷ The ciphertext is derived from the challenge
4: **if** $j \in \mathcal{C}$ **then** HALT **else** add C to $\mathcal{L}^*$ ▷ $\mathtt{abort}_3^*$ **end if**
5: **end if**
6: Add $(C, j, m, (c_j, \ell + 1))$ to $\mathcal{L}$
7: **return** $(c_j, \ell + 1)$

Oracle $(\mathtt{challenge}, i^*, (m_0^*, m_1^*), \ell^*)$ ▷ Single access

1: Compute $(c_{i^*}, \ell^*) \leftarrow \mathsf{PRE.E}(\mathtt{pk}_{i^*}, (m_b^*, \ell^*))$ and increment C
2: **if** $i^* \in \mathcal{C}$ **then** HALT **else** add C to $\mathcal{L}^*$ ▷ $\mathtt{abort}_1$ **end if**
3: Add $(C, i^*, m_b^*, (c_{i^*}, \ell^*))$ to $\mathcal{L}$
4: **return** (c_{i^*}, ℓ^*)

Game 2: sPRE-HRA

Security against honest-reencryption attack. A stronger security definition was introduced in [Coh17] to address some of the restrictions that sPRE-CPA imposes on the adversary. The idea is to allow re-encryptions from honest to corrupt keys, if the ciphertexts to re-encrypt were honestly generated. The adversary can obtain such honest ciphertexts via an `encrypt` oracle, which stores them in a list. The `reencrypt` oracle now takes the index of an honestly generated ciphertext. It was shown in [Coh17] that (selective) HRA-security implies (selective) CPA-security and also that if the PRE scheme is re-encryption-simulatable (a generalisation of Definition 9) then (selective) CPA-security implies (selective) HRA-security. In sPRE-HRA, which we formally define in Game 2, $\mathtt{abort}_3$ is relaxed to

– $\mathtt{abort}_3^*$: The adversary queries the re-encryption of a ciphertext that is the result of a chain of re-encryptions of the challenge ciphertext from an honest to a corrupt key.

Definition 4 (sPRE-HRA-security). *A PRE scheme is (s, ϵ)-selectively secure against honest-reencryption attack if* $\mathsf{sHRA}^0 \approx_{(s,\epsilon)} \mathsf{sHRA}^1$*, where* sHRA^b *is defined in Game 2.*

2.2.2 Modelling Adaptive Corruption. The adaptive security games corresponding to Games 1 and 2 are given in Games 3 and 4, respectively. To model adaptive corruption, we think of the game being played on a directed graph $G = (\mathcal{V}, \mathcal{E})$ called the "recoding" graph. The vertices of the recoding graph correspond to the public keys, i.e., $\mathcal{V} = [n]$. The edges are derived from the re-keys and re-encryptions issued to the adversary in the security game, and their purpose is to ensure that the adversary does not win the game in a trivial manner. In particular, the recoding graph is defined so that *no* corrupt key is reachable from the challenge key. To be precise, in CPA an edge (i, j) is added to $\mathcal{E}$ if the adversary made either a $(\texttt{rekey}, i, j)$ or $(\texttt{reencrypt}, i, j, \cdot)$ query (see Game 3 and Figure 1). Consequently, the adversary is forbidden from making *any* re-key or re-encryption queries to a corrupt user that is reachable from the challenge key.[5]

For HRA, on the other hand, (i, j) is added to $\mathcal{E}$ if the adversary made either a $(\texttt{rekey}, i, j)$ query or a $(\texttt{reencrypt}, i, j, k)$ query where the k-th ciphertext is a re-encryption of the challenge ciphertext (see Game 4 and Figure 1). This is less restrictive than in CPA: the adversary can make re-encryption queries to a corrupt user that is reachable from the challenge key *unless* it is related to the challenge ciphertext.

For comparison we have reformulated the selective notions defined in Games 1 and 2 using a recoding graph instead of explicit aborts. Games 9 and 10 given in Appendix A define the exact same notions as Games 1 and 2, respectively.

Definition 5 (PRE-CPA-security). *A PRE scheme is (s, ϵ)-adaptively secure against chosen-plaintext attack if* $\mathsf{CPA}^0 \approx_{(s,\epsilon)} \mathsf{CPA}^1$*, where* CPA^b *is defined in Game 3.*

Definition 6 (PRE-HRA-security). *A PRE scheme is (s, ϵ)-adaptively secure against honest-reencryption attack if* $\mathsf{HRA}^0 \approx_{(s,\epsilon)} \mathsf{HRA}^1$*, where* HRA^b *is defined in Game 4.*

3 Preliminaries

This section provides the background necessary for the main results in §4. We start with the security assumptions on PREs that allow us to prove adaptive security (§3.1) and then give an overview of the framework of [JKK+17] (§3.2), the description of the pebbling game that is used in the design of the hybrids (§3.3).

[5] The selective CPA notion (Game 1) is in fact more restrictive in that it does not allow re-keys and re-encryptions from *any* honest user to a corrupt user.

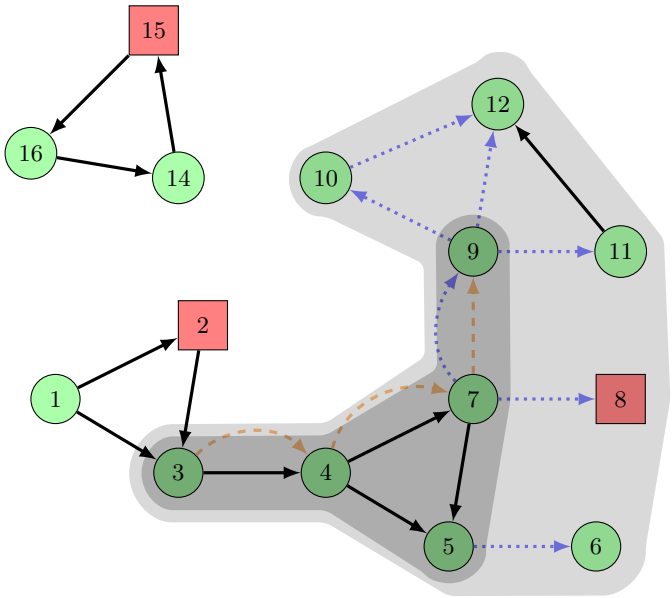

Fig. 1: Recoding graph. The (round) green nodes represent the honest users, whereas the (square) red nodes are the corrupted ones. The edges denote the recoding information. In particular, the (solid) black edges are the re-keys, the (dashed) orange edges are the re-encryptions related to the challenge ciphertext (therefore, 3 is the challenge vertex) and (dotted) blue edges represent the remaining re-encryptions. For CPA, all the edges are counted, but for HRA the blue (dotted) edges are *not* counted. The subgraph of the recoding graph that forms the challenge graph (cf. §4) is shaded: the darker inner shading for HRA, whereas the lighter outer shading is the challenge graph for CPA. Note that the edge $(7, 8)$ is valid in the case of HRA, but invalid for CPA (and therefore the CPA challenger would abort at the end of such an execution.)

Challenger $\mathsf{CPA}^b(1^\kappa, 1^\lambda, n)$

1: Set $\mathcal{C}, \mathcal{E} = \emptyset$ ▷ Stores corrupt keys and issued re-keys and re-encryptions
2: $\mathtt{pp} \leftarrow \mathsf{PRE.S}(1^\kappa, 1^\lambda)$, $(\mathtt{pk}_1, \mathtt{sk}_1), \ldots, (\mathtt{pk}_n, \mathtt{sk}_n) \leftarrow \mathsf{PRE.K}(\mathtt{pp})$ ▷ Generate keys
3: $\forall i, j \in [n], i \neq j : \mathtt{rk}_{i,j} \leftarrow \mathsf{PRE.RK}((\mathtt{pk}_i, \mathtt{sk}_i), \mathtt{pk}_j)$ ▷ Generate re-keys
4: $b' \leftarrow \mathsf{A}^{(\mathtt{corrupt},\cdot),(\mathtt{rekey},\cdot,\cdot),(\mathtt{reencrypt},\cdot,\cdot,\cdot),(\mathtt{challenge},\cdot,\cdot,\cdot)}(\mathtt{pp}, \mathtt{pk}_1, \ldots, \mathtt{pk}_n)$
5: **if** A made call $(\mathtt{challenge}, i^*, \cdot, \cdot)$ for some i^* **then** ▷ Check abort conditions
6: **if** $\exists\, i \in \mathcal{C} : i^*$ is connected to i in $([n], \mathcal{E})$ **then return** 0 **end if**
7: **end if**
8: **return** b'

Oracle $(\mathtt{corrupt}, i)$

1: Add i to $\mathcal{C}$
2: **return** $\mathtt{sk}_i$

Oracle $(\mathtt{rekey}, i, j)$

1: Add (i, j) to $\mathcal{E}$ ▷ Add to recoding graph
2: **return** $\mathtt{rk}_{i,j}$

Oracle $(\mathtt{reencrypt}, i, j, (c_i, \ell))$

1: Add (i, j) to $\mathcal{E}$ ▷ Add to recoding graph
2: **return** $(c_j, \ell + 1) \leftarrow \mathsf{PRE.RE}(\mathtt{rk}_{i,j}, \mathtt{pk}_i, \mathtt{pk}_j, (c_i, \ell))$

Oracle $(\mathtt{challenge}, i^*, (m_0^*, m_1^*), \ell^*)$ ▷ Single access

1: **return** $(c_{i^*}, \ell^*) \leftarrow \mathsf{PRE.E}(\mathtt{pk}_{i^*}, (m_b^*, \ell^*))$

Game 3: PRE-CPA

3.1 Security Assumptions on PRE

We describe the three security properties of PRE schemes that allow us to prove adaptive security: indistinguishability, key-privacy and source-hiding.

Indistinguishability of ciphertexts. For proxy re-encryption, we require the notion of indistinguishability, as defined for public-key encryption in [GM82], to hold on *all* levels:

Definition 7 (Indistinguishability). *A proxy re-encryption scheme* PRE *has* (s, ϵ)*-indistinguishable ciphertexts if* $\mathsf{IND}^0 \approx_{(s,\epsilon)} \mathsf{IND}^1$ *with* IND *as in Game 5.*

Key-Privacy. The original notion of key-privacy for PREs, which we refer to as "strong" key-privacy, was introduced in [ABH09]. It is modelled by a security game similar to sPRE-CPA: the adversary has access to `corrupt`, `rekey` and `reencrypt` oracles, but as a challenge it has to distinguish a real re-key from a re-key sampled *uniformly at random* from the support of re-keys. We refer the readers to [ABH09] for the details.

We only need a weaker definition stating that re-keys should hide the source keys. That is, the re-key $\mathtt{rk}_{0,1}$ from source $(\mathtt{pk}_0, \mathtt{sk}_0)$ to a target key $\mathtt{pk}_1$ should be indistinguishable from a random source to $\mathtt{pk}_1$. In addition, we need this property to hold with respect to multiple re-keys. More formally, the security game for weak key-privacy is given in Game 6 where the simulator RK^* is defined as

$$\mathsf{RK}^*(\mathtt{pp}, \mathtt{pk}_1) := \mathsf{RK}((\mathtt{pk}_0, \mathtt{sk}_0), \mathtt{pk}_1) : (\mathtt{pk}_0, \mathtt{sk}_0) \leftarrow \mathsf{K}(\mathtt{pp}).$$

Challenger $\mathsf{HRA}^b(1^\kappa, 1^\lambda, n)$
1: Set $\mathcal{C}, \mathcal{L}, \mathcal{L}^* = \emptyset$ and $C = 0$ ▷ $\mathcal{L}$ stores honest enc's, $\mathcal{L}^*$ marks challenge reenc's
2: $\mathcal{E} = \emptyset$ ▷ The edges of the recoding graph
3: $\mathtt{pp} \leftarrow \mathsf{PRE.S}(1^\kappa, 1^\lambda)$, $(\mathtt{pk}_1, \mathtt{sk}_1), \ldots, (\mathtt{pk}_n, \mathtt{sk}_n) \leftarrow \mathsf{PRE.K}(\mathtt{pp})$ ▷ Generate keys
4: $\forall i, j \in [n], i \neq j : \mathtt{rk}_{i,j} \leftarrow \mathsf{PRE.RK}((\mathtt{pk}_i, \mathtt{sk}_i), \mathtt{pk}_j)$ ▷ Generate re-keys
5: $b' \leftarrow \mathsf{A}^{(\mathtt{corrupt},\cdot),(\mathtt{rekey},\cdot,\cdot),(\mathtt{encrypt},\cdot,\cdot),(\mathtt{reencrypt},\cdot,\cdot,\cdot),(\mathtt{challenge},\cdot,\cdot,\cdot)}(\mathtt{pp}, \mathtt{pk}_1, \ldots, \mathtt{pk}_n)$
6: **if** A made call $(\mathtt{challenge}, i^*, \cdot, \cdot)$ for some i^* **then** ▷ Check abort conditions
7: **if** $\exists\, i \in \mathcal{C} : i^*$ is connected to i **then return** 0 **end if**
8: **end if**
9: **return** b'

Oracle $(\mathtt{corrupt}, i)$
1: Add i to $\mathcal{C}$
2: **return** $\mathtt{sk}_i$

Oracle $(\mathtt{rekey}, i, j)$
1: Add (i, j) to $\mathcal{E}$ ▷ Add to recoding graph
2: **return** $\mathtt{rk}_{i,j}$

Oracle $(\mathtt{encrypt}, i, (m, \ell))$
1: $c \leftarrow \mathsf{PRE.E}(\mathtt{pk}_i, (m, \ell))$, increment C and add $(C, i, m, (c, \ell))$ to $\mathcal{L}$
2: **return** c

Oracle $(\mathtt{reencrypt}, i, j, k)$
1: Retrieve $(k, i, m, (c_i, \ell))$ from $\mathcal{L}$
2: $(c_j, \ell + 1) \leftarrow \mathsf{PRE.RE}(\mathtt{rk}_{i,j}, \mathtt{pk}_i, \mathtt{pk}_j, (c_i, \ell))$
3: Increment C and add $(C, j, m, (c_j, \ell + 1))$ to $\mathcal{L}$
4: **if** $k \in \mathcal{L}^*$ **then** ▷ c_j derived from challenge
5: Add C to $\mathcal{L}^*$ and add (i, j) to $\mathcal{E}$ ▷ Add to recoding graph
6: **end if**
7: **return** $(c_j, \ell + 1)$

Oracle $(\mathtt{challenge}, i^*, (m_0^*, m_1^*), \ell^*)$ ▷ Single access
1: Compute $(c_{i^*}, \ell^*) \leftarrow \mathsf{PRE.E}(\mathtt{pk}_{i^*}, (m_b^*, \ell^*))$
2: Increment C, add $(C, i^*, m_b^*, (c_{i^*}, \ell^*))$ to $\mathcal{L}$ and C to $\mathcal{L}^*$
3: **return** (c_{i^*}, ℓ^*)

Game 4: PRE-HRA

Challenger $\mathsf{IND}^b(1^\kappa, 1^\lambda)$
1: $\mathtt{pp} \leftarrow \mathsf{PRE.S}(1^\kappa, 1^\lambda)$, $(\mathtt{pk}, \mathtt{sk}) \leftarrow \mathsf{PRE.K}(\mathtt{pp})$
2: **return** $b' \leftarrow \mathsf{A}^{(\mathtt{challenge},\cdot,\cdot)}(\mathtt{pp}, \mathtt{pk})$

Oracle $(\mathtt{challenge}, (m_0^*, m_1^*), \ell^*)$
1: **return** $\mathsf{PRE.E}(\mathtt{pk}, (m_b^*, \ell^*))$

Game 5: Security game IND for ciphertext indistinguishability

Definition 8 (Weak key-privacy). *Let $\delta \in \mathbb{N}$. A proxy re-encryption scheme* PRE *is (s, ϵ, δ)-weakly key-private if* $\mathsf{KP}^0 \approx_{(s,\epsilon)} \mathsf{KP}^1$ *with* KP *as in Game 6.*

Source-hiding. Source-hiding is a special case of re-encryption-simulatability, a notion that was introduced in [Coh17]. It requires that re-encryptions can be simulated without knowledge of the secret key. In particular, the simulated re-encryptions should be indistinguishable from re-encrypted ciphertexts even when given the secret keys for the source and target public keys, as well as the

re-key that was used for re-encryption (hence the notion of indistinguishability is at least that of statistical indistinguishability). A PRE scheme is called *source-hiding* if re-encrypted ciphertexts have the same distribution as "fresh" ciphertexts, i.e., the encryption algorithm can be used as a simulator for re-encryption.

Definition 9 (Source-hiding). *A proxy re-encryption scheme* PRE *is* (s, ϵ)*-source-hiding if* $\mathsf{SH}^0 \approx_{(s,\epsilon)} \mathsf{SH}^1$*, with* SH *as defined in Game 7.*

3.2 Overview of [JKK+17]

Random guessing. A standard way to prove adaptive security is to first show security in a "selective" version of the adaptive game, in which the adversary commits to some of its future choices, and then use random guessing of the adversary's commitment to reduce adaptive security to selective security. For instance, consider the indistinguishability game for identity-based encryption: in the selective counterpart the adversary commits to the challenge identity at the start of the game, and the adaptive to selective reduction then works by guessing the challenge identity. More formally, let G^0 and G^1 denote the two adaptive games. For some function $g\colon \{0,1\}^* \to \mathcal{W}$ we define below the selective games $\mathsf{H}^0 = \mathsf{SEL}_{\mathcal{W}}[\mathsf{G}^0, g]$ and $\mathsf{H}^1 = \mathsf{SEL}_{\mathcal{W}}[\mathsf{G}^1, g]$ where the adversary commits to some information $w \in \mathcal{W}$ – for the case of IBE, $\mathcal{W}$ is the set of all identities. Note that the selective game gets a commitment w from the adversary but essentially ignores it during the rest of the game. It checks that the commitment matches what actually happened during the game only at the very end of the game; whether w matches is defined via the function g.

Definition 10 (Fully selectivised game [JKK+17]). *Given an (adaptive) game* G *and some function* $g\colon \{0,1\}^* \to \mathcal{W}$*, the selectivised game* $\mathsf{H} = \mathsf{SEL}_{\mathcal{W}}[\mathsf{G}, g]$ *is defined as follows. The adversary* A *first sends a commitment* $w \in \mathcal{W}$ *to* H*. Then* H *runs the challenger* G *against* A*, at the end of which* G *outputs a bit* $\hat{b}$*. Let* `transcript` *denote all communication exchanged between* G *and* A*. If* $g(\texttt{transcript}) = w$ *then* H *outputs the bit* $\hat{b}$ *and else it outputs* 0*.*

Next, suppose that the selective security is proved using a hybrid argument. That is, to show the indistinguishability of H^0 and H^1 suppose we have a sequence of $\tau + 1$ (selective) hybrid games $\mathsf{H}^0 = \mathsf{H}_0, \mathsf{H}_1, \ldots, \mathsf{H}_\tau = \mathsf{H}^1$ (see Figure 2). If

Challenger $\mathsf{KP}^b(1^\kappa, 1^\lambda)$
1: $\mathtt{pp} \leftarrow \mathsf{PRE.S}(1^\kappa, 1^\lambda)$, $(\mathtt{pk}_0, \mathtt{sk}_0), \ldots, (\mathtt{pk}_\delta, \mathtt{sk}_\delta) \leftarrow \mathsf{K}(\mathtt{pp})$
2: $\forall j \in [\delta] : \mathtt{rk}^{(0)}_{0,j} \leftarrow \mathsf{RK}((\mathtt{pk}_0, \mathtt{sk}_0), \mathtt{pk}_j)$
3: $\mathtt{rk}^{(1)}_{0,j} \leftarrow \mathsf{RK}^*(\mathtt{pp}, \mathtt{pk}_j)$
4: **return** $b' \leftarrow \mathsf{A}(\mathtt{pp}, \mathtt{pk}_0, \ldots, \mathtt{pk}_\delta, \mathtt{rk}^{(b)}_{0,1}, \ldots, \mathtt{rk}^{(b)}_{0,\delta})$

Game 6: Security game KP for weak key-privacy

Challenger $\mathsf{SH}^b(1^\kappa, 1^\lambda)$

1: $\mathtt{pp} \leftarrow \mathsf{PRE.S}(1^\kappa, 1^\lambda)$
2: $(\mathtt{pk}_0, \mathtt{sk}_0), (\mathtt{pk}_1, \mathtt{sk}_1) \leftarrow \mathsf{PRE.K}(\mathtt{pp})$
3: $\mathtt{rk}_{0,1} \leftarrow \mathsf{PRE.RK}((\mathtt{pk}_0, \mathtt{sk}_0), \mathtt{pk}_1)$
4: $b' \leftarrow \mathsf{A}^{(\mathtt{challenge},\cdot,\cdot)}(\mathtt{pp}, (\mathtt{pk}_0, \mathtt{sk}_0), (\mathtt{pk}_1, \mathtt{sk}_1), \mathtt{rk}_{0,1})$
5: **return** b'

Oracle $(\mathtt{challenge}, m^*, \ell^*)$ ▷ $\ell^* \in [\lambda - 1]$

1: $c_0 \leftarrow \mathsf{PRE.E}(\mathtt{pk}_0, (m^*, \ell^*))$
2: $c_1^{(0)} \leftarrow \mathsf{PRE.RE}(\mathtt{rk}_{0,1}, \mathtt{pk}_0, \mathtt{pk}_1, c_0)$ ▷ Real re-encryption
3: $c_1^{(1)} \leftarrow \mathsf{PRE.E}(\mathtt{pk}_1, (m^*, \ell^* + 1))$ ▷ Simulate re-encryption
4: **return** $(c_0, c_1^{(b)})$

Game 7: Security game SH for source hiding

we only assume that neighbouring hybrids $\mathsf{H}_i, \mathsf{H}_{i+1}$ are indistinguishable, then by combining the hybrid argument and random guessing we get that G^0 and G^1 are indistinguishable with a loss in distinguishing advantage of $\tau \cdot |\mathcal{W}|$. The factor of $|\mathcal{W}|$ is the cost of the random guessing, whereas the factor of τ is due to the hybrid argument. This is stated in the following (recall that $s_{\mathcal{W}}$ denotes the complexity of sampling from $\mathcal{W}$):

Theorem 3 ([BB04, JKK+17]). *Assume we have two games defined via (adaptive) challengers G^0 and G^1 respectively. Let $g\colon \{0,1\}^* \to \mathcal{W}$ be an arbitrary function and define the selectivised games $\mathsf{H}^b = \mathsf{SEL}_{\mathcal{W}}[\mathsf{G}^b, g]$ for $b \in \{0,1\}$. Also assume that for each $i \in [\tau]$, the games $\mathsf{H}_{i-1}, \mathsf{H}_i$ are (s, ϵ)-indistinguishable. Then, G^0 and G^1 are $(s - s_{\mathcal{W}},\ \epsilon \cdot \tau \cdot |\mathcal{W}|)$-indistinguishable.*

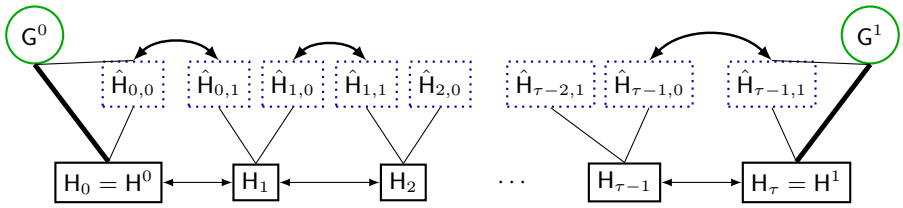

Fig. 2: A schematic diagram showing the relationship between adaptive, fully selective and partially selective hybrids. The adaptive games G^0 and G^1 are in green (circles); the fully selective games $\mathsf{H}_0, \ldots, \mathsf{H}_\tau$ are in solid black (boxes); and the partially selective games $\hat{\mathsf{H}}_{0,0}, \hat{\mathsf{H}}_{0,1}, \ldots, \hat{\mathsf{H}}_{\tau-1,0}, \hat{\mathsf{H}}_{\tau-1,1}$ are in (dotted) blue boxes. The arrows indicate indistinguishability.

The framework. In some cases only *part* of the information that the adversary commits to is used in simulating the intermediate hybrids, but when considering

all the hybrids the whole commitment is being used. For example, the simulation of an intermediate hybrid in the case of IBE could rely on only certain bits of the challenge identity. It is shown in [JKK+17] that the security loss in such cases can be limited to the maximum size of the information used across any two successive hybrids.

More formally, [JKK+17] makes a stronger assumption: not only are neighbouring hybrids $\mathsf{H}_i, \mathsf{H}_{i+1}$ indistinguishable, but they are "selectivised" versions, of "partially" selective games $\hat{\mathsf{H}}_{i,0}, \hat{\mathsf{H}}_{i,1}$ which are already indistinguishable. In particular, for each pair of neighbouring hybrids $\mathsf{H}_i, \mathsf{H}_{i+1}$ there exists a pair of partially selective hybrids $\hat{\mathsf{H}}_{i,0}, \hat{\mathsf{H}}_{i,1}$ (see Figure 2) in which the adversary commits to much less information $h_i(w) \in \mathcal{U}$ instead of $w \in \mathcal{W}$. The selectivised game essentially ignores w and only relies on the partial information $u = h_i(w)$ during the course of the game but at the very end it still checks that the full commitment w matches what actually happened during the game.

Definition 11 (Partially selectivised game [JKK+17]). *Assume $\hat{\mathsf{H}}$ is a game which expects to receive some commitment $u \in \mathcal{U}$ from the adversary in the beginning. Given functions $g\colon \{0,1\}^* \to \mathcal{W}$ and $h\colon \mathcal{W} \to \mathcal{U}$ the partially selectivised game $\mathsf{H} = \mathsf{SEL}_{\mathcal{U}\to\mathcal{W}}[\hat{\mathsf{H}}, g, h]$ is defined as follows. The adversary A first sends a commitment $w \in \mathcal{W}$ to H and H begins running $\hat{\mathsf{H}}$ and passes it $u = h(w)$. It then continues running the game between $\hat{\mathsf{H}}$ and A at the end of which $\hat{\mathsf{H}}$ outputs a bit $\hat{b}$. Let $\texttt{transcript}$ denote all communication exchanged between $\hat{\mathsf{H}}$ and A. If $g(\texttt{transcript}) = w$ then H outputs the bit $\hat{b}$ and else it outputs 0.*

Note that different pairs of partially selective hybrids $\hat{\mathsf{H}}_{i,0}, \hat{\mathsf{H}}_{i,1}$ might rely on completely different partial information $h_i(w)$ about the adversary's choices. The partially selective hybrid associated to each H_i can thus be different when we compare $\mathsf{H}_{i-1}, \mathsf{H}_i$ (in which case it is $\hat{\mathsf{H}}_{i-1,1}$) and when we compare H_i and H_{i+1} (in which case it is $\hat{\mathsf{H}}_{i,0}$) – see Figure 2. The next theorem shows that we only incur a security loss proportional to $|\mathcal{U}|$ rather than $|\mathcal{W}|$ if we can define a sequence of partially selective hybrids which only require commitments from $\mathcal{U}$.

Theorem 4 ([JKK+17]). *Let G^0 and G^1 be two adaptive games. For some function $g\colon \{0,1\}^* \to \mathcal{W}$ we define the selectivised games $\mathsf{H}^0 = \mathsf{SEL}_{\mathcal{W}}[\mathsf{G}^0, g]$, $\mathsf{H}^1 = \mathsf{SEL}_{\mathcal{W}}[\mathsf{G}^1, g]$. Let $\mathsf{H}^0 = \mathsf{H}_0, \mathsf{H}_1, \ldots, \mathsf{H}_\tau = \mathsf{H}^1$ be some sequence of hybrid games. Assume that for each $i \in [0, \tau - 1]$, there exists a function $h_i\colon \mathcal{W} \to \mathcal{U}$ and games $\hat{\mathsf{H}}_{i,0}, \hat{\mathsf{H}}_{i,1}$ such that*

$$\mathsf{H}_i \equiv \mathsf{SEL}_{\mathcal{U}\to\mathcal{W}}[\hat{\mathsf{H}}_{i,0}, g, h_i] \quad \text{and} \quad \mathsf{H}_{i+1} \equiv \mathsf{SEL}_{\mathcal{U}\to\mathcal{W}}[\hat{\mathsf{H}}_{i,1}, g, h_i]. \tag{1}$$

Furthermore, if $\hat{\mathsf{H}}_{i,0}, \hat{\mathsf{H}}_{i,1}$ are (s, ϵ)-indistinguishable for all $i \in [0, \tau - 1]$, then G^0 and G^1 are $(s - s_{\mathcal{U}}, \epsilon \cdot \tau \cdot |\mathcal{U}|)$-indistinguishable.

3.3 Pebbling Games

The reversible pebbling game on DAGs was introduced in [Ben89] to model reversible computation. We define a variant in which the pebbling rules have

been adapted for application to PREs. In particular, the rule is the opposite of that in [Ben89]: a pebble can be placed on or removed from a vertex if all its *children* carry a pebble.[6]

Definition 12. *A reversible pebbling of a directed acyclic graph $G = (\mathcal{V}, \mathcal{E})$ with a unique source vertex i^* is a sequence $\mathcal{P} := (\mathcal{P}_0, \ldots, \mathcal{P}_\tau)$ of pebbling configurations $\mathcal{P}_t \subseteq \mathcal{V}$. Two subsequent configurations differ only in one vertex and the following rule is respected in a move: a pebble can be placed on or removed from a vertex iff all its children carry a pebble. That is, $\mathcal{P}$ is a valid sequence iff*

$$\forall t \in [\tau]\ \exists!\, i \in \mathcal{P}_{t-1} \triangle \mathcal{P}_t \quad \textit{and} \quad \mathsf{children}(i, G) \subseteq \mathcal{P}_{t-1}.$$

Starting with an empty graph (i.e., $\mathcal{P}_0 = \emptyset$), the goal of the game is to place a pebble on the source (i.e., $i^* \in \mathcal{P}_\tau$).

For a DAG G, let $\mathcal{P}_G$ denote the set of all valid reversible pebbling sequences (as per Definition 12) for G. The *time complexity* of a *particular* sequence $\mathcal{P} = (\mathcal{P}_0, \ldots, \mathcal{P}_\tau)$ for a DAG G is defined as $\tau_G(\mathcal{P}) := \tau$, whereas its space complexity is defined as

$$\sigma_G(\mathcal{P}) := \max_{t \in [0,\tau]} |\mathcal{P}_t|.$$

Definition 13 (Space- and time-complexity of a class of DAGs). *We say that a class of DAGs $\mathcal{G}$ has time complexity τ and space complexity σ if*

$$\forall G \in \mathcal{G}\ \exists \mathcal{P} \in \mathcal{P}_G : \tau_G(\mathcal{P}) \leq \tau \wedge \sigma_G(\mathcal{P}) \leq \sigma.$$

Concrete Bounds. We compute the pebbling complexity for the following classes of *single-source* graphs on n vertices:

[Lemma 2]: $\mathcal{G}(n, \delta, d)$, DAGs with outdegree δ and depth d;
[Lemma 3]: $\mathcal{B}(n) = \mathcal{G}(n, 2, \log n)$, complete binary trees of size n; and
[Lemma 4]: $\mathcal{C}(n) = \mathcal{G}(n, 1, n)$, chains of length n.

Lemma 2 (Arbitrary DAGs). *$\mathcal{G}(n, \delta, d)$ has space-complexity $(\delta + 1) \cdot d$ and time-complexity $(2\delta)^d$.*

Proof. The pebbling algorithm P_1 that pebbles any graph in $\mathcal{G}(n, \delta, d)$ using at most $(\delta + 1) \cdot d$ pebbles in at most $(2\delta)^d$ moves is given in Algorithm 1. The strategy is recursive in the depth, and to pebble a vertex i, P_1 (recursively) pebbles all of i's children. We consider the time and space complexity of the pebbling sequence defined by P_1 as functions $\tau(d)$ and $\sigma(d)$ of the depth d. Then the number of moves incurred is captured by the expression $\tau(d) \leq 2\delta \cdot \tau(d-1)$ with $\tau(1) \leq 2\delta$, and hence $\tau(\delta) \leq (2\delta)^d$. The number of pebbles, on the other hand, is captured by the recursion $\sigma(d) < (\delta + 1) + \sigma(d-1)$ with $\sigma(1) = \delta + 1$; hence $\sigma(d) = (\delta + 1) \cdot d$. □

[6] Alternatively, one can think of the pebbling game in Definition 12 as the classical reversible pebbling game played on a DAG whose edges have their direction flipped.

Lemma 3 (Complete binary trees). *$\mathcal{B}(n)$ has space-complexity $3 \cdot \log n$ and time-complexity n^2.*

Proof. This follows from Lemma 2 on substituting $\delta = 2$ and $d = \log n$. □

Lemma 4 (Chains). *$\mathcal{C}(n)$ has space-complexity $\log n + 1$ and time-complexity $3^{\log n}$.*

Proof. A pebbling algorithm P_2 for pebbling the source vertex of $\mathcal{C}(n)$, where n is a power of two, is given in Algorithm 2 – the argument can be easily extended for arbitrary n and vertex. Let $\sigma(n)$ and $\tau(n)$ denote the space and time complexity of the pebbling defined by P_2 for chains of length n. Then the number of pebbles used by P_2 is captured by the recursion $\sigma(n) = \sigma(n/2) + 1$, with $\sigma(1) = 1$. The number of moves, on the other hand, is captured by $\tau(n) = 3 \cdot \tau(n/2)$ with $\tau(1) = 1$. Therefore, $\sigma(n) = \log n + 1$ and $\tau(n) = 3^{\log n}$. □

$\mathsf{P}_1(G, t)$
1: Set $T = 0$ and $\mathcal{P} = \emptyset$ ▷ Global variables
2: Let i^* denote the source of the graph G
3: **return** $\mathsf{P}'_1(G, t, i^*, i^*)$

$\mathsf{P}'_1(G, t, i^*, i)$ ▷ i^* denotes the source; i is the vertex currently pebbled
1: **for** $j \in \mathsf{children}(i, G)$ **do** $\mathsf{P}'_1(G, t, i^*, j)$ ▷ Pebble children recursively
2: **if** $i \in \mathcal{P}$ **then** $\mathcal{P} := \mathcal{P} \setminus \{i\}$ ▷ Unpebble if i already pebbled
3: **else**
4: $\quad \mathcal{P} := \mathcal{P} \cup \{i\}$ ▷ Place pebble on i
5: $\quad$ **if** $i = i^*$ **then return** $\mathcal{P}$ ▷ Placed pebble on source
6: **end if**
7: Increment T
8: **if** $T = t$ **then return** $\mathcal{P}$ ▷ $\mathcal{P}$ currently stores the t-th pebbling configuration
9: **for** $j \in \mathsf{children}(i, G)$ **do** $\mathsf{P}'_1(G, t, i^*, j)$ ▷ Unpebble children

Algorithm 1: A pebbling strategy for general DAGs.

4 Framework for Adaptive Security

In this section we demonstrate, using the framework of [JKK+17], how adaptive security can be achieved for PREs. In particular, we show that for CPA and derive an analogous result for HRA. As for the applications given in [JKK+17], we use pebbling games on DAGs to design the hybrid games. Each pebbling configuration uniquely determines a hybrid game bridging the two real games CPA^0 and CPA^1. The DAG that we pebble in the proof is the subgraph of the recoding graph that is reachable from the challenge i^* (via the edges $\mathcal{E}$ defined during the game); it is thus a subgraph of the recoding graph with one unique source i^*,

$\mathsf{P}_2(n,t)$ ▷ n denotes the length of the chain
1: Set $T=0$ and $\mathcal{P}=\emptyset$ ▷ Global variables
2: **return** $\mathsf{P}'_2(1,n,t)$

$\mathsf{P}'_2(i,j,t)$ ▷ i and j denote the end points of the active chain
1: **if** $i=j$ **then** ▷ End of recursion
2: **if** $i\in\mathcal{P}$ **then** $\mathcal{P}:=\mathcal{P}\setminus\{i\}$ ▷ Unpebble if i already pebbled
3: **else**
4: $\mathcal{P}:=\mathcal{P}\cup\{i\}$ ▷ Place pebble on i
5: **if** $i=1$ **then return** $\mathcal{P}$ ▷ Placed pebble on challenge
6: **end if**
7: Increment T
8: **else**
9: $\mathsf{P}'_2(i,(i+j-1)/2,t)$ ▷ Recursively pebble left half
10: $\mathsf{P}'_2((i+j+1)/2,j,t)$ ▷ Recursively pebble right half
11: $\mathsf{P}'_2(i,(i+j-1)/2,t)$ ▷ Recursively unpebble left half
12: **end if**
13: **if** $T=t$ **then return** $\mathcal{P}$ ▷ $\mathcal{P}$ currently stores the t-th pebbling configuration

Algorithm 2: Pebbling strategy for chains with n vertices, for n a power of two.

which we call the *challenge graph*. A pebble on a vertex allows the simulation of the hybrid to be carried out *without* the knowledge of the secret key associated with that vertex. The pebbling rules will ensure that hybrids corresponding to two successive pebbling configurations can be proven indistinguishable assuming key-privacy.

4.1 Adaptive Security Against Chosen-Plaintext Attack

We first show how a pebbling sequence on the challenge graph defines a sequence of fully selective hybrids (Lemma 5), and then prove that these hybrids are partially selectivised (Lemma 6).

4.1.1 Fully Selective Hybrids. In the fully selectivised version of PRE-CPA (Game 3), A first makes a commitment $\hat{G}$ to the challenge graph. Any correct commitment $\hat{G}$ must therefore have one unique source, which we denote by $\hat{\imath}$. The selective challenger is thus $\mathsf{SEL}_{\mathcal{G}}[\mathsf{CPA}^b, g]$, where g is the function that extracts the recoding graph G and the challenge user i^* from the transcript and returns the challenge graph, i.e., the subgraph of G reachable from i^*. Note that this is fundamentally different from the original selective game (i.e., sCPA in Game 1) where the adversary commits, beforehand, to *the set of corrupt public keys*.

Each hybrid is associated with a pebbling configuration $\mathcal{P}_t$ and a bit b, and we consider the sequence of hybrids $\mathsf{H}^0_0,\ldots,\mathsf{H}^0_\tau,\mathsf{H}^1_\tau,\ldots,\mathsf{H}^1_0$. The pebbling state of a vertex dictates how the outgoing re-key and re-encrypt queries are simulated, whereas the bit determines how the challenge query is answered. To be precise, in game H^b_t, for each pebbled vertex in $\mathcal{P}_t$ all used re-keys outgoing from that vertex are faked, and the challenge query is answered by an encryption of m^*_b.

(Rekeys outgoing from pebbled vertices that are not used for any queries are defined as real re-keys.) Observe that the secret key corresponding to a vertex is used only for the generation of the re-keys outgoing from that vertex; the simulation of a hybrid can thus be carried out *without* knowledge of the secret keys corresponding to the pebbled vertices (as the non-queried re-keys need not be generated).

Since the initial pebbling configuration is the empty set, H_0^0 and H_0^1 correspond to the (fully selectivised) games $\mathsf{SEL}_{\mathcal{G}}[\mathsf{CPA}^0, g]$ and $\mathsf{SEL}_{\mathcal{G}}[\mathsf{CPA}^1, g]$, respectively. Now, consider the middle hybrids H_τ^0 and H_τ^1: they are the same except for the response to the challenge query which is the encryption of m_0^* in the former and the encryption of m_1^* in the latter. Since the pebbling configuration $\mathcal{P}_\tau$, by definition, contains a pebble on the challenge vertex i^*, the simulation of this hybrid can be carried out without knowledge of the secret key corresponding to i^*. This means we can reduce indistinguishability of the PRE to the indistinguishability of these two hybrids. To be precise, the reduction embeds the challenge public key at $\hat{\imath}$, which is defined by the commitment $\hat{G}$ and replies to the challenge query (in the CPA game) by sending the challenge ciphertext (of the indistinguishability game). Note that if $i^* \neq \hat{\imath}$, that is, the commitment $\hat{G}$ doesn't coincide with the transcript of the CPA game, then the hybrid returns 0 anyway. The reduction is formally defined in Algorithm 4.

Next, consider any two hybrids H_t^b and H_{t+1}^b, $t \in [0, \tau - 1]$ and $b \in \{0, 1\}$. Also, assume $\mathcal{P}_{t+1}$ results from $\mathcal{P}_t$ by placing a pebble on the vertex i_0 (the case when a pebble is removed can be argued analogously). The simulation of H_t^b and H_{t+1}^b is the same except for the (used) re-keys outgoing from i_0: in H_t^b they are all real whereas in H_{t+1}^b they are all fake. By the rules of the pebbling game, the children of i_0 all carry pebbles in the configurations $\mathcal{P}_t$ and $\mathcal{P}_{t+1}$; therefore the simulation need not know the corresponding secret keys. This means that we can prove indistinguishability of H_t^b and H_{t+1}^b from weak key-privacy: the reduction embeds the (key-privacy) challenge public keys $\mathtt{pk}_0, \ldots, \mathtt{pk}_\delta$ at i_0 and its children, and uses the challenge re-keys $\mathtt{rk}_{0,1}, \ldots, \mathtt{rk}_{0,\delta}$ to simulate the re-key oracle for queries from i_0 to its children. The reduction is formally defined in Algorithm 5. (Note that the simulation of the reduction in Algorithm 5 is perfect: if the commitment $\hat{G}$ does not match with the transcript, it returns 0; else, we have $\hat{\imath} = i^*$ and by definition of the pebbling, i_0 is reachable from $\hat{\imath} = i^*$ and so are its children $i_1, \ldots, i_\delta$. If the adversary corrupts any of these, then the game returns 0.)

In summary, we get a sequence of hybrids $\mathsf{SEL}_{\mathcal{G}}[\mathsf{CPA}^0, g] = \mathsf{H}_0^0, \ldots, \mathsf{H}_\tau^0, \mathsf{H}_\tau^1, \ldots, \mathsf{H}_0^1 = \mathsf{SEL}_{\mathcal{G}}[\mathsf{CPA}^1, g]$, where each pair of subsequent hybrids can be proved indistinguishable. Security in the fully selectivised CPA game follows by Lemma 1. We state this formally in Lemma 5 below.

Lemma 5 (Security against fully selectivised PRE-CPA). *Consider the sequence of hybrids $\mathsf{H}_0^0, \ldots, \mathsf{H}_\tau^0, \mathsf{H}_\tau^1, \ldots, \mathsf{H}_0^1$, where H_t^b is defined in Algorithm 3 using the pebbling configuration $\mathcal{P}_t$. H_0^b is the fully selectivised game of CPA^b: i.e., $\mathsf{H}_0^b = \mathsf{SEL}_{\mathcal{G}}[\mathsf{CPA}^b, g]$ where g extracts the challenge graph (subgraph reachable from the challenge vertex) from a transcript. Moreover, if the adversary*

Hybrid $\mathsf{H}_t^b(1^\kappa, 1^\lambda, n)$

1: Obtain the challenge graph $\hat{G} \in \mathcal{G}(n, \delta, d)$ from A
2: Compute $\mathcal{P}_t \leftarrow \mathsf{P}(\hat{G}, t)$ ▷ The t-th pebbling configuration
3: Set $\mathcal{C}, \mathcal{E} = \emptyset$ ▷ Stores corrupt keys and issued re-keys and re-encryptions
4: $\mathtt{pp} \leftarrow \mathsf{PRE.S}(1^\kappa, 1^\lambda)$, $(\mathtt{pk}_1, \mathtt{sk}_1), \ldots, (\mathtt{pk}_n, \mathtt{sk}_n) \leftarrow \mathsf{PRE.K}(\mathtt{pp})$
5: $\forall i \in \mathcal{P}_t, \forall j \in \mathsf{children}(i, \hat{G})$: $\mathtt{rk}_{i,j} \leftarrow \mathsf{RK}^*(\mathtt{pp}, \mathtt{pk}_j)$ ▷ Fake re-keys
6: $\forall i \in \mathcal{P}_t, \forall j \in [n] \setminus \{\mathsf{children}(i, \hat{G}) \cup i\}$: $\mathtt{rk}_{i,j} \leftarrow \mathsf{PRE.RK}((\mathtt{pk}_i, \mathtt{sk}_i), \mathtt{pk}_j)$ ▷ Real re-keys
7: $\forall i \in [n] \setminus \mathcal{P}_t, \forall j \neq i$: $\mathtt{rk}_{i,j} \leftarrow \mathsf{PRE.RK}((\mathtt{pk}_i, \mathtt{sk}_i), \mathtt{pk}_j)$ ▷ Real re-keys
8: $b' \leftarrow \mathsf{A}^{(\mathtt{corrupt},\cdot),(\mathtt{rekey},\cdot,\cdot),(\mathtt{reencrypt},\cdot,\cdot,\cdot),(\mathtt{challenge},\cdot,\cdot,\cdot)}(\mathtt{pp}, \mathtt{pk}_1, \ldots, \mathtt{pk}_n)$
9: **if** A made call $(\mathtt{challenge}, i^*, \cdot, \cdot)$ for some i^* **then** ▷ Check abort conditions
10: **if** $\exists\, i \in \mathcal{C} : i^*$ is connected to i in $([n], \mathcal{E})$ **then return** 0 **end if**
11: **end if**
12: **if** $\hat{G}$ is the subgraph of $([n], \mathcal{E})$ reachable from i^* **then return** b' **end if**
13: **return** 0

Algorithm 3: Template for generating fully selective PRE-CPA hybrids given a pebbling configuration. All the oracles are defined like in Game 3.

makes at most Q_{RE} re-encryption queries, then a PRE scheme that is (s_1, ϵ_1)-indistinguishable and $(s_2, \epsilon_2, \delta)$-weakly key-private is (s, ϵ)-secure against fully selectivised PRE-CPA restricted to challenge graphs in $\mathcal{G}(n, \delta, d)$ with

$$s := \min(s_1, s_2) - s_{\mathsf{CPA}} \quad \textit{and} \quad \epsilon := \epsilon_1 + 2\tau \cdot \epsilon_2,$$

where $s_{\mathsf{CPA}} \approx O(s_{\mathsf{P}} + n^2 \cdot s_{\mathsf{RK}} + Q_{\mathsf{RE}} \cdot s_{\mathsf{RE}})$ denotes the complexity of simulating the CPA game.

PRE-CPA-security follows from random guessing (Theorem 3) but with a security loss of 2^{n^2}, where n^2 is an upper bound on the number of bits required to encode the challenge subgraph:

Corollary 1 (PRE-CPA-security by random guessing). *A PRE scheme that is (s_1, ϵ_1)-indistinguishable and $(s_2, \epsilon_2, \delta)$-weakly key-private is (s, ϵ)-secure against PRE-CPA restricted to challenge graphs in $\mathcal{G}(n, \delta, d)$, where*

$$s := \min(s_1, s_2) - s_{\mathsf{CPA}} - s_{\mathcal{G}} \quad \textit{and} \quad \epsilon := (\epsilon_1 + 2\tau \cdot \epsilon_2) \cdot 2^{n^2}.$$

4.1.2 Partially Selective Hybrids.

In hybrid H_t^b described in Algorithm 3, we observe that not all information on the committed recoding graph $\hat{G}$ is actually required for the simulation. In fact, only the pebbling configuration $\mathcal{P}_t$ is required to simulate the hybrid: re-keys are only required once a corresponding re-key or a re-encrypt query is issued; for a pebbled node, such queries lead to an edge added in $\mathcal{E}$; thus the re-key is simulated (while the "not-queried" re-keys are never used during the experiment).

In addition to the pebbling configuration $\mathcal{P}_\tau$, the reduction from ciphertext indistinguishability (cf. Algorithm 4) also needs to know the challenge vertex

Reduction $\mathsf{R}_\tau^{(\mathsf{IND.challenge},\cdot,\cdot)}(\mathtt{pp}^*, \mathtt{pk}^*)$ ▷ $\mathtt{pk}^*$ denotes the challenge public key

1: Obtain the challenge graph $\hat{G} \in \mathcal{G}(n, \delta, d)$ from A
2: Compute $\mathcal{P}_\tau \leftarrow \mathsf{P}(\hat{G}, \tau)$ ▷ The τ-th pebbling configuration
3: Set $\mathcal{C}, \mathcal{E} = \emptyset$ ▷ Stores corrupt keys and issued re-keys and re-encryptions
4: $(\mathtt{pk}_1, \mathtt{sk}_1), \ldots, (\mathtt{pk}_{\hat{\imath}-1}, \mathtt{sk}_{\hat{\imath}-1}), (\mathtt{pk}_{\hat{\imath}+1}, \mathtt{sk}_{\hat{\imath}+1}), \ldots, (\mathtt{pk}_n, \mathtt{sk}_n) \leftarrow \mathsf{PRE.K}(\mathtt{pp}^*)$
5: Let $\hat{\imath}$ be the source of $\hat{G}$, set $\mathtt{pk}_{\hat{\imath}} := \mathtt{pk}^*$ ▷ Embed challenge public key
6: $\forall i \in \mathcal{P}_\tau, \forall j \in \mathsf{children}(i, \hat{G})$: $\mathtt{rk}_{i,j} \leftarrow \mathsf{RK}^*(\mathtt{pp}, \mathtt{pk}_j)$ ▷ Fake re-keys
7: $\forall i \in [n] \setminus \mathcal{P}_\tau, \forall j \in \mathsf{children}(i, \hat{G})$: $\mathtt{rk}_{i,j} \leftarrow \mathsf{PRE.RK}((\mathtt{pk}_i, \mathtt{sk}_i), \mathtt{pk}_j)$ ▷ Real re-keys
8: $\forall i \in \mathcal{P}_\tau, \forall j \in [n] \setminus \{\mathsf{children}(i, \hat{G}) \cup i\}$: $\mathtt{rk}_{i,j} \leftarrow \mathsf{PRE.RK}((\mathtt{pk}_i, \mathtt{sk}_i), \mathtt{pk}_j)$ ▷ Real re-keys
9: $\forall i \in [n] \setminus \mathcal{P}_\tau, \forall j \neq i$: $\mathtt{rk}_{i,j} \leftarrow \mathsf{PRE.RK}((\mathtt{pk}_i, \mathtt{sk}_i), \mathtt{pk}_j)$ ▷ Real re-keys
10: $b' \leftarrow \mathsf{A}^{(\mathtt{corrupt},\cdot),(\mathtt{rekey},\cdot,\cdot),(\mathtt{reencrypt},\cdot,\cdot,\cdot),(\mathtt{challenge},\cdot,\cdot,\cdot)}(\mathtt{pp}^*, \mathtt{pk}_1, \ldots, \mathtt{pk}_n)$
11: **if** A made call $(\mathtt{challenge}, i^*, \cdot, \cdot)$ for some i^* **then** ▷ Check abort conditions
12: **if** $\exists\, i \in \mathcal{C} : i^*$ is connected to i in $([n], \mathcal{E})$ **then return** 0 **end if**
13: **end if**
14: **if** $\hat{G}$ is the subgraph of $([n], \mathcal{E})$ reachable from i^* **then return** b' **end if**
15: **return** 0

Oracles `rekey` and `reencrypt` are defined like in Game 3.

Oracle $(\mathtt{corrupt}, i)$

1: **if** $i = \hat{\imath}$ **then** HALT: R_τ returns 0 **end if** ▷ Commitment $\hat{G}$ doesn't match or...
2: Add i to $\mathcal{C}$ and **return** $\mathtt{sk}_i$ ▷ ... i^* corrupted

Oracle $(\mathtt{challenge}, i^*, (m_0^*, m_1^*), \ell^*)$ ▷ Single access

1: $(c_{i^*}, \ell^*) \leftarrow \mathsf{IND.challenge}((m_0^*, m_1^*), \ell^*)$ ▷ Embed challenge ciphertext
2: **return** (c_{i^*}, ℓ^*)

Algorithm 4: The reduction showing that the hybrids H_τ^0 and H_τ^1 are indistinguishable by indistinguishability of ciphertexts.

Reduction $\mathsf{R}_t^b(\mathtt{pp}^*, \mathtt{pk}_0^*, \ldots, \mathtt{pk}_\delta^*, \mathtt{rk}_{0,1}^*, \ldots, \mathtt{rk}_{0,\delta}^*)$

1: Obtain the challenge graph $\hat{G} \in \mathcal{G}(n, \delta, d)$ from A
2: Compute $\mathcal{P}_t \leftarrow \mathsf{P}(\hat{G}, t)$, $\mathcal{P}_{t+1} \leftarrow \mathsf{P}(\hat{G}, t+1)$ ▷ The t-th and $(t+1)$-th configurations
3: Set $\mathcal{C}, \mathcal{E} = \emptyset$ ▷ Stores corrupt keys and issued re-keys and re-encryptions
4: $i_0 := \mathcal{P}_t \Delta \mathcal{P}_{t+1}$, $i_1, \ldots, i_\delta := \mathsf{children}(i_0, \hat{G})$ ▷ i_0 denotes pebbled/unpebbled vertex
5: $\forall k \in [0, \delta]$: $\mathtt{pk}_{i_k} := \mathtt{pk}_k^*$ ▷ Embed the challenge public keys
6: $\forall k \in [n] \setminus \{i_0, \ldots, i_\delta\}$: $(\mathtt{pk}_k, \mathtt{sk}_k) \leftarrow \mathsf{PRE.K}(\mathtt{pp}^*)$ ▷ Real keys
7: $\forall k \in [\delta] : \mathtt{rk}_{i_0, i_k} := \mathtt{rk}_{0,k}^*$ ▷ Embed challenge re-keys
8: $\forall i \in \mathcal{P}_t \setminus \{i_0\}, \forall j \in \mathsf{children}(i, \hat{G})$: $\mathtt{rk}_{i,j} \leftarrow \mathsf{RK}^*(\mathtt{pp}^*, \mathtt{pk}_j)$ ▷ Fake re-keys
9: $\forall i \in \mathcal{P}_t, \forall j \in [n] \setminus \{\mathsf{children}(i, \hat{G}) \cup i\}$: $\mathtt{rk}_{i,j} \leftarrow \mathsf{PRE.RK}((\mathtt{pk}_i, \mathtt{sk}_i), \mathtt{pk}_j)$ ▷ Real re-keys
10: $\forall i \in [n] \setminus (\mathcal{P}_t \cup \{i_0\}), \forall j \neq i$: $\mathtt{rk}_{i,j} \leftarrow \mathsf{PRE.RK}((\mathtt{pk}_i, \mathtt{sk}_i), \mathtt{pk}_j)$ ▷ Real re-keys
11: $b' \leftarrow \mathsf{A}^{(\mathtt{corrupt}, \cdot), (\mathtt{rekey}, \cdot, \cdot), (\mathtt{reencrypt}, \cdot, \cdot, \cdot), (\mathtt{challenge}, \cdot, \cdot, \cdot)}(\mathtt{pp}^*, \mathtt{pk}_1, \ldots, \mathtt{pk}_n)$
12: **if** A made call $(\mathtt{challenge}, i^*, \cdot, \cdot)$ for some i^* **then** ▷ Check abort conditions
13: **if** $\exists\, i \in \mathcal{C} : i^*$ is connected to i in $([n], \mathcal{E})$ **then return** 0 **end if**
14: **end if**
15: **if** $\hat{G}$ is the subgraph of $([n], \mathcal{E})$ reachable from i^* **then return** b' **end if**
16: **return** 0

Oracles `rekey`, `reencrypt` and `challenge` are defined like in Game 3.

Oracle $(\mathtt{corrupt}, i)$

1: **if** $i \in \{i_0, \ldots, i_\delta\}$ **then** HALT: R_τ returns 0 **end if** ▷ Commitment $\hat{G}$ doesn't match
2: Add i to $\mathcal{C}$ and **return** $\mathtt{sk}_i$ ▷ ... or i reachable from i^*

Algorithm 5: The reduction showing that the hybrids H_t^b and H_{t+1}^b, for $t \in [0, \tau - 1]$ and $b \in \{0, 1\}$, are indistinguishable by weak key-privacy.

$\hat{\imath}$ in order to embed the challenge public key. The reduction from weak key-privacy (cf. Algorithm 5) requires, in addition to $\mathcal{P}_t$, the vertex that is pebbled or unpebbled in $\mathcal{P}_{t+1}$ (i.e., the vertex i_0) and its children, so it can embed its challenge public keys and re-keys.

To sum up, two consecutive hybrids H^b_t and H^b_{t+1} can be shown to be indistinguishable using a lot less information than what the adversary commits to. We thus have the following:

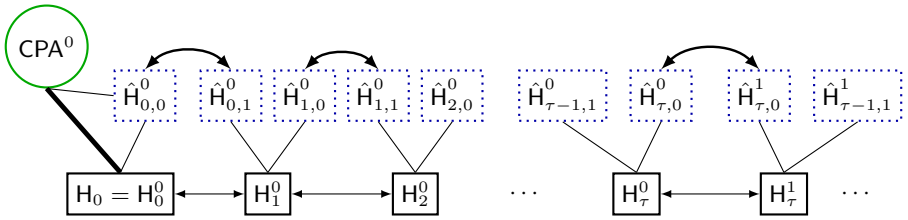

Fig. 3: Diagram showing the partially selectivised hybrids for PRE-CPA.

Lemma 6 (Partially selectivised hybrids). *Let $\mathcal{P}_0, \ldots, \mathcal{P}_\tau$ and $\mathsf{H}^0_0, \ldots, \mathsf{H}^0_\tau$, $\mathsf{H}^1_\tau, \ldots, \mathsf{H}^1_0$ be defined as in Lemma 5, and let σ denote the space complexity of the pebbling sequence. Then, for $t \in [0, \tau - 1]$ and $b, \beta \in \{0, 1\}$,*

$$\mathsf{H}^b_{t+\beta} \equiv \mathsf{SEL}_{\mathcal{U}\to\mathcal{G}}[\hat{\mathsf{H}}^b_{t,\beta}, g, h_t] \quad \text{and} \quad \mathsf{H}^b_\tau \equiv \mathsf{SEL}_{\mathcal{U}\to\mathcal{G}}[\hat{\mathsf{H}}^b_{\tau,0}, g, h_\tau],$$

where $\hat{\mathsf{H}}^b_{t,\beta}$ is defined in Algorithm 6 (see also Figure 3), g extracts the challenge graph from the transcript (as in Lemma 5). For $t \in [0, \tau - 1]$, h_t is the function that extracts the pebbling configuration $\mathcal{P}_t$, the pebbled/unpebbled vertex in $\mathcal{P}_{t+1}$ and its children; h_τ extracts the pebbling configuration $\mathcal{P}_\tau$ and the challenge node i^*. Thus, $\mathcal{U}$ corresponds to the set $\mathcal{V}^{\sigma+\delta+1}$.

The tighter bound for PRE-CPA-security now results by applying Theorem 4:

Theorem 5 (main, PRE-CPA security). *Let σ and τ denote, respectively, the pebbling space and time complexity for the class $\mathcal{G}(n, \delta, d)$. Then a PRE scheme that is (s_1, ϵ_1)-indistinguishable and $(s_2, \epsilon_2, \delta)$-weakly key-private is (s, ϵ)-PRE-CPA-secure restricted to challenge graphs in $\mathcal{G}(n, \delta, d)$, where*

$$s := \min(s_1, s_2) - s_{\mathsf{CPA}} - s_{\mathcal{G}} \quad \text{and} \quad \epsilon := (\epsilon_1 + 2\tau \cdot \epsilon_2) \cdot n^{\sigma+\delta+1}.$$

4.2 Adaptive Security Against Honest-Reencryption Attack

Cohen [Coh17] showed that if a PRE scheme is re-encryption-simulatable then selective security against HRA reduces to showing selective security against CPA.

Hybrid $\mathsf{H}^b_{t+\beta}$

Obtain the challenge graph $\hat{G} \in \mathcal{G}(n, \delta, d)$ from A and let $\hat{\imath}$ be its source
Compute $\mathcal{P}_t \leftarrow \mathsf{P}(\hat{G}, t)$, $\mathcal{P}_{t+1} \leftarrow \mathsf{P}(\hat{G}, t+1)$ ▷ The t-th and $(t+1)$-th configurations
$i_0 := \mathcal{P}_t \Delta \mathcal{P}_{t+1}$, $i_1, \ldots, i_\delta := \mathsf{children}(i_0, \hat{G})$ ▷ i_0 denotes pebbled/unpebbled vertex
if $t < \tau$ **then** $\hat{b} \leftarrow \hat{\mathsf{H}}^b_{t,\beta}(\mathcal{P}_t, \{i_0, \ldots, i_\delta\})$ ▷ Key-privacy hybrid
else $\hat{b} \leftarrow \hat{\mathsf{H}}^b_{\tau,0}(\mathcal{P}_\tau, \{\hat{\imath}, \bot, \ldots, \bot\})$ **end if** ▷ $t = \tau$, $\beta = 0$: ind hybrid
if $\hat{G}$ is subgraph of $([n], \mathcal{E})$ reachable from i^* **then return** b' **end if**
return 0

$\hat{\mathsf{H}}^b_{t,\beta}(\mathcal{P}_t, \{i_0, \ldots, i_\delta\})$

Set $\mathcal{C}, \mathcal{E} = \emptyset$ ▷ Stores corrupt keys and issued re-keys and re-encryptions
if $t < \tau$ **then**
  if $i_0 \in \mathcal{P}_t$ **then** $\mathcal{P}_{t+1} := \mathcal{P}_t \setminus \{i_0\}$ **else** $\mathcal{P}_{t+1} := \mathcal{P}_t \cup \{i_0\}$ **end if**
end if
$\mathsf{pp} \leftarrow \mathsf{PRE.S}(1^\kappa, 1^\lambda)$, $(\mathsf{pk}_1, \mathsf{sk}_1), \ldots, (\mathsf{pk}_n, \mathsf{sk}_n) \leftarrow \mathsf{PRE.K}(\mathsf{pp})$
$\forall i, j \in [n], i \neq j : \mathtt{rk}_{i,j} = \bot$ ▷ Delay re-key generation till the query
$b' \leftarrow \mathsf{A}^{(\mathtt{corrupt},\cdot),(\mathtt{rekey},\cdot,\cdot),(\mathtt{reencrypt},\cdot,\cdot,\cdot),(\mathtt{challenge},\cdot,\cdot,\cdot)}(\mathsf{pp}, \mathsf{pk}_1, \ldots, \mathsf{pk}_n)$
if A made call $(\mathtt{challenge}, i^*, \cdot, \cdot)$ for some i^* **then** ▷ Check abort conditions
  if $\exists\, i \in \mathcal{C} : i^*$ is connected to i in $([n], \mathcal{E})$ **then return** 0 **end if**
end if
return b'

Oracles `corrupt` and `challenge` are defined like in Game 3.

Oracle $(\mathtt{rekey}, i, j)$

if $\mathtt{rk}_{i,j} = \bot$ **then** ▷ Re-key not generated
  if $i \in \mathcal{P}_{t+\beta}$ **then** $\mathtt{rk}_{i,j} \leftarrow \mathsf{RK}^*(\mathsf{pp}, \mathsf{pk}_j)$ ▷ Fake re-key
  else $\mathtt{rk}_{i,j} \leftarrow \mathsf{RK}((\mathsf{pk}_i, \mathsf{sk}_i), \mathsf{pk}_j)$ **end if** ▷ Real re-key
end if
Add (i, j) to $\mathcal{E}$ ▷ Add to recoding graph
return $\mathtt{rk}_{i,j}$

Oracle $(\mathtt{reencrypt}, i, j, (c_i, \ell))$

if $\mathtt{rk}_{i,j} = \bot$ **then** ▷ Re-key not generated
  if $i \in \mathcal{P}_{t+\beta}$ **then** $\mathtt{rk}_{i,j} \leftarrow \mathsf{RK}^*(\mathsf{pp}, \mathsf{pk}_j)$ ▷ Fake re-key
  else $\mathtt{rk}_{i,j} \leftarrow \mathsf{RK}((\mathsf{pk}_i, \mathsf{sk}_i), \mathsf{pk}_j)$ **end if** ▷ Real re-key
end if
Add (i, j) to $\mathcal{E}$ ▷ Add to recoding graph
return $(c_j, \ell + 1) \leftarrow \mathsf{PRE.RE}(\mathtt{rk}_{i,j}, \mathsf{pk}_i, \mathsf{pk}_j, (c_i, \ell))$

Algorithm 6: Partially selectivised hybrids. For $t \in [0, \tau - 1]$ and $b, \beta \in \{0, 1\}$: $\mathsf{H}^b_{t+\beta} = \mathsf{SEL}_{\mathcal{U} \to \mathcal{G}}[\hat{\mathsf{H}}^b_{t,\beta}, g, h_t]$ and $\mathsf{H}^b_\tau = \mathsf{SEL}_{\mathcal{U} \to \mathcal{G}}[\hat{\mathsf{H}}^b_{\tau,0}, g, h_\tau]$. Moreover, $\mathcal{U}$ is the set $\mathcal{V}^{\sigma+\delta+1}$. Note that the sampling of the re-keys has been deferred to the actual calls.

We now consider such a reduction in the adaptive setting. This is not immediate because the reduction in [Coh17] simulates all re-encryption queries from the

Hybrid shHRA^b

Set $\mathcal{C}, \mathcal{L}, \mathcal{L}^*, \mathcal{E} = \emptyset$ and $C = 0$ ▷ As in Game 2, $\mathcal{E}$... edges of recoding graph
$\mathtt{pp} \leftarrow \mathsf{PRE.S}(1^\kappa, 1^\lambda)$, $(\mathtt{pk}_1, \mathtt{sk}_1), \ldots, (\mathtt{pk}_n, \mathtt{sk}_n) \leftarrow \mathsf{PRE.K}(\mathtt{pp})$
$\forall i, j \in [n], i \neq j : \mathtt{rk}_{i,j} \leftarrow \mathsf{PRE.RK}((\mathtt{pk}_i, \mathtt{sk}_i), \mathtt{pk}_j)$
$b' \leftarrow \mathsf{A}^{(\mathtt{corrupt},\cdot),(\mathtt{rekey},\cdot,\cdot),(\mathtt{encrypt},\cdot,\cdot),(\mathtt{reencrypt},\cdot,\cdot,\cdot),(\mathtt{challenge},\cdot,\cdot,\cdot)}(\mathtt{pp}, \mathtt{pk}_1, \ldots, \mathtt{pk}_n)$
if A made call $(\mathtt{challenge}, i^*, \cdot, \cdot)$ for some i^* **then** ▷ Check abort conditions
  if $\exists\, i \in \mathcal{C} : i^*$ is connected to i **then return** 0 **end if**
end if
return b'

Oracles `corrupt`, `rekey`, `encrypt` and `challenge` are defined like Game 4.

Oracle $(\mathtt{reencrypt}, i, j, k)$

Retrieve $(k, i, m, (c_i, \ell))$ from $\mathcal{L}$ and increment C
if $k \notin \mathcal{L}^*$ **then** ▷ Not a re-encryption of challenge ciphertext
  $(c_j, \ell + 1) \leftarrow \mathsf{PRE.E}(\mathtt{pk}_j, (m, \ell + 1))$ ▷ Simulate re-encryption
else
  $(c_j, \ell + 1) \leftarrow \mathsf{PRE.RE}(\mathtt{rk}_{i,j}, \mathtt{pk}_i, \mathtt{pk}_j, (c_i, \ell))$ ▷ Real re-encryption
  Add C to $\mathcal{L}^*$ and add (i, j) to $\mathcal{E}$ ▷ c_j derived from challenge
end if
Add $(C, j, m, (c_j, \ell + 1))$ to $\mathcal{L}$
return $(c_j, \ell + 1)$

Game 8: Intermediate game shHRA in the proof of HRA.

honest keys to the corrupt keys. This works in the selective setting, where the set of corrupt users is known in advance, but not in the adaptive setting.

The recoding graphs $([n], \mathcal{E})$ for CPA and HRA are defined differently: for HRA, only re-encryptions of the challenge ciphertexts lead to edges in $\mathcal{E}$, whereas for CPA all re-encryption queries do. To prove that CPA implies HRA we need to define a reduction playing the CPA game and simulating the HRA game for an adversary. It cannot forward the adversary's re-encryption queries to its own challenger, as this might create edges in the CPA game, but not in the HRA game. An adversary that then corrupts the target key of such a re-encryption query might still win the HRA game, while the reduction loses the CPA game.

Assuming source-hiding, the reduction could answer all re-encryption queries by fresh encryptions. But then every re-encryption of the challenge ciphertext would also be freshly encrypted, meaning that the reduction needs to make multiple challenge queries. This would require a multi-challenge notion of CPA and thus worsen the security guarantees; we proceed thus differently. Instead of replacing all re-encryptions by fresh encryptions, the reduction only replaces those that do *not* concern the challenge ciphertext, while still forwarding re-encryption queries of the ciphertext to its own re-encryption oracle. The vertices created in the HRA game correspond then precisely to those created in the CPA game and the adversary's success probability translates directly to that of the reduction.

4.2.1 HRA from CPA and Source-Hiding. We start by defining, in Game 8, the intermediate game shHRA^b just discussed. It proceeds like HRA^b, except that all re-encryption queries which do *not* re-encrypt the challenge ciphertext are simulated. The games HRA^b and shHRA^b are shown to be indistinguishable assuming source-hiding by a standard hybrid argument (without pebbling) which includes a moderate amount of guessing: when replacing a re-encryption by a fresh encryption, it guesses the two concerned users and which ciphertext will be re-encrypted. This loses a factor of $n(n-1)(Q_{\mathsf{RE}}+Q_{\mathsf{E}})$ in the distinguishing advantage. As there are Q_{RE} hybrids, we get the following:

Lemma 7. *If a PRE scheme is (s_3, ϵ_3)-source-hiding then HRA^b and shHRA^b are (s, ϵ)-indistinguishable, where*

$$s := s_3 - s_{\mathsf{HRA}} \quad \textit{and} \quad \epsilon := n(n-1)(Q_{\mathsf{E}}+Q_{\mathsf{RE}})Q_{\mathsf{RE}} \cdot \epsilon_3$$

where Q_{E} and Q_{RE} are upper bounds on the number of the adversary's encryption and re-encryption queries.

Proof. We define a sequence of intermediate hybrids $\mathsf{shHRA}^b_0, \ldots, \mathsf{shHRA}^b_{Q_{\mathsf{RE}}}$ between HRA^b and shHRA^b where certain re-encrypt queries are simulated by computing fresh encryptions. In particular, the simulation in shHRA^b_q is similar to that in HRA^b, except that the first q queries $(\texttt{reencrypt}, \cdot, \cdot, k)$ with $k \notin \mathcal{L}^*$ are replied by fresh encryptions. Therefore, we have

$$\mathsf{shHRA}^b_0 \equiv \mathsf{HRA}^b \quad \text{and} \quad \mathsf{shHRA}^b_{Q_{\mathsf{RE}}} \equiv \mathsf{shHRA}^b.$$

We show that two neighbouring hybrids shHRA^b_{q-1} and shHRA^b_q are indistinguishable assuming that the PRE is source-hiding. The reduction receives a (source-hiding) challenge $(\mathtt{pp}, (\mathtt{pk}, \mathtt{sk}), (\mathtt{pk}', \mathtt{sk}'), \mathtt{rk})$ and has (one-time) access to an oracle $(\mathsf{SH}.\texttt{challenge}, \cdot, \cdot)$. The reduction proceeds as follows:

1. It first makes a guess $(i^*, j^*, k^*) \in [n] \times ([n] \setminus \{i^*\}) \times [Q_{\mathsf{E}} + q - 1]$ that the q-th re-encryption query will be of the form $(\texttt{reencrypt}, i^*, j^*, k^*)$
2. It simulates game shHRA^b_k setting $(\mathtt{pk}_{i^*}, \mathtt{sk}_{i^*}) := (\mathtt{pk}, \mathtt{sk})$, $(\mathtt{pk}_{j^*}, \mathtt{sk}_{j^*}) := (\mathtt{pk}', \mathtt{sk}')$ and $\mathtt{rk}_{i^*,j^*} := \mathtt{rk}$.
3. When the adversary makes the k^*-th encrypt or re-encrypt query, the reduction does the following:
   (a) $(\texttt{encrypt}, i, (m, \ell))$: if $i \neq i^*$ abort; else query $(\mathsf{SH}.\texttt{challenge}, m, \ell)$ to receive (c, c'); reply (c, ℓ) after adding the new entry to $\mathcal{L}$.
   (b) $(\texttt{reencrypt}, i, j, k)$: if $k \in \mathcal{L}^*$ then proceed as in Game 4. Otherwise: if $j \neq i^*$ abort; else retrieve $(k, i, m, (c_i, \ell-1))$ from $\mathcal{L}$, query $(\mathsf{SH}.\texttt{challenge}, m, \ell)$ to receive (c, c'); reply (c, ℓ) after adding the new entry to $\mathcal{L}$.
4. When the adversary makes the q-th re-encrypt query $(\texttt{reencrypt}, i, j, k)$, the reduction aborts if $(i, j, k) \neq (i^*, j^*, k^*)$. Otherwise it replies $(c', \ell + 1)$, with c' received on its $\mathsf{SH}.\texttt{challenge}$ query.
5. If the reduction aborted the simulation, it returns a random bit, otherwise it returns the adversary's output bit b'.

Assuming the reduction's guess is right, if the c' returned by the source-hiding challenger is a re-encryption then the reduction simulated shHRA^b_{q-1} while if c' was a fresh encryption, it simulated shHRA^b_q. □

We next show that shHRA^0 and shHRA^1 are indistinguishable assuming CPA^0 and CPA^1 are. To do so, we construct a reduction R in Algorithm 7, which simulates game shHRA to an adversary A (denoted $\mathsf{R}[\mathsf{A}]$); the reduction runs in CPA. It is easily seen that R perfectly simulates the oracles of shHRA to A and that at the end of the game the sets $\mathcal{C}$ and the graphs $([n], \mathcal{E})$ which were implicitly defined by the adversary's oracle calls in the simulated HRA game and the reduction's calls in the CPA game are the same. The CPA game thus returns the reduction's output b' (which is A's output) whenever the shHRA game would. We have thus:

$$\langle \mathsf{shHRA}^b, \mathsf{R}[\mathsf{A}] \rangle \equiv \langle \mathsf{CPA}^b, \mathsf{A} \rangle, \tag{2}$$

that is, the two games are equally distributed. (This can also be seen by replacing A in the definition of CPA (Game 3) by the code of $\mathsf{R}[\mathsf{A}']$ (Algorithm 7), which yields game HRA (Game 4) played by A'.)

Reduction $\mathsf{R}^{b,(\mathsf{CPA.corrupt},\cdot),(\mathsf{CPA.rekey},\cdot,\cdot),(\mathsf{CPA.reencrypt},\cdot,\cdot,\cdot),(\mathsf{CPA.challenge},\cdot,\cdot,\cdot)}(\mathtt{pp}, \mathtt{pk}_1, \ldots, \mathtt{pk}_n)$

1: Set $\mathcal{L}, \mathcal{L}^* = \emptyset$ and $C = 0$ ▷ Stores honestly created ciphertexts
2: $b' \leftarrow \mathsf{A}^{(\mathtt{corrupt},\cdot),(\mathtt{rekey},\cdot,\cdot),(\mathtt{encrypt},\cdot,\cdot),(\mathtt{reencrypt},\cdot,\cdot,\cdot),(\mathtt{challenge},\cdot,\cdot,\cdot)}(\mathtt{pp}, \mathtt{pk}_1, \ldots, \mathtt{pk}_n)$
3: **return** b'

Oracle $(\mathtt{corrupt}, i)$
1: **return** $\mathtt{sk}_i \leftarrow (\mathsf{CPA}.\mathtt{corrupt}, i)$

Oracle $(\mathtt{rekey}, i, j)$
1: **return** $\mathtt{rk}_{i,j} \leftarrow (\mathsf{CPA}.\mathtt{rekey}, i, j)$

Oracle $(\mathtt{encrypt}, i, (m, \ell))$
1: $c \leftarrow \mathsf{PRE.E}(\mathtt{pk}_i, (m, \ell))$, increment C and add $(C, i, m, (c, \ell))$ to $\mathcal{L}$
2: **return** c

Oracle $(\mathtt{reencrypt}, i, j, k)$
1: Retrieve $(k, i, m, (c_i, \ell))$ from $\mathcal{L}$ and increment C
2: **if** $k \notin \mathcal{L}^*$ **then** $(c_j, \ell + 1) \leftarrow \mathsf{PRE.E}(\mathtt{pk}_j, (m, \ell + 1))$
3: **else** $(c_j, \ell + 1) \leftarrow (\mathsf{CPA}.\mathtt{reencrypt}, i, j, k)$ and add C to $\mathcal{L}^*$ **end if**
4: Add $(C, j, m, (c_j, \ell + 1))$ to $\mathcal{L}$
5: **return** $(c_j, \ell + 1)$

Oracle $(\mathtt{challenge}, i^*, (m_0^*, m_1^*), \ell^*)$ ▷ Single access
1: $(c_{i^*}, \ell^*) \leftarrow (\mathsf{CPA}.\mathtt{challenge}, i^*, (m_0^*, m_1^*), \ell^*)$
2: Increment C, add $(C, i^*, m_b^*, (c_{i^*}, \ell^*))$ to $\mathcal{L}$ and C to $\mathcal{L}^*$
3: **return** (c_{i^*}, ℓ^*)

Algorithm 7: The reduction relating shHRA^b to CPA^b.

Combining Lemma 7 and (2), we get that HRA^b and CPA^b are $(s_3 - s_{\mathsf{HRA}},\ n(n-1)(Q_{\mathsf{E}} + Q_{\mathsf{RE}})Q_{\mathsf{RE}} \cdot \epsilon_3)$-indistinguishable, and together with Theorem 4 this finally yields:

Table 2: Space and time complexity for different classes of DAGs and approximate security loss implied by Theorem 5.

Family	Bounds		
	Space (σ)	Time (τ)	Security loss ($\approx \epsilon/\epsilon'$)
Arbitrary DAGs $\mathcal{G}(n, \delta, d)$ (Lemma 2)	$(\delta + 1) \cdot d$	$(2\delta)^d$	$n^{O(d \cdot \delta)}$
Complete binary trees $\mathcal{B}(n)$ (Lemma 3)	$\log n$	n^2	$n^{O(\log n)}$
Chains $\mathcal{C}(n)$ (Lemma 4)	$\log n + 1$	$3^{\log n}$	$n^{O(\log n)}$

Theorem 6 (main, PRE-HRA security). *Let σ and τ denote, respectively, the upper bound on time and space complexity for the class $\mathcal{G} = \mathcal{G}(n, \delta, d)$. Then a PRE scheme that is (s_1, ϵ_1)-indistinguishable, $(s_2, \epsilon_2, \delta)$-weakly key-private and (s_3, ϵ_3)-source-hiding is (s, ϵ)-PRE-HRA-secure restricted to challenge graphs in $\mathcal{G}$, where $s := \min(s_1, s_2, s_3) - s_{\mathsf{HRA}} - s_{\mathcal{G}}$ and*

$$\epsilon := 2n(n-1)(Q_{\mathsf{E}} + Q_{\mathsf{RE}})Q_{\mathsf{RE}} \cdot \epsilon_3 + n^{\sigma+\delta+1}\big(\epsilon_1 + 2\tau \cdot \epsilon_2\big).$$

Remark 1. All the schemes that we inspect in §5 turn out to be *statistically* source-hiding, and therefore ϵ_3 is exponentially small. In such cases, assuming that the adversary is allowed to make only polynomially many queries to the encryption and re-encryption oracle, the term $2n(n-1)(Q_{\mathsf{E}} + Q_{\mathsf{RE}})Q_{\mathsf{RE}} \cdot \epsilon_3$ is negligible and therefore $\epsilon = O(n^{\sigma+\delta+1}(\epsilon_1 + 2\tau \cdot \epsilon_2))$ (just like in CPA).

4.3 Corollaries

We calculate concrete bounds to Theorems 5 and 6 for the following families of recoding graphs: arbitrary DAGs in $\mathcal{G}(n, \delta, d)$, complete binary trees $\mathcal{B}(n)$ and chains $\mathcal{C}(n)$. Table 2 lists the space and time complexity for these classes (from Lemmas 2, 3 and 4) and approximate security loss (assuming $\epsilon_1 = \epsilon_2 = \epsilon'$) that results when substituting these bounds for CPA in Theorem 5. The same bounds hold for HRA if one assumes that Q_{RE} and Q_{E} (i.e., number of queries) are polynomial and $\epsilon_3 = 2^{-\kappa}$ (i.e., the PRE scheme is *statistically* source-hiding).

5 Adaptively Secure PRE Schemes

We show that several existing PRE schemes satisfy the requirements in §3.1 and, therefore, can be proven adaptively secure using Theorems 5 and 6.

5.1 Single-Hop Schemes from Bilinear Maps

We start with the unidirectional, *single-hop* schemes based on bilinear maps from [AFGH05] and [ABH09]. The definition of bilinear maps is given below in

Definition 14; the hardness assumptions on which security of the constructions is based are then listed in Definitions 16 through 18.

Definition 14 (Bilinear maps). *Let* BS' *be an algorithm that on input a security parameter* 1^κ *outputs the description of cyclic groups* $\mathbb{G}_1$, $\mathbb{G}_2$ *and* $\mathbb{G}_T$, *all of prime order* $q \in \Theta(2^\kappa)$, *generators* $g_1 \in \mathbb{G}_1$, $g_2 \in \mathbb{G}_2$, *and an asymmetric cryptographic bilinear map* $e' \colon \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ *– i.e.,* e' *is efficiently computable, bilinear (i.e.,* $e'(g_1^a, g_2^b) = e'(g_1, g_2)^{ab}$*) and non-degenerate (i.e., for a generator* g_1 *for* $\mathbb{G}_1$ *and a generator* g_2 *for* $\mathbb{G}_2$*:* $e'(g_1, g_2) \neq 1_{\mathbb{G}_T}$*).*

The algorithm BS *for symmetric bilinear groups is defined as* BS'*, except for having* $\mathbb{G}_1 = \mathbb{G}_2$ *and* $g_1 = g_2$*.*

Definition 15 (eDBDH in asymmetric groups [AFGH05]). *Let* $Grp = (q, g_1, g_2, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e') \leftarrow \mathsf{BS}'(1^\kappa)$ *and* $a, b, c, r \leftarrow \mathbb{Z}_q$*. The extended decisional bilinear Diffie-Hellman problem is* (s, ϵ)*-hard in the asymmetric setting if*

$$(g_2^a, g_1^b, g_1^c, e'(g_1, g_2)^{bc^2}, e'(g_1, g_2)^{abc}, Grp) \approx_{(s,\epsilon)} (g_2^a, g_1^b, g_1^c, e'(g_1, g_2)^{bc^2}, e'(g_1, g_2)^r, Grp).$$

The symmetric variant of the above is obtained by replacing $\mathbb{G}_1$ and $\mathbb{G}_2$ by $\mathbb{G}$, and g_1 and g_2 by g.

Definition 16 (eDBDH assumption [AFGH05]). *Let* $Grp := (q, g, \mathbb{G}, \mathbb{G}_T, e) \leftarrow \mathsf{BS}(1^\kappa)$ *and* $a, b, c, r \leftarrow \mathbb{Z}_q$*. The extended decisional bilinear Diffie-Hellman problem is* (s, ϵ)*-hard if*

$$(g^a, g^b, g^c, e(g, g)^{bc^2}, e(g, g)^{abc}, Grp) \approx_{(s,\epsilon)} (g^a, g^b, g^c, e(g, g)^{bc^2}, e(g, g)^r, Grp).$$

Definition 17 (XDH [Sco02, BBS04]). *Let* $Grp = (q, g_1, g_2, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e') \leftarrow \mathsf{BS}'(1^\kappa)$ *and* $a, b, r \leftarrow \mathbb{Z}_q$*. The external Diffie-Hellman problem is* (s, ϵ)*-hard if the decisional Diffie-Hellman problem is* (s, ϵ)*-hard in the group* $\mathbb{G}_1$*, that is*

$$(g_1^a, g_1^b, g_1^{ab}, Grp) \approx_{(s,\epsilon)} (g_1^a, g_1^b, g_1^r, Grp).$$

Definition 18 (DLin [BBS04]). *Let* $Grp = (q, g, \mathbb{G}, \mathbb{G}_T, e) \leftarrow \mathsf{BS}(1^\kappa)$*,* $a, b, r \leftarrow \mathbb{Z}_q$ *and* h, f *be two random generators of* $\mathbb{G}$*. The decision linear problem is* (s, ϵ)*-hard if*

$$(h, f, g^a, h^b, f^{a+b}, Grp) \approx_{(s,\epsilon)} (h, f, g^a, h^b, f^r, Grp).$$

5.1.1 [AFGH05] Scheme. The original scheme (given as Construction 1) encrypts messages from $\mathbb{G}_T$ and was shown selectively secure against CPA assuming eDBDH, but [AFGH05] did not consider key-privacy. We prove that when instantiated over asymmetric bilinear groups, their scheme is key-private assuming XDH. To additionally make the scheme source-hiding we *re-randomise* the re-encryption algorithm. The modified scheme is given in Construction 2.

1. $\mathsf{S}(1^\kappa)$: $(q, g, \mathbb{G}, \mathbb{G}_T, e) \leftarrow \mathsf{BS}(1^\kappa)$. Compute $Z = e(g, g)$ and return the public parameters $\mathtt{pp} = ((q, g, \mathbb{G}, \mathbb{G}_T, e), Z)$.
2. $\mathsf{K}(\mathtt{pp})$: Pick $a, b \leftarrow \mathbb{Z}_q$ and set $\mathtt{pk} := (Z^a, g^b)$ and $\mathtt{sk} := (a, b)$. Return the keys $(\mathtt{pk}, \mathtt{sk})$.
3. $\mathsf{RK}((\mathtt{pk}_i, \mathtt{sk}_i), \mathtt{pk}_j)$: Parse $\mathtt{sk}_i$ as $(a_i, b_i) \in \mathbb{Z}_q^2$ and $\mathtt{pk}_j$ as $(\mathtt{pk}_{j,1}, \mathtt{pk}_{j,2})$. Return the re-key $\mathtt{rk}_{i,j} := \mathtt{pk}_{j,2}^{a_i} = g^{b_j \cdot a_i}$.
4. $\mathsf{E}(\mathtt{pk}, (m, \ell))$: Parse $\mathtt{pk}$ as $(\mathtt{pk}_1, \mathtt{pk}_2) \in \mathbb{G}_T \times \mathbb{G}$ and pick $k \leftarrow \mathbb{Z}_q$. If $\ell = 1$ return the (level-1) ciphertext $((g^k,\ m \cdot \mathtt{pk}_1^k = m \cdot Z^{ak}),\ 1)$;
otherwise return the (level-2) ciphertext $((e(\mathtt{pk}_2, g)^k,\ m \cdot Z^k) = (Z^{bk},\ m \cdot Z^k),\ 2)$.
5. $\mathsf{RE}((c_i, 1), \mathtt{rk}_{i,j})$: Parse c_i as $(c_{i,1}, c_{i,2}) \in \mathbb{G} \times \mathbb{G}_T$ and return the level-2 ciphertext $((e(\mathtt{rk}_{i,j}, c_{i,1}),\ c_{i,2}) = (Z^{b_j \cdot (a_i k)},\ m \cdot Z^{a_i k}),\ 2)$.
6. $\mathsf{D}((c, \ell), \mathtt{sk})$: Parse the secret key $\mathtt{sk}$ as $(a, b) \in \mathbb{Z}_q^2$. Parse a level-1 ciphertext as $(c_1, c_2) \in \mathbb{G} \times \mathbb{G}_T$ and return $c_2 / e(c_1, g)^a$. Parse a level-2 ciphertext as $(c_1, c_2) \in \mathbb{G}_T^2$ and return $c_2 / c_1^{1/b}$.

Construction 1: Unidirectional, single-hop PRE from [AFGH05]; basis for Construction 2.

1. $\mathsf{S}(1^\kappa)$: $\boxed{(q, g_1, g_2, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e') \leftarrow \mathsf{BS}'(1^\kappa)}$. Compute $Z = e'(g_1, g_2)$ and return the public parameters $\mathtt{pp} = ((q, g_1, g_2, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e'), Z)$.
2. $\mathsf{K}(\mathtt{pp})$: Pick $a, b \leftarrow \mathbb{Z}_q$ and set $\mathtt{pk} := (Z^a, g_1^b)$ and $\mathtt{sk} := (a, b)$. Return the keys $(\mathtt{pk}, \mathtt{sk})$.
3. $\mathsf{RK}((\mathtt{pk}_i, \mathtt{sk}_i), \mathtt{pk}_j)$: Parse $\mathtt{sk}_i$ as $(a_i, b_i) \in \mathbb{Z}_q^2$ and $\mathtt{pk}_j$ as $(\mathtt{pk}_{j,1}, \mathtt{pk}_{j,2}) \in \mathbb{G}_T \times \mathbb{G}_1$. Return the re-key $\mathtt{rk}_{i,j} := \mathtt{pk}_{j,2}^{a_i} = g_1^{b_j \cdot a_i}$.
4. $\mathsf{E}(\mathtt{pk}, (m, \ell))$: Parse $\mathtt{pk}$ as $(\mathtt{pk}_1, \mathtt{pk}_2) \in \mathbb{G}_T \times \mathbb{G}_1$ and pick $k \leftarrow \mathbb{Z}_q$. If $\ell = 1$ return the (level-1) ciphertext $((g_2^k,\ m \cdot \mathtt{pk}_1^k = m \cdot Z^{ak}), 1)$;
otherwise return the (level-2) ciphertext $((e'(\mathtt{pk}_2, g_2)^k = Z^{bk},\ m \cdot Z^k), 2)$.
5. $\mathsf{RE}(\mathtt{rk}_{i,j}, \mathtt{pk}_i, (c_i, 1))$: Parse c_i as $(c_{i,1}, c_{i,2}) \in \mathbb{G}_2 \times \mathbb{G}_T$ and $\mathtt{pk}_i$ as $(\mathtt{pk}_{i,1}, \mathtt{pk}_{i,2}) \in \mathbb{G}_T \times \mathbb{G}_1$. Pick $k' \leftarrow \mathbb{Z}_q$ for re-randomisation and return the level-2 ciphertext $\boxed{((e'(\mathtt{rk}_{i,j}, c_{i,1} \cdot g_2^{k'}), c_{i,2} \cdot \mathtt{pk}_{i,1}^{k'}), 2)} = (Z^{b_j \cdot a_i (k + k')}, m \cdot Z^{a_i (k + k')})$.
6. $\mathsf{D}(\mathtt{sk}, (c, \ell))$: Parse the secret key $\mathtt{sk}$ as (a, b). Parse a level-1 ciphertext as $c = (c_1, c_2) \in \mathbb{G}_2 \times \mathbb{G}_T$ and return $c_2 / e'(g_1, c_1)^a$. Parse a level-2 ciphertext as $c = (c_1, c_2) \in \mathbb{G}_T^2$ and return $c_2 / c_1^{1/b}$.

Construction 2: Rerandomised [AFGH05] in the asymmetric setting. The differences from Construction 1 are highlighted in boxes.

Security. If the eDBDH problem is (s_1, ϵ_1)-hard then the original scheme (Construction 1) is $(s_1, 2\epsilon_1)$-sPRE-CPA-secure– and hence has indistinguishable ciphertexts [AFGH05, Theorem 3.1]. They also claim security when instantiated in the asymmetric setting, presumably assuming that the eDBDH problem is

hard in the asymmetric setting: this reduction works for Construction 2 too. It moreover satisfies the remaining two properties:

Lemma 8. *Construction 2 is statistically source-hiding.*

Proof. A level-2 ciphertext under $\mathtt{pk}_j$ that results from the re-encryption of a level-1 ciphertext $c_i = (c_{i,1}, c_{i,2})$ under $\mathtt{pk}_i = (\mathtt{pk}_{i,1}, \mathtt{pk}_{i,2})$ is of the form $c_j = (c_{j,1}, c_{j,2})$

$$\begin{aligned} c_{j,1} &= e'(\mathtt{rk}_{i,j}, c_{i,1} \cdot g_2^{k'}) = e'(g_1^{b_j \cdot a_i}, g_2^{k} \cdot g_2^{k'}) = e'(g_1^{b_j}, g_2)^{a_i \cdot (k+k')} \\ c_{j,2} &= c_{i,2} \cdot \mathtt{pk}_{i,1}^{k'} = m \cdot Z^{a_i \cdot k} \cdot Z^{a_i \cdot k'} = m \cdot Z^{a_i \cdot (k+k')}. \end{aligned} \tag{3}$$

From (3) it is clear that the distribution of the re-encrypted ciphertext is statistically close to a fresh level-2 ciphertext under $\mathtt{pk}_j$. □

Lemma 9. *If XDH is (s_2, ϵ_2)-hard then Construction 2 is $(s_2 - \delta \cdot s_{\mathsf{Exp}}, \epsilon_2, \delta)$-weakly key-private, where s_{Exp} is the complexity of four exponentiations in $\mathbb{G}_1$.*

Proof. For $(q, g_1, g_2, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e') \leftarrow \mathsf{BS}'(1^\kappa)$ and $Z := e'(g_1, g_2)$, our goal is to show that $\mathsf{KP}^0 \approx_{(s_2, \epsilon_2)} \mathsf{KP}^1$, where

$$\begin{aligned} \mathsf{KP}^0 &:= ((Z^{a_0}, g_1^{b_0}), (Z^{a_1}, g_1^{b_1}), \ldots, (Z^{a_\delta}, g_1^{b_\delta}), g_1^{b_1 \cdot a_0}, \ldots, g_1^{b_\delta \cdot a_0}) \text{ and} \\ \mathsf{KP}^1 &:= ((Z^{a_0}, g_1^{b_0}), (Z^{a_1}, g_1^{b_1}), \ldots, (Z^{a_\delta}, g_1^{b_\delta}), g_1^{b_1 \cdot r_1}, \ldots, g_1^{b_\delta \cdot r_\delta}) \end{aligned}$$

with $a_0, b_0, a_1, b_1, \ldots, a_\delta, b_\delta, r_1, \ldots, r_\delta \leftarrow \mathbb{Z}_q$. Let A be an adversary of size s_2 that distinguishes KP^0 from KP^1 with probability at least ϵ_2. Given an XDH instance (A, B, C, Grp), where $Grp := (q, g_1, g_2, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e')$, the reduction first uses random self-reducibility of XDH to generate δ instances (A, B_i, C_i, Grp): it picks $u_i, v_i \leftarrow \mathbb{Z}_q$ and sets $B_i := B^{u_i} g_1^{v_i}$ and $C_i := C^{u_i} A^{v_i}$. (If $A = g_1^a, B = g_1^b, C = g_1^c$ then $B_i = g_1^{b_i}$ with $b_i = bu_i + v_i$ and $C_i = g_1^{c_i}$ with $c_i = u_i(c - ab) + ab_i$; thus if $c = ab$ then (A, B_i, C_i) is a DH tuple, otherwise B_i and C_i are independently random.) The reduction picks $b_0, a_1, \ldots, a_\delta \leftarrow \mathbb{Z}_q$ and sends

$$\mathsf{KP} := ((e'(A, g_2), g_1^{b_0}), (Z^{a_1}, B_1), \ldots, (Z^{a_\delta}, B_\delta), C_1, \ldots, C_\delta)$$

to A. Depending on whether C was real or random, the adversary sees either KP^0 or KP^1, and any distinguishing advantage it has is translated to that of the reduction. □

Note that the reduction of XDH to δ-weak key-privacy is without any security loss due to the use of random self-reducibility of XDH. The adaptive security of Construction 2 against CPA (resp., HRA) is a corollary to Theorem 5 (resp., Theorem 6), [AFGH05, Theorem 3.1], Lemma 8 and Lemma 9.

Theorem 7 (PRE-CPA and PRE-HRA security of Construction 2). *Let σ and τ denote, respectively, the space and time complexity for the class $\mathcal{G} = \mathcal{G}(n, \delta, d)$. Assume BS' generates asymmetric bilinear groups for which eDBDH*

1. $\mathsf{S}(1^\kappa)$: $(q, g, \mathbb{G}, \mathbb{G}_T, e) \leftarrow \mathsf{BS}(1^\kappa)$. Pick a random generator $h \in \mathbb{G}$ and compute $Z = e(g, h)$. Return the public parameters $\mathtt{pp} = ((q, g, \mathbb{G}, \mathbb{G}_T, e), h, Z)$.
2. $\mathsf{K}(\mathtt{pp})$: Pick $a, b \leftarrow \mathbb{Z}_q$ and set the public key as $\mathtt{pk} := (Z^a, g^b)$ and the secret key as $\mathtt{sk} := (a, b)$. Return the keys $(\mathtt{pk}, \mathtt{sk})$.
3. $\mathsf{RK}((\mathtt{pk}_i, \mathtt{sk}_i), \mathtt{pk}_j)$: Parse $\mathtt{sk}_i$ as $(a_i, b_i) \in \mathbb{Z}_q^2$ and $\mathtt{pk}_j$ as $(\mathtt{pk}_{j,1}, \mathtt{pk}_{j,2}) \in \mathbb{G}_T \times \mathbb{G}$. Pick $r, \boxed{w} \leftarrow \mathbb{Z}_q$ and return the re-key:

$$\mathtt{rk}_{i,j} := ((\mathtt{pk}_{j,2})^{a_i + r}, h^r, \boxed{e(\mathtt{pk}_{j,2}, h)^w, e(g, h)^w}) = ((g^{b_j})^{a_i + r}, h^r, \boxed{Z^{w b_j}, Z^w}).$$

4. $\mathsf{E}(\mathtt{pk}, (m, \ell))$: Parse the public key as $\mathtt{pk} = (\mathtt{pk}_1, \mathtt{pk}_2) \in \mathbb{G}_T \times \mathbb{G}$ and pick $k \leftarrow \mathbb{Z}_q$. If $\ell = 1$ return the (level-1) ciphertext $((g^k, h^k, m \cdot \mathtt{pk}_1^k), 1)$; otherwise, return the (level-2) ciphertext $((e(\mathtt{pk}_2, h)^k, m \cdot Z^k), 2)$.
5. $\mathsf{RE}(\mathtt{rk}_{i,j}, (c_i, 1))$: Parse the re-key $\mathtt{rk}_{i,j}$ as $(\mathtt{rk}_{i,j,1}, \ldots, \mathtt{rk}_{i,j,4}) \in \mathbb{G}^2 \times \mathbb{G}_T^2$, and the ciphertext c_i as $(c_{i,1}, c_{i,2}, c_{i,3}) \in \mathbb{G}^2 \times \mathbb{G}_T$. Verify that the ciphertext is well-formed by checking if $e(c_{i,1}, h) = e(g, c_{i,2})$ – if it is not, halt. Compute $t_1 = e(\mathtt{rk}_{i,j,1}, c_{i,2}) = Z^{b_j (a_i + r) k}$ and $t_2 = c_{i,3} \cdot e(c_{i,1}, \mathtt{rk}_{i,j,2}) = m \cdot Z^{a_i k} \cdot Z^{kr}$; choose $w' \leftarrow \mathbb{Z}_q$ and re-randomise t_1, t_2 by setting $t_1' = t_1 \cdot \boxed{\mathtt{rk}_{i,j,3}^{w'}}$ and $t_2' = t_2 \cdot \boxed{\mathtt{rk}_{i,j,4}^{w'}}$. Return $((t_1', t_2'), 2)$.
6. $\mathsf{D}(\mathtt{sk}, (c, \ell))$: Parse the secret key $\mathtt{sk}$ as $(a, b) \in \mathbb{Z}_q^2$. Parse a level-1 ciphertext as $(c_1, c_2, c_3) \in \mathbb{G}^2 \times \mathbb{G}_T$; halt if $e(c_1, h) \neq e(g, c_2)$, and otherwise return $c_3 / e(c_1, h)^a$. Parse a level-2 ciphertext as as $(c_1, c_2) \in \mathbb{G}_T^2$ and return $c_2 / c_1^{1/b}$.

Construction 3: Unidirectional, single-hop PRE from [ABH09]: the differences to our simplified version (Construction 4) are boxed.

is (s_1, ϵ_1)-hard, XDH is (s_2, ϵ_2)-hard and where four exponentiations in $\mathbb{G}_1$ cost s_{Exp}. Then Construction 2 is (s, ϵ)-PRE-CPA-secure and (s', ϵ')-PRE-HRA-secure restricted to challenge graphs in $\mathcal{G}$ where

$$\begin{aligned} s &:= \min(s_1, s_2 - \delta \cdot s_{\mathsf{Exp}}) - s_{\mathsf{CPA}} - s_{\mathcal{G}}, \\ \epsilon &:= 2(\epsilon_1 + \tau \cdot \epsilon_2) \cdot n^{\sigma + \delta + 1}, \\ s' &:= \min(s_1, s_2 - \delta \cdot s_{\mathsf{Exp}}) - s_{\mathsf{HRA}} - s_{\mathcal{G}} \quad \textit{and} \\ \epsilon' &:= 2n(n-1)(Q_{\mathsf{E}} + Q_{\mathsf{RE}}) Q_{\mathsf{RE}} \cdot 2^{-\kappa} + \epsilon. \end{aligned}$$

5.1.2 [ABH09] Scheme. This scheme, given in Construction 3, can be thought of as a variant of Construction 1 with a *randomised* re-key generation algorithm, and it is this feature that enabled [ABH09] to show (strong) key-privacy assuming DLin. In Construction 4, we simplify their scheme and show that it is *weakly* key-private (still) assuming DLin. The main difference from the original construction is the way the re-randomisation of a re-encrypted ciphertext is carried out: it is now done just like in Construction 2, and this allows for shorter re-keys (just two group elements compared to four).

1. $\mathsf{S}(1^\kappa)$: $(q, g, \mathbb{G}, \mathbb{G}_T, e) \leftarrow \mathsf{BS}(1^\kappa)$. Pick a random generator $h \in \mathbb{G}$ and compute $Z = e(g, h)$. Return the public parameters $\mathtt{pp} = ((q, g, \mathbb{G}, \mathbb{G}_T, e), h, Z)$.
2. $\mathsf{K}(\mathtt{pp})$: Pick $a, b \leftarrow \mathbb{Z}_q$ and set the public key as $\mathtt{pk} := (Z^a, g^b)$ and the secret key as $\mathtt{sk} := (a, b)$. Return the keys $(\mathtt{pk}, \mathtt{sk})$.
3. $\mathsf{RK}((\mathtt{pk}_i, \mathtt{sk}_i), \mathtt{pk}_j)$: Parse $\mathtt{sk}_i$ as $(a_i, b_i) \in \mathbb{Z}_q^2$ and $\mathtt{pk}_j$ as $(\mathtt{pk}_{j,1}, \mathtt{pk}_{j,2}) \in \mathbb{G}_T \times \mathbb{G}$. Pick $r \leftarrow \mathbb{Z}_q$ and return the re-key:
$$\mathtt{rk}_{i,j} := ((\mathtt{pk}_{j,2})^{a_i + r}, h^r) = ((g^{b_j})^{a_i + r}, h^r).$$
4. $\mathsf{E}(\mathtt{pk}, (m, \ell))$: Parse the public key as $\mathtt{pk} = (\mathtt{pk}_1, \mathtt{pk}_2) \in \mathbb{G}_T \times \mathbb{G}$ and pick $k \leftarrow \mathbb{Z}_q$. If $\ell = 1$ return the (level-1) ciphertext $(c, 1) = ((g^k, h^k, m \cdot \mathtt{pk}_1^k), 1)$; otherwise, return the (level-2) ciphertext $(c, 2) = ((e(\mathtt{pk}_2, h)^k = Z^{b \cdot k}, m \cdot Z^k), 2)$.
5. $\mathsf{RE}(\mathtt{rk}_{i,j}, \mathtt{pk}_i, (c_i, 1))$: Parse the re-key $\mathtt{rk}_{i,j}$ as $(\mathtt{rk}_{i,j,1}, \mathtt{rk}_{i,j,2}) \in \mathbb{G}^2$, the public key $\mathtt{pk}_i$ as $(\mathtt{pk}_{i,1}, \mathtt{pk}_{i,2}) \in \mathbb{G}_T \times \mathbb{G}$ and the ciphertext c_i as $(c_{i,1}, c_{i,2}, c_{i,3}) \in \mathbb{G}^2 \times \mathbb{G}_T$. Verify that the ciphertext is well-formed by checking if $e(c_{i,1}, h) = e(g, c_{i,2})$ – if it is not, halt. Pick $k' \leftarrow \mathbb{Z}_q$ and return $((c_{j,1}, c_{j,2}), 2)$, where
$$c_{j,1} := e(\mathtt{rk}_{i,j,1},\ c_{i,2} \cdot h^{k'}) = Z^{b_j (a_i + r)(k + k')} \quad \text{and}$$
$$c_{j,2} := c_{i,3} \cdot \mathtt{pk}_{i,1}^{k'} \cdot e(c_{i,1} \cdot g^{k'},\ \mathtt{rk}_{i,j,2}) = m \cdot Z^{a_i k} \cdot Z^{a_i k'} \cdot Z^{(k + k')r}.$$
6. $\mathsf{D}(\mathtt{sk}, (c, \ell))$: Parse the secret key $\mathtt{sk}$ as $(a, b) \in \mathbb{Z}_q$. Parse a level-1 ciphertext as $(c_1, c_2, c_3) \in \mathbb{G}^2 \times \mathbb{G}_T$; halt if $e(c_1, h) \neq e(g, c_2)$, and otherwise return $c_3 / e(c_1, h)^a$. Parse a level-2 ciphertext as $(c_1, c_2) \in \mathbb{G}_T^2$ and return $c_2 / c_1^{1/b}$.

Construction 4: Simplified version of scheme from [ABH09] (Construction 3).

Security. If the eDBDH problem is (s_1, ϵ_1)-hard then Construction 3 is $(s_1, 2\epsilon_1)$-sPRE-CPA-secure [ABH09, Theorem 3.1]. The sPRE-CPA security of Construction 4 follows by the same reduction: we refer the readers to [ABH09] for the details. It was also shown in [ABH09, Theorem 3.4] that if the DLin problem is (s_2, ϵ_2)-hard then Construction 3 is $(s_2, 4n^2 \cdot \epsilon_2)$ (strongly) key-private. Below we simplify this reduction to show that Construction 4 is $(s_2, \delta \cdot \epsilon_2, \delta)$ weakly key-private (Lemma 10). We also show (Lemma 11) that it is statistically source-hiding.

Lemma 10. *If the DLin problem is (s_2, ϵ_2)-hard then Construction 4 is $(s_2, \delta \cdot \epsilon_2, \delta)$-weakly key-private.*

Proof. For $(q, g, \mathbb{G}, \mathbb{G}_T, e) \leftarrow \mathsf{BS}(1^\kappa)$, a random generator $h \in \mathbb{G}$ and $Z := e(g, h)$, our goal is to show that $\mathsf{KP}^0 \approx_{(s_2, \epsilon_2)} \mathsf{KP}^1$, where

$$K^0 := \left((Z^{a_0}, g^{b_0}), \ldots, (Z^{a_\delta}, g^{b_\delta}), ((g^{b_1})^{a_0 + r_1}, h^{r_1}), \ldots, ((g^{b_\delta})^{a_0 + r_\delta}, h^{r_\delta})\right) \quad \text{and}$$
$$K^1 := \left((Z^{a_0}, g^{b_0}), \ldots, (Z^{a_\delta}, g^{b_\delta}), ((g^{b_1})^{a'_1 + r_1}, h^{r_1}), \ldots, ((g^{b_\delta})^{a'_\delta + r_\delta}, h^{r_\delta})\right)$$

with $a_0, b_0, a_1, a'_1, b_1, r_1, \ldots, a_\delta, a'_\delta, b_\delta, r_\delta \leftarrow \mathbb{Z}_q$. Unfortunately, in contrast to Lemma 9, random self-reducibility of DLin does not apply here, since the DLin

instances share one of their exponents (a_0); we therefore proceed via a hybrid argument. Let A be an adversary of size s_2 that distinguishes KP^0 from KP^1 with probability at least ϵ_2. With probability ϵ_2/δ it must therefore distinguish two of the hybrid games $\mathsf{KP}^0 = K_0, \ldots, K_\delta = \mathsf{KP}^1$ defined as

$$K_i := \big((Z^{a_0}, g^{b_0}), \ldots, (Z^{a_\delta}, g^{b_\delta}), ((g^{b_1})^{a'_1+r_1}, h^{r_1}), \ldots, ((g^{b_i})^{a'_i+r_i}, h^{r_i}), \\ ((g^{b_{i+1}})^{a_0+r_{i+1}}, h^{r_{i+1}}), \ldots ((g^{b_\delta})^{a_0+r_\delta}, h^{r_\delta})\big).$$

Given a DLin instance (h, f, A, B, C, Grp), where $Grp := (q, g, \mathbb{G}, \mathbb{G}_T, e)$, the reduction picks $a_1, \ldots, a_\delta, b_0, r_0, \ldots, b_{i-1}, r_{i-1}, b_{i+1}, r_{i+1}, \ldots, b_\delta, r_\delta \leftarrow \mathbb{Z}_q$ and runs A on

$$\big((e(A,h), g^{b_0}), (Z^{a_1}, g^{b_1}), \ldots, (Z^{a_{i-1}}, g^{b_{i-1}}), (Z^{a_i}, f), (Z^{a_{i+1}}, g^{b_{i+1}}), \ldots, (Z^{a_\delta}, g^{b_\delta}), \\ ((g^{b_1})^{a'_1+r_1}, h^{r_1}), \ldots, ((g^{b_{i-1}})^{a'_{i-1}+r_{i-1}}, h^{r_{i-1}}), (C, B), \\ (A^{b_{i+1}} g^{b_{i+1} r_{i+1}}, h^{r_{i+1}}), \ldots, (A^{b_\delta} g^{b_\delta r_\delta}, h^{r_\delta})\big).$$

Letting a_0, r_i be such that $A = g^{a_0}$ and $B = h^{r_i}$, the challenge C is either $f^{a_0+r_i}$ (real) or $f^{a'_i+r_i}$ (random). Thus the above is either distributed as K_{i-1} (real) or K_i (random). Its distinguishing advantage thus translates to that of the reduction. □

Lemma 11. *Construction 4 is statistically source-hiding.*

Proof. A level-2 ciphertext under $\mathtt{pk}_j$ that results from the re-encryption of a level-1 ciphertext $c_i = (c_{i,1}, c_{i,2}, c_{i,3})$ under $\mathtt{pk}_i = (\mathtt{pk}_{i,1}, \mathtt{pk}_{i,2})$ is of the form

$$\begin{aligned} c_j &= (c_{j,1}, c_{j,2}) \\ &= (e(\mathtt{rk}_{i,j,1}, c_{i,2} \cdot h^{k'}), c_{i,3} \cdot \mathtt{pk}_{i,1}^{k'} \cdot e(c_{i,1} \cdot g^{k'}, \mathtt{rk}_{i,j,2})) \\ &= (e(g^{b_j(a_i+r)}, h^k \cdot h^{k'}), m \cdot Z^{a_i k} \cdot Z^{a_i k'} \cdot e(g^k \cdot g^{k'}, h^r)) \\ &= (Z^{b_j(k+k')(a_i+r)}, m \cdot Z^{(k+k')(a_i+r)}). \end{aligned} \tag{4}$$

From (4) it is clear that the distribution of the re-encrypted ciphertext is statistically close to a fresh level-2 ciphertext under $\mathtt{pk}_j$. □

The adaptive security of Construction 4 against CPA (resp., HRA) is then a corollary to Theorem 5 (resp., Theorem 6) and [ABH09, Theorem 3.1] using the above lemmas.

Theorem 8 (PRE-CPA and PRE-HRA security of Construction 4). *Let σ and τ denote, resp., the space and time complexity for the class $\mathcal{G} = \mathcal{G}(n, \delta, d)$. Assume BS generates bilinear groups for which eDBDH is (s_1, ϵ_1)-hard and DLin is (s_2, ϵ_2)-hard. Then Construction 4 is (s, ϵ)-PRE-CPA-secure and (s', ϵ')-PRE-HRA-secure restricted to challenge graphs in $\mathcal{G}$ where*

$$s := \min(s_1, s_2) - s_{\mathsf{CPA}} - s_{\mathcal{G}} \qquad \epsilon := 2(\epsilon_1 + \tau \cdot \delta \cdot \epsilon_2) \cdot n^{\sigma+\delta+1}$$
$$s' := \min(s_1, s_2) - s_{\mathsf{HRA}} - s_{\mathcal{G}} \qquad \epsilon' := 2n(n-1)(Q_{\mathsf{E}} + Q_{\mathsf{RE}})Q_{\mathsf{RE}} \cdot 2^{-\kappa} + \epsilon$$

1. Algorithms S,K and D for PRE are defined the same as their counterparts in FHE.
2. $\mathsf{PRE.RK}((\mathtt{pk}_i, \mathtt{sk}_i), \mathtt{pk}_j)$: The re-key $\mathtt{rk}_{i,j}$ is an encryption of $\mathtt{sk}_i$ under $\mathtt{pk}_j$:

$$\mathsf{PRE.RK}((\mathtt{pk}_i, \mathtt{sk}_i), \mathtt{pk}_j) := \mathsf{FHE.E}(\mathtt{pk}_j, \mathtt{sk}_i).$$

3. $\mathsf{PRE.E}(\mathtt{pk}, m)$: Given a message m and a public key $\mathtt{pk}$, the encryption algorithm outputs a "sanitized" FHE encryption:

$$(\mathsf{PRE.E}(\mathtt{pk}, m) \leftarrow \boxed{\mathsf{Sanitize}\ (\mathtt{pk},} \mathsf{FHE.E}(\mathtt{pk}, m)).$$

4. $\mathsf{PRE.RE}(\mathtt{rk}_{i,j}, \mathtt{pk}_i, \mathtt{pk}_j, (c_i, \ell)) \to (c_j, \ell{+}1)$: Given a re-key $\mathtt{rk}_{i,j}$ and a level-ℓ ciphertext c_i that was encrypted under $\mathtt{pk}_i$, the re-encryption algorithm homomorphically decrypts the ciphertext c_i and "sanitizes" the result:

$$c_j \leftarrow \boxed{\mathsf{Sanitize}(\mathtt{pk}_j,} \mathsf{FHE.F}(\mathsf{FHE.D}, (\mathtt{rk}_{i,j}, \mathsf{FHE.E}(\mathtt{pk}_j, c_i)))).$$

Construction 5: PRE from sanitizable FHE. $\mathsf{Sanitize}$ denotes the sanitization algorithm. We refer to the construction without the boxes by Construction 5.a and the construction with blurring (including the boxes) by Construction 5.b.

Remark 2. We note that since the proofs for key-privacy in both Constructions 2 and 4 proceed via a hybrid argument, by using a trick of Panjwani [Pan07], one can improve the bound for ϵ to $2(\epsilon_1{+}\tau{\cdot}\epsilon_2){\cdot}n^{\sigma+\log\delta+1}$ and $2(\epsilon_1{+}\tau{\cdot}\delta{\cdot}\epsilon_2){\cdot}n^{\sigma+\log\delta+1}$, respectively. We refer the reader to [JKK+17] for further reading.

5.2 Multi-Hop Scheme from Fully Homomorphic Encryption.

We now describe the generic construction of a unidirectional multi-hop PRE scheme from fully homomorphic encryption (FHE) due to Gentry [Gen09].

5.2.1 [Gen09] Scheme. A fully homomorphic encryption scheme consists of a five-tuple of algorithms $(\mathsf{S}, \mathsf{K}, \mathsf{E}, \mathsf{D}, \mathsf{F})$, where S is the setup algorithm, K the key-generation algorithm, E the encryption algorithm, D the decryption algorithm and F the homomorphic evaluation algorithm, which takes a function f and encrypted inputs for f and returns an encryption of the evaluation of f on the inputs.

Gentry [Gen09] gave a generic construction of PRE from FHE. We show that this scheme, given in Construction 5.a, is adaptively secure against CPA. In Construction 5.b, we "sanitize" the previous construction and prove that the resulting construction is adaptively secure against HRA.

Security. The ciphertext indistinguishability of Construction 5.a directly follows from the semantic security of the underlying FHE scheme. We show that weak key-privacy also follows from semantic security. If, in addition, the FHE

scheme is *sanitizable*, then Construction 5.a is source-hiding and thus satisfies PRE-HRA. In particular, if sanitizability is as defined and discussed in [DS16] (see Definition 19), then Construction 5.b is *statistically* source-hiding.

Lemma 12. *If* FHE *is* (s_1, ϵ_1)*-semantically secure then Construction 5.a is* $(s_1 - \delta s_{\mathsf{E}} - n s_{\mathsf{K}}, \delta \cdot \epsilon_1, \delta)$*-weakly key-private.*

Proof. Our goal is to show that $\mathsf{KP}^0 \approx_{(s_1,\epsilon_1)} \mathsf{KP}^1$ where

$$\mathsf{KP}^0 := (\mathtt{pp}, \mathtt{pk}_0, \mathtt{pk}_1, \ldots, \mathtt{pk}_\delta, \mathsf{FHE.E}(\mathtt{pk}_1, \mathtt{sk}_0), \ldots, \mathsf{FHE.E}(\mathtt{pk}_\delta, \mathtt{sk}_0)) \text{ and}$$
$$\mathsf{KP}^1 := (\mathtt{pp}, \mathtt{pk}_0, \mathtt{pk}_1, \ldots, \mathtt{pk}_\delta, \mathsf{FHE.E}(\mathtt{pk}_1, \mathtt{sk}'_0), \ldots, \mathsf{FHE.E}(\mathtt{pk}_\delta, \mathtt{sk}'_\delta)),$$

where $\mathtt{pp} \leftarrow \mathsf{FHE.S}(1^\kappa)$ and $(\mathtt{pk}_0, \mathtt{sk}_0), \ldots, (\mathtt{pk}_\delta, \mathtt{sk}_\delta), (\mathtt{pk}'_1, \mathtt{sk}'_1), \ldots, (\mathtt{pk}'_\delta, \mathtt{sk}'_\delta) \leftarrow \mathsf{FHE.K}(\mathtt{pp})$. We use a sequence of hybrid distributions $\mathsf{KP}^0 = K_0, \ldots, K_\delta = \mathsf{KP}^1$, where in the i-th hybrid the first i re-keys are random and the rest real. That is,

$$K_i := \big(\mathtt{pp}, \mathtt{pk}_0, \mathtt{pk}_1, \ldots, \mathtt{pk}_\delta, \mathsf{FHE.E}(\mathtt{pk}_1, \mathtt{sk}'_0), \ldots, \mathsf{FHE.E}(\mathtt{pk}_i, \mathtt{sk}'_i), \mathsf{FHE.E}(\mathtt{pk}_{i+1}, \mathtt{sk}_0), \ldots, \mathsf{FHE.E}(\mathtt{pk}_\delta, \mathtt{sk}_0)\big).$$

Let A be an adversary of size s_1 that distinguishes KP^0 from KP^1 with probability at least ϵ_1. Given a challenge for FHE semantic security containing parameters $\mathtt{pp}^*$ and public key $\mathtt{pk}^*$, the reduction sets $\mathtt{pk}_i := \mathtt{pk}^*$, picks $(\mathtt{pk}_0, \mathtt{sk}_0), (\mathtt{pk}'_i, \mathtt{sk}'_i) \leftarrow \mathsf{FHE.K}(\mathtt{pp}^*)$, sends $(\mathtt{sk}_0, \mathtt{sk}'_i)$ to its own challenger and receives c^*. The reduction embeds c^* at position i and sends

$$K_{i-1,i} := \big(\mathtt{pp}, \mathtt{pk}_0, \mathtt{pk}_1, \ldots, \mathtt{pk}_\delta, \mathsf{FHE.E}(\mathtt{pk}_1, \mathtt{sk}'_0), \ldots, \mathsf{FHE.E}(\mathtt{pk}_{i-1}, \mathtt{sk}'_{i-1}), c^*, \mathsf{FHE.E}(\mathtt{pk}_{i+1}, \mathtt{sk}_0), \ldots, \mathsf{FHE.E}(\mathtt{pk}_\delta, \mathtt{sk}_0)\big)$$

to A. Depending on whether c^* encrypts $\mathtt{sk}_0$ or $\mathtt{sk}'_i$, $K_{i-1,i}$ is distributed as K_{i-1} or K_i. □

Definition 19 (Sanitizability of encryptions [DS16]). *An encryption scheme* $(\mathsf{S}, \mathsf{K}, \mathsf{E}, \mathsf{D})$ *is called sanitizable if there exists a polynomial-time algorithm* $\mathsf{Sanitize}$ *which takes as input a public key and a ciphertext, outputs a ciphertext, and satisfies the following two properties (with all but negligible probability).*

- *Message-preserving. For any key pair* $(\mathtt{pk}, \mathtt{sk}) \leftarrow \mathsf{K}(1^\kappa)$ *and any* c *in the ciphertext space*
$$\mathsf{D}(\mathtt{sk}, \mathsf{Sanitize}(\mathtt{pk}, c)) = \mathsf{D}(\mathtt{sk}, c).$$
- *Sanitizing. For all* c, c' *such that* $\mathsf{D}(\mathtt{sk}, c) = \mathsf{D}(\mathtt{sk}, c')$
$$\Delta((\mathsf{Sanitize}(\mathtt{pk}, c), \mathtt{pk}, \mathtt{sk}), (\mathsf{Sanitize}(\mathtt{pk}, c'), \mathtt{pk}, \mathtt{sk})) \geq 2^{-\kappa}.$$

Theorem 9 (PRE-CPA security of Construction 5.a). *Let* σ *and* τ *denote the space and time complexity for* $\mathcal{G} = \mathcal{G}(n, \delta, d)$*. If* FHE *is* (s_1, ϵ_1)*-semantically secure then Construction 5.a is* (s, ϵ)*-PRE-CPA-secure restricted to challenge graphs in* $\mathcal{G}$*, where*

$$s := s_1 - (\delta s_{\mathsf{E}} + n s_{\mathsf{K}} + s_{\mathsf{CPA}} + s_{\mathcal{G}}) \quad \text{and} \quad \epsilon := (2\tau \cdot \delta + 1) \cdot \epsilon_1 \cdot n^{\sigma+\delta+1}.$$

1. $\mathsf{S}(1^\kappa)$: Pick lattice parameters N, M, $q \in \mathbb{N}$ and a B-bounded error distribution χ on $\mathbb{Z}_q^M$. Sample $\boldsymbol{A} \leftarrow \mathbb{Z}_q^{M \times N}$ uniformly at random and return the public parameters $\mathtt{pp} = (\boldsymbol{A}, N, M, q, \chi)$.
2. $\mathsf{K}(\mathtt{pp})$: Sample $\boldsymbol{s} \leftarrow \mathbb{Z}_q^N$ uniformly at random and compute $\boldsymbol{b} = \boldsymbol{A} \cdot \boldsymbol{s} + \boldsymbol{e}$, where $\boldsymbol{e} \leftarrow \chi$. Set $\mathtt{pk} := \boldsymbol{b}$ as the public key and $\mathtt{sk} := \boldsymbol{s}$ as the secret key. Return $(\mathtt{pk}, \mathtt{sk})$.
3. $\mathsf{E}(\mathtt{pk}, m)$: On input $\mathtt{pk} \in \mathbb{Z}_q^M$ and a message bit $m \in \{0,1\}$, sample $\boldsymbol{r} \leftarrow \{0,1\}^M$ and output
$$\boldsymbol{c} := \boldsymbol{r}^T(\boldsymbol{A}, \boldsymbol{b}) + (0^N, m \cdot \lceil q/2 \rfloor) \in \mathbb{Z}_q^{N+1}.$$
4. $\mathsf{D}(\mathtt{sk}, \boldsymbol{c})$: On input a secret key $\mathtt{sk} = \boldsymbol{s} \in \mathbb{Z}_q^N$ and a ciphertext $\boldsymbol{c} = (\boldsymbol{\alpha}, \beta) \in \mathbb{Z}_q^N \times \mathbb{Z}_q$, output 0 if $\beta - \langle \boldsymbol{\alpha}, \boldsymbol{s} \rangle$ is closer to 0 than to $\lceil q/2 \rfloor$, else output 1.

Construction 6: Regev's Encryption scheme RGV [Reg05].

Theorem 10 (PRE-HRA-security of Construction 5.b). *Let ϵ be defined as in Theorem 9 and let σ and τ denote the space and time complexity for $\mathcal{G} = \mathcal{G}(n, \delta, d)$. If FHE is a sanitizable FHE scheme that is (s_1, ϵ_1)-semantically secure, then Construction 5.b is (s', ϵ')-PRE-HRA-secure restricted to challenge graphs in $\mathcal{G}$, where*

$$s' := s_1 - (\delta s_{\mathsf{E}} + n s_{\mathsf{K}} + s_{\mathsf{HRA}} + s_{\mathcal{G}}) \quad \text{and} \quad \epsilon' := n(n-1)(Q_{\mathsf{E}} + Q_{\mathsf{RE}})Q_{\mathsf{RE}} \cdot 2^{-\kappa} + \epsilon.$$

5.3 Lattice-based Multi-Hop Schemes

Here, we decribe the lattice-based unidirectional multi-hop PRE scheme from [CCL+14]. Being based directly on the decision LWE (DLWE) problem it achieves better parameters than the construction from FHE above.

5.3.1 [CCL+14] Scheme. In [CCL+14], Chandran et al. propose two lattice-based unidirectional multi-hop proxy re-encryption schemes. The schemes are built upon Regev's encryption [Reg05] and its dual version [GPV07], respectively. Here, we will describe the former one, which is inspired by the fully homomorphic encryption scheme of [BV11]. Security can be proven assuming the hardness of the decisional learning with errors (DLWE) problem (cf. Definition 20 below).

We recall Regev's encryption scheme in Construction 6. We can now define the PRE scheme from [CCL+14] using RGV in Construction 7.a. To achieve source-hiding, Chandran et al. propose the variant given as Construction 7.b.

In both schemes, the LWE error will grow with each re-encryption and the level bound λ needs to be chosen appropriately so that correctness of decryption is still guaranteed (with overwhelming probability). The second variant achieves the stronger notion of PRE-HRA-security (see below) at the cost of worse parameters; only a small number λ of re-encryptions is supported by this scheme and the underlying security assumption is very strong.

1. $\mathsf{S}(1^\kappa)$: Get parameters $\mathtt{pp}' \leftarrow \mathsf{RGV.S}(1^\kappa)$, level bound λ and "blurring error" bound E_ℓ for each level $\ell \in [\lambda]$. Return the parameters $\mathtt{pp} = (\mathtt{pp}', \lambda, (E_\ell)_{\ell\in[\lambda]})$.
2. $\mathsf{K}(\mathtt{pp})$: Run $\mathsf{RGV.K}(\mathtt{pp}')$ and output the result.
3. $\mathsf{E}(\mathtt{pk}, (m, \ell))$: Compute $\boldsymbol{c} = \mathsf{RGV.E}(\mathtt{pk}, m) + (0^N, f_\ell)$, where $f_\ell \leftarrow [-E_\ell, E_\ell] \cap \mathbb{Z}$, and return the level-$\ell$ ciphertext $(\boldsymbol{c}, \ell)$.
4. $\mathsf{RK}((\mathtt{pk}_i, \mathtt{sk}_i), \mathtt{pk}_j)$: Parse $\mathtt{sk}_i$ as $\boldsymbol{s}_i = (s_{i,1}, \ldots, s_{i,N}) \in \mathbb{Z}_q^N$. For $k \in [N]$ and $l \in [\lceil \log q \rceil]$, compute $K_{k,l} \leftarrow \mathsf{RGV.E}(\mathtt{pk}_j, 0) + (0^N, s_{i,k} \cdot 2^l)$. Return the re-key:
$$\mathtt{rk}_{i,j} := \{K_{k,l}\}_{k\in[N], l\in[\lceil \log q \rceil]}.$$
5. $\mathsf{RE}(\mathtt{rk}_{i,j}, \mathtt{pk}_i, \mathtt{pk}_j, (\boldsymbol{c}_i, \ell)) \to (c_j, \ell + 1)$: If $\ell \geq \lambda$, abort. Otherwise, parse the level-ℓ ciphertext $\boldsymbol{c}_i$ as $(\boldsymbol{\alpha}, \beta) \in \mathbb{Z}_q^N \times \mathbb{Z}_q$ and $\mathtt{rk}_{i,j}$ as $\{K_{k,l}\}_{k\in[N], l\in[\lceil \log q \rceil]}$. Denote by α_k the k-th component of $\boldsymbol{\alpha}$, and denote the bit decomposition of α_k as $\{\alpha_{k,l}\}_{l\in[\lceil \log q \rceil]}$, i.e., $\alpha_k = \sum_{l\in[\lceil \log q \rceil]} \alpha_{k,l} 2^l$, where each $\alpha_{k,l} \in \{0, 1\}$. Compute
$$\boldsymbol{c}_j = (0^N, \beta) + \textstyle\sum_{k,l} \alpha_{k,l} \cdot K_{k,l} + \mathsf{RGV.E}(\mathtt{pk}_j, 0) + (0^N, f_{\ell+1}),$$
where $f_{\ell+1} \leftarrow [-E_{\ell+1}, E_{\ell+1}] \cap \mathbb{Z}$, and return $(\boldsymbol{c}_j, \ell + 1)$.
6. $\mathsf{D}(\mathtt{sk}, (\boldsymbol{c}, \ell))$: Run $\mathsf{RGV.D}(\mathtt{sk}, \boldsymbol{c})$ and output the result.

Construction 7: source-hiding unidirectional multi-hop PRE from [CCL+14]. We refer to the construction without the blurring (ignoring the boxes) by Construction 7.a and the construction with blurring (including the boxes) by Construction 7.b.

Security. The PRE scheme in Construction 7.a can be proven secure assuming the hardness of decisional learning with errors (DLWE). We will first show PRE-CPA-security of Construction 7.a and then consider PRE-HRA-security of Construction 7.b.

Definition 20 (DLWE [Reg05]). *Let N, M, $q \in \mathbb{N}$. For a matrix $\boldsymbol{A} \leftarrow \mathbb{Z}_q^{M\times N}$ and a secret vector $\boldsymbol{s} \leftarrow \mathbb{Z}_q^N$, each sampled uniformly at random, and a vector $\boldsymbol{e} \leftarrow \chi$ for an error distribution χ on $\mathbb{Z}_q^M$, the decisional LWE problem $\mathrm{DLWE}_{N,M,q,\chi}$ is to distinguish $(\boldsymbol{A}, \boldsymbol{A} \cdot \boldsymbol{s} + \boldsymbol{e})$ from $(\boldsymbol{A}, \boldsymbol{b})$ for a uniformly random sample $\boldsymbol{b} \leftarrow \mathbb{Z}_q^M$.*

To prove adaptive security for the two variants of Construction 7, we will need the following lemma [BV11].

Lemma 13 (matrix-vector leftover hash lemma). *Let κ, N, $q \in \mathbb{N}$, and $M \geq N \cdot \log q + 2\kappa$. For $\boldsymbol{A} \leftarrow \mathbb{Z}_q^{M\times N}$, $\boldsymbol{r} \leftarrow \{0, 1\}^M$, and $\boldsymbol{y} \leftarrow \mathbb{Z}_q^N$ each sampled uniformly at random, it holds $\Delta((\boldsymbol{A}, \boldsymbol{A}^T\boldsymbol{r}), (\boldsymbol{A}, \boldsymbol{y})) \leq 2^{-\kappa}$.*

Assuming the computational hardness of $\mathrm{DLWE}_{N,M,q,\chi}$ for appropriate parameters, by the above lemma we get for any $\mathtt{pp} = (\boldsymbol{A}, N, M, q, \chi, \lambda)$, $\mathtt{pk} = \boldsymbol{b}$ and

$m \in \{0,1\}$: $\mathsf{RGV.E}(\mathtt{pk}, m) = \boldsymbol{r}^T(\boldsymbol{A}, \boldsymbol{b}) + (0^N, m \cdot \lceil q/2 \rfloor)$ is computationally indistinguishable from $\boldsymbol{r}^T(\boldsymbol{A}, \boldsymbol{b}') + (0^N, m \cdot \lceil q/2 \rfloor)$, where $\boldsymbol{r} \leftarrow \{0,1\}^M$ and $\boldsymbol{b}' \leftarrow \mathbb{Z}_q^M$. The latter distribution is, in turn, statistically close to the uniform distribution on $\mathbb{Z}_q^{N+1}$. Informally, since $\mathsf{RGV.E}(\mathtt{pk}, 0)$ is computationally indistinguishable from uniformly random, ciphertexts, re-keys and re-encrypted ciphertexts all look uniformly random; in particular Construction 7.a satisfies indistinguishability of ciphertexts as well as δ-weak key privacy.

Lemma 14. *Assuming* $\mathrm{DLWE}_{N,M,q,\chi}$ *is* (s_1, ϵ_1)*-hard for parameters* N, M, q *as in Lemma 13, Construction 7.a satisfies* $(s_1 - s_{\mathsf{E}}, 2(\epsilon_1 + 2^{-\kappa}))$*-indistinguishability and* $(s_1 - O(\delta\, N \lceil \log q \rceil\, (s_{\mathbb{Z}_q^{N+1}} + s_{\mathsf{RGV.E}})), \delta N \lceil \log q \rceil \epsilon_1, \delta)$*-weak key-privacy.*

Theorem 11 (PRE-CPA-security of Construction 7.a). *Let* σ *and* τ *denote the space and time complexity for the class* $\mathcal{G} = \mathcal{G}(n, \delta, d)$*. Assume the* $\mathrm{DLWE}_{N,M,q,\chi}$ *problem is* (s_1, ϵ_1)*-hard for parameters* N, M, q *as in Lemma 13. Then Construction 7.a is* (s, ϵ)*-PRE-CPA-secure restricted to challenge graphs in* $\mathcal{G}$*, where*

$$s := s_1 - O(\delta\, N \lceil \log q \rceil\, (s_{\mathbb{Z}_q^{N+1}} + s_{\mathsf{RGV.E}})) - s_{\mathsf{CPA}} - s_{\mathcal{G}} \quad \text{and}$$
$$\epsilon := (2\tau \cdot \delta N \lceil \log q \rceil + 1) \cdot \epsilon_1 \cdot n^{\sigma+\delta+1}.$$

Construction 7.a clearly does not satisfy source-hiding and, thus cannot be proven PRE-HRA-secure using our results. Fortunately, Construction 7.b solves this issue, but at the cost of only allowing for a constant level bound λ. The additional uniform error $f_\ell \leftarrow [-E_\ell, E_\ell] \cap \mathbb{Z}$ added in E and RE in Construction 7.b is used to "blur out" the different errors caused by encryption or re-encryption, respectively. Choosing the error bounds E_ℓ appropriately guarantees the source-hiding property of the scheme while still preserving correctness.[7] Chandran et al. refer to this rerandomisation technique as *strong blurring*; a more detailed analysis can be found in [DS16, Section 4.1], where the same method for rerandomization of Regev ciphertexts is used to discuss sanitizability of the FHE scheme from [BV11].

To prove PRE-HRA-security of Construction 7.b, note that, as above, semantic security and δ-weak key-privacy of (E, D) directly follow by the security of Regev's encryption scheme. We get a result similar to Lemma 14.

Lemma 15. *For large enough (see Footnote 7) error ranges* E_ℓ*,* $\ell \in [\lambda]$*, Construction 7.b is (statistically) source-hiding.*

Theorem 12 (PRE-HRA-security of Construction 7.b). *Let* ϵ *be as in Theorem 11 and let* σ *and* τ *denote the space and time complexity for* $\mathcal{G} = \mathcal{G}(n, \delta, d)$*. If* $\mathrm{DLWE}_{N,M,q,\chi}$ *is* (s_1, ϵ_1)*-hard for parameters* N, M, q *as in Lemma 13*

[7] In fact, we need to choose the error bounds $(E_\ell)_{\ell \in [\lambda]}$ exponentially large, eg., $E_1 \geq (M+1)B2^\kappa$. Thus, to provide correctness of the scheme, one needs to choose the modulus q to be of size $\exp(O(\kappa))$ and the level bound λ of size $O(1)$.

and E_ℓ ($\ell \in [\lambda]$), λ are chosen appropriately, then Construction 7.b is (s', ϵ')-PRE-HRA-secure restricted to challenge graphs in $\mathcal{G}$, where

$$s' := s_1 - O(\delta\, N \lceil \log q \rceil\, (s_{\mathbb{Z}_q^{N+1}} + s_{\mathsf{RGV.E}})) - s_{\mathsf{HRA}} - s_{\mathcal{G}}, \quad \textit{and}$$
$$\epsilon' := 2n(n-1)(Q_{\mathsf{E}} + Q_{\mathsf{RE}})Q_{\mathsf{RE}} \cdot 2^{-\kappa} + \epsilon.$$

6 Open Problems

We leave as open problems to find adaptively secure PREs (either via the [JKK+17] framework or using a new technique) for more general settings, which includes unidirectional PREs on general graphs, bidirectional PREs and CCA-secure PRE(the schemes above only satisfy CPA, and the slightly stronger HRA security notion).

References

ABH09. Giuseppe Ateniese, Karyn Benson, and Susan Hohenberger. Key-private proxy re-encryption. In Marc Fischlin, editor, *Topics in Cryptology – CT-RSA 2009*, pages 279–294, Springer, 2009.

AFGH05. Giuseppe Ateniese, Kevin Fu, Matthew Green, and Susan Hohenberger. Improved proxy re-encryption schemes with applications to secure distributed storage. In *Proceedings of the Network and Distributed System Security Symposium, NDSS 2005, San Diego, California, USA*. The Internet Society, 2005.

BB04. Dan Boneh and Xavier Boyen. Secure identity based encryption without random oracles. In Matt Franklin, editor, *Advances in Cryptology – CRYPTO 2004*, volume 3152 of LNCS, pages 443–459. Springer, 2004.

BBS98. Matt Blaze, Gerrit Bleumer, and Martin Strauss. Divertible protocols and atomic proxy cryptography. In Kaisa Nyberg, editor, *Advances in Cryptology — EUROCRYPT'98*, pages 127–144, 1998. Springer.

BBS04. Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures. In Matt Franklin, editor, *Advances in Cryptology – CRYPTO 2004*, volume 3152 of LNCS, pages 41–55. Springer 2004.

Ben89. Charles H. Bennett. Time/space trade-offs for reversible computation. *SIAM Journal on Computing*, 18(4):766–776, 1989.

BV11. Z. Brakerski and V. Vaikuntanathan. Efficient fully homomorphic encryption from (standard) LWE. In *2011 IEEE 52nd Annual Symposium on Foundations of Computer Science*, pages 97–106, Oct 2011.

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 *Public-Key Cryptography - PKC 2014*, volume 8383, pages 95–112. Springer, March 2014.

CCV12. Nishanth Chandran, Melissa Chase, and Vinod Vaikuntanathan. Functional re-encryption and collusion-resistant obfuscation. In Ronald Cramer, editor, *Theory of Cryptography*, pages 404–421, 2012. Springer.

CH07. Ran Canetti and Susan Hohenberger. Chosen-ciphertext secure proxy re-encryption. In *Proceedings of the 14th ACM Conference on Computer and Communications Security*, CCS '07, pages 185–194, 2007. ACM.

Coh17. Aloni Cohen. What about Bob? The inadequacy of CPA security for proxy reencryption. Cryptology ePrint Report 2017/785, 2017. https://ia.cr/2017/785.

DS16. Léo Ducas and Damien Stehlé. Sanitization of FHE ciphertexts. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology – EUROCRYPT 2016*, pages 294–310, 2016. Springer.

FL17. Xiong Fan and Feng-Hao Liu. Proxy re-encryption and re-signatures from lattices. Cryptology ePrint Report 2017/456, https://ia.cr/2017/456.

Gen09. Craig Gentry. Fully homomorphic encryption using ideal lattices. In *Proceedings of the Forty-first Annual ACM Symposium on Theory of Computing*, STOC '09, pages 169–178, 2009. ACM.

GM82. Shafi Goldwasser and Silvio Micali. Probabilistic encryption & how to play mental poker keeping secret all partial information. In *Proceedings of the fourteenth annual ACM symposium on Theory of computing*, STOC '82, pages 365–377, 1982. ACM.

GPV07. Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. *Electronic Colloquium on Computational Complexity (ECCC)*, 14(133), 2007.

HRsV07. Susan Hohenberger, Guy N. Rothblum, abhi shelat, and Vinod Vaikuntanathan. Securely obfuscating re-encryption. In Salil P. Vadhan, editor, *Theory of Cryptography*, pages 233–252, 2007. Springer.

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*, pages 133–163. Springer, 2017.

LV08. Benoît Libert and Damien Vergnaud. Unidirectional chosen-ciphertext secure proxy re-encryption. In Ronald Cramer, editor, *Public Key Cryptography – PKC 2008*, volume 4939 of LNCS, pages 360–379. Springer 2008.

Pan07. Saurabh Panjwani. Tackling adaptive corruptions in multicast encryption protocols. In Salil P. Vadhan, editor, *Theory of Cryptography*, volume 4392 of LNCS, pages 21–40. Springer 2007.

PWA+16. Le Trieu Phong, Lihua Wang, Yoshinori Aono, Manh Ha Nguyen, and Xavier Boyen. Proxy re-encryption schemes with key privacy from lwe. Cryptology ePrint Report 2016/327, 2016. https://ia.cr/2016/327.

Reg05. Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *Proceedings of the Thirty-seventh Annual ACM Symposium on Theory of Computing*, STOC '05, pages 84–93, 2005. ACM.

Sco02. Mike Scott. Authenticated ID-based key exchange and remote log-in with simple token and PIN number. Cryptology ePrint Report 2002/164, 2002. https://ia.cr/2002/164.

A Selective Security Definitions

In order to relate the selective security notions sCPA and sHRA to their adaptive variants, we reformulate the former using recoding graphs instead of explicit abort conditions. Inspection of the games yields the following:

Lemma 16. *A PRE scheme is (s, ϵ)-selectively secure against chosen-plaintext attack (i.e, $\mathsf{sCPA}^0 \approx_{(s,\epsilon)} \mathsf{sCPA}^1$ with sCPA from Game 1) if and only if*

$$\mathsf{sCPA}^0_* \approx_{(s,\epsilon)} \mathsf{sCPA}^1_*$$

Challenger $\mathsf{sCPA}^b_*(1^\kappa, 1^\lambda, n)$

1: Set $\mathcal{C}, \mathcal{E} = \emptyset$ ▷ Stores corrupt keys and issued re-keys and re-encryptions
2: $\mathsf{pp} \leftarrow \mathsf{PRE.S}(1^\kappa, 1^\lambda)$, $(\mathtt{pk}_1, \mathtt{sk}_1), \ldots, (\mathtt{pk}_n, \mathtt{sk}_n) \leftarrow \mathsf{PRE.K}(\mathsf{pp})$ ▷ Generate keys
3: $\forall i, j \in [n], i \neq j : \mathtt{rk}_{i,j} \leftarrow \mathsf{PRE.RK}((\mathtt{pk}_i, \mathtt{sk}_i), \mathtt{pk}_j)$ ▷ Generate re-keys
4: $\mathsf{state} \leftarrow \mathsf{A}_1^{(\mathtt{corrupt}, \cdot)}(\mathsf{pp})$ ▷ Phase 1
5: $b' \leftarrow \mathsf{A}_2^{(\mathtt{rekey}, \cdot, \cdot), (\mathtt{reencrypt}, \cdot, \cdot, \cdot), (\mathtt{challenge}, \cdot, \cdot, \cdot)}(\mathtt{pk}_1, \ldots, \mathtt{pk}_n, \mathsf{state})$ ▷ Phase 2
6: **if** $i^* \in \mathcal{C}$ or $\exists\, i \notin \mathcal{C}\, \exists\, j \in \mathcal{C}$: i is connected to j in $([n], \mathcal{E})$ **then return** 0
7: **end if**
8: **return** b'

Oracle $(\mathtt{corrupt}, i)$

1: Add i to $\mathcal{C}$
2: **return** $\mathtt{sk}_i$

Oracle $(\mathtt{rekey}, i, j)$

1: Add (i, j) to $\mathcal{E}$ ▷ Add to recoding graph
2: **return** $\mathtt{rk}_{i,j}$

Oracle $(\mathtt{reencrypt}, i, j, (c_i, \ell))$

1: Add (i, j) to $\mathcal{E}$ ▷ Add to recoding graph
2: **return** $(c_j, \ell + 1) \leftarrow \mathsf{PRE.RE}(\mathtt{rk}_{i,j}, \mathtt{pk}_i, \mathtt{pk}_j, (c_i, \ell))$

Oracle $(\mathtt{challenge}, i^*, (m_0^*, m_1^*), \ell^*)$ ▷ Single access

1: **return** $(c_{i^*}, \ell^*) \leftarrow \mathsf{PRE.E}(\mathtt{pk}_{i^*}, (m_b^*, \ell^*))$

Game 9: sPRE-CPA using recoding graph to check illegal behaviour.

with sCPA_* *defined in Game 9.*

Lemma 17. *A PRE scheme is* (s, ϵ)*-selectively secure against honest-reencryption attack (i.e,* $\mathsf{sHRA}^0 \approx_{(s,\epsilon)} \mathsf{sHRA}^1$ *with* sHRA *from Game 2) if and only if*

$$\mathsf{sHRA}^0_* \approx_{(s,\epsilon)} \mathsf{sHRA}^1_*$$

with sHRA_* *defined in Game 10.*

Challenger $\mathsf{sHRA}^b_*(1^\kappa, 1^\lambda, n)$
1: Set $\mathcal{C}, \mathcal{L}, \mathcal{L}^* = \emptyset$ ▷ Corrupt keys, ciphertexts and which derive from challenge
2: Set $C = 0$ ▷ Counts ciphertexts generated
3: $\mathtt{pp} \leftarrow \mathsf{PRE.S}(1^\kappa, 1^\lambda)$, $(\mathtt{pk}_1, \mathtt{sk}_1), \ldots, (\mathtt{pk}_n, \mathtt{sk}_n) \leftarrow \mathsf{PRE.K}(\mathtt{pp})$ ▷ Generate keys
4: $\forall i, j \in [n], i \neq j : \mathtt{rk}_{i,j} \leftarrow \mathsf{PRE.RK}((\mathtt{pk}_i, \mathtt{sk}_i), \mathtt{pk}_j)$ ▷ Generate re-keys
5: $\mathsf{state} \leftarrow \mathsf{A}_1^{(\mathtt{corrupt}, \cdot)}(\mathtt{pp})$ ▷ Phase 1
6: $b' \leftarrow \mathsf{A}_2^{(\mathtt{encrypt}, \cdot, \cdot), (\mathtt{rekey}, \cdot, \cdot), (\mathtt{reencrypt}, \cdot, \cdot, \cdot), (\mathtt{challenge}, \cdot, \cdot, \cdot)}(\mathtt{pk}_1, \ldots, \mathtt{pk}_n, \mathsf{state})$ ▷ Phase 2
7: **if** A made call $(\mathtt{challenge}, i^*, \cdot, \cdot)$ for some i^* **then** ▷ Check abort conditions
8: **if** $\exists\, i \in \mathcal{C} : i^*$ is connected to i **then return** 0 **end if**
9: **end if**
10: **return** b'

Oracles `corrupt` and `rekey` are defined like in Game 9.

Oracle $(\mathtt{encrypt}, i, (m, \ell))$
1: $(c, \ell) \leftarrow \mathsf{PRE.E}(\mathtt{pk}_i, (m, \ell))$
2: Increment C and add $(C, i, m, (c, \ell))$ to $\mathcal{L}$
3: **return** (c, ℓ)

Oracle $(\mathtt{reencrypt}, i, j, k)$
1: Retrieve $(k, i, m, (c_i, \ell))$ from $\mathcal{L}$ and increment C
2: $(c_j, \ell + 1) \leftarrow \mathsf{PRE.RE}(\mathtt{rk}_{i,j}, \mathtt{pk}_i, \mathtt{pk}_j, (c_i, \ell))$
3: **if** $k \in \mathcal{L}^*$ **then** ▷ The ciphertext is derived from the challenge
4: Add C to $\mathcal{L}^*$ and add (i, j) to $\mathcal{E}$ ▷ Add to recoding graph
5: **end if**
6: Add $(C, j, m, (c_j, \ell + 1))$ to $\mathcal{L}$
7: **return** $(c_j, \ell + 1)$

Oracle $(\mathtt{challenge}, i^*, (m_0^*, m_1^*), \ell^*)$ ▷ Single access
1: Compute $(c_{i^*}, \ell^*) \leftarrow \mathsf{PRE.E}(\mathtt{pk}_{i^*}, (m_b^*, \ell^*))$
2: Increment C, add $(C, i^*, m_b^*, (c_{i^*}, \ell^*))$ to $\mathcal{L}$ and C to $\mathcal{L}^*$
3: **return** (c_{i^*}, ℓ^*)

Game 10: sPRE-HRA using recoding graph to check illegal behaviour.

B Application to Key Rotation

An interesting application of unidirectional multi-hop PRE is for key rotation in remote storage systems, for which it can mitigate the risk of key compromise. Consider a user that stores content on a server which is encrypted under her public key $\mathtt{pk}$. Using key rotation, from time to time the user creates a new key pair $(\mathtt{pk}', \mathtt{sk}')$ and would like to replace the encryption of the content under $\mathtt{pk}$ by an encryption under $\mathtt{pk}'$.

To do so, it seems she must either download the content, re-encrypt it and upload it again; or give the secret key $\mathtt{sk}$ for $\mathtt{pk}$ to the server, so the latter can obtain the content and encrypt it under the new key. PRE allows avoiding both costly transfer of content *and* unnecessary trust in the server: the user simply creates $\mathtt{rk} \leftarrow \mathsf{RK}((\mathtt{pk}, \mathtt{sk}), \mathtt{pk}')$ and sends $\mathtt{rk}$ to the server. The latter uses $\mathtt{rk}$ to re-encrypt the content to the new key $\mathtt{pk}'$ without knowing either content nor any secret keys.

There are several attack scenarios to consider: an adversary could (i) obtain the encrypted content or (ii) a re-key by breaking into the server; and it could obtain (iii) the secret key by attacking the user. We would like to guarantee that, as long as the adversary does not see a ciphertext which it can decrypt by either obtaining the secret key for it or re-keys that allow it to (consecutively) re-encrypt it to a key it knows, nothing is leaked on the plaintext.

We formalize this via a game that considers n epochs in each of which a new key pair is generated and the content is re-encrypted. The adversary obtains all public keys and can ask for messages to be encrypted at certain epochs; one of which is its challenge query which lets it choose two messages (m_0, m_1). The adversary can at any time query secret keys for any epoch and re-keys between two epochs. It can also ask to see (re-)encryptions of (the challenge) messages. If it asks to see the challenge ciphertext in some epoch i and corrupts a secret key in some epoch j such that either $j = i$ or it has obtained all re-keys $\mathtt{rk}_{i,i+1}, \ldots, \mathtt{rk}_{j-1,j}$ then it loses. Otherwise, the adversary wins if it guesses the bit b chosen by the challenger that determines whether m_0 or m_1 was encrypted.

Definition 21 (KRot security). *A PRE scheme is (s, ϵ)-adaptively secure against key rotation attacks if $\mathsf{KRot}^0 \approx_{(s,\epsilon)} \mathsf{KRot}^1$, where KRot^b is defined in Game 11.*

Note that, as with all other notions, restricting the adversary to a single `challenge` call is without loss of generality: a standard hybrid argument shows that this notion implies a more general multi-challenge variant of this definition. (In the jth hybrid, the first $j-1$ calls $(\mathtt{challenge}, i, (m_0, m_1))$ are answered like $(\mathtt{encrypt}, i, m_1)$ and the remaining ones like $(\mathtt{encrypt}, i, m_0)$.)

The adaptive notion of key-rotation security we defined is not immediately implied by HRA security: in the former the adversary is allowed to ask for m_b to be encrypted under $\mathtt{pk}_1$, to see its re-encryption under $\mathtt{pk}_2$ and to corrupt $\mathtt{pk}_1$. This is not allowed by the HRA game, as a challenge ciphertext is immediately revealed by the oracle and the corresponding key must therefore not be corrupted. We show however that if the PRE scheme is source-hiding then HRA security implies key-rotation (KRot) security.

Theorem 13 (KRot security). *Let $n \in \mathbb{N}$. Then a PRE scheme that is (s_1, ϵ_1)-PRE-HRA-secure (under multiple challenges) and (s_2, ϵ_2)-source-hiding is (s, ϵ)-KRot-secure, where*

$$s := \min\{s_1, s_2\} - s_{\mathsf{KRot}}, \quad \textit{and} \quad \epsilon := \epsilon_1 + (n-1)\epsilon_2,$$

where s_{KRot} denotes the complexity of simulating game KRot.

Proof. To prove the theorem we show that if the encryption scheme is source-hiding then the game KRot is indistinguishable from a game where, instead of being re-encrypted, the challenge is freshly encrypted when the adversary queries its `reveal` oracle on it.

Consider the hybrid game shKRot^b_t defined as KRot^b, except that the challenge oracle is defined as follows:

Challenger $\mathsf{KRot}^b(1^\kappa, 1^n)$

1: Set $\mathcal{C}, \mathcal{E}, \mathcal{S} = \emptyset$ ▷ Stores corrpt. keys ($\mathcal{C}$), re-keys ($\mathcal{E}$) and revealed challenges ($\mathcal{S}$)
2: Set $\mathcal{L}, \mathcal{L}^* = \emptyset$ ▷ Stores ciphertexts and challenge ciphertexts
3: Set $C = 0$ ▷ Counter for generated ciphertexts
4: $\mathsf{pp} \leftarrow \mathsf{PRE.S}(1^\kappa, 1^n)$ ▷ n determines the number of levels
5: $(\mathtt{pk}_1, \mathtt{sk}_1), \ldots, (\mathtt{pk}_n, \mathtt{sk}_n) \leftarrow \mathsf{PRE.K}(\mathsf{pp})$
6: $\forall i \in [2, n] : \mathtt{rk}_{i-1,i} \leftarrow \mathsf{PRE.RK}((\mathtt{pk}_{i-1}, \mathtt{sk}_{i-1}), \mathtt{pk}_i)$
7: $b' \leftarrow \mathsf{A}^{(\mathtt{corrupt},\cdot),(\mathtt{rekey},\cdot,\cdot),(\mathtt{encrypt},\cdot,\cdot),(\mathtt{challenge},\cdot,\cdot,\cdot),(\mathtt{reveal},\cdot,\cdot)}(\mathsf{pp}, \mathtt{pk}_1, \ldots, \mathtt{pk}_n)$
8: **if** $\exists i \in \mathcal{S} \exists j \in \mathcal{C} : i$ is connected to j **then return** 0 **end if**
9: **return** b'

Oracle $(\mathtt{corrupt}, i)$

1: Add i to $\mathcal{C}$
2: **return** $\mathtt{sk}_i$

Oracle $(\mathtt{rekey}, i)$

1: Add $(i-1, i)$ to $\mathcal{E}$
2: **return** $\mathtt{rk}_{i-1,i}$

Oracle $(\mathtt{encrypt}, i, m)$

1: Increment C, compute $(c_i, i) \leftarrow \mathsf{PRE.E}(\mathtt{pk}_i, (m, i))$
2: **for** $j = i+1 \ldots n$ **do** $(c_j, j) \leftarrow \mathsf{PRE.RE}(\mathtt{rk}_{j-1,j}, \mathtt{pk}_{j-1}, \mathtt{pk}_j, (c_{j-1}, j-1))$
3: **end for**
4: Increment C, add $(C, i, m, (c_i, \ldots, c_n))$ to $\mathcal{L}$

Oracle $(\mathtt{challenge}, i, (m_0^*, m_1^*))$ ▷ Single access

1: Compute $(c_i, i) \leftarrow \mathsf{PRE.E}(\mathtt{pk}_i, (m_b^*, i))$
2: **for** $j = i+1 \ldots n$ **do** $(c_j, j) \leftarrow \mathsf{PRE.RE}(\mathtt{rk}_{j-1,j}, \mathtt{pk}_{j-1}, \mathtt{pk}_j, (c_{j-1}, j-1))$ **end for**
3: Increment C, add $(C, i, m_b^*, (c_i, \ldots, c_n))$ to $\mathcal{L}$ and C to $\mathcal{L}^*$

Oracle $(\mathtt{reveal}, k, j)$

1: Retrieve $(k, i, m, (c_i, \ldots, c_n))$ from $\mathcal{L}$
2: **if** $k \in \mathcal{L}^*$ **then** add j to $\mathcal{S}$ **end if**
3: **return** c_j

Game 11: Game KRot for key rotation using a (unidirectional multi-hop) PRE scheme PRE.

Oracle $(\mathtt{challenge}, i, (m_0^*, m_1^*))$

1: Compute $(c_i, i) \leftarrow \mathsf{PRE.E}(\mathtt{pk}_i, (m_b^*, i))$
2: **for** $j = i+1 \ldots t$ **do** $(c_j, j) \leftarrow \mathsf{PRE.E}(\mathtt{pk}_j, (m_b^*, j))$ **end for**
3: **for** $j = t+1 \ldots n$ **do** $(c_j, j) \leftarrow \mathsf{PRE.RE}(\mathtt{rk}_{j-1,j}, \mathtt{pk}_{j-1}, \mathtt{pk}_j, (c_{j-1}, j-1))$ **end for**
4: Increment C, add $(C, i, m_b^*, (c_i, \ldots, c_n))$ to $\mathcal{L}$ and C to $\mathcal{L}^*$

The difference between shKRot_{t-1}^b and shKRot_t^b is whether the challenge oracle computes c_t by reencrypting or by freshly encrypting. An adversary distinguishing shKRot_{t-1}^b from shKRot_t^b can thus be turned into an adversary against source-hiding (Definition 9): the reduction receives $(\mathsf{pp}, (\mathtt{pk}, \mathtt{sk}), (\mathtt{pk}', \mathtt{sk}'), \mathtt{rk})$, simulates the experiment setting $(\mathtt{pk}_{t-1}, \mathtt{sk}_{t-1}) := (\mathtt{pk}, \mathtt{sk})$, $(\mathtt{pk}_t, \mathtt{sk}_t) := (\mathtt{pk}', \mathtt{sk}')$, and $\mathtt{rk}_{t-1,t} := \mathtt{rk}$. When the adversary queries $(\mathtt{challenge}, i, (m_0^*, m_1^*))$, the reduction first computes for $j = i+1 \ldots t-2$: $(c_j, j) \leftarrow \mathsf{PRE.E}(\mathtt{pk}_j, (m_b^*, j))$; it then queries $(\mathtt{challenge}, m_b^*, t-1)$ to receive c_{t-1} (a fresh encryption) and c_t

(a re-encryption of c_{t-1} or a fresh encryption depending on the challenger); for $j = t+1 \ldots n$ it computes $(c_j, j) \leftarrow \mathsf{PRE.RE}(\mathtt{rk}_{j-1,j}, \mathtt{pk}_{j-1}, \mathtt{pk}_j, (c_{j-1}, j-1))$. The reduction continues the simulation of the game (it has all secret keys) and returns whatever the adversary does.

shKRot_1^b is the original game KRot^b, whereas shKRot_n^b is a game where the challenge message is freshly re-encrypted for every $\mathtt{pk}_i$. Moreover, shKRot_n^b is equivalently distributed to a game shKRot_{n+1}^b where the challenge oracle does not compute anything yet, and the show oracle directly computes challenge ciphertexts. shKRot_{n+1}^b is thus defined like KRot^b except for the following oracles:

Oracle $(\mathtt{challenge}, i, (m_0^*, m_1^*))$
1: Increment C, add $(C, i, m_b^*, \emptyset)$ to $\mathcal{L}$ and C to $\mathcal{L}^*$

Oracle $(\mathtt{reveal}, k, j)$
1: Retrieve $(k, i, m, (c_i, \ldots, c_n))$ from $\mathcal{L}$ ▷ $(c_i, \ldots, c_n)$ could be empty
2: **if** $k \in \mathcal{L}^*$ **then**
3: Add j to $\mathcal{S}$
4: Compute $(c_j, j) \leftarrow \mathsf{PRE.E}(\mathtt{pk}_j, (m, j))$
5: **end if**
6: **return** c_j

Game shKRot_{n+1}^b can now be simulated for an adversary A in a straightforward way by a reduction R that plays (the multi-challenge version of) game HRA^b: R forwards all `corrupt` and `rekey` queries; when A makes a query `encrypt`, then R generates $(c_i, c_{i+1}, \ldots, c_n)$ as follows: it queries its `encrypt` oracle to get c_i and its `reencrypt` oracle to obtain $c_{i+1}, \ldots, c_n$. (Note that this does not lead to any edges being added to $\mathcal{E}$.) When A makes its query `challenge` then R faithfully simulates shKRot_{n+1}^b and stores the message in $\mathcal{L}$ and the current counter value in $\mathcal{L}^*$. Queries $(\mathtt{reveal}, k, j)$ with $k \notin \mathcal{L}^*$ are answered as specified in shKRot_{n+1}^b; if $k \in \mathcal{L}^*$ then R queries $(\mathtt{challenge}, j, (m_0^*, m_1^*), j)$ and forwards the reply. Finally R returns A's final output b'.

R's probability of winning KRot^b is the same as A's probability of winning shKRot_{n+1}^b, as R only violates the winning condition that no key that was challenged is connected to a corrupt key via re-keys (represented by edges in $\mathcal{E}$ in both games) when and only when A does in game shKRot_{n+1}^b. □