The Related-Key Analysis of Feistel Constructions

Manuel Barbosa[1] and Pooya Farshim[2]

[1] HASLab – INESC TEC and Universidade do Minho, Portugal
[2] Fachbereich Informatik, Technische Universität Darmstadt, Germany
mbb@di.uminho.pt, pooya.farshim@gmail.com

Abstract. It is well known that the classical three- and four-round Feistel constructions are provably secure under chosen-plaintext and chosen-ciphertext attacks, respectively. However, irrespective of the number of rounds, no Feistel construction can resist related-key attacks where the keys can be offset by a constant. In this paper we show that, under suitable reuse of round keys, security under related-key attacks can be provably attained. Our modification is substantially simpler and more efficient than alternatives obtained using generic transforms, namely the PRG transform of Bellare and Cash (CRYPTO 2010) and its random-oracle analogue outlined by Lucks (FSE 2004). Additionally we formalize Luck's transform and show that it does not *always* work if related keys are derived in an oracle-dependent way, and then prove it sound under appropriate restrictions.

Keywords. Feistel construction, Luby–Rackoff, Related-key attack, Pseudorandom permutation, Random oracle.

1 Introduction

Cryptographic algorithms deployed in the real world are subject to a multitude of threats. Many of these threats are accounted for in the theoretical security analysis carried out by cryptographers, but not all. Indeed, many documented cases [31,14,13,39] show that theoretically secure cryptographic algorithms can be vulnerable to relatively simple physical attacks, when these exploit implementation aspects that were abstracted away in the security analysis. For this reason, an enormous research effort has been undertaken in recent years to bridge the gap between physical security and theoretical security.

An important part of this effort has been dedicated to *related-key attacks* (RKAs), which were first identified by Knudsen and Biham [26,8] as an important risk on implementations of symmetric-key cryptosystems. The idea behind these attacks is as follows. The security of cryptographic algorithms depends fundamentally on keeping secret keys hidden from attackers for extended periods of time. For this reason, secret keys are typically stored and manipulated in protected memory areas and dedicated hardware components. When these mechanisms can be influenced by intrusive techniques (such as fault injection [2]) an adversary may be able to disturb the value of a secret key and observe results computed using the manipulated (and likely correlated) key value.

Since the original work of Knudsen and Biham, there have been many reported cases of successful related-key cryptanalysis [9,27,7], and notably of the Advanced Encryption Standard (AES) [10,11]. These results led to the consensual view that RKA resilience should be a standard design goal for low-level cryptographic primitives such as block ciphers and hash functions. For example, in the recent SHA-3 competition, candidates were analyzed with respect to such attacks (cf. the work of Khovratovich et al. [25]), which played an important role in the selection process.

The importance of including RKA security as a design goal for basic cryptographic components is further heightened by the fact that such low-level primitives are often *assumed* to provide RKA security when used in higher-level protocols. Prominent examples are the key derivation procedures in standard protocols such as EMV [15] and the 3GPP integrity and confidentiality algorithms [24], where efficiency considerations lead to the use of the same block cipher under closely related keys.

Similar assumptions arise in constructions of *tweakable ciphers* [28], where a block cipher is called on keys which are offset by XOR-ing tweak values.

Provable RKA security. Bellare and Kohno [5] initiated the theoretical treatment of security under related-key attacks by proposing definitions for RKA-secure pseudorandom functions (PRFs) and pseudorandom permutations (PRPs), and presenting possibility and impossibility results for these primitives. The models proposed in [5] were subsequently extended by Albrecht et al. [1] to address the possibility of oracle-dependent attacks in idealized models of computation.

Various important positive results for provably RKA-secure constructions of complex cryptographic primitives were subsequently published in the literature. Bellare and Cash [3] obtained a breakthrough result by presenting a concrete construction of an RKA-secure pseudorandom function based on standard computational assumptions and in the standard model. Bellare, Cash, and Miller [4] present a comprehensive treatment of RKA security for various cryptographic primitives, focusing on the problem of leveraging the RKA resilience of one primitive to construct RKA-secure instances of another. In particular, Bellare et al. present a generic transformation in which an RKA-secure pseudorandom generator can be used to convert instances of standard primitives such as digital signatures and identity-based encryption into RKA-secure ones. Concrete constructions of RKA-secure public-key primitives were given by Wee [42] and Bellare et al. [6].

Feistel networks. A Feistel network [16,17] is a construction that permits obtaining an efficiently computable and invertible permutation from an efficiently computable function. The network is a cascade of simple Feistel permutations, each relying on a *round function* (f, g, and h) mapping bit strings of length n to outputs of the same length. Here the input and output are shown as tuples (L, R) and (L', R'), where each component is a string of length n. For any number of rounds, these networks provide an invertible permutation over bit strings of length $2n$. Figure 1 shows an example of a Feistel network with three rounds.

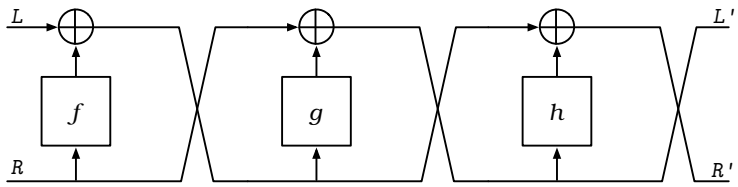

Fig. 1: A three-round Feistel network.

Feistel networks (and generalized variants such as those discussed by Hoang and Rogaway [22]) have been extensively used in the construction of symmetric cryptosystems (and even asymmetric ones such as RSA-OAEP), since the notable case of the Data Encryption Standard (DES) in the 1970s [17]. In particular, a multitude of block ciphers include Feistel-like constructions in their design, including GOST, MYSTY1, Skipjack, BEAR and LION, CAST-256, RC6, and MARS [38]. For this reason, the security properties of Feistel networks received significant attention in the last decades.

Security of the Feistel construction. In their seminal paper, Luby and Rackoff [29] showed that instantiating the round functions in a Feistel construction with independently keyed secure PRFs is sufficient to obtain a secure PRP. For three rounds of cascading, this result applies when the adversary has access to results of forward computations (i.e., under chosen-plaintext attacks), and for four rounds, the result holds even if the adversary can additionally observe the results of inverse computations (i.e., under chosen-ciphertext attacks).

Following Luby and Rackoff's result, many subsequent works looked at the security of Feistel networks and generalized variants thereof. Important results were obtained with respect to the efficiency of the construction, for example by reducing the necessary key material (cf. the work of Patarin [36])

and by weakening the security assumptions for some of the round functions as in the work of Naor and Reingold [35]. In a different direction, the security offered by Feistel networks with increasing numbers of rounds was precisely characterized in a sequence of works by Vaudenay [41], Maurer and Pietrzak [32], Patarin [37], and Hoang and Rogaway [22]. Holenstein, Künzler, and Tessaro [23] used the Feistel construction with fourteen rounds to establish the equivalence of the random-oracle and the ideal-cipher models in a broad range of applications via the indifferentiability framework [33].

RKA security of Feistel networks. Despite this large body of work on the provable security of the Feistel construction and the positive results on the RKA security of advanced cryptographic primitives referred above, the RKA security of the Feistel construction has received little attention. Indeed, to the best of our knowledge, only the work of Bellare and Kohno [5] touches upon this topic, where a strong negative result is shown: the Feistel construction *irrespective of the number of rounds* is vulnerable to related-key attacks, provided that the attacker is able to modify as little as a single bit in the key used in the last round function.

Referring to Figure 1, the attacker would proceed as follows. It would first observe the output (L'_1, R'_1) of the permutation computed on an input (L, R). Then, the adversary would modify round function h to some other function h' by manipulating its key, and observe the output (L'_2, R'_2) computed over the same input. The adversary can now determine whether or not it is interacting with the ideal permutation: If interacting with Feistel, the outputs will *always* satisfy $L'_1 = L'_2$, whereas in for the ideal (keyed) permutation the two outputs will be different with overwhelming probability. This attack is possible whenever the adversary is able to independently tweak the round function of the output stage in the network, independently of the number of rounds, and even if the round functions are instantiated with RKA-secure PRFs.

This vulnerability is relevant for practical applications of Feistel constructions, since many important cryptanalytic results such as those presented by Biryukov et al. [10,11] can be described as utilizing related keys that are derived by xor-ing the original key with a constant. This in particular permits an attacker to selectively modify the secret key for the output round in a Feistel network and break the security of the construction. In this work we initiate the treatment of provable RKA security of the Feistel constructions. Our main result is that specific instances of Feistel networks that reuse round keys offer *intrinsic* RKA security against practically relevant classes of RKD functions, and thus overcome the negative result by Bellare and Kohno described above. We now present our contributions in more detail.

Contributions. Lucks [30] proposes a general solution to the RKA security of any cryptographic primitive in the random-oracle model: hash the secret key before applying it to the cryptosystem. The intuition is that, modeling the hash function as a random oracle, any modification to the secret key will result in a new independent key being used in the cryptosystem, confining the RKA adversary to standard attacks. The RKA-secure PRG transform of Bellare, Cash, and Miller (BCM) [4] that we discussed above can be seen as a special standard-model analogue of this transform. Somewhat surprisingly, we show that the original random oracle transform does not *always* result in an RKA-secure construction. We amend this by first showing that, under certain restrictions on the RKD set, the random oracle is an RKA-secure PRG, and then extending the BCM result to the random-oracle model. The set of necessary restrictions is permissive enough to include offsetting keys by constants (even if those keys were hashed) as a particular case. This solution, however, in addition to relying on strong assumptions on the hash function, gives rise to decreased efficiency with respect to the original primitive.

Moreover, the above result only applies to a transformed construction and says nothing about the RKA security of Feistel constructions (which could be present in the construction of the hash function itself!). We therefore revisit the Bellare–Kohno (BK) negative result and complement it by

characterizing the class of related-key attacks that *can* be sustained by three and four rounds Feistel networks with independent round keys (i.e., the original Luby–Rackoff constructions). The class of tolerated attacks is highly restrictive and, in particular, it excludes the XOR-with-constants set. (This was to be expected, since the BK attack can be launched using these RKD functions.)

We next consider variants of Feistel constructions in which the keys to round functions in different stages of the network may be *reused*. These variants were already proposed in the literature (cf. the work by Patarin [36]) due to the efficiency and security benefits of reducing the necessary secret key material. However, we observe that key reuse has the added effect of *limiting* the power of an RKA adversary in targeting individual round keys. We build on this intuition to obtain our main results: we show that Feistel networks with three (resp., four) rounds can be proven CPA (resp., CCA) RKA secure by relying on an RKA-secure PRF and using specific key assignments that reuse some of the round keys.

Intuitively, our selection of key reusing assignments can be described as follows. It is well known that reusing the same key in all rounds of the Feistel network or, more generally any palindromic assignment of the keys, leads to totally insecure constructions. Additionally, the BK attack rules out key assignments where the key to the output round (in both forward and inverse computations) can be independently thwarted. These restrictions leave few plausible key assignments for intrinsic RKA security of three- and four-round Feistel networks. From these candidates we selected two specific assignments based on *two* PRF keys K_1 and K_2: we consider the key assignment (K_1, K_2, K_2) for the three-round variant, and the (K_1, K_2, K_1, K_2) key assignment for the four-round variant. We prove that the three-round variant is CPA secure and that the four-round variant is CCA secure, both in the RKA setting, assuming that the underlying PRF is RKA secure, and that the RKD set satisfies natural restrictions akin to those adopted, e.g., in [5].

Our results require no other modification to the original constructions in addition to the key assignment, and therefore come with minimal modifications to deployed implementations.[3] We are able to prove the RKA security of the three-stage (CPA) and four-stage (CCA) Luby–Rackoff constructions, whilst *reducing* the amount of key material and potentially improving the efficiency of the resulting implementations.

For practical applications, the most important aspect of our results is perhaps that they cover the standard classes of RKD functions considered in literature, namely those which offset the key by XOR-ing a constant. However, for the sake of generality our presentation relies on a slightly more abstract framework, where we characterize the covered classes of covered RKD functions by defining a set of sufficient restrictions that they must satisfy. This approach also enables a clearer and more modular presentation. For example, as an intermediate step, we formalize a notion of multi-key RKA security that may be of independent interest, and relate it to the standard single-key variant.

From a foundational perspective, our result can be seen as one bringing RKA security analysis to the classical constructions of pseudorandom objects. Goldberg and Liskov [18] study this question for building RKA-secure pseudorandom generators (where the seed is interpreted as the key) from one-way functions via Goldreich–Levin [19]. However, the natural questions of transforming RKA-secure PRGs to RKA-secure PRFs via the GGM construction [20] or RKA-secure PRFs to PRPs via the Luby–Rackoff constructions [29] have not been addressed yet. Our results can been seen as giving a positive answer to the latter question for the XOR set.

[3] Albeit imposing a stronger security assumption on the underlying PRF.

2 Preliminaries

NOTATION. We write $x \leftarrow y$ for the action of assigning the value y to the variable x. We write $x_1, \ldots, x_n \leftarrow\$ \mathsf{X}$ for sampling $x_1, \ldots, x_n$ from a finite set X uniformly at random. If $\mathcal{A}$ is a probabilistic algorithm we write $y_1, \ldots, y_n \leftarrow\$ \mathcal{A}(x_1, \ldots, x_n)$ for the action of running $\mathcal{A}$ on inputs $x_1, \ldots, x_n$ with independently chosen coins, and assigning the result to $y_1, \ldots, y_n$. For a vector $\mathbf{x} = (x_1, \ldots, x_n)$, we define $\mathbf{x}|_i = x_i$. We let $[n] := \{1, \ldots, n\}$. A function $\epsilon(\lambda)$ is negligible if $|\epsilon(\lambda)| \in \lambda^{-\omega(1)}$. PPT as usual abbreviates probabilistic polynomial-time.

KEYED FUNCTIONS AND PERMUTATIONS. Let Dom_λ, Rng_λ, and KSp_λ be three families of finite sets parametrized by a security parameter $\lambda \in \mathbb{N}$. We denote the set of all functions $\rho : \mathsf{Dom}_\lambda \longrightarrow \mathsf{Rng}_\lambda$ by $\mathrm{Func}(\mathsf{Dom}_\lambda, \mathsf{Rng}_\lambda)$. A keyed function is a set of functions in $\mathrm{Func}(\mathsf{Dom}_\lambda, \mathsf{Rng}_\lambda)$ indexed by the elements of the key space KSp_λ. We denote the set of all keyed functions by $\mathrm{Func}(\mathsf{KSp}_\lambda, \mathsf{Dom}_\lambda, \mathsf{Rng}_\lambda)$. By the ideal keyed function, we mean the family of distributions corresponding to choosing a function uniformly at random from $\mathrm{Func}(\mathsf{KSp}_\lambda, \mathsf{Dom}_\lambda, \mathsf{Rng}_\lambda)$. The random oracle is the ideal keyed function where KSp_λ for each $\lambda \in \mathbb{N}$ contains a single key. We denote the set of all permutations on Dom_λ by $\mathrm{Perm}(\mathsf{Dom}_\lambda)$. Note that each permutation uniquely defines its inverse permutation (which is also a member of this set). We define a family of keyed permutations analogously by indexing a set of permutations according to keys in some space KSp_λ. We denote the set of all such keyed permutations by $\mathrm{Perm}(\mathsf{KSp}_\lambda, \mathsf{Dom}_\lambda)$. The ideal keyed permutation (aka the ideal cipher) is defined as the family of distributions that choose a random element of $\mathrm{Perm}(\mathsf{KSp}_\lambda, \mathsf{Dom}_\lambda)$.

PSEUDORANDOM FUNCTION AND PERMUTATION FAMILY. A pseudorandom function family $\mathsf{PRF} := \{\mathsf{PRF}_\lambda\}_{\lambda \in \mathbb{N}}$ is a family of efficiently implementable keyed functions, i.e., functions $\mathsf{PRF}_\lambda : \mathsf{KSp}_\lambda \times \mathsf{Dom}_\lambda \longrightarrow \mathsf{Dom}_\lambda$, where PRP_λ can be computed in polynomial time in λ, together with an efficient procedure for sampling of keys and domain points which by a slight abuse of notation we denote by $\mathsf{KSp}(1^\lambda)$ and $\mathsf{Dom}(1^\lambda)$, respectively. A pseudorandom permutation family is defined analogously with the extra requirement that the inverse of each permutation in the family is also efficiently computable.

3 RKA-Secure Pseudorandom Functions and Permutations

In this section we introduce the formal framework in which we will analyze the RKA security of Feistel constructions. We begin by formalizing the notion of a family of related-key deriving (RKD) functions, which will parametrize our RKA security notions. Subsequently we introduce a generalization of the standard security model for RKA-secure pseudorandom functions and permutations to a scenario where multiple secret keys may be present in the system and influence the secret key derived by an RKD function. This is the natural setting for analyzing Feistel networks, as they use multiple instances of the same PRF.

FAMILY OF RKD SETS. A family of n-ary related-key deriving (RKD) sets Φ is a family of RKD sets $\{\Phi_\lambda\}$ consisting of RKD functions ϕ (viewed as circuits) which map an n-tuple of keys in some key space KSp_λ to a new key in KSp_λ, i.e., $\phi : \mathsf{KSp}_\lambda^n \to \mathsf{KSp}_\lambda$. Throughout the paper we assume that membership in any RKD set can be efficiently decided.

MULTI-KEY RKA SECURITY. Let $\mathsf{PRP} := \{\mathsf{PRP}_\lambda : \mathsf{KSp}_\lambda \times \mathsf{Dom}_\lambda \longrightarrow \mathsf{Dom}_\lambda\}$ be a pseudorandom permutation family and let $\Phi := \{\Phi_\lambda\}$ be a family of n-ary RKD sets where the implicit key space of the RKD functions in Φ_λ is KSp_λ. Let game $\mathsf{RKCCA}_{\mathsf{PRP},\mathcal{A},\Phi}(1^\lambda)$ be as shown in Figure 2. We say that PRP is Φ-RKCCA secure if the advantage of any legitimate PPT adversary $\mathcal{A}$ defined as

$$\mathbf{Adv}^{\mathsf{rkcca}}_{\mathsf{PRP},\mathcal{A},\Phi}(\lambda) := 2 \cdot \Pr\left[\mathsf{RKCCA}_{\mathsf{PRP},\mathcal{A},\Phi}(1^\lambda)\right] - 1$$

is negligible as a function of λ. An adversary is legitimate if it queries the RKFN and RKFN^{-1} oracles with functions ϕ in Φ_λ only.[4] We say PRP is Φ-RKCPA secure if the above advantage is negligible for any legitimate PPT adversary $\mathcal{A}$ that never queries its RKFN^{-1} oracle.

$\underline{\mathsf{RKCCA}_{\mathsf{PRP},\mathcal{A},\Phi}(1^\lambda)}$:	$\underline{\text{RKFN}(\phi, x)}$:	$\underline{\text{RKFN}^{-1}(\phi, x)}$:
$b \leftarrow\$ \{0,1\}$	$K' \leftarrow \phi(K_1, \ldots, K_n)$	$K' \leftarrow \phi(K_1, \ldots, K_n)$
$\pi \leftarrow\$ \text{Perm}(\mathsf{KSp}_\lambda, \mathsf{Dom}_\lambda)$	If $b = 0$ Return $\pi(K', x)$	If $b = 0$ Return $\pi^{-1}(K', x)$
$K_1, \ldots, K_n \leftarrow\$ \mathsf{KSp}(1^\lambda)$	Return $\mathsf{PRP}(K'x)$	Return $\mathsf{PRP}^{-1}(K', x)$
$b' \leftarrow\$ \mathcal{A}^{\text{RKFN},\text{RKFN}^{-1}}(1^\lambda)$		
Return $(b' = b)$		

Fig. 2: Game defining the Φ-RKCCA security of a PRP.

In Appendix A we prove that under the natural (but strong) restriction that any $\phi \in \Phi_\lambda$ is of the form $\phi : (K_1, \ldots, K_n) \mapsto \psi(K_i)$, where $i \in [n]$ and $\psi : \mathsf{KSp}_\lambda \longrightarrow \mathsf{KSp}_\lambda$ is a unary RKD function, the single-key and multi-key RKA models are equivalent.

Remark. The multi-key RKA model for PRFs (under chosen-plaintext attacks) is recovered when π is sampled from $\text{Func}(\mathsf{KSp}_\lambda, \mathsf{Dom}_\lambda, \mathsf{Rng}_\lambda)$ and oracle RKFN^{-1} is no longer present. When $n = 1$, we recover the single-key RKA model for PRPs and PRFs as in [5]. The standard model for PRPs/PRFs is one where the RKD sets Φ_λ contain the identity functions $id_\lambda : \mathsf{KSp}_\lambda \longrightarrow \mathsf{KSp}_\lambda;\ K \mapsto K$ only. The above definition is not the strongest multi-key security model that one can envision. (For instance consider a model where the adversary can choose the arity n.) However, since the applications that we consider in this paper have a fixed number of keys, the simpler definition above is sufficient for our purposes.

4 The Random-Oracle Transform

One way to transform a standard pseudorandom permutation to one which resists related-key attacks is to hash the PRP key before using it in the construction [30]. We call this the "Hash-then-PRP" transform. Bellare and Cash [3, Theorem 6.1] prove the soundness of this approach in the *standard* model for a restricted class of RKD functions, when the hash function is replaced by an RKA-secure pseudorandom generator. At first sight it appears that the ideal hash function (i.e., the random oracle) should be a valid instantiation of this construction. However, in the random-oracle model (ROM) the security proof should be carried out in a setting where *all* parties have access to the random oracle (which models the hash function). In this section we consider the implications of this observation, and show that the random oracle does *not* always give rise to a good instantiation of the construction. We also provide a set of sufficient conditions that allows us to formally prove that the heuristic transform is sound in the ROM.

RKA-secure PRG in ROM.[5] We define an *oracle* RKD function to be a circuit which contains special oracle gates, and we write an n-ary oracle RKD function as $\phi^H : \mathsf{KSp}^n \to \mathsf{KSp}$. Families of oracle RKD sets are defined in the obvious way.

[4] Throughout the paper, we assume all the adversaries are, in this sense, legitimate.

[5] We remark that this game can also be seen as extension of correlated-input secure hashing [21] to the random-oracle model.

$\mathsf{RKA}_{\mathsf{PRG},\mathcal{A},\Phi}(1^\lambda)$:	$\mathrm{RKF{\scriptsize N}}(\phi)$:
$\rho \leftarrow\$ \mathrm{Func}(\mathsf{Dom}, \mathsf{Rng})$	$K' \leftarrow \phi^H(K_1, \ldots, K_n)$
$H \leftarrow\$ \mathrm{Func}(\mathsf{Dom}', \mathsf{Rng}')$	If $b = 0$ Return $\rho(K')$
$K_1, \ldots, K_n \leftarrow\$ \mathsf{Dom}(1^\lambda)$	Return $\mathsf{PRG}^H(K')$
$b \leftarrow\$ \{0,1\}$	
$b' \leftarrow\$ \mathcal{A}^{\mathrm{RKF{\scriptsize N}},\mathrm{RO}}(1^\lambda)$	$\mathrm{RO}(X)$:
Return $(b' = b)$	Return $H(X)$

Fig. 3: Game defining the Φ-RKA security of a PRG. An adversary is legitimate if it queries RKFN with a $\phi \in \Phi_\lambda$ only.

Let $\mathsf{PRG}^H : \mathsf{Dom} \longrightarrow \mathsf{Rng}$ be a pseudorandom generator in the ROM. Let game $\mathsf{RKA}_{\mathsf{PRG},\mathcal{A},\Phi}$ be as shown in Figure 3. We say that PRG is Φ-RKA secure if the advantage of any PPT adversary $\mathcal{A}$ as defined below is negligible in λ.

$$\mathbf{Adv}^{\mathsf{rka}}_{\mathsf{PRG},\Phi,\mathcal{A}}(\lambda) := 2 \cdot \Pr\left[\mathsf{RKA}_{\mathsf{PRF},\mathcal{A},\Phi}(1^\lambda)\right] - 1 \ .$$

The question that we wish to answer is under which conditions does the random oracle itself (i.e., when $\mathsf{PRG}^H(X) := H(X)$) constitute an RKA-secure PRG. The attack we now show and the ensuing discussion demonstrate that this is only the case if we exclude certain forms of *oracle-dependent* related-key attacks.

The attack. Consider a unary RKD set containing the identity function and an oracle-dependent RKD function [1] ϕ^H given by $\Phi := \{id : K \mapsto K,\ \phi^H : K \mapsto H(K)\}$. Here, H denotes the random oracle. Now consider an adversary that first requests a PRG value of the seed by querying id to the RKFN oracle. It receives as response a value y which is either $H(K)$, when $b = 1$, or $\rho(K)$ when $b = 0$, where ρ is an independent random oracle. The adversary now queries y to RO to get a new value z which is either $H(H(K))$ or $H(\rho(K))$. Finally, the adversary queries ϕ^H to RKFN to get a value z' which is either $H(H(K))$ or $\rho(H(K))$. Now, when $b = 1$, then $z = z'$ with probability 1. When $b = 0$ the values z and z' would only match if $H(\rho(K)) = \rho(H(K))$. The probability of this event is negligible, and thus the adversary wins with overwhelming probability by returning $(z = z')$.

We now define a sufficient set of restrictions on oracle RKD sets that allow us to prove a ROM analogue of the result by Bellare and Cash [3]. Intuitively the restrictions are strong enough to rule out attacks that follow the above pattern.

Output unpredictability. A family of oracle RKD sets Φ is output unpredictable (UP) if the following definition of advantage is negligible in λ for any PPT adversary $\mathcal{A}$ outputting a list of RKD functions and a list of keys.

$$\mathbf{Adv}^{\mathsf{up}}_{\mathcal{A},\Phi}(\lambda) := \Pr\left[\exists\ (\phi^H, \mathbf{K}^*) \in \mathsf{L}_1 \times \mathsf{L}_2 \ \text{ s.t. } \ \phi^H(\mathbf{K}) = \mathbf{K}^* : \right.$$
$$\left. H \leftarrow\$ \mathsf{Func}(\mathsf{KSp}_\lambda, \mathsf{KSp}_\lambda); \mathbf{K} \leftarrow\$ \mathsf{KSp}^n_\lambda; (\mathsf{L}_1, \mathsf{L}_2) \leftarrow\$ \mathcal{A}^H(1^\lambda)\right]$$

Claw-freeness. A family of oracle RKD sets Φ is claw-free (CF) if the following definition of advantage is negligible in λ for any PPT adversary $\mathcal{A}$ outputting a list of RKD functions.

$$\mathbf{Adv}^{\mathsf{cf}}_{\mathcal{A},\Phi}(\lambda) := \Pr\left[\exists\ \phi^H_1, \phi^H_2 \in \mathsf{L} \ \text{ s.t. } \ \phi^H_1(\mathbf{K}) = \phi^H_2(\mathbf{K}) \wedge \phi^H_1 \neq \phi^H_2 : \right.$$
$$\left. H \leftarrow\$ \mathsf{Func}(\mathsf{KSp}_\lambda, \mathsf{KSp}_\lambda); \mathbf{K} \leftarrow\$ \mathsf{KSp}^n_\lambda; \mathsf{L} \leftarrow\$ \mathcal{A}^H(1^\lambda)\right]$$

Query independence. A family of oracle RKD sets Φ is query independent (QI) if the following definition of advantage is negligible in λ for any PPT adversary $\mathcal{A}$ outputting a list of RKD functions.

$$\mathbf{Adv}^{\mathsf{qi}}_{\mathcal{A},\Phi}(\lambda) := \Pr\left[\exists\ \phi^H_1, \phi^H_2 \in \mathsf{L} \ \text{ s.t. } \ \phi^H_1(\mathbf{K}) \in \mathsf{Qry}[\phi^H_2(\mathbf{K})] : \right.$$
$$\left. H \leftarrow\$ \mathsf{Func}(\mathsf{KSp}_\lambda, \mathsf{KSp}_\lambda); \mathbf{K} \leftarrow\$ \mathsf{KSp}^n_\lambda; \mathsf{L} \leftarrow\$ \mathcal{A}^H(1^\lambda)\right]$$

Here, $\mathsf{Qry}[\phi_2^H(\mathbf{K})]$ denotes the set of queries placed to H by ϕ_2^H when run on a vector of keys $\mathbf{K}$. Note that RKD functions ϕ_1^H and ϕ_2^H need not be distinct.

We recover the standard (non-oracle) definition of output unpredictability and claw-freeness [5], when the RKD functions do not make any oracle queries: the random oracle can be simulated using lazy sampling. Query independence is trivially satisfied for such non-oracle RKD functions.

We now prove that the random oracle is an RKA-secure pseudorandom generator under the above restrictions on the oracle RKD set, and then build on this result to establish security of the Hash-then-PRP transform in the random-oracle model. Looking ahead, this result allows us to take a Luby–Rackoff PRP and generically transform it to obtain an RKA-secure PRP. In subsequent sections we will explore less intrusive, more efficient alternatives that take advantage of the inner structure of the Feistel construction.

Theorem 1 (RKA security of the random oracle). *Let Φ be a family of oracle RKD sets. Then for any Φ-RKCCA adversary $\mathcal{A}$ against the pseudorandom generator $\mathsf{PRG}^H(K) := H(K)$, there are adversaries $\mathcal{A}_1$, $\mathcal{A}_2$, and $\mathcal{A}_3$ such that*

$$\mathbf{Adv}^{\mathsf{rkcpa}}_{\mathsf{PRG},\mathcal{A},\Phi}(\lambda) \leq \mathbf{Adv}^{\mathsf{up}}_{\mathcal{A}_1,\Phi}(\lambda) + 2 \cdot \mathbf{Adv}^{\mathsf{cf}}_{\mathcal{A}_2,\Phi}(\lambda) + \mathbf{Adv}^{\mathsf{qi}}_{\mathcal{A}_3,\Phi}(\lambda) \ .$$

Proof (Sketch). We give only the intuition; the details of the proof can be found in Appendix B. Assume, without loss of generality, that the adversary never places repeat queries to its RKFn and RO oracles. Let Game_0 denote the RKA game where H is used in the RKFn oracle (i.e., the challenge bit is 1).

We modify Game_0 to Game_1 by implementing the H oracle in the RKFn oracle in a forgetful way (i.e., we won't keep track of repetitions), but leaving it unchanged for the explicit queries made through RO and the indirect queries placed by the oracle RKD functions. Note that in this game the adversary receives independently and uniformly distributed strings from either of its oracles.

Games Game_0 and Game_1 are identical unless one of the following events takes place: 1) A repeat H query is placed as a result of an explicit RO query and an output of an oracle RKD function queried to RKFn: this leads to a violation of the output unpredictability. 2) There is a repeat query to H as a result of two distinct RKFn queries: this leads to a claw-freeness break. 3) There is a repeat H query as a result of a query to RKFn and an indirect query placed by an oracle RKD function to H: this breaks the query-independence property.

We now modify Game_1 to Game_2 by changing the forgetful oracle and implementing it using an independently chosen (non-forgetful) random oracle. The games are identical unless there is a claw among the RKD functions queried to RKFn, which by the above analysis happens with negligible probability. Finally note that Game_2 is identical to the RKA game conditioned on $b = 0$.[6] □

In Appendix C we state and prove a random-oracle model analogue of the RKA-secure PRG transform of Bellare, Cash, and Miller [4], which in combination with Theorem 1 establishes security of the Hash-then-PRP transform (in the random-oracle model).

5 The Feistel Construction

In this section we recall the formal definitions related to the Feistel constructions and introduce our notion for key assignments. We also establish a general result that permits shifting the analysis of

[6] This transition my be avoided by observing that Game_0 and Game_2 are also identical until the same bad events which separate Game_0 and Game_1 take place.

Feistel networks with any number of rounds where the round functions are instantiated with an RKA-secure PRF to a more convenient setting where the round functions are instantiated with the ideal keyed function.

Feistel networks. The one-round Feistel construction and its inverse with respect to a function f is defined as

$$\mathbf{F}[f](L,R) := (R, L \oplus f(R)) \quad \text{and} \quad \mathbf{F^{-1}}[f](L,R) := (R \oplus f(L), L) \ .$$

The n-round Feistel construction with respect to functions $f_1, \ldots, f_n$ is defined recursively via the following equations (see Figure 1 for a pictorial representation).

$$\mathbf{F}[f_1, \ldots, f_n](L,R) := \mathbf{F}[f_2, \ldots, f_n](\mathbf{F}[f_1](L,R)) \ ,$$
$$\mathbf{F^{-1}}[f_1, \ldots, f_n](L,R) := \mathbf{F^{-1}}[f_1, \ldots, f_{n-1}](\mathbf{F^{-1}}[f_n](L,R))$$

Typically, functions $f_1, \ldots, f_n$ are implemented using a PRF under independently generated keys $K_1, \ldots, K_n$. In our analysis we will also consider the conceptual setting in which these functions are instantiated by the ideal keyed function ρ, again under independently generated keys $K_1, \ldots, K_n$. We denote these constructions by $\mathbf{F}^{\mathsf{PRF}}[K_1, \ldots, K_n]$ and $\mathbf{F}^{\rho}[K_1, \ldots, K_n]$, respectively.

Key assignment. A key assignment is a family of circuits $\kappa_\lambda : \overline{\mathsf{KSp}}_\lambda \longrightarrow \mathsf{KSp}_\lambda^n$, where $\overline{\mathsf{KSp}}$ is an arbitrary key space. Given $\kappa := \{\kappa_\lambda\}$ and $K \in \overline{\mathsf{KSp}}_\lambda$, we consider the associated n-round Feistel construction $\mathbf{F}^{\mathsf{PRF}}[\kappa(K)]$. When the key $K \in \overline{\mathsf{KSp}}_\lambda$ is randomly generated, we denote the construct by $\mathbf{F}^{\mathsf{PRF}}[\kappa]$. For example, the Hash-then-PRP transform of the previous section can be viewed as $\mathbf{F}^{\mathsf{PRF}}[H]$. We are, however, interested in simple key assignments of the form $\kappa : (K_1, \ldots, K_m) \mapsto (K_{i_1}, \ldots, K_{i_n})$, where $i_1, \ldots, i_n$ are fixed indices in $[m]$. We will therefore compactly write the Feistel construction associated to the simple key assignment above by $\mathbf{F}^{\mathsf{PRF}}[i_1, \ldots, i_n]$. For example, when $\kappa(K_1, K_2) := (K_1, K_2, K_2)$, the associated Feistel construction is written as $\mathbf{F}^{\mathsf{PRF}}[1, 2, 2]$.

When the round functions in a 3-round Feistel construction are instantiated with a PRF under independent keys, we obtain the classic CPA-secure Luby–Rackoff pseudorandom permutation. When 4 rounds are used, we obtain its CCA-secure counterpart. As stated in the introduction, Bellare and Kohno [5] observed that if an adversary can arbitrarily tamper with the key used in the last round of any Feistel network, then a successful related-key attack is possible (even if the underlying PRF is RKA secure).

As discussed in the previous section, using Theorems 5 and 1 we can obtain a PRP which resists related-key attacks by applying the transform to the Luby–Rackoff construction. The underlying PRG can be instantiated in the standard model via an RKA-secure PRF (e.g., that used in the Luby–Rackoff construction) as suggested in [3] or, stepping outside the standard model, using random oracles.

Both transformations, however, come with two major drawbacks. The first drawback is the performance penalty. The standard-model approach incurs a total of six PRF computations in the 3-round network: 3 calls to generate the keys and another 3 to compute the PRP.[7] (The total number of calls is eight for the CCA case.) Note that the amortized complexity of the construction cannot be brought down back to 3 by storing the generated keys, as related-key attacks can be applied to these keys. In the ROM transform (on top of strong assumptions) the penalty will be smaller if the hash function is more efficient than the PRF. However, this leads to a second drawback: the transform is software/hardware intrusive, as extra circuitry for the implementation key-derivation procedure need to be added.

For these reasons, in the remainder of the paper, we will consider more efficient alternatives to obtaining RKA-secure PRPs by exploring directly the structure of Feistel constructions via simple key

[7] The overall tightness of security obtained via [3, Theorem 6.1] is also worse than what we obtain here, although it is possible that it can be improved via a direct analysis.

assignments. Before doing so, we prove a general theorem that allows us to move from the security analysis of a Feistel construction with respect to an RKA-secure PRF to a setting in which the round functions are instantiated by the ideal keyed function. Our result holds for any number of rounds and any key assignment.

Theorem 2 (Computational RKA transition). *Let Φ be a family of RKD sets containing functions of the form $\mathsf{KSp}^m \longrightarrow \mathsf{KSp}^m$ and let $\kappa : \mathsf{KSp}^m \longrightarrow \mathsf{KSp}^n$ be a key assignment. Define $\Psi := \cup_i(\kappa \circ \Phi)_i$, where $(\kappa \circ \Phi)_i$ is the RKD set obtained by composing function in Φ by κ on the right and then projecting to i-th component for $1 \leq i \leq n$. Let ρ denote the ideal keyed function, and let* PRF *denote be a pseudorandom function. Then for any* PPT *adversary $\mathcal{A}$ against the* Φ-RKCCA *security of $\mathbf{F}^{\mathsf{PRF}}[\kappa]$, there is an adversary $\mathcal{B}$ against the* Ψ-RKCPA *security of* PRF *such that*

$$\mathbf{Adv}^{\mathsf{rkcca}}_{\mathbf{F}^{\mathsf{PRF}}[\kappa],\mathcal{A},\Phi}(\lambda) \leq \mathbf{Adv}^{\mathsf{rkcca}}_{\mathbf{F}^{\rho}[\kappa],\mathcal{A},\Phi}(\lambda) + \mathbf{Adv}^{\mathsf{rkcpa}}_{\mathsf{PRF},\mathcal{B},\Psi}(\lambda) \ .$$

An analogous result holds for Φ-RKCPA *adversaries.*

Proof (Sketch). We start with the Φ-RKCCA game for $\mathbf{F}^{\mathsf{PRF}}[\kappa]$ and replace all n rounds function in the Feistel construction with the ideal keyed function. Any change in an adversary $\mathcal{A}$'s advantage in the two games can be used to break the (multi-key) Ψ-RKCCA security of PRF via an adversary $\mathcal{B}$. Algorithm $\mathcal{B}$ runs $\mathcal{A}$ and answers its forward queries to the Feistel construction as follows. On input (ϕ, x) where $\phi \in \Phi$, algorithm $\mathcal{B}$ sets $\psi_1 := (\kappa \circ \phi)|_1$ and calls the RKFn oracle on (ψ_1, x) to get x_1. It then sets $\psi_2 := (\kappa \circ \phi)|_2$, queries RKFn on (ψ_2, x_1) to get x_2. Algorithm $\mathcal{B}$ continues in this way for all n rounds and returns the final output. Backward queries can be also handled similarly using RKFn in the reverse direction. Clearly, according to the challenge bit b used in the Ψ-RKCPA game, $\mathcal{B}$ simulates the Φ-RKCCA game with the same challenge bit b for algorithm $\mathcal{A}$. □

6 CPA Security: The 3-Round Constructions

As we discussed in the introduction, no palindromic assignment of keys in a three-round Feistel construction can result in a CPA-secure PRP, since the construction in the forward direction can be used to compute inverses, and a trivial distinguishing attack emerges. Moreover, if the key used in the third round is independent of those used in first and second rounds, then the BK attack applies. Under these restriction, for simple key assignments and up to relabeling of the indices, we are left with only one 3-round construction which can potentially achieve CPA security under related-key attacks: $\mathbf{F}^{\mathsf{PRF}}[1, 2, 2]$.

The main proof of this section is an information-theoretic argument showing that $\mathbf{F}^{\rho}[1, 2, 2]$ is Φ-RKCPA secure for Φ's which are claw-free and switch-free. Combined with Theorem 2 in the previous section, this implies that $\mathbf{F}^{\mathsf{PRF}}[1, 2, 2]$ offers *intrinsic* RKA resilience, in the sense that it permits leveraging the RKA-security properties of its underlying PRF.

For the security proof in this and the next sections we need to rely on an additional restriction on RKD sets.

Switch-freeness. A family of RKD sets Φ with arity $n > 2$ is called switch-free (SF) if the advantage of any PPT adversary $\mathcal{A}$ as defined below is negligible as a function of λ.

$$\mathbf{Adv}^{\mathsf{sf}}_{\mathcal{A},\Phi}(\lambda) := \Pr\left[(\exists \phi_1, \phi_2 \in \mathsf{L})(\exists i \neq j \in [n]) \ \phi_1(\mathbf{K})|_i = \phi_2(\mathbf{K})|_j : \mathbf{K} \leftarrow_{\$} \mathsf{KSp}^n_\lambda; \mathsf{L} \leftarrow_{\$} \mathcal{A}(1^\lambda)\right]$$

We note that the switch-free and claw-free properties are in general incomparable. Consider, for example, the set consisting of *id* and a function which agrees with *id* on all but one point. This set

is switch-free but not claw-free. Conversely, consider the set consisting of id and the map $(K_1, K_2) \mapsto (K_2, K_1)$. This set is claw-free but not switch-free.

Theorem 3 ($\mathbf{F}^\rho[1,2,2]$ security). *Let Φ be a family of RKD sets. Then the $\mathbf{F}^\rho[1,2,2]$ construction is Φ-RKCPA secure in the ideal keyed function model if Φ is claw-fee and switch-free. More precisely, for every Φ-RKCPA adversary $\mathcal{A}$ placing at most $Q(\lambda)$ queries to* RKFn*, there exist adversaries $\mathcal{B}_1$ and $\mathcal{B}_2$ such that*

$$\mathbf{Adv}^{\mathsf{rkcpa}}_{\mathbf{F}^\rho[1,2,2],\mathcal{A},\Phi}(\lambda) \leq \mathbf{Adv}^{\mathsf{rf/rp}}_{\mathcal{A},\Phi}(\lambda) + 2 \cdot \mathbf{Adv}^{\mathsf{sf}}_{\mathcal{B}_1,\Phi}(\lambda) + 4 \cdot \mathbf{Adv}^{\mathsf{cf}}_{\mathcal{B}_2,\Phi}(\lambda) + \frac{2^5 \cdot Q(\lambda)^2}{|\mathsf{Dom}_\lambda|} \ .$$

Proof (Intuition). We give a high-level description of the proof and refer the reader to Appendix E for the full details. We assume, without loss of generality, that the adversary is non-repeating in the sense that it not place redundant repeat queries to its oracle. We start with the Φ-RKCPA game, and consider a modified game where the round functions are implemented as follows. The first round is implemented using a consistent ideal keyed function (as in the original construction). The second and third round functions, however, will be forgetful and return independent random values on each invocation irrespective of the input values. Note that the outputs of the network computed according to this game are random, and, by an appropriate strengthening of the classical PRP/PRF switching lemma (Lemma 1 in Appendix D), they are also indistinguishable from the ideal keyed permutation. Furthermore, in this game the values of the outputs of the first round function remain hidden as they are masked by random values generated in the third round.

Now the game above differs from the original CPA game due to inconsistencies occurring in computing round function values both across and within the same round, when the adversary is able to cause collisions in round function inputs in the original CPA game that are ignored in the game above. There are five such pairs of inconsistencies possible (we keep track of queries to the first round, so inconsistencies wont happen there). If there is a collision in inputs, which include the keys, to the first and second or first and third rounds, then the keys collide and this event leads to a violation of switch-freeness. Now suppose the inconsistency is due to a collision between the inputs to the third round function. Since the outputs of the second round function are randomly chosen at each invocation, this event happens with probability roughly $Q(\lambda)^2/|\mathsf{Dom}_\lambda|$ by the birthday bound. Collisions between the inputs to the second and third rounds also happen with negligible probably as the outputs of the first round remain hidden from the adversary. Finally, we are left with collisions in the inputs to the second round function. Note that this means that the keys input to this function are identical. Now if the keys or right halves of the inputs used in the first round in the two colliding queries were different, then the outputs of the first round function would be random and independent, and a collision would happen with a negligible probability (as first-round outputs are hidden). If the keys and right halves were identical, a collision can only take place if the left halves are also identical. However, due to the non-repeating condition, in this case we must have that the queried RKD functions are distinct, and consequently a claw in the RKD set is discovered. □

We emphasize that we do not claim the switch-free and claw-free restrictions are *necessary* for non-existence of attacks. On the other hand, these restrictions are akin to those adopted in previous works on RKA security, and do not overly constrain the practical applicability of our results. For example, the n-ary RKD sets for XOR-ing with constants defined by

$$\Phi^{\oplus}_{m,\lambda} := \{\phi_{C_1,\ldots,C_m} : (K_1,\ldots,K_m) \mapsto (K_1 \oplus C_1,\ldots,K_m \oplus C_m) \ : \ (C_1,\ldots,C_m) \in \mathsf{KSp}^m_\lambda\}$$

can be easily shown to satisfy these restrictions. Unpredictability follows from the fact that each map in the set induces a permutation over the keys (and hence output distribution is uniform). For claw-freeness suppose we are given two distinct RKD functions. Suppose they differ in their i-th component,

i.e., $C_i \neq C'_i$. Then, since the keys K_i and K_j are chosen independently and uniformly at random, the probability that the i-th output keys match, i.e., that $K_i \oplus C_i = K_j \oplus C'_i$, is negligible. Switch-freeness follows from a similar argument. Note finally that the restrictions needed in Theorems 1 and 2 are easily shown to be satisfied by the above set, as the key assignment is simple. We obtain the following corollary.

Corollary 1. $\mathbf{F}^{\mathsf{PRF}}[1,2,2]$ *is a* $\Phi_2^{\oplus}$-RKCPA-*secure PRP, if* PRF *is a* $\Phi_1^{\oplus}$-RKCPA-*secure PRF.*

In Appendix F we characterize the RKA security of the original three-round Luby–Rackoff construction, where three independent round keys are used.

7 CCA Security: The 4-Round Constructions

It is well known that the $\mathbf{F}^{\rho}[1,2,3]$ construction is CCA insecure. For example, the attacker can proceed as follows: 1) Choose arbitrary L, R, L', query $\mathrm{RKFN}(L,R)$ to obtain C_1 and query $\mathrm{RKFN}(L',R)$ to obtain C_2; 2) Query $\mathrm{RKFN}^{-1}(C_2 \oplus (0, L \oplus L'))$ to obtain C_3; 3) Check if $(C_1 \oplus C_2 \oplus \mathsf{Swap}(C_3))$ is same as R. The same attack applies to all Feistel networks with three rounds, independently of the key assignment, and so there is no hope that such constructions can achieve any form of CCA security.

In this section we investigate the CCA security of 4-round constructions under related-key attacks. Due to the generic related-key attacks that we listed in the previous section (insecurity of palindromic key assignment and tampering with the last key), and the fact the in the CCA model the construction can be accessed in both the forward and backward directions, the only candidates than can potentially satisfy RKCCA security are: $\mathbf{F}^{\rho}[1,1,2,1]$, its inverse $\mathbf{F}^{\rho}[1,2,1,1]$, $\mathbf{F}^{\rho}[1,1,2,2]$, $\mathbf{F}^{\rho}[1,2,1,2]$, and $\mathbf{F}^{\rho}[1,2,3,1]$. In this work, we look at $\mathbf{F}^{\rho}[1,2,1,2]$.

The proof of RKCCA security for the $\mathbf{F}[1,2,1,2]$ construction, as in the RKCPA case, has two components: a computational part allowing transition from PRFs to ideal keyed functions, and an information-theoretic argument that establishes security when the construction is instantiated with the ideal keyed function. The first part of the proof follows from Theorem 2. We now prove the second part.

Theorem 4 ($\mathbf{F}[1,2,1,2]$ security). *Let* Φ *be a family of RKD sets. Suppose* Φ *is claw-fee and switch-free. Then the* $\mathbf{F}^{\rho}[1,2,1,2]$ *construction is* Φ-RKCCA *secure in the ideal keyed function model. More precisely, for every* Φ-RKCCA *adversary* $\mathcal{A}$ *placing at most* $Q(\lambda)$ *queries to* RKFN *or* RKFN^{-1}*, there are* $\mathcal{B}_1$ *and* $\mathcal{B}_2$ *such that*

$$\mathbf{Adv}^{\mathsf{rkcca}}_{\mathbf{F}^{\rho}[1,2,1,2],\mathcal{A},\Phi}(\lambda) \leq \mathbf{Adv}^{\mathsf{rf/rp}}_{\mathcal{A},\Phi}(\lambda) + 2 \cdot \mathbf{Adv}^{\mathsf{sf}}_{\mathcal{B}_1,\Phi}(\lambda) + 8 \cdot \mathbf{Adv}^{\mathsf{cf}}_{\mathcal{B}_2,\Phi}(\lambda) + \frac{2^8 \cdot Q(\lambda)^2}{|\mathsf{Dom}_{\lambda}|} \ .$$

Proof (Intuition). We give a high-level description of the proof and refer the reader to Appendix E for the full details. The proof follows the same structure as Theorem 3, but it is slightly more complex due to the possibility of collisions occurring in the inputs of the round functions when they are used in the RKFN and RKFN^{-1} oracles. We assume, without loss of generality, that the adversary is non-repeating in the sense that it does not place repeat queries to either of its oracles, does not decipher an enciphered value, and does not enciphered a deciphered value.

We start with the Φ-RKCCA game where the round functions faithfully implement the ideal keyed function. We then consider a game where all round functions are implemented in a *forgetful* way except that 1) the input round function in RKFN is consistent and also keeps track of the entries contributed from RKFN^{-1}'s output round; and 2) the input round function in RKFN^{-1} is consistent and also keeps track of the entries contributed from RKFN's output round. In this game the output values of the

construction are random and hence indistinguishable from those from the ideal keyed permutation by Lemma 1. Furthermore, the outputs of the input round functions in the RKFN and RKFN^{-1} oracles remain hidden as they are masked by the forgetful action of the remaining round functions.

As in the CPA setting, we need to keep track of collisions in the inputs to various pairs of round functions with lead to inconsistencies, as follows. 1) First forward and fourth backward rounds are consistent with previous queries due to their implementation. 2) Collisions between even and odd numbered round functions in both directions happen with negligible probability due to switch-freeness. 3) Inputs to the third and fourth forward rounds collide with negligible probability with the previous inputs of all other round functions due to the randomness of their respective inputs. A similar argument applies to the first and second backward rounds. 4) Collisions between first forward and third forward/backward rounds happen with negligible probability as the outputs of the fourth backward round are random and remain hidden from the adversary. A similar argument applies to the fourth/second rounds in the backward direction. 5) Collisions between second forward and fourth forward/backward rounds happen with negligible probability as outputs of the first forward round are random and remain hidden. A similar argument applies to the second round in the backward direction. 6) Finally, collisions between the second forward round and itself or second backward can be bounded using the fact that outputs of the first forward round are random remain hidden, combined with claw-freeness, similarly to the CPA case. A similar argument applies to the third backward round. □

As in the CPA setting, the $\Phi_4^{\oplus}$ family satisfies all the prerequisites of Theorems 1, 2, and 4, and we obtain the following corollary.

Corollary 2. $\mathbf{F}^{\mathsf{PRF}}[1,2,1,2]$ *is a* $\Phi_2^{\oplus}$-RKCCA-*secure PRP, if* PRF *is a* $\Phi_1^{\oplus}$-RKCPA-*secure PRF.*

In Appendix H we give a positive result for the RKA security of the original 4-round Luby–Rackoff construction.

8 Directions for Further Research

This works takes a first step in the construction of RKA-secure symmetric cryptosystems based on Feistel networks, and leaves open a number of directions for future research. From a conceptual point of view, the RKA-security of many-round Feistel networks (including beyond-birthday-type concrete security) are important open questions. From a practical point of view, the RKA security of alternative constructions of PRPs such as generalized Feistel networks [22] and key-alternating ciphers [12], along with their potential (dis)advantages over Feistel networks are another interesting direction for future work.

We conclude the paper with a conjecture about the RKA security of Feistel networks with respect to arbitrary numbers of rounds and key assignments, which generalizes the CCA characterization studied in [34], and generalizes our result in Section 7 to the other plausible key assignments.

Conjecture. Let $n > 3$ be an integer, $\kappa : \mathsf{KSp}^m \longrightarrow \mathsf{KSp}^n$ be a simple key assignment, and Φ be a family of RKD sets consisting of functions $\phi : \mathsf{KSp}^m \longrightarrow \mathsf{KSp}^m$. Suppose that the following requirements are satisfied.

1. $\kappa \circ \Phi$ is output unpredictable and claw-free.
2. (κ, Φ) is palindrome-fee: for any $\phi, \phi' \in \Phi$ the probability over a random $(K_1, \ldots, K_m)$ that $\kappa \circ \phi'(K_1, \ldots, K_m) = \sigma \circ \kappa \circ \phi(K_1, \ldots, K_m)$ is negligible, where $\sigma(K_1, \ldots, K_m) := (K_m, \ldots, K_1)$.
3. (κ, Φ) is first-key repeating: for any two distinct $\phi, \phi' \in \Phi$ the probability over a random $(K_1, \ldots, K_m)$ that $[\kappa \circ \phi(K_1, \ldots, K_m)]_1 \neq [\kappa \circ \phi'(K_1, \ldots, K_m)]_1$ and $[\kappa \circ \phi(K_1, \ldots, K_m)]_i = [\kappa \circ \phi'(K_1, \ldots, K_m)]_i$ for all $1 < i \leq n$ is negligible.

4. (κ, Φ) is last-key repeating: for any two distinct $\phi, \phi' \in \Phi$ the probability over a random $(K_1, \ldots, K_m)$ that $[\kappa \circ \phi(K_1, \ldots, K_m)]_n \neq [\kappa \circ \phi'(K_1, \ldots, K_m)]_n$ and $[\kappa \circ \phi(K_1, \ldots, K_m)]_i = [\kappa \circ \phi'(K_1, \ldots, K_m)]_i$ for all $1 \leq i < n$ is negligible.

Then the $\mathbf{F}^{\rho}[\kappa]$ construction is Φ-RKCCA secure in the ideal keyed function model and hence, combined with Theorem 2, the $\mathbf{F}^{\mathsf{PRF}}[\kappa]$ construction is Φ-RKCCA secure for a Ψ-RKCPA-secure PRF, for Ψ as in the statement of Theorem 2.

We note that among the above restrictions claw-freeness is the only requirement which is not known to be necessary. Hence we obtain an "almost" characterization. Note, however, that the RKA security of a deterministic cryptosystems seems difficult to be established without assuming claw-freeness (nevertheless, cf. [4] for the weaker ICR notion). The conjecture strengthens and extends some of the results presented in the previous sections.

Acknowledgements

Manuel Barbosa was supported by Project Best Case, co-financed by the North Portugal Regional Operational Programme (ON.2 – O Novo Norte), under the National Strategic Reference Framework (NSRF), through the European Regional Development Fund (ERDF). Pooya Farshim was supported by grant Fi 940/4-1 of the German Research Foundation (DFG).

References

1. Martin R. Albrecht, Pooya Farshim, Kenneth G. Paterson, and Gaven J. Watson: On cipher-dependent related-key attacks in the ideal-cipher model. In Antoine Joux, editor, *FSE 2011*, vol. 6733 of *LNCS*, pp. 128–145. Springer, 2011.
2. Ross J. Anderson and Markus G. Kuhn: Low cost attacks on tamper resistant devices. In Bruce Christianson, Bruno Crispo, T. Mark A. Lomas, and Michael Roe, editors, *Security Protocols Workshop*, vol. 1361 of *LNCS*, pp. 125–136. Springer, 1997.
3. Mihir Bellare and David Cash: Pseudorandom functions and permutations provably secure against related-key attacks. In *Cryptology ePrint Archive*, Report 2010/397, 2010.
4. Mihir Bellare, David Cash, and Rachel Miller: Cryptography secure against related-key attacks and tampering. In Dong Hoon Lee and Xiaoyun Wang, editors, *ASIACRYPT 2011*, vol. 7073 of *LNCS*, pp. 486–503. Springer, 2011.
5. Mihir Bellare and Tadayoshi Kohno: A theoretical treatment of related-key attacks: RKA-PRPs, RKA-PRFs, and applications. In Eli Biham, editor, *EUROCRYPT 2003*, vol. 2656 of *LNCS*, pp. 491–506. Springer, 2003.
6. Mihir Bellare, Kenneth G. Paterson, and Susan Thomson: RKA Security beyond the linear barrier: IBE, encryption and signatures. In Xiaoyun Wang and Kazue Sako, editors, *ASIACRYPT 2012*, vol. 7658 of *LNCS*, pp. 331–348. Springer, 2012.
7. Eli Biham: How to decrypt or even substitute DES-encrypted messages in 228 steps. *Information Processing Letters*, 84(3), pp. 117–124, 2002.
8. Eli Biham: New types of cryptoanalytic attacks using related keys. *Journal of Cryptology*, 7(4):229–246, 1994.
9. Eli Biham, Orr Dunkelman, and Nathan Keller: Related-key boomerang and rectangle attacks. In Ronald Cramer, editor, *EUROCRYPT 2005*, vol. 3494 of *LNCS*, pp. 507–525. Springer, 2005
10. Alex Biryukov and D. Khovratovich: Related-key cryptanalysis of the full AES-192 and AES-256. In Mitsuru Matsui, editor, *ASIACRYPT 2009*, vol. 5912 of *LNCS*, pp. 1–18. Springer, 2009.
11. Alex Biryukov, Dmitry Khovratovich, and Ivica Nikolic: Distinguisher and related-key attack on the full AES-256. In Shai Halevi, editor, *CRYPTO 2009*, vol. 5677 of *LNCS*, pp. 231–249. Springer, 2009.
12. Andrey Bogdanov, Lars R. Knudsen, Gregor Leander, François-Xavier Standaert, John P. Steinberger, and Elmar Tischhauser: Key-alternating ciphers in a provable setting: Encryption using a small number of public permutations - (extended abstract). In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, vol. 7237 of *LNCS*, pp. 45–62. Springer, 2012.
13. Eric Brier, Christophe Clavier, and Francis Olivier: Correlation power analysis with a leakage model. In Marc Joye and Jean-Jacques Quisquater, editors *CHES 2004*, vol. 3156 of *LNCS*, pp. 16–29, Springer 2004.
14. Suresh Chari, Josyula R. Rao, and Pankaj Rohatgi: Template attacks. In Burton S. Kaliski Jr., Çetin Kaya Koç, and Christof Paar, editors, *CHES 2002*, vol. 2523 of *LNCS*, pp. 13–28. Springer, 2002

15. EMV integrated circuit card specifications for payment systems. Book 2 Security and Key Management, Version 4.2, June 2008.
16. Horst Feistel: Cryptography and computer privacy. *Sc. American*, 228 (May 1973), pp. 15–23.
17. Horst Feistel, William A. Notz, and J. Lynn Smithm: Some cryptographic techniques for machine-to-machine data communications. In *Proc. of the IEEE*, 63(11), pp. 1545–1554, 1975.
18. David Goldenberg and Moses Liskov: On related-secret pseudorandomness. In Daniele Micciancio, editor, *TCC 2010*, vol. 7785 of *LNCS*, pp. 255–272. Springer, 2010.
19. Oded Goldreich and Leonid Levin: A hard-core predicate for all one-way functions. In Jeffrey Scott Vitter, editor, *STOC*, pp. 25–32. ACM, 1989.
20. Oded Goldreich, Shafi Goldwasser, and Silvio Micali: How to construct random functions. *Journal of the ACM*, 33(4):792–807 (1986).
21. Vipul Goyal, Adam O'Neill, and Vanishree Rao: Correlated-input secure hash functions. In Yuval Ishai, editor, *TCC 2011*, vol. 6597 of *LNCS*, pp. 182–200. Springer, 2011.
22. Viet Tung Hoang and Phillip Rogaway: On generalized feistel networks. In Tal Rabin, editor, *CRYPTO 2010*, vol. 6223 of *LNCS*, pp. 613–630. Springer, 2010.
23. Thomas Holenstein, Robin Künzler, and Stefano Tessaro: The equivalence of the random oracle model and the ideal cipher model, revisited. In Lance Fortnow and Salil P. Vadhan, editors, *STOC 2011*, pp. 89–98, ACM, 2011.
24. Tetsu Iwata and Tadayoshi Kohno: New security proofs for the 3GPP confidentiality and integrity algorithms. In Bimal K. Roy and Willi Meier, editors, *FSE 2004*, vol. 3017 of *LNCS*, pp. 427–445. Springer, 2004.
25. Dmitry Khovratovich, Ivica Nikolic, and Christian Rechberger: Rotational rebound attacks on reduced skein. In Masayuki Abe, editor, *ASIACRYPT 2010*, vol. 6477 of *LNCS*, pp. 1–19. Springer, 2010.
26. Lars R. Knudsen: Cryptanalysis of LOKI91. In Jennifer Seberry and Yuliang Zheng, editors, *AUSCRYPT '92*, vol. 718 of *LNCS*, pp. 196–208. Springer, 1993.
27. Lars R. Knudsen and Tadayoshi Kohno: Analysis of RMAC. In Thomas Johansson, editor, *FSE 2003*, vol. 2887 of *LNCS*, pp. 182–191. Springer, 2003.
28. Moses Liskov, Ronald L. Rivest, and David Wagner: Tweakable Block Ciphers. In Moti Yung, editor, *ĂŤ CRYPTO 2002*, vol. 2442 of *LNCS*, pp. 31–46. Springer, 2002.
29. Michael Luby and Charles Rackoff: How to construct pseudo-random permutations from pseudo-random functions. *SIAM J. Comput.* 17(2):373–386 (1988).
30. Stefan Lucks: Ciphers secure against related-key attacks. In Bimal K. Roy and Willi Meier, editors, *FSE 2004*, vol. 3017 of *LNCS*, pp. 359–370. Springer, 2004.
31. Paul C. Kocher, Joshua Jaffe, and Benjamin Jun: Differential power analysis. In Michael J. Wiener, editor, *CRYPTO '99*, vol. 1666 of *LNCS*, pp. 388–397. Springer, 1999.
32. Ueli M. Maurer and Krzysztof Pietrzak: The security of many-round Luby–Rackoff pseudo-random permutations. In Eli Biham, editor, *EUROCRYPT 2003*, vol. 2656 of *LNCS*, pp. 544–561. Springer, 2003.
33. Ueli M. Maurer, Renato Renner, and Clemens Holenstein: Indifferentiability, impossibility results on reductions, and applications to the random oracle methodology. In Moni Naor, editor, *TCC 2004*, vol. 2951 of *LNCS*, pp. 521–39. Springer, 2004.
34. Mridul Nandi: The characterization of Luby–Rackoff and its optimum single-key variants. In Guang Gong and Kishan Chand Gupta, editors, *INDOCRYPT 2010*, vol. 6498 of *LNCS*, pp. 82–97. Springer, 2010.
35. Moni Naor and Omer Reingold: On the construction of pseudorandom permutations: Luby–Rackoff revisited. *Journal of Cryptology*, 12(1):29–66, 1999.
36. Jacques Patarin: How to construct pseudorandom permutations and super pseudorandom permutations from one single pseudorandom functions. In Rainer A. Rueppel, editor, *EUROCRYPT '92*, vol. 658 of *LNCS*, pp. 56–266. Springer, 1992.
37. Jacques Patarin: Security of random feistel schemes with 5 or more rounds. In Matthew K. Franklin, editor, *CRYPTO 2004*, vol. 3152 of *LNCS*, pp. 106–122. Springer, 2004.
38. Gilles Piret: *Block Ciphers: Security Proofs, Cryptanalysis, Design, and Fault Attacks*. Ph.D. Thesis, Université catholique de Louvain, 2005.
39. Santanu Sarkar and Subhamoy Maitra: Side channel attack to actual cryptanalysis: breaking CRT-RSA with low weight decryption exponents. In Emmanuel Prouff and Patrick Schaumont, editors, *CHES 2012*, vol. 7428 of *LNCS*, pp. 476–493. Springer, 2012.
40. Victor Shoup: Sequences of games: a tool for taming complexity in security proofs. In *Cryptology ePrint Archive*, Report 2004/332, 2004.
41. Serge Vaudenay: Feistel ciphers with L_2-decorrelation. In Stafford E. Tavares and Henk Meijer, editors, *SAC 1998*, vol. 1556 of *LNCS*, pp. 1–14. Springer, 1998.
42. Hoeteck Wee: Public key encryption against related key attacks. In Marc Fischlin, Johannes Buchmann, and Mark Manulis, editors, *PKC 2012*, vol. 7293 of *LNCS*, pp. 262–279. Springer, 2012.

A Equivalence of Multi-Key and Single-Key RKA Security

The implication from multi-key security to single-key security follows from a simple direct reduction. The reverse direction is given by the following proposition.

Proposition 1 (Multi-key to single-key reduction). *Let* $\mathsf{PRP}_\lambda : \mathsf{KSp}_\lambda \times \mathsf{Dom}_\lambda \longrightarrow \mathsf{Dom}_\lambda$ *be a pseudorandom permutation family, and let* Φ *be a family of* n*-ary RKD sets. Suppose that any* $\phi \in \Phi_\lambda$ *is of the form* $\phi : (K_1, \ldots, K_n) \mapsto \psi(K_i)$*, where* $i \in [n]$ *and* $\psi : \mathsf{KSp}_\lambda \longrightarrow \mathsf{KSp}_\lambda$ *is a unary RKD function. We denote the unary function corresponding to* ϕ *by* $(i, \psi)[\phi]$*, and assume that a description of it can be effectively computed given* ϕ*. Now let* $\Psi_{i,\lambda} := \bigcup_{\phi \in \Phi_\lambda} (i, \psi)[\phi]$*, and define* $\Psi_i := \{\Psi_{i,\lambda}\}$*. Then for any multi-key* Φ-RKCCA *adversary* $\mathcal{A}$*, there exist single-key adversaries* $\mathcal{B}_1, \ldots, \mathcal{B}_n$ *such that*

$$\mathbf{Adv}^{\mathsf{rkcca}}_{\mathsf{PRP},\mathcal{A},\Phi}(\lambda) \leq \sum_{i=1}^{n} \mathbf{Adv}^{\mathsf{rkcca}}_{\mathsf{PRP},\mathcal{B}_i,\Psi_i}(\lambda) \ .$$

Furthermore, if $\mathcal{A}$ *is a* CPA *adversary, then so are the adversaries* $\mathcal{B}_i$*. An analogous result for the* Φ-RKCPA *security of PRFs also holds.*

Proof (Sketch). The proof proceeds via a standard hybrid argument as follows. Game_0 is the multi-key RKA game where the challenge bit is set to 1 (i.e., PRP is used in the two oracles). We modify this game to Game_1 and answer all RKA queries (which map to) $(1, \psi)$ in either the forward or backward direction using the ideal keyed permutation rather than the PRP. All other queries (i.e., those which map to (i, ψ) for $i > 1$) are answered as in Game_0 using the PRP. Using the single-key Ψ-RKCCA security of the PRP it is easy to show that $\mathcal{A}$'s advantage changes negligibly from Game_0 to Game_1 by constructing Ψ-RKCCA adversary $\mathcal{B}_1$ that interpolates between the two games. We now continue in this manner with a sequence of games for functions $(2, \psi), \ldots, (n, \psi)$ until we reach a terminal game Game_n where all the queries (i, ψ) are answered using the ideal keyed permutation. This game is identical to the Φ-RKCCA game where the challenge bit is set to 0. □

B The RKA Security of the Random Oracle

We refer the reader to Section 4 for an intuitive description of the proof.

Proof. The proof proceeds along four game hops as follows. In each game we let S_i denote the event that the adversary $\mathcal{A}$ returns 1 in Game_i.

Game_0 : is the RKA security game conditioned on $b = 1$ (i.e., when the RKFn oracle uses the random oracle H) and where the random oracle H is implemented via a lazy sampling procedure T.

Game_1 : keeps track of the queries made to H as result of outputs of oracle RKD functions computed in the RKFn oracle on a separate list T', and keeps full consistency between T and T'. (Direct queries to H coming from the adversary or the RKD functions are still handled through T.) This change is purely conceptual change and the two games are identical:

$$\Pr[S_0] = \Pr[S_1] \ .$$

Game_2 : sets flag bad if procedure T is queried on a point which exists on list T', or procedure T' is queried on a point which exists on list T. No action is taken and the two games are identical.

$$\Pr[S_1] = \Pr[S_2] \ .$$

Game_1:	$\boxed{\mathsf{Game}_2}$ and Game_3:	$\boxed{\mathsf{Game}_4}$ and Game_5:
$K \leftarrow\$ \mathsf{Dom}(1^\lambda)$ $b' \leftarrow\$ \mathcal{A}^{\mathrm{RKFn},\mathrm{RO}}(1^\lambda)$ Return $(b' = 1)$	$K \leftarrow\$ \mathsf{Dom}(1^\lambda)$ $b' \leftarrow\$ \mathcal{A}^{\mathrm{RKFn},\mathrm{RO}}(1^\lambda)$ Return $(b' = 1)$	$K \leftarrow\$ \mathsf{Dom}(1^\lambda)$ $b' \leftarrow\$ \mathcal{A}^{\mathrm{RKFn},\mathrm{RO}}(1^\lambda)$ Return $(b' = 1)$
$\mathrm{RKFn}(\phi)$: $K' \leftarrow \phi^T(K)$ Return $T'(K')$	$\mathrm{RKFn}(\phi)$: $K' \leftarrow \phi^T(K)$ Return $T'(K')$	$\mathrm{RKFn}(\phi)$: $K' \leftarrow \phi^T(K)$ Return $T'(K')$
$\mathrm{RO}(X)$: Return $T(X)$	$\mathrm{RO}(X)$: Return $T(X)$	$\mathrm{RO}(X)$: Return $T(X)$
$T(X)$: If $T[X] \neq \perp$ Return $T[X]$ If $T'[X] \neq \perp$ Return $T'[X]$ $T[X] \leftarrow\$ \mathsf{Rng}$ Return $T[X]$	$T(X)$: If $T[X] \neq \perp$ Return $T[X]$ If $T'[X] \neq \perp$ Then $\mathsf{bad} \leftarrow \mathsf{true}$; $\boxed{\text{Return } T'[X]}$ $T[X] \leftarrow\$ \mathsf{Rng}$ Return $T[X]$	$T(X)$: If $T[X] \neq \perp$ Return $T[X]$ If $T'[X] \neq \perp$ Then $\mathsf{bad} \leftarrow \mathsf{true}$ $T[X] \leftarrow\$ \mathsf{Rng}$ Return $T[X]$

$T'(X)$:
If $T[X] \neq \perp$ Return $T[X]$
If $T'[X] \neq \perp$ Return $T'[X]$
$T'[X] \leftarrow\$ \mathsf{Rng}$
Return $T'[X]$
	$T'(X)$:
	If $T[X] \neq \perp$ Then
	$\mathsf{bad} \leftarrow \mathsf{true}$; $\boxed{\text{Return } T[X]}$
	If $T'[X] \neq \perp$ Return $T'[X]$
	$T'[X] \leftarrow\$ \mathsf{Rng}$
	Return $T'[X]$
		$T'(X)$:
		If $T[X] \neq \perp$ Then
		$\mathsf{bad} \leftarrow \mathsf{true}$
		If $T'[X] \neq \perp$ Then
		$\mathsf{bad} \leftarrow \mathsf{true}$; $\boxed{\text{Return } T'[X]}$
		$T'[X] \leftarrow\$ \mathsf{Rng}$
		Return $T'[X]$

Fig. 4: Sequence of games for the proof of Theorem 1.

Game_3 : ignores the code after bad, i.e., T and T' only check consistency using their respective lists. Let E be the event that bad is set. Event E can happen in one of the following ways. Event E_1: the adversary queries its RO oracle on a point which coincides with the output of an oracle RKD function. Event E_2: an RKD function queries its RO oracle on a point which coincides with the output of an oracle RKD function. These events will be bounded by the output-unpredictability and query-independence advantages in the final game. Note that Game_3 is identical to the RKA security game conditioned on $b = 0$ (i.e., when the RKFn oracle uses an independent random oracle ρ):

$$\mathbf{Adv}^{\mathsf{rka}}_{\mathsf{Dom},\mathsf{Rng},\mathcal{A},\Phi}(\lambda) = \Pr[S_1] - \Pr[S_2] \leq \Pr[E] \leq \Pr[E_1] + \Pr[E_2] \ .$$

Game_4 : sets flag bad if procedure T' is queried on a point which exists on list T'. No action is taken. Game_4 and Game_3 are identical.

Game_5 : ignores the code after the new flag bad, i.e., T' no longer checks consistency with any of the lists. Game_5 and Game_4 are identical until the adversary queries the RKFn query on two RKD functions which result in the same output. We denote this by F'. Let E'_1 and E'_2 denote the equivalent of events E_1 and E_2 in Game_5. We have:

$$\Pr[E_1] - \Pr[E'_1] \leq \Pr[F'] \ \text{ and } \ \Pr[E_2] - \Pr[E'_2] \leq \Pr[F'] \ .$$

And hence

$$\mathbf{Adv}^{\mathsf{rka}}_{\mathsf{Dom},\mathsf{Rng},\mathcal{A},\Phi}(\lambda) \leq \Pr[E'_1] + \Pr[E'_2] + 2 \cdot \Pr[F'] \ .$$

Bounding of the probabilities in the right-hand side of the above inequality in Game_5 are straightforward:

Event E_1' : Based on $\mathcal{A}$ we build adversary $\mathcal{B}_1$ against output unpredictability of Φ as follows. Algorithm $\mathcal{B}_1$ runs $\mathcal{A}$ and answers its oracle queries (according to game Game_5) by simply returning new random values. When $\mathcal{A}$ terminates $\mathcal{B}_1$ outputs the list of explicit queries to H by $\mathcal{A}$ as List_1 and the list of RKFn queries as List_2. We have

$$\Pr[E_1'] \leq \mathbf{Adv}^{\mathsf{up}}_{\Phi,\mathcal{B}_1}(\lambda) \ .$$

Event E_2' : Based on $\mathcal{A}$ we build adversary $\mathcal{B}_2$ against query independence of Φ as follows. Algorithm $\mathcal{B}_2$ runs $\mathcal{A}$ and answers its oracle queries (according to game Game_5) by returning new random values. When $\mathcal{A}$ terminates $\mathcal{B}_2$ outputs the queries made to RKFn queries as List. We have

$$\Pr[E_2'] \leq \mathbf{Adv}^{\mathsf{qi}}_{\Phi,\mathcal{B}_2}(\lambda) \ .$$

Event F' : Based on $\mathcal{A}$ we build adversary $\mathcal{B}_3$ against claw-freeness of Φ as follows. Algorithm $\mathcal{B}_3$ runs $\mathcal{A}$ and answers its oracle queries (according to game Game_5) by returning new random values. When $\mathcal{A}$ terminates $\mathcal{B}_3$ outputs the queries made to RKFn queries as List. We have

$$\Pr[F'] \leq \mathbf{Adv}^{\mathsf{cf}}_{\Phi,\mathcal{B}_3}(\lambda) \ .$$

The theorem follows form the last four inequalities. □

C The RKA-Secure PRG Transform in the ROM

Theorem 5 (ROM transform). *Let* $\mathsf{PRP} := (\mathsf{E}, \mathsf{E}^{-1})$ *be a* CCA*-secure PRP. Let* Φ *be a family of oracle RKD sets, and suppose* PRG^H *is a* Φ-RKA*-secure PRG in the random-oracle model. Suppose the range of* PRG^H *and the key space of* PRP *are identical. Define transformed scheme* $\overline{\mathsf{PRP}}^H$ *by*

$$\overline{\mathsf{E}}^H(K, M) := \mathsf{E}(\mathsf{PRG}^H(K), M) \quad \textit{and} \quad \overline{\mathsf{E}}^{-1^H}(K, M) := \mathsf{E}^{-1}(\mathsf{PRG}^H(K), M) \ .$$

Then for any Φ-RKCCA *adversary* $\mathcal{A}$ *against* $\overline{\mathsf{PRP}}^H$ *in the random-oracle model there is a* Φ-RKA *adversary* $\mathcal{B}_1$ *against* PRG^H *in the random-oracle model, an adversary* $\mathcal{B}_3$ *against the claw-freeness of* Φ*, and a* CCA *adversary* $\mathcal{B}_2$ *against* PRP *such that*

$$\mathbf{Adv}^{\mathsf{rkcca}}_{\overline{\mathsf{PRP}},\mathcal{A},\Phi}(\lambda) \leq \mathbf{Adv}^{\mathsf{rka}}_{\mathsf{PRG},\mathcal{B}_1,\Phi}(\lambda) + \mathbf{Adv}^{\mathsf{cf}}_{\mathcal{B}_2,\Phi}(\lambda) + Q(\lambda) \cdot \mathbf{Adv}^{\mathsf{cca}}_{\mathsf{PRP},\mathcal{B}_3}(\lambda) + \frac{Q(\lambda)^2}{|\mathsf{KSp}_\lambda|}$$

where $Q(\lambda)$ *denotes the number related-key oracle queries that* $\mathcal{A}$ *places to its oracles in either the forward or backward direction.*

Proof (Sketch). The proof follows that for the PRG transform of Bellare and Cash [3, Theorem 6.1]. Let Game_0 denote the Φ-RKCCA game conditioned on $b = 0$, i.e., when the PRP and PRG^H are used to answer oracle queries. In Game_1 we use a random function R instead of PRG^H to preprocess the PRP key. Any change in adversarial advantage can be upper-bounded using the RKA security of PRG. In Game_2, instead of PRP, we use a random permutation H in oracle queries. Any change of adversarial success probability in this transition, can be bounded, via a hybrid argument, using the CCA security of the PRP, provided all the keys output by the queried RKD functions are distinct, an event whose probably is overwhelming down to claw-freeness. Finally in Game_3, we use a random permutation, and do not use R. The last two games are identical unless distinct keys are mapped to the same point by R. By the birthday bound, the probability of this event is at most $Q(\lambda)^2/|\mathsf{KSp}_\lambda|$. □

D RKA Switching Lemma

In this section we provide a strengthening of the classical PRF/PRP switching lemma to the RKA setting. The switching lemma we will use in the paper establishes an upper-bound on the distinguishing capability of an RKA adversary when it is interacting with either the ideal keyed permutation or a forgetful random oracle that returns random values on each query.

Lemma 1 (RKA switching lemma). *Let Φ be a family of RKD sets. Define the advantage of a* PPT *adversary $\mathcal{A}$ in game $\mathsf{RF/RP}_{\mathcal{A},\Phi}(1^\lambda)$ shown in Figure 5 by*

$$\mathbf{Adv}^{\mathsf{rf/rp}}_{\mathcal{A},\Phi}(\lambda) := 2 \cdot \Pr\left[\mathsf{RF/RP}_{\mathcal{A},\Phi}(1^\lambda)\right] - 1 \ ,$$

where $\mathcal{A}$ is assumed to be non-repeating: it never repeats a query to either of its oracles and whenever it queries (ϕ, x) to RKFN *(resp.,* RKFN^{-1}*) and receives y, it does not subsequently query (ϕ, y) to* RKFN^{-1} *(resp.,* RKFN*). Then there exist* PPT *adversaries $\mathcal{B}_1$ and $\mathcal{B}_2$ such that*

$$\mathbf{Adv}^{\mathsf{rf/rp}}_{\mathcal{A},\Phi}(\lambda) \leq \mathbf{Adv}^{\mathsf{cf}}_{\mathcal{B}_1,\Phi}(\lambda) + \mathbf{Adv}^{\mathsf{cf}}_{\mathcal{B}_2,\Phi}(\lambda) + \frac{Q(\lambda)^2}{2|\mathsf{Dom}_\lambda|} \ ,$$

where $Q(\lambda)$ is the number of queries (in either the forward or backward direction) that $\mathcal{A}$ places to its oracles.

$\mathsf{RF/RP}_{\mathcal{A},\Phi}(1^\lambda)$:	RKFN(ϕ, x):	RKFN$^{-1}(\phi, x)$:
$b \leftarrow\$ \{0,1\}$	$Y \leftarrow\$ \mathsf{Dom}(1^\lambda)$	$Y \leftarrow\$ \mathsf{Dom}(1^\lambda)$
$\pi \leftarrow\$ \mathrm{Perm}(\mathsf{KSp}_\lambda, \mathsf{Dom}_\lambda)$	$K' \leftarrow \phi(K_1, \ldots, K_n)$	$K' \leftarrow \phi(K_1, \ldots, K_n)$
$K_1, \ldots, K_n \leftarrow\$ \mathsf{KSp}(1^\lambda)$	If $b = 0$ Return Y	If $b = 0$ Return Y
$b' \leftarrow\$ \mathcal{A}^{\mathrm{RKFN},\mathrm{RKFN}^{-1}}(1^\lambda)$	Return $\pi(K', x)$	Return $\pi^{-1}(K', x)$
Return $(b' = b)$		

Fig. 5: An adversary is legitimate if it queries RKFN with a $\phi \in \Phi_\lambda$ only.

Proof (Sketch). The main difference with the standard RF/RP switching lemma [40] is that here we are only guaranteed that the implicit inputs to the oracles are distinct (and not the actual key/point pairs computed using the queried RKD functions). To deal with this, we use a sequence of games.

We let Game_0 denote the game above conditioned on $b = 0$. We define Game_1 by setting a bad flag bad^1 when two distinct queries $(\phi_1, x_1) \neq (\phi_2, x_2)$ are made to either of the oracles such that $(K'_1, x_1) = (K'_2, x_2)$. Furthermore, if the bad^1 flag is set, the adversary's guess is taken to be a purely random bit, so that it can have no advantage in winning the game. Note that the two games are identical until bad, and the adversary receives independently generated random values from the two oracles. The probability that the bad flag is set in Game_1 can therefore be reduced to the claw-freeness property by an adversary $\mathcal{B}_1$ that succeeds in breaking it whenever this event occurs.

We now let Game_2 denote Game_1, but we introduce a new bad flag bad^2 that is activated whenever on a query (ϕ, x) the oracle is about to return a value that collides with a previous query (ϕ, x') with $x \neq x'$. Furthermore, if the bad^2 flag is set, the adversary's guess is taken to be a purely random bit, so that it can have no advantage in winning the game.

Transition from Game_1 to Game_2 is analyzed by observing that the two games are identical, unless the adversary causes a collision in the output of the oracle that might occur in Game_1, but which causes

bad^2 to be set in Game_2. Suppose the adversary splits its q queries into r different RKD functions $\phi_1, \ldots, \phi_r$, and denote by $q_1, \ldots, q_r$ the number of queries dedicated to each ϕ_i (with the obvious restriction that $q_1 + \ldots + q_r = q$). Then, the probability of this event occurring can be upper-bounded using the union bound by

$$\sum_{i=1}^{r} \frac{q_i^2}{2|\mathsf{Dom}_\lambda|} \leq \frac{q^2}{2|\mathsf{Dom}_\lambda|} \ .$$

Finally, we let Game_3 be the game above conditioned on $b = 1$. We observe that the operation of bad^2 flag in Game_2 ensures that the adversary can only have a distinguishing advantage when interacting with a ideal keyed permutation. This means that Game_3 can be re-written as a game that is identical to Game_2, until the bad^1 flag is activated.

Luckily, the probability that the bad^1 flag is set in Game_2 can be reduced to the claw-freeness property by an adversary $\mathcal{B}_2$ that succeeds in breaking it whenever this event occurs. Combining the previous results gives us an upper-bound on the adversary's distinguishing advantage between games Game_0 and Game_3, and the theorem follows. □

E Security Analysis of $\mathbf{F}^\rho[1, 2, 2]$

Proof. We prove the theorem via a sequence of five games. Without loss of generality we assume that all of the adversary's queries to the RKFN oracle are distinct. Each game's main and RKFN_0 procedures are shown in Figure 6. We also assume that all lists used in the games are initialized to empty, and all bad flags are initialized to false. In each game we modify the RKFN_1 procedure. We denote the event that Game_i returns true by S_i, and let Bad_i^j denote the event that flag bad^j is set to true in game Game_i.

Game_i	$\text{RKFN}_0(\phi, M)$:
$\pi \leftarrow\$ \text{Perm}(\mathsf{KSp}_\lambda^2, \mathsf{Dom}_\lambda^2)$ $\rho \leftarrow\$ \text{Func}(\mathsf{KSp}_\lambda, \mathsf{Dom}_\lambda, \mathsf{Dom}_\lambda)$ $K_1, K_2 \leftarrow\$ \mathsf{KSp}(1^\lambda)$ $b' \leftarrow\$ \mathcal{A}^{\text{RKFN}_b}(1^\lambda)$ Return $(b' = b)$	$(K_1', K_2') \leftarrow \phi(K_1, K_2)$ $C \leftarrow \pi((K_1', K_2'), M)$ Return C

Fig. 6: Procedures common to Game_0–Game_5 used in the proof of Theorem 3.

Game_0 is the Φ-RKCPA game. We change Game_0 to Game_1 by implementing the ideal keyed function via lazy sampling using a list List managed by a new procedure $\text{RO}(\cdot, \cdot)$. The view of the adversary in the two games is identical:

$$\Pr[S_1] = \Pr[S_0] \ .$$

From Game_1 to Game_2 we separate the list used to manage the ideal keyed function into two lists List_1 (for round 1 queries) and List_2 (for round 2 and 3 queries). Two procedures $\text{RO}_1(\cdot, \cdot)$ and $\text{RO}_2(\cdot, \cdot)$ are now used to manage the two lists independently. We also introduce a flag bad^2 that is activated whenever an inconsistency in the simulation of RO occurs in which case random values are returned to the adversary in the output of the RKFN_1 oracle. We detect such inconsistencies using a third list List_ϕ which keeps track of all keys used within the game, and ensures that RO_1 and RO_2 are always queried on distinct keys, i.e., these procedures are never called on a common key. This check is sufficient to ensure consistency of simulation. Game_2 is identical to Game_1 until Bad_2^2:

$$\Pr[S_1] - \Pr[S_2] \leq \Pr[\mathsf{Bad}_2^2] \ .$$

Game_0	Game_1	Game_2
$\underline{\mathrm{RKF}_{\mathrm{N}1}(\phi, M)}$: $(K_1', K_2') \leftarrow \phi(K_1, K_2)$ $\mathbf{K}' \leftarrow (K_1', K_2', K_2')$ $C \leftarrow \mathbf{F}^{\rho}[\mathbf{K}'](M)$ Return C	$\underline{\mathrm{RKF}_{\mathrm{N}1}(\phi, M)}$: $(K_1', K_2') \leftarrow \phi(K_1, K_2)$ $\mathbf{K}' \leftarrow (K_1', K_2', K_2')$ $C \leftarrow \mathbf{F}^{\mathrm{RO}}[\mathbf{K}'](M)$ Return C	$\underline{\mathrm{RKF}_{\mathrm{N}1}(\phi, M)}$: $(K_1', K_2') \leftarrow \phi(K_1, K_2)$ $\mathbf{K}' \leftarrow (K_1', K_2', K_2')$ If $(\star, \star, K_1') \in \mathsf{List}_\phi \vee (\star, K_2', \star) \in \mathsf{List}_\phi$ $\quad \mathsf{bad}^2 \leftarrow \mathsf{true}$ $\quad C \leftarrow\$ \mathsf{Dom}(1^\lambda) \times \mathsf{Dom}(1^\lambda)$ Else $\quad C \leftarrow \mathbf{F}^{\mathrm{RO}_1, \mathrm{RO}_2, \mathrm{RO}_2}[\mathbf{K}'](M)$ $\mathsf{List}_\phi \leftarrow (\phi, K_1', K_2') : \mathsf{List}_\phi$ Return C
	$\underline{\mathrm{RO}(K, X)}$: If $(K, X, Y) \in \mathsf{List}$ Return Y $Y \leftarrow\$ \mathsf{Dom}(1^\lambda)$ $\mathsf{List} \leftarrow (K, X, Y) : \mathsf{List}$ Return Y	$\underline{\mathrm{RO}_i(K, X)}$: $\quad$ // $i = 1, 2$ If $(K, X, Y) \in \mathsf{List}_i$ Return Y $Y \leftarrow\$ \mathsf{Dom}(1^\lambda)$ $\mathsf{List}_i \leftarrow (K, X, Y) : \mathsf{List}_i$ Return Y

Fig. 7: Game_0–Game_2 used in the proof of Theorem 3.

The analysis of the probability of setting the bad^2 flag is deferred until Game_5, where one can show an upper-bound using the switch-free property of the RKD function family.

In Game_3 we prepare the ground for a case analysis that will be carried out in Game_4 by excluding the possibility that the adversary explores claws in the RKD function family. On every query to the $\mathrm{RKF}_{\mathrm{N}1}$ oracle, we check whether the adversary found a claw in the RKD function family. If so, we set a new bad^3 flag and return random values to the adversary. If not, we proceed as in Game_2. The analysis of the probability of setting the bad^3 flag is deferred until Game_5, where one can establish an upper-bound using the claw-free property of the RKD function family. Game_3 is identical to Game_2 until Bad_3^3 occurs. Hence,

$$\Pr[S_2] - \Pr[S_3] \leq \Pr[\mathsf{Bad}_3^3] \;\text{ and }\; \Pr[\mathsf{Bad}_2^2] - \Pr[\mathsf{Bad}_3^2] \leq \Pr[\mathsf{Bad}_3^3] \;.$$

In Game_4 we modify rounds 2 and 3, i.e., procedure RO_2, so that the ideal keyed function in these rounds is now computed in a *forgetful* way. This means that fresh outputs are chosen for every query. A new bad^4 flag is associated to the event that an inconsistency with Game_3 could occur, i.e., a repeat query is placed on the round function. Game_4 is identical to Game_3 until Bad_4^4 occurs. Hence,

$$\begin{aligned} \Pr[S_3] - \Pr[S_4] &\leq \Pr[\mathsf{Bad}_4^4] \;, \\ \Pr[\mathsf{Bad}_3^2] - \Pr[\mathsf{Bad}_4^2] &\leq \Pr[\mathsf{Bad}_4^4] \;, \\ \Pr[\mathsf{Bad}_3^3] - \Pr[\mathsf{Bad}_4^3] &\leq \Pr[\mathsf{Bad}_4^4] \;. \end{aligned}$$

Combining the inequalities we have obtained so far, we get that

$$\Pr[S_0] - \Pr[S_4] \leq \Pr[\mathsf{Bad}_4^2] + 2 \cdot \Pr[\mathsf{Bad}_4^3] + 4 \cdot \Pr[\mathsf{Bad}_4^4] \;.$$

The bulk of the proof consists of an information-theoretic argument that permits upper-bounding the probability of Bad_4^4. This is proved in Lemma 5.

Finally, Game_5 is a very simple game in which all of the adversary's queries to the $\mathrm{RKF}_{\mathrm{N}1}$ oracle are answered using fresh and independent random values. The view of the adversary in this game is identical to its view in Game_4. To see this, observe that, in Game_4 rounds 2 and 3 of the Feistel network in the $\mathrm{RKF}_{\mathrm{N}1}$ oracle are outputting fresh independent random values on every new query due to the action of RO_2. As a result

$$\Pr[S_4] = \Pr[S_5], \quad \Pr[\mathsf{Bad}_4^2] = \Pr[\mathsf{Bad}_5^2] \quad \text{and} \quad \Pr[\mathsf{Bad}_4^3] = \Pr[\mathsf{Bad}_5^3] \;.$$

Game_3	Game_4	Game_5

$\underline{\mathrm{RKF}_{\mathrm{N}1}(\phi, M)}$:
$(K_1', K_2') \leftarrow \phi(K_1, K_2)$
$\mathbf{K}' \leftarrow (K_1', K_2', K_2')$
If $(\star, \star, K_1') \in \mathsf{List}_\phi \vee (\star, K_2', \star) \in \mathsf{List}_\phi$
  $\mathsf{bad}^2 \leftarrow \mathsf{true}$
  $C \leftarrow\$ \mathsf{Dom}(1^\lambda) \times \mathsf{Dom}(1^\lambda)$
Else If $(\phi', K_1', K_2') \in \mathsf{List}_\phi \wedge \phi' \neq \phi$
  $\mathsf{bad}^3 \leftarrow \mathsf{true}$
  $C \leftarrow\$ \mathsf{Dom}(1^\lambda) \times \mathsf{Dom}(1^\lambda)$
Else
  $C \leftarrow \mathbf{F}^{\mathrm{RO}_1, \mathrm{RO}_2, \mathrm{RO}_2}[\mathbf{K}'](M)$
$\mathsf{List}_\phi \leftarrow (\phi, K_1', K_2') : \mathsf{List}_\phi$
Return C
 	
$\underline{\mathrm{RO}_i(K, X)}$: // $i = 1, 2$
If $(K, X, Y) \in \mathsf{List}_i$ Return Y
$Y \leftarrow\$ \mathsf{Dom}(1^\lambda)$
$\mathsf{List}_i \leftarrow (K, X, Y) : \mathsf{List}_i$
Return Y
	$\underline{\mathrm{RKF}_{\mathrm{N}1}(\phi, M)}$:
	$(K_1', K_2') \leftarrow \phi(K_1, K_2)$
	$\mathbf{K}' \leftarrow (K_1', K_2', K_2')$
	If $(\star, \star, K_1') \in \mathsf{List}_\phi \vee (\star, K_2', \star) \in \mathsf{List}_\phi$
	  $\mathsf{bad}^2 \leftarrow \mathsf{true}$
	  $C \leftarrow\$ \mathsf{Dom}(1^\lambda) \times \mathsf{Dom}(1^\lambda)$
	Else If $(\phi', K_1', K_2') \in \mathsf{List}_\phi \wedge \phi' \neq \phi$
	  $\mathsf{bad}^3 \leftarrow \mathsf{true}$
	  $C \leftarrow\$ \mathsf{Dom}(1^\lambda) \times \mathsf{Dom}(1^\lambda)$
	Else
	  $C \leftarrow \mathbf{F}^{\mathrm{RO}_1, \mathrm{RO}_2, \mathrm{RO}_2}[\mathbf{K}'](M)$
	$\mathsf{List}_\phi \leftarrow (\phi, K_1', K_2') : \mathsf{List}_\phi$
	Return C
	
	$\underline{\mathrm{RO}_1(K, X)}$:
	If $(K, X, Y) \in \mathsf{List}_1$ Return Y
	$Y \leftarrow\$ \mathsf{Dom}(1^\lambda)$
	$\mathsf{List}_1 \leftarrow (K, X, Y) : \mathsf{List}_1$
	Return Y
	
	$\underline{\mathrm{RO}_2(K, X)}$:
	$Y \leftarrow\$ \mathsf{Dom}(1^\lambda)$
	If $(K, X, \star) \in \mathsf{List}_2$ Then $\mathsf{bad}^4 \leftarrow \mathsf{true}$
	Else $\mathsf{List}_2 \leftarrow (K, X, Y) : \mathsf{List}_2$
	Return Y
		$\underline{\mathrm{RKF}_{\mathrm{N}1}(\phi, M)}$:
		$(K_1', K_2') \leftarrow \phi(K_1, K_2)$
		$\mathbf{K}' \leftarrow (K_1', K_2', K_2')$
		If $(\star, \star, K_1') \in \mathsf{List}_\phi \vee (\star, K_2', \star) \in \mathsf{List}_\phi$
		  $\mathsf{bad}^2 \leftarrow \mathsf{true}$
		Else If $(\phi', K_1', K_2') \in \mathsf{List}_\phi \wedge \phi' \neq \phi$
		  $\mathsf{bad}^3 \leftarrow \mathsf{true}$
		$C \leftarrow\$ \mathsf{Dom}(1^\lambda) \times \mathsf{Dom}(1^\lambda)$
		$\mathsf{List}_\phi \leftarrow (\phi, K_1', K_2') : \mathsf{List}_\phi$
		Return C

Fig. 8: Game_3 and Game_4 used in the proof of Theorem 3 and Game_5.

In Lemmas 2 and 3 we upper-bound the probabilities of Bad_5^2 and Bad_5^3.

Finally, by construction, the adversary's probability of success in Game_5 is exactly that established in Lemma 1. So we have,

$$2 \cdot \Pr[S_5] - 1 \ = \mathbf{Adv}^{\mathsf{rf/rp}}_{\mathcal{A},\Phi}(\lambda) \ .$$

The theorem follows from combining the above results and using the upper-bounds that we establish in the lemmas below.

$$\mathbf{Adv}^{\mathsf{rkcpa}}_{\mathbf{F}^{\rho[1,2,2]},\mathcal{A},\Phi}(\lambda) = 2 \cdot \Pr[S_0] - 1 \leq \mathbf{Adv}^{\mathsf{rf/rp}}_{\mathcal{A},\Phi}(\lambda) + 2 \cdot \Pr[\mathsf{Bad}_5^2] + 4 \cdot \Pr[\mathsf{Bad}_5^3] + 8 \cdot \Pr[\mathsf{Bad}_4^4] \ .$$

□

Lemma 2. *For any* Game_5 *adversary* $\mathcal{A}$*, there exists an adversary* $\mathcal{B}_1$ *against the switch-free property of* Φ *such that*

$$\Pr[\mathsf{Bad}_5^2] \leq \mathbf{Adv}^{\mathsf{sf}}_{\mathcal{B}_1,\Phi}(\lambda) \ .$$

Proof. We build adversary $\mathcal{B}_1$ against the switch-free property of Φ as follows. Adversary $\mathcal{B}_1$ runs adversary $\mathcal{A}$ in the Game_4 environment, always returning random outputs on queries to $\mathrm{RKF}_{\mathrm{N}1}$. When $\mathcal{A}$ terminates, $\mathcal{B}_1$ returns the list of RKD functions queried to the $\mathrm{RKF}_{\mathrm{N}1}$ oracle by $\mathcal{A}$. If event Bad_5^2 occurred, then the list of functions will satisfy the winning condition in the switch-free game. To see this, note that the bad^2 flag is set if and only if a key K_1' occurred that collided with a previous value of K_2' *or* a key K_2' occurred that collided with a previous value of K_1'. Then, if Bad_5^2 occurred, it must be the case that in the list of ϕ's queried by $\mathcal{A}$ to the $\mathrm{RKF}_{\mathrm{N}1}$ oracle, there exist ϕ and ϕ' such that $\phi(K_1, K_2)|_1 = \phi'(K_1, K_2)|_2$. □

Lemma 3. *For any* Game_5 *adversary* $\mathcal{A}$*, there exists an adversary* $\mathcal{B}_2$ *against the claw-free property of* Φ *such that*

$$\Pr[\mathsf{Bad}_5^3] \leq \mathbf{Adv}^{\mathsf{cf}}_{\mathcal{B}_2,\Phi}(\lambda) \ .$$

Proof. We build an adversary $\mathcal{B}_2$ against the claw-free property of Φ as follows. Adversary $\mathcal{B}$ runs adversary $\mathcal{A}$ in the Game_5 environment, returning always random outputs on queries to RKFN_1. When $\mathcal{A}$ terminates, $\mathcal{B}_2$ returns the list of functions queried to the RKFN_1 oracle by $\mathcal{A}$. If event Bad_5^3 occurred, then the list of functions will satisfy the winning condition in the claw-free game. To see this, note that the bad^3 flag is set if and only if an entry was added to List_ϕ such that ϕ was different from a previous ϕ' and yet the value of (K'_1, K'_2) was the same. □

The following preliminary lemma is critical to upper-bounding the probability of Bad_4^4.

Lemma 4. *Throughout the execution of* Game_4 *the values of* Y *stored in* List_1*, i.e., the third component of the entries of* List_1*, are mutually independent and uniformly distributed. Furthermore, these values are independent from the view of the adversary, which comprises its random coins and the answers it receives from the queries to the* RKFN_1 *procedure.*

Proof. The proof is based on the alternative view of the RKFN_1 procedure shown in Figure 9. Throughout the proof of this lemma (and subsequent lemmas where we refer to this figure) we will denote the variables manipulated by this procedure when computing the answer to the adversary's i-th query by adding the query index to the superscript: (ϕ^i, M^i) is adversary's i-th query.

The first query placed by the adversary is fully determined by its random coins. The Y_1^1 value is sampled uniformly at random and added to List_1. However, R_3^1 is the only variable that depends on Y_1^1 and $\mathcal{A}$ observes. But since Y_1^1 is masked by a uniformly and independently chosen value Y_3^1 (which is never reused in the remainder of the game) in the computation of R_3^1 it remains information-theoretically hidden from the adversary.

Let us now move to the second query. Conditioning on the view of the adversary in the first query, the second query is fully determined by the random coins of the adversary and the answers provided by the previous query. Once again Y_1^2 is masked by a uniform and independently chosen value Y_3^2, and hence Y_1^2 remains hidden from the adversary's view.

This inductive argument establishes the independence claim in the lemma, and as a side result, we also obtain that the answers provided to the adversary by the RKFN_1 oracle are uniformly and independently distributed. □

Lemma 5. *For all* Game_4 *adversaries making at most* $Q(\lambda)$ *oracle queries, we have*

$$\Pr[\mathsf{Bad}_4^4] \leq \frac{5}{2} \cdot \frac{Q(\lambda)^2}{|\mathsf{Dom}_\lambda|} \ .$$

Proof. In the following description we refer to Figure 9. We will consider the probability of bad^4 being activated in query i, due to a collision with a value stored in List_2 in a previous query $j < i$. We simplify the analysis by defining three events:

F_1 : bad^4 is activated in the third round.

F_2 : bad^4 is activated in the second round due to a collision with a value of the form (K'^j_2, R_2^j) coming from the third round of the computation in query j.

F_3 : bad^4 is activated immediately after the generation of Y_2^i due to a collision with a value of the form (K'^j_2, R_1^j) coming from the second round of the computation in query j.

```
RKFN1(φ, (L, R)):
(K'_1, K'_2) ← φ(K_1, K_2)
If (⋆, ⋆, K'_1) ∈ List_φ ∨ (⋆, K'_2, ⋆) ∈ List_φ
    bad^2 ← true
    C ←$ Dom(1^λ) × Dom(1^λ)
    List_φ ← (φ, K'_1, K'_2) : List_φ
    Return C
Else If (φ', K'_1, K'_2) ∈ List_φ ∧ φ' ≠ φ
    bad^3 ← true
    C ←$ Dom(1^λ) × Dom(1^λ)
    List_φ ← (φ, K'_1, K'_2) : List_φ
    Return C

L_1 ← R                          // Round 1
If (K'_1, R, Y_1) ∈ List_1 Then R_1 ← L ⊕ Y_1
Else
    Y_1 ←$ Dom(1^λ)
    List_1 ← (K'_1, R, Y_1) : List_1
    R_1 ← L ⊕ Y_1

L_2 ← R_1                        // Round 2
Y_2 ←$ Dom(1^λ)
If (K'_2, R_1, ⋆) ∈ List_2 Then bad^4 ← true
Else List_2 ← (K'_2, R_1, Y_2) : List_2
R_2 ← L_1 ⊕ Y_2

L_3 ← R_2                        // Round 3
Y_3 ←$ Dom(1^λ)
If (K'_2, R_2, ⋆) ∈ List_2 Then bad^4 ← true
Else List_2 ← (K'_2, R_2, Y_3) : List_2
R_3 ← L_2 ⊕ Y_3

C ← (L_3, R_3); Return C
```

Fig. 9: Expanded view of the $\mathrm{RKF}_{\mathrm{N}_1}$ oracle in Game_4.

By the union bound,

$$\Pr[\mathsf{Bad}_4^4] \leq \Pr[F_1] + \Pr[F_2] + \Pr[F_3] \ .$$

We first bound the probability of F_1. Observe that at each query i the value of R_2^i is uniformly distributed and independent from all of the previous R_1^j and R_2^j values stored in List_2 for $j < i$ (because R_2^i are XOR-ed with random values Y_2^i). Furthermore, each query adds at most two entries to List_2, which means that a conflict between queries i and j can occur with probability at most $2/|\mathsf{Dom}_\lambda|$. Therefore, by the union bound, we obtain that

$$\Pr[F_1] \ \leq \ \sum_{i=1}^{Q(\lambda)} \sum_{j=1}^{i-1} \frac{2}{|\mathsf{Dom}_\lambda|} \leq \frac{Q(\lambda)^2}{|\mathsf{Dom}_\lambda|} \ .$$

Next we establish a bound on the probability of F_2. For this to happen we would need to have $R_2^j = R_1^i = L^i \oplus Y_1^i$, where R_2^j and L^i can be fixed, while Y_1^i is independently distributed as proven in Lemma 4. So, the probability of this event occurring is at most $1/|\mathsf{Dom}_\lambda|$. Thus, over all $Q(\lambda)$ queries we have

$$\Pr[F_2] \ \leq \ \sum_{i=1}^{Q(\lambda)} \sum_{j=1}^{i-1} \frac{1}{|\mathsf{Dom}_\lambda|} \leq \frac{Q(\lambda)^2}{2|\mathsf{Dom}_\lambda|} \ .$$

To analyze the probability of event F_3, we will consider several cases and use the assumption that all of the adversary's queries are distinct. We also observe that, under the rules of Game_4, due to the action of the bad^2 flag, the RKFN_1 procedure is not called when two queried functions $\phi \neq \phi'$ yield the same value (K'_1, K'_2). The case analysis is as follows.

1. Case $\phi^i \neq \phi^j$.
   We have that $(K'^i_1, K'^i_2) \neq (K'^j_1, K'^j_2)$. We consider two sub-cases.
   (a) Case $K'^i_2 \neq K'^j_2$.
   By definition, F_3 cannot happen since K'_2 is used to check for repetitions.
   (b) Case $K'^i_2 = K'^j_2$.
   We must have that $K'^i_1 \neq K'^j_1$. In this case, event F_3 will occur due to a collision with (K'^j_2, R^j_1) if and only if $L^i \oplus L^j = Y^i_1 \oplus Y^j_1$. However, we know Y^i_1 and Y^j_1 were sampled independently from each other (since $K'^i_1 \neq K'^j_1$) and are independently distributed from the adversary's view by Lemma 4. This means that the probability of this happening is upper-bounded by $1/|\mathsf{Dom}_\lambda|$.
2. Case $\phi^i = \phi^j$.
   In this it must be the case that $(L^i, R^i) \neq (L^j, R^j)$. Again we consider two sub-cases.
   (a) Case $R^i = R^j$.
   We must have $L^i \neq L^j$ and, by construction, it follows that $R^j_1 \neq R^i_1$ and therefore event F_3 cannot occur due to a collision with (K'^j_2, R^j_1).
   (b) Case $R^i \neq R^j$.
   In this case, event F_3 will occur due to a collision with (K'^j_2, R^j_1) if and only if $L^i \oplus L^j = Y^i_1 \oplus Y^j_1$. However, we know Y^i_1 and Y^j_1 were sampled independently from each other (since $R^i \neq R^j$) and are information-theoretically hidden from the adversary according to Lemma 4. This means that the probability of this happening is upper-bounded by $1/|\mathsf{Dom}_\lambda|$.

Putting together the results of our case analysis, and using the union bound over all query combinations, we get that

$$\Pr[F_3] \;\leq\; \sum_{i=1}^{Q(\lambda)} \sum_{j=1}^{i-1} \frac{2}{|\mathsf{Dom}_\lambda|} \leq \frac{Q(\lambda)^2}{|\mathsf{Dom}_\lambda|} \;.$$

The lemma follows from the combination of the above results. □

F Security Analysis of $\mathbf{F}^\rho[1,2,3]$

In this section we answer the natural question of what level of RKA security does the original $\mathbf{F}^\rho[1,2,3]$ construction achieve. Recall that the Bellare–Kohno attack [5] ruled out $\Phi_3^\oplus$-RKCPA, as it is possible to use XOR-ing to tamper with the key in the last round. We therefore define a family of RKD sets $\Phi_3^\star \subset \Phi_3^\oplus$, which we prove to be both sufficient and necessary for the RKCPA security of the $\mathbf{F}^\rho[1,2,3]$ construction. This family includes all functions of the form

$$\phi : (K_1, K_2, K_3) \mapsto (K_1 \oplus C_1, K_2 \oplus C_3, K_3 \oplus C_3)$$

such that, for any two $(C_1, C_2, C_3), (C'_1, C'_2, C'_3) \in \Phi_3^\star$, whenever $C_3 \neq C'_3$, then $(C_1, C_2) \neq (C'_1, C'_2)$. Necessity follows from the observation that any two RKD functions which fail to satisfy this property can be used to launch a BK-type attack. We next prove sufficiency.

Proposition 2. *The $\mathbf{F}^\rho[1,2,3]$ construction is a $\Phi_3^\star$-RKCPA-secure PRP. More precisely, for any* PPT *adversary $\mathcal{A}$ we have that*

$$\mathbf{Adv}^{\mathsf{rkcpa}}_{\mathbf{F}^\rho[1,2,3],\mathcal{A},\Phi_3^\star}(\lambda) \leq \mathbf{Adv}^{\mathsf{rf/rp}}_{\mathcal{A},\Phi_3^\star}(\lambda) + \frac{2^6 \cdot Q(\lambda)^2}{|\mathsf{Dom}_\lambda|} \;,$$

where $Q(\lambda)$ is the maximum number of queries that $\mathcal{A}$ to its oracle.

Proof (Sketch). The proof of this proposition uses a sequence of games similar to those in the proof of Theorem 3. The main difference lies in the fact that the restrictions imposed on the RKD set enforce exactly the same restrictions that we derived from the claw-free and switch-free conditions. As we have seen, the XOR-based RKD sets are intrinsically switch-free. For the claw-freeness property, however, we need extra restrictions to be imposed. Observe that in the $\mathbf{F}^{\rho}[1,2,2]$ case, the absence of claws guaranteed that different RKD functions imply that either the first round key or the other key is modified. For the case analysis in the proof of Theorem 3 it is critical that one can argue that either the key for the first round changed, or the key for the *second* round changed. To obtain a similar implication in the $\mathbf{F}^{\rho}[1,2,3]$ case we impose the $C_3 \neq C_3' \Rightarrow (C_1, C_2) \neq (C_1', C_2')$ restriction. The proposition then follows from the same information-theoretic arguments that conclude the proof of Theorem 3 together with an application of Lemma 1. □

G Security Analysis of $\mathbf{F}^{\rho}[1,2,1,2]$

We prove the theorem via a sequence of seven games. Without loss of generality we assume that all of the adversary's queries to the RKFN and RKFN^{-1} oracle are distinct. The adversary will also refrain from inverting a value that it obtained from an forward oracle query or computing in the forward direction a value it has already received from the inversion oracle. Each game's main and RKFN_0 and RKFN_0^{-1} procedures are shown in Figure 10. We also assume that all lists used in the games are initialized to empty, and all bad flags are initialized to false. In each game we modify the RKFN_1 and RKFN_1^{-1} procedures. We denote the event that Game_i returns true by S_i and let Bad_i^j denote the event that flag bad^j is set to true in game Game_i.

Game_i

$\pi \leftarrow\$ \text{Perm}(\mathsf{KSp}_\lambda^2, \mathsf{Dom}_\lambda^2)$
$\rho \leftarrow\$ \text{Func}(\mathsf{KSp}_\lambda, \mathsf{Dom}_\lambda, \mathsf{Dom}_\lambda)$
$K_1, K_2 \leftarrow\$ \mathsf{KSp}(1^\lambda)$
$b' \leftarrow\$ \mathcal{A}^{\text{RKFN}_b, \text{RKFN}_b^{-1}}(1^\lambda)$
Return $(b' = b)$

$\underline{\text{RKFN}_0(\phi, M)}$:
$(K_1', K_2') \leftarrow \phi(K_1, K_2)$
$C \leftarrow \pi((K_1', K_2'), M)$
Return C

$\underline{\text{RKFN}_0^{-1}(\phi, M)}$:
$(K_1', K_2') \leftarrow \phi(K_1, K_2)$
$C \leftarrow \pi^{-1}((K_1', K_2'), M)$
Return C

Fig. 10: Procedures common to Game_0–Game_7 used in the proof of Theorem 4.

We will first intuitively explain the modifications introduced in each game. Then we will present a series of lemmas that establish intermediate results that we require for our proof. The proof of the theorem will be presented last.

- Game_0 is the original security game in the ideal keyed function model. Our goal is to prove that the adversary's distinguishing probability in this game is negligible.
- Game_7 is the terminal game. All of the adversary's queries to the construction are answered using values sampled independently and uniformly at random. The adversary's probability of success in this game is exactly that established in Lemma 1, and therefore negligible.
- Game_1 introduces the conceptual change of implementing the ideal keyed function in the rounds of the Feistel construction using lazy sampling. An initially empty list is maintained by a new procedure $\text{RO}(\cdot,\cdot)$. Whenever the function should be evaluated, the list is checked for a repeat input to ensure consistent answers. If a fresh input is given, then the output sampled independently and uniformly at random and added to the list. The view of the adversary is identical in Game_0 and Game_1.
- Game_2 separates the list used to manage the ideal keyed function into two different ones: List_{13} (corresponding to rounds 1 and 3) and List_{24} (corresponding to rounds and 2 and 4). Two procedures $\text{RO}_{13}(\cdot,\cdot)$ and $\text{RO}_{24}(\cdot,\cdot)$ are used to manage the two lists independently. We also introduce a bad^2

flag that is activated whenever an inconsistency in the simulation of the ideal keyed function occurs. Furthermore, once this flag is activated, random values are returned to the adversary in the output of the construction in both forward and backward directions. The analysis of the probability of setting the bad^2 flag is deferred until Game_7, where one can show an upper-bound using the switch-free property of the RKD function family. Game_2 is identical to Game_1 until Bad_2^2 occurs.

Game_3 introduces another conceptual change. It further divides List_{13} and List_{24}. There are now four lists List_1, List_2, List_3 and List_4 and four respective procedures RO_1, RO_2, RO_3 and RO_4 for each round. However, at this point the game still keeps track of possible collisions between List_1 and List_3 and between List_2 and List_4. Therefore, the view of the adversary is identical in Game_2 and Game_3.

Game_4 modifies the round 1 oracle in forward computations to make sure that it only uses values stored in List_1. Similarly, it modifies round 4 in inverse computations to make sure that it only uses values stored in List_4. This means that we now need to maintain six different procedures to manage lists List_1 to List_4. Intuitively, this is needed to ensure that later in Game_6 the values in List_1 and List_4 are information-theoretically hidden from the adversary. A new bad^4 flag is associated to the event that an inconsistency with Game_3 could occur. Game_4 is identical to Game_3 until Bad_4^4 occurs. The analysis of the probability of bad^4 being activated is deferred until Game_6.

Game_5 prepares the ground for a case analysis that will be carried out in Game_6 by excluding the possibility that the adversary explores claws in the RKD function family. On every query to the RKFN_1 and RKFN_1^{-1} oracles, it checks whether the adversary found a claw in the RKD function family. If so, it sets a new bad^5 flag and returns a random value to the adversary. If not, it proceeds as in Game_4. The analysis of the probability of setting the bad^5 flag is deferred until Game_7, where one can easily show an upper-bound using the claw-freeness of the RKD set. Game_5 is identical to Game_4 until Bad_5^5.

Game_6 modifies rounds 2–4 in forward computations and rounds 1–3 in backward computations so that the ideal keyed function in these rounds is now computed in a *forgetful* way. This means that fresh outputs are chosen for every query. A new bad^6 flag is associated to the event that an inconsistency with Game_5 occurs, so that the two games are identical until Bad_6^6. The bulk of the proof consists of an argument that permits upper-bounding the probabilities of Bad_4^6 or Bad_5^6. The fact that the adversary obtains independent and uniformly distributed random values in all oracle answers computed in Game_7 implies that the adversary's view is identical in Game_6 and Game_7.

Game_0	Game_1	Game_2	Game_3
$\underline{\mathrm{RKF}_{\mathrm{N}_1}(\phi, M)}$: $(K_1', K_2') \leftarrow \phi(K_1, K_2)$ $\mathbf{K}' \leftarrow (K_1', K_2', K_1', K_2')$ $C \leftarrow \mathbf{F}^{\rho}[\mathbf{K}'](M)$ Return C	$\underline{\mathrm{RKF}_{\mathrm{N}_1}(\phi, M)}$: $(K_1', K_2') \leftarrow \phi(K_1, K_2)$ $\mathbf{K}' \leftarrow (K_1', K_2', K_1', K_2')$ $C \leftarrow \mathbf{F}^{\mathrm{RO}}[\mathbf{K}'](M)$ Return C	$\underline{\mathrm{RKF}_{\mathrm{N}_1}(\phi, M)}$: $(K_1', K_2') \leftarrow \phi(K_1, K_2)$ $\mathbf{K}' \leftarrow (K_1', K_2', K_1', K_2')$ If $(\star, \star, K_1') \in \mathsf{List}_\phi \vee (\star, K_2', \star) \in \mathsf{List}_\phi$ $\quad \mathsf{bad}^2 \leftarrow \mathsf{true}$ $\quad C \leftarrow\$ \mathsf{Dom}(1^\lambda) \times \mathsf{Dom}(1^\lambda)$ Else $\quad C \leftarrow \mathbf{F}^{\mathrm{RO}_{13}, \mathrm{RO}_{24}, \mathrm{RO}_{13}, \mathrm{RO}_{24}}[\mathbf{K}'](M)$ $\mathsf{List}_\phi \leftarrow (\phi, K_1', K_2') : \mathsf{List}_\phi$ Return C	$\underline{\mathrm{RKF}_{\mathrm{N}_1}(\phi, M)}$: $(K_1', K_2') \leftarrow \phi(K_1, K_2)$ $\mathbf{K}' \leftarrow (K_1', K_2', K_1', K_2')$ If $(\star, \star, K_1') \in \mathsf{List}_\phi \vee (\star, K_2', \star) \in \mathsf{List}_\phi$ $\quad \mathsf{bad}^2 \leftarrow \mathsf{true}$ $\quad C \leftarrow\$ \mathsf{Dom}(1^\lambda) \times \mathsf{Dom}(1^\lambda)$ Else $\quad C \leftarrow \mathbf{F}^{\mathrm{RO}_1, \mathrm{RO}_2, \mathrm{RO}_3, \mathrm{RO}_4}[\mathbf{K}'](M)$ $\mathsf{List}_\phi \leftarrow (\phi, K_1', K_2') : \mathsf{List}_\phi$ Return C
$\underline{\mathrm{RKF}_{\mathrm{N}_1}^{-1}(\phi, M)}$: $(K_1', K_2') \leftarrow \phi(K_1, K_2)$ $\mathbf{K}' \leftarrow (K_1', K_2', K_1', K_2')$ $C \leftarrow \mathbf{F}^{-\mathbf{1}\rho}[\mathbf{K}'](M)$ Return C	$\underline{\mathrm{RKF}_{\mathrm{N}_1}^{-1}(\phi, M)}$: $(K_1', K_2') \leftarrow \phi(K_1, K_2)$ $\mathbf{K}' \leftarrow (K_1', K_2', K_1', K_2')$ $C \leftarrow \mathbf{F}^{-\mathbf{1}\mathrm{RO}}[\mathbf{K}'](M)$ Return C	$\underline{\mathrm{RKF}_{\mathrm{N}_1}^{-1}(\phi, M)}$: $(K_1', K_2') \leftarrow \phi(K_1, K_2)$ $\mathbf{K}' \leftarrow (K_1', K_2', K_1', K_2')$ If $(\star, \star, K_1') \in \mathsf{List}_\phi \vee (\star, K_2', \star) \in \mathsf{List}_\phi$ $\quad \mathsf{bad}^2 \leftarrow \mathsf{true}$ $\quad C \leftarrow\$ \mathsf{Dom}(1^\lambda) \times \mathsf{Dom}(1^\lambda)$ Else $\quad C \leftarrow \mathbf{F}^{-\mathbf{1}\mathrm{RO}_{13}, \mathrm{RO}_{24}, \mathrm{RO}_{13}, \mathrm{RO}_{24}}[\mathbf{K}'](M)$ $\mathsf{List}_\phi \leftarrow (\phi, K_1', K_2') : \mathsf{List}_\phi$ Return C	$\underline{\mathrm{RKF}_{\mathrm{N}_1}^{-1}(\phi, M)}$: $(K_1', K_2') \leftarrow \phi(K_1, K_2)$ $\mathbf{K}' \leftarrow (K_1', K_2', K_1', K_2')$ If $(\star, \star, K_1') \in \mathsf{List}_\phi \vee (\star, K_2', \star) \in \mathsf{List}_\phi$ $\quad \mathsf{bad}^2 \leftarrow \mathsf{true}$ $\quad C \leftarrow\$ \mathsf{Dom}(1^\lambda) \times \mathsf{Dom}(1^\lambda)$ Else $\quad C \leftarrow \mathbf{F}^{-\mathbf{1}\mathrm{RO}_1, \mathrm{RO}_2, \mathrm{RO}_3, \mathrm{RO}_4}[\mathbf{K}'](M)$ $\mathsf{List}_\phi \leftarrow (\phi, K_1', K_2') : \mathsf{List}_\phi$ Return C
	$\underline{\mathrm{RO}(K, X)}$: If $(K, X, Y) \in \mathsf{List}$ Return Y $Y \leftarrow\$ \mathsf{Dom}(1^\lambda)$ $\mathsf{List} \leftarrow (K, X, Y) : \mathsf{List}$ Return Y	$\underline{\mathrm{RO}_{13}(K, X)}$: If $(K, X, Y) \in \mathsf{List}_{13}$ Return Y $Y \leftarrow\$ \mathsf{Dom}(1^\lambda)$ $\mathsf{List}_{13} \leftarrow (K, X, Y) : \mathsf{List}_{13}$ Return Y $\underline{\mathrm{RO}_{24}(K, X)}$: If $(K, X, Y) \in \mathsf{List}_{24}$ Return Y $Y \leftarrow\$ \mathsf{Dom}(1^\lambda)$ $\mathsf{List}_{24} \leftarrow (K, X, Y) : \mathsf{List}_{24}$ Return Y	$\underline{\mathrm{RO}_i(K, X)}$: $\quad /\!/\ i = 1, 3$ If $(K, X, Y) \in \mathsf{List}_1 \cup \mathsf{List}_3$ Return Y $Y \leftarrow\$ \mathsf{Dom}(1^\lambda)$ $\mathsf{List}_i \leftarrow (K, X, Y) : \mathsf{List}_i$ Return Y $\underline{\mathrm{RO}_i(K, X)}$: $\quad /\!/\ i = 2, 4$ If $(K, X, Y) \in \mathsf{List}_2 \cup \mathsf{List}_4$ Return Y $Y \leftarrow\$ \mathsf{Dom}(1^\lambda)$ $\mathsf{List}_i \leftarrow (K, X, Y) : \mathsf{List}_i$ Return Y

Fig. 11: Game_0–Game_3 used in the proof of Theorem 4.

Game_4

$\underline{\mathrm{RKF}_{\mathrm{N}_1}(\phi, M)}$:
$(K_1', K_2') \leftarrow \phi(K_1, K_2)$
$\mathbf{K}' \leftarrow (K_1', K_2', K_1', K_2')$
If $(\star, \star, K_1') \in \mathsf{List}_\phi \vee (\star, K_2', \star) \in \mathsf{List}_\phi$
  $\mathsf{bad}^2 \leftarrow \mathsf{true}$
  $C \leftarrow\$ \mathsf{Dom}(1^\lambda) \times \mathsf{Dom}(1^\lambda)$
Else
  $C \leftarrow \mathbf{F}^{\mathrm{RO}_{11}, \mathrm{RO}_2, \mathrm{RO}_3, \mathrm{RO}_4}[\mathbf{K}'](M)$
$\mathsf{List}_\phi \leftarrow (\phi, K_1', K_2') : \mathsf{List}_\phi$
Return C

$\underline{\mathrm{RKF}_{\mathrm{N}_1}^{-1}(\phi, M)}$:
$(K_1', K_2') \leftarrow \phi(K_1, K_2)$
$\mathbf{K}' \leftarrow (K_1', K_2', K_1', K_2')$
If $(\star, \star, K_1') \in \mathsf{List}_\phi \vee (\star, K_2', \star) \in \mathsf{List}_\phi$
  $\mathsf{bad}^2 \leftarrow \mathsf{true}$
  $C \leftarrow\$ \mathsf{Dom}(1^\lambda) \times \mathsf{Dom}(1^\lambda)$
Else
  $C \leftarrow \mathbf{F}^{-\mathbf{1}\,\mathrm{RO}_1, \mathrm{RO}_2, \mathrm{RO}_3, \mathrm{RO}_{44}}[\mathbf{K}'](M)$
$\mathsf{List}_\phi \leftarrow (\phi, K_1', K_2') : \mathsf{List}_\phi$
Return C

$\underline{\mathrm{RO}_{11}(K, X)}$:
If $(K, X, Y) \in \mathsf{List}_1$ Return Y
$Y \leftarrow\$ \mathsf{Dom}(1^\lambda)$
If $(K, X, \star) \in \mathsf{List}_3$ Then $\mathsf{bad}^4 \leftarrow \mathsf{true}$
Else $\mathsf{List}_1 \leftarrow (K, X, Y) : \mathsf{List}_1$
Return Y

$\underline{\mathrm{RO}_{44}(K, X)}$:
If $(K, X, Y) \in \mathsf{List}_4$ Return Y
$Y \leftarrow\$ \mathsf{Dom}(1^\lambda)$
If $(K, X, \star) \in \mathsf{List}_2$ Then $\mathsf{bad}^4 \leftarrow \mathsf{true}$
Else $\mathsf{List}_4 \leftarrow (K, X, Y) : \mathsf{List}_4$
Return Y

$\underline{\mathrm{RO}_i(K, X)}$: // $i = 1, 3$
If $(K, X, Y) \in \mathsf{List}_1 \cup \mathsf{List}_3$ Return Y
$Y \leftarrow\$ \mathsf{Dom}(1^\lambda)$
$\mathsf{List}_i \leftarrow (K, X, Y) : \mathsf{List}_i$
Return Y

$\underline{\mathrm{RO}_i(K, X)}$: // $i = 2, 4$
If $(K, X, Y) \in \mathsf{List}_2 \cup \mathsf{List}_4$ Return Y
$Y \leftarrow\$ \mathsf{Dom}(1^\lambda)$
$\mathsf{List}_i \leftarrow (K, X, Y) : \mathsf{List}_i$
Return Y

Game_5

$\underline{\mathrm{RKF}_{\mathrm{N}_1}(\phi, M)}$:
$(K_1', K_2') \leftarrow \phi(K_1, K_2)$
$\mathbf{K}' \leftarrow (K_1', K_2', K_1', K_2')$
If $(\star, \star, K_1') \in \mathsf{List}_\phi \vee (\star, K_2', \star) \in \mathsf{List}_\phi$
  $\mathsf{bad}^2 \leftarrow \mathsf{true}$
  $C \leftarrow\$ \mathsf{Dom}(1^\lambda) \times \mathsf{Dom}(1^\lambda)$
Else If $(\phi', K_1', K_2') \in \mathsf{List}_\phi \wedge \phi' \neq \phi$
  $\mathsf{bad}^5 \leftarrow \mathsf{true}$
  $C \leftarrow\$ \mathsf{Dom}(1^\lambda) \times \mathsf{Dom}(1^\lambda)$
Else
  $C \leftarrow \mathbf{F}^{\mathrm{RO}_{11}, \mathrm{RO}_2, \mathrm{RO}_3, \mathrm{RO}_4}[\mathbf{K}'](M)$
$\mathsf{List}_\phi \leftarrow (\phi, K_1', K_2') : \mathsf{List}_\phi$
Return C

$\underline{\mathrm{RKF}_{\mathrm{N}_1}^{-1}(\phi, M)}$:
$(K_1', K_2') \leftarrow \phi(K_1, K_2)$
$\mathbf{K}' \leftarrow (K_1', K_2', K_1', K_2')$
If $(\star, \star, K_1') \in \mathsf{List}_\phi \vee (\star, K_2', \star) \in \mathsf{List}_\phi$
  $\mathsf{bad}^2 \leftarrow \mathsf{true}$
  $C \leftarrow\$ \mathsf{Dom}(1^\lambda) \times \mathsf{Dom}(1^\lambda)$
Else If $(\phi', K_1', K_2') \in \mathsf{List}_\phi \wedge \phi' \neq \phi$
  $\mathsf{bad}^5 \leftarrow \mathsf{true}$
  $C \leftarrow\$ \mathsf{Dom}(1^\lambda) \times \mathsf{Dom}(1^\lambda)$
Else
  $C \leftarrow \mathbf{F}^{-\mathbf{1}\,\mathrm{RO}_1, \mathrm{RO}_2, \mathrm{RO}_3, \mathrm{RO}_{44}}[\mathbf{K}'](M)$
$\mathsf{List}_\phi \leftarrow (\phi, K_1', K_2') : \mathsf{List}_\phi$
Return C

$\underline{\mathrm{RO}_{11}(K, X)}$:
If $(K, X, Y) \in \mathsf{List}_1$ Return Y
$Y \leftarrow\$ \mathsf{Dom}(1^\lambda)$
If $(K, X, \star) \in \mathsf{List}_3$ Then $\mathsf{bad}^4 \leftarrow \mathsf{true}$
Else $\mathsf{List}_1 \leftarrow (K, X, Y) : \mathsf{List}_1$
Return Y

$\underline{\mathrm{RO}_{44}(K, X)}$:
If $(K, X, Y) \in \mathsf{List}_4$ Return Y
$Y \leftarrow\$ \mathsf{Dom}(1^\lambda)$
If $(K, X, \star) \in \mathsf{List}_2$ Then $\mathsf{bad}^4 \leftarrow \mathsf{true}$
Else $\mathsf{List}_4 \leftarrow (K, X, Y) : \mathsf{List}_4$
Return Y

$\underline{\mathrm{RO}_i(K, X)}$: // $i = 1, 3$
If $(K, X, Y) \in \mathsf{List}_1 \cup \mathsf{List}_3$ Return Y
$Y \leftarrow\$ \mathsf{Dom}(1^\lambda)$
$\mathsf{List}_i \leftarrow (K, X, Y) : \mathsf{List}_i$
Return Y

$\underline{\mathrm{RO}_i(K, X)}$: // $i = 2, 4$
If $(K, X, Y) \in \mathsf{List}_2 \cup \mathsf{List}_4$ Return Y
$Y \leftarrow\$ \mathsf{Dom}(1^\lambda)$
$\mathsf{List}_i \leftarrow (K, X, Y) : \mathsf{List}_i$
Return Y

Fig. 12: Game_4 and Game_5 used in the proof of Theorem 4.

Game_6

$\underline{\mathrm{RKF}_{\mathrm{N}_1}(\phi, M)}$:
$(K_1', K_2') \leftarrow \phi(K_1, K_2)$
$\mathbf{K}' \leftarrow (K_1', K_2', K_1', K_2')$
If $(\star, \star, K_1') \in \mathsf{List}_\phi \vee (\star, K_2', \star) \in \mathsf{List}_\phi$
  $\mathsf{bad}^2 \leftarrow \mathsf{true}$
  $C \leftarrow\$ \mathsf{Dom}(1^\lambda) \times \mathsf{Dom}(1^\lambda)$
Else If $(\phi', K_1', K_2') \in \mathsf{List}_\phi \wedge \phi' \neq \phi$
  $\mathsf{bad}^5 \leftarrow \mathsf{true}$
  $C \leftarrow\$ \mathsf{Dom}(1^\lambda) \times \mathsf{Dom}(1^\lambda)$
Else
  $C \leftarrow \mathbf{F}^{\mathrm{RO}_{11}, \mathrm{RO}_2, \mathrm{RO}_3, \mathrm{RO}_4}[\mathbf{K}'](M)$
$\mathsf{List}_\phi \leftarrow (\phi, K_1', K_2') : \mathsf{List}_\phi$
Return C

$\underline{\mathrm{RO}_{11}(K, X)}$:
If $(K, X, Y) \in \mathsf{List}_1$ Return Y
$Y \leftarrow\$ \mathsf{Dom}(1^\lambda)$
If $(K, X, \star) \in \mathsf{List}_3$ Then $\mathsf{bad}^4 \leftarrow \mathsf{true}$
Else $\mathsf{List}_1 \leftarrow (K, X, Y) : \mathsf{List}_1$
Return Y

$\underline{\mathrm{RO}_i(K, X)}$: // $i = 1, 3$
$Y \leftarrow\$ \mathsf{Dom}(1^\lambda)$
If $(K, X, Y) \in \mathsf{List}_1 \cup \mathsf{List}_3$ Then $\mathsf{bad}^6 \leftarrow \mathsf{true}$
Else $\mathsf{List}_i \leftarrow (K, X, Y) : \mathsf{List}_i$
Return Y

$\underline{\mathrm{RKF}_{\mathrm{N}_1}^{-1}(\phi, M)}$:
$(K_1', K_2') \leftarrow \phi(K_1, K_2)$
$\mathbf{K}' \leftarrow (K_1', K_2', K_1', K_2')$
If $(\star, \star, K_1') \in \mathsf{List}_\phi \vee (\star, K_2', \star) \in \mathsf{List}_\phi$
  $\mathsf{bad}^2 \leftarrow \mathsf{true}$
  $C \leftarrow\$ \mathsf{Dom}(1^\lambda) \times \mathsf{Dom}(1^\lambda)$
Else If $(\phi', K_1', K_2') \in \mathsf{List}_\phi \wedge \phi' \neq \phi$
  $\mathsf{bad}^5 \leftarrow \mathsf{true}$
  $C \leftarrow\$ \mathsf{Dom}(1^\lambda) \times \mathsf{Dom}(1^\lambda)$
Else
  $C \leftarrow \mathbf{F}^{-\mathbf{1}\,\mathrm{RO}_1, \mathrm{RO}_2, \mathrm{RO}_3, \mathrm{RO}_{44}}[\mathbf{K}'](M)$
$\mathsf{List}_\phi \leftarrow (\phi, K_1', K_2') : \mathsf{List}_\phi$
Return C

$\underline{\mathrm{RO}_{44}(K, X)}$:
If $(K, X, Y) \in \mathsf{List}_4$ Return Y
$Y \leftarrow\$ \mathsf{Dom}(1^\lambda)$
If $(K, X, \star) \in \mathsf{List}_2$ Then $\mathsf{bad}^4 \leftarrow \mathsf{true}$
Else $\mathsf{List}_4 \leftarrow (K, X, Y) : \mathsf{List}_4$
Return Y

$\underline{\mathrm{RO}_i(K, X)}$: // $i = 2, 4$
$Y \leftarrow\$ \mathsf{Dom}(1^\lambda)$
If $(K, X, Y) \in \mathsf{List}_2 \cup \mathsf{List}_4$ Then $\mathsf{bad}^6 \leftarrow \mathsf{true}$
Else $\mathsf{List}_i \leftarrow (K, X, Y) : \mathsf{List}_i$
Return Y

Game_7

$\underline{\mathrm{RKF}_{\mathrm{N}_1}(\phi, M)}$:
$(K_1', K_2') \leftarrow \phi(K_1, K_2)$
$\mathbf{K}' \leftarrow (K_1', K_2', K_1', K_2')$
If $(\star, \star, K_1') \in \mathsf{List}_\phi \vee (\star, K_2', \star) \in \mathsf{List}_\phi$
  $\mathsf{bad}^2 \leftarrow \mathsf{true}$
Else If $(\phi', K_1', K_2') \in \mathsf{List}_\phi \wedge \phi' \neq \phi$
  $\mathsf{bad}^5 \leftarrow \mathsf{true}$
$C \leftarrow\$ \mathsf{Dom}(1^\lambda) \times \mathsf{Dom}(1^\lambda)$
$\mathsf{List}_\phi \leftarrow (\phi, K_1', K_2') : \mathsf{List}_\phi$
Return C

$\underline{\mathrm{RKF}_{\mathrm{N}_1}^{-1}(\phi, M)}$:
$(K_1', K_2') \leftarrow \phi(K_1, K_2)$
$\mathbf{K}' \leftarrow (K_1', K_2', K_1', K_2')$
If $(\star, \star, K_1') \in \mathsf{List}_\phi \vee (\star, K_2', \star) \in \mathsf{List}_\phi$
  $\mathsf{bad}^2 \leftarrow \mathsf{true}$
Else If $(\phi', K_1', K_2') \in \mathsf{List}_\phi \wedge \phi' \neq \phi$
  $\mathsf{bad}^5 \leftarrow \mathsf{true}$
$C \leftarrow\$ \mathsf{Dom}(1^\lambda) \times \mathsf{Dom}(1^\lambda)$
$\mathsf{List}_\phi \leftarrow (\phi, K_1', K_2') : \mathsf{List}_\phi$
Return C

Fig. 13: Game_6 and Game_7 used in the proof of Theorem 4.

We start by proving two lemmas in the environment of Game_7.

Lemma 6. *For any* Game_7 *adversary* $\mathcal{A}$*, there exists an adversary* $\mathcal{B}_1$ *against the switch-free property of* Φ *such that*

$$\Pr[\mathsf{Bad}_7^2] \leq \mathbf{Adv}^{\mathsf{sf}}_{\mathcal{B}_1,\Phi}(\lambda) \ .$$

Proof. We build adversary $\mathcal{B}_1$ against the switch-free property of Φ as follows. Adversary $\mathcal{B}_1$ runs adversary $\mathcal{A}$ in the Game_7 environment, returning always random outputs on queries to RKFN_1 or RKFN_1^{-1}. When $\mathcal{A}$ terminates, $\mathcal{B}_1$ returns the list of RKD functions queried to the RKFN_1 or RKFN_1^{-1} oracles by $\mathcal{A}$. If event Bad_7^2 occurred, then the list of functions will satisfy the winning condition in the switch-free game. To see this, note that the bad^2 flag is set if and only if a key K_1' occurred that collided with a previous value of K_2' *or* a key K_2' occurred that collided with a previous value of K_1'. Then, if Bad_7^2 occurred, it must be the case that in the list of ϕ's queried by $\mathcal{A}$ to the RKFN_1 or RKFN_1^{-1} oracles, there exist ϕ and ϕ' such that $\phi(K_1, K_2)|_1 = \phi'(K_1, K_2)|_2$. □

Lemma 7. *For any* Game_7 *adversary* $\mathcal{A}$*, there exists an adversary* $\mathcal{B}_2$ *against the claw-free property of* Φ *such that*

$$\Pr[\mathsf{Bad}_7^5] \leq \mathbf{Adv}^{\mathsf{cf}}_{\mathcal{B}_2,\Phi}(\lambda) \ .$$

Proof. We build an adversary $\mathcal{B}_2$ against the claw-free property of Φ. Adversary $\mathcal{B}_2$ runs adversary $\mathcal{A}$ in the Game_7 environment, returning always random outputs on queries to RKFN_1 or RKFN_1^{-1}. When $\mathcal{A}$ terminates, $\mathcal{B}_2$ returns the list of functions queried to the RKFN_1 or RKFN_1^{-1} oracles by $\mathcal{A}$. If event Bad_7^5 occurred, then the list of functions will satisfy the winning condition in the claw-free game. To see this, note that the bad^5 flag is set if and only if an entry was added to List_ϕ such that ϕ was different from a previous ϕ' and yet the value of (K_1', K_2') was the same. □

The next lemma is central to the analysis of the bad events in Game_6.

Lemma 8. *Throughout the execution of* Game_6 *the values of* Y *stored in* List_1 *and* List_4*, i.e., the third component of the entries of* List_1 *and* List_4*, are uniformly and mutually independently distributed. Furthermore, they are independent from the view of the adversary, which comprises its random coins and the answers it receives from the queries to the* RKFN_1 *and* RKFN_1^{-1} *oracles.*

Proof. The proof is based on the alternative view of the RKFN_1 and RKFN_1^{-1} oracles shown in Figure 14.

Throughout the proof we will refer to the variables manipulated by these oracles when computing the answer to the adversary's i-th query by adding the query index as a superscript, referring to the i-th query by (ϕ^i, M^i).

The first query placed by the adversary is fully determined by its random coins. Suppose this is a RKFN_1 query. The Y_1^1 value is sampled uniformly at random and added to List_1. However, we can see that the value of Y_1^1 is information-theoretically hidden from the adversary as it is masked by Y_3^1, which is sampled in this query and never reused in the remainder of the game. Similarly, the value of Y_4^1 is also sampled uniformly at random, added to List_4 and is masked by the value of Y_2^1.

Suppose now the first query was to RKFN_1^{-1}. The Y_4^1 value is sampled uniformly at random and added to List_4. However, we can see that the value of L_1^1 is information-theoretically hidden from the adversary by the value of Y_2^1, which is sampled in this query and never reused in the remainder of the game. Similarly, the value of Y_1^1 is also sampled uniformly at random, added to List_1 and masked by the value of Y_3^1.

Let us now move to the second query. Conditioning on the view of the adversary in the first query, the second query is fully determined by the random coins of the adversary and the answers provided

$\underline{\mathrm{RKFN}_1(\phi,(L,R))}$:
$(K_1',K_2') \leftarrow \phi(K_1,K_2)$
If $(\star,\star,K_1') \in \mathsf{List}_\phi \vee (\star,K_2',\star) \in \mathsf{List}_\phi$
  $\mathsf{bad}^2 \leftarrow \mathsf{true}$
  $C \leftarrow\$ \mathsf{Dom}(1^\lambda) \times \mathsf{Dom}(1^\lambda)$
  $\mathsf{List}_\phi \leftarrow (\phi,K_1',K_2') : \mathsf{List}_\phi$
  Return C
Else If $(\phi',K_1',K_2') \in \mathsf{List}_\phi \wedge \phi' \neq \phi$
  $\mathsf{bad}^5 \leftarrow \mathsf{true}$
  $C \leftarrow\$ \mathsf{Dom}(1^\lambda) \times \mathsf{Dom}(1^\lambda)$
  $\mathsf{List}_\phi \leftarrow (\phi,K_1',K_2') : \mathsf{List}_\phi$
  Return C

$L_1 \leftarrow R$ // Round 1
If $(K_1',R,Y_1) \in \mathsf{List}_1$ Then $R_1 \leftarrow L \oplus Y_1$
Else
  $Y_1 \leftarrow\$ \mathsf{Dom}(1^\lambda)$
  If $(K_1',R,\star) \in \mathsf{List}_3$ Then $\mathsf{bad}^4 \leftarrow \mathsf{true}$
  Else $\mathsf{List}_1 \leftarrow (K_1',R,Y_1) : \mathsf{List}_1$
  $R_1 \leftarrow L \oplus Y_1$

$L_2 \leftarrow R_1$ // Round 2
$Y_2 \leftarrow\$ \mathsf{Dom}(1^\lambda)$
If $(K_2',R_1,\star) \in \mathsf{List}_2 \cup \mathsf{List}_4$ Then $\mathsf{bad}^6 \leftarrow \mathsf{true}$
Else $\mathsf{List}_2 \leftarrow (K_2',R_1,Y_2) : \mathsf{List}_2$
$R_2 \leftarrow L_1 \oplus Y_2$

$L_3 \leftarrow R_2$ // Round 3
$Y_3 \leftarrow\$ \mathsf{Dom}(1^\lambda)$
If $(K_1',R_2,\star) \in \mathsf{List}_1 \cup \mathsf{List}_3$ Then $\mathsf{bad}^6 \leftarrow \mathsf{true}$
Else $\mathsf{List}_3 \leftarrow (K_1',R_2,Y_3) : \mathsf{List}_3$
$R_3 \leftarrow L_2 \oplus Y_3$

$L_4 \leftarrow R_3$ // Round 4
$Y_4 \leftarrow\$ \mathsf{Dom}(1^\lambda)$
If $(K_2',R_3,\star) \in \mathsf{List}_2 \cup \mathsf{List}_4$ Then $\mathsf{bad}^6 \leftarrow \mathsf{true}$
Else $\mathsf{List}_4 \leftarrow (K_2',R_3,Y_4) : \mathsf{List}_4$
$R_4 \leftarrow L_3 \oplus Y_4$

$C \leftarrow (L_4,R_4)$; Return C

$\underline{\mathrm{RKFN}_1^{-1}(\phi,(L_4,R_4))}$:
$(K_1',K_2') \leftarrow \phi(K_1,K_2)$
If $(\star,\star,K_1') \in \mathsf{List}_\phi \vee (\star,K_2',\star) \in \mathsf{List}_\phi$
  $\mathsf{bad}^2 \leftarrow \mathsf{true}$
  $C \leftarrow\$ \mathsf{Dom}(1^\lambda) \times \mathsf{Dom}(1^\lambda)$
  $\mathsf{List}_\phi \leftarrow (\phi,K_1',K_2') : \mathsf{List}_\phi$
  Return C
Else If $(\phi',K_1',K_2') \in \mathsf{List}_\phi \wedge \phi' \neq \phi$
  $\mathsf{bad}^5 \leftarrow \mathsf{true}$
  $C \leftarrow\$ \mathsf{Dom}(1^\lambda) \times \mathsf{Dom}(1^\lambda)$
  $\mathsf{List}_\phi \leftarrow (\phi,K_1',K_2') : \mathsf{List}_\phi$
  Return C

$R_3 \leftarrow L_4$ // Round 4
If $(K_2',R_3,Y_4) \in \mathsf{List}_4$ Then $L_3 \leftarrow R_4 \oplus Y_4$
Else
  $Y_4 \leftarrow\$ \mathsf{Dom}(1^\lambda)$
  If $(K_2',R_3,\star) \in \mathsf{List}_2$ Then $\mathsf{bad}^4 \leftarrow \mathsf{true}$
  Else $\mathsf{List}_4 \leftarrow (K_2',R_3,Y_4) : \mathsf{List}_4$
  $L_3 \leftarrow R_4 \oplus Y_4$

$R_2 \leftarrow L_3$ // Round 3
$Y_3 \leftarrow\$ \mathsf{Dom}(1^\lambda)$
If $(K_1',R_2,\star) \in \mathsf{List}_1 \cup \mathsf{List}_3$ Then $\mathsf{bad}^6 \leftarrow \mathsf{true}$
Else $\mathsf{List}_3 \leftarrow (K_1',R_2,Y_3) : \mathsf{List}_3$
$L_2 \leftarrow R_3 \oplus Y_3$

$R_1 \leftarrow L_2$ // Round 2
$Y_2 \leftarrow\$ \mathsf{Dom}(1^\lambda)$
If $(K_2',R_1,\star) \in \mathsf{List}_2 \cup \mathsf{List}_4$ Then $\mathsf{bad}^6 \leftarrow \mathsf{true}$
Else $\mathsf{List}_2 \leftarrow (K_2',R_1,Y_2) : \mathsf{List}_2$
$L_1 \leftarrow R_2 \oplus Y_2$

$R \leftarrow L_1$ // Round 1
$Y_1 \leftarrow\$ \mathsf{Dom}(1^\lambda)$
If $(K_1',R,\star) \in \mathsf{List}_1 \cup \mathsf{List}_3$ Then $\mathsf{bad}^6 \leftarrow \mathsf{true}$
Else $\mathsf{List}_1 \leftarrow (K_1',R,Y_1) : \mathsf{List}_1$
$L \leftarrow R_1 \oplus Y_1$

$C \leftarrow (L,R)$; Return C

Fig. 14: Expanded view of the RKFN_1 and RKFN_1^{-1} oracles in Game_6.

by the previous query. Once again Y_1^2 is masked by a uniform and independently chosen value Y_3^2, and hence Y_1^2 remains hidden from the adversary's view. Similarly Y_4^2 remains hidden from the adversary.

This inductive argument can be applied to all subsequent queries, establishing the independence claim in the lemma. It also implies that the outputs of the RKFN_1 and RKFN_1^{-1} provided to the adversary in this game are all uniformly and independently distributed. □

The next lemma gives an upper-bound on the probability of the bad^4 flag being activated in Game_6.

Lemma 9. *For any* Game_6 *adversary* $\mathcal{A}$ *placing at most* $Q(\lambda)$ *oracle queries in total*

$$\Pr[\mathsf{Bad}_6^4] \leq \frac{Q(\lambda)^2}{2|\mathsf{Dom}_\lambda|} \ .$$

Proof. We will refer again to Figure 14. Take any two queries i and j, with $j < i$ that would cause the bad^4 flag to be set when query i is placed, due to a conflict with a value stored in either List_2 or List_3 due to query j.

For a conflict to occur in List_3 during the computation of RKFN_1, we would need to have $R^i = R_2^j$. There are two cases:

- If R_2^j resulted from a forward query, then we would need to have $R_2^j = R^j \oplus Y_2^j$. This means the adversary could place a query that would satisfy the following equation: $R^j \oplus R^i = Y_2^j$. However, Y_2^j occurred only once in the game and was given to the adversary in the form of $Y_2^j = R_4^j \oplus R^j \oplus Y_4^j$. Rewriting the equations, the adversary would need to come up with a query that satisfies the following equation: $R^j \oplus R^i = R_4^j \oplus R^j \oplus Y_4^j$. This simplifies to $R^i = R_4^j \oplus Y_4^j$.
- If R_2^j resulted from a backward query, then the adversary would need to satisfy exactly the same equation $R^i = R_4^j \oplus Y_4^j$.

In both of the equations above, all of the terms are fixed in the adversary's view, except the Y_4^j value. However, from Lemma 8, we know that the Y_4 values stored in List_4 are independent from the adversary's view, which means that the probability of this event for queries i and j is at most $1/|\mathsf{Dom}_\lambda|$.

The probability of a conflict occurring in List_2 during the computation of RKFN_1^{-1}, can be analyzed in exactly the same way. We conclude that, for any two queries i and j with $j < i$, the probability of the bad^4 flag being set is upper-bounded by $1/|\mathsf{Dom}_\lambda|$ (the flag can only be activated in one of the oracles in each query). Applying the union bound, we obtain

$$\Pr[\mathsf{Bad}_6^4] \leq \sum_{i=1}^{Q(\lambda)} \sum_{j=1}^{i-1} \frac{1}{|\mathsf{Dom}_\lambda|} \leq \frac{Q(\lambda)^2}{2|\mathsf{Dom}_\lambda|} \ .$$

□

The next lemma provides a bound on the probability of the bad^6 flag being activated in Game_6.

Lemma 10. *For any* Game_6 *adversary* $\mathcal{A}$ *placing at most* $Q(\lambda)$ *oracle queries in total*

$$\Pr[\mathsf{Bad}_6^6] \leq \frac{5Q(\lambda)^2}{|\mathsf{Dom}_\lambda|} \ .$$

Proof. We will refer again to Figure 14. Take any two queries i and j with $j < i$ that would cause the bad^6 flag to be set when query i is placed due to a conflict with a value stored in one of the lists in the game during the execution of query j. We split the analysis by defining various sub-events:

F_1 : bad^6 is activated in the third and fourth rounds of the RKFN_1 oracle, *or* in the first and second rounds of the RKFN_1^{-1} oracle, i.e., the output rounds in both oracles.

F_2 : bad^6 is activated in the second round of the RKFN_1 oracle due to a collision with a value of the form (K'^j_2, R_3^j) stored in List_4.

F_3 : bad^6 is activated in the third round RKFN_1^{-1} oracle due to a collision with a value of the form (K'^j_1, R^j) stored in List_1.

F_4 : bad^6 is activated in the second round of the RKFN_1 oracle due to a collision with a value of the form (K'^j_2, R_1^j) stored in List_2.

F_5 : bad^6 is activated in the third round of the RKFN_1^{-1} oracle due to a collision with a value of the form (K'^j_1, R_2^j) stored in List_3.

By the union bound

$$\Pr[\mathsf{Bad}_6^6] = \Pr[F_1 \vee F_2 \vee F_3 \vee F_4 \vee F_5] \leq \Pr[F_1] + \Pr[F_2] + \Pr[F_3] + \Pr[F_4] + \Pr[F_5] \ .$$

We first bound the probability of F_1. There are four places in which the bad^6 flag can be activated to cause this event, and we will upper-bound the probability of each of them occurring in exactly the same way. Observe that any query j adds at most two values to the composed lists $\mathsf{List}_1 \cup \mathsf{List}_3$ and $\mathsf{List}_2 \cup \mathsf{List}_4$. Furthermore, due to the action of Y_2^i and Y_3^i, the values of R_2^i and R_3^i in $\mathrm{RKF}_{\mathrm{N}_1}$ are uniformly distributed and independent from all of the previous values stored in $\mathsf{List}_1 \cup \mathsf{List}_3$ and $\mathsf{List}_2 \cup \mathsf{List}_4$ in all queries $j < i$; similarly, the values of R_1^i and R^i in $\mathrm{RKF}_{\mathrm{N}_1}^{-1}$ are uniformly distributed and independent from all of the previous values stored in $\mathsf{List}_1 \cup \mathsf{List}_3$ and $\mathsf{List}_2 \cup \mathsf{List}_4$ in all queries $j < i$. This means that the probability of the bad^6 flag being activated due to a conflict between values added in queries i and j is $2/|\mathsf{Dom}_\lambda|$ at each of the four points relevant for F_1. By the union bound, and observing that in query i only one of the oracles will be executed (meaning that the bad^6 can only be activated in two positions), we obtain that

$$\Pr[F_1] \ \leq \ \sum_{i=1}^{Q(\lambda)} \sum_{j=1}^{i-1} \frac{2 \cdot 2}{|\mathsf{Dom}_\lambda|} \leq \ \frac{2Q(\lambda)^2}{|\mathsf{Dom}_\lambda|} \ .$$

We move on to analyze F_2. For this event to occur, we would need to have $R_1^i = R_3^j$ or $L^i \oplus Y_1^i = R_3^j$, where $R_3^j = L_4^j$ and L^i can be fixed, leaving Y_1^i independently distributed according to Lemma 8. We note that due to our labeling of variables in the $\mathrm{RKF}_{\mathrm{N}_1}^{-1}$ procedure, this equation is also valid for values of R_3^j and L_4^j in backward queries. So, the probability of this occurring is at most $1/|\mathsf{Dom}_\lambda|$. Again, over all $Q(\lambda)$ queries, we have

$$\Pr[F_2] \ \leq \ \sum_{i=1}^{Q(\lambda)} \sum_{j=1}^{i-1} \frac{1}{|\mathsf{Dom}_\lambda|} \leq \ \frac{Q(\lambda)^2}{2|\mathsf{Dom}_\lambda|} \ .$$

The analysis of F_3 is analogous and leads to

$$\Pr[F_3] \leq \frac{Q(\lambda)^2}{2|\mathsf{Dom}_\lambda|} \ .$$

Let us now consider the probability of F_4. Recall that this refers to a collision between values of the form $(K_2'^i, R_1^i)$ and $(K_2'^j, R_1^j)$ in the second round of the $\mathrm{RKF}_{\mathrm{N}_1}$ oracle, immediately after the generation of Y_2.

We need to split the analysis into several cases. We observe that the j-th query may have been a forward or a backward query. However, in the case analysis below this will be irrelevant due to the assumption that the adversary does not place redundant queries and due to the fact that, in Figure 14, the labeling of variables is consistent between the two oracles, leading to identical equations in both case. We consider the following cases:

1. Case $\phi^i \neq \phi^j$.
   Given the operation of initial lines in the $\mathrm{RKF}_{\mathrm{N}_1}$ oracle, we know that the Feistel construction will only be computed if $(K_1'^i, K_2'^i) \neq (K_1'^j, K_2'^j)$, so we can consider two sub-cases.
   (a) Case $K_2'^i \neq K_2'^j$.
   In this case, it is clear that, by definition, F_4 cannot be caused due to a conflict between queries i and j.

(b) Case ${K'_2}^i = {K'_2}^j$.
In this case we must have that ${K'_1}^i \neq {K'_1}^j$ and event F_4 will occur due to a collision with $({K'_2}^j, R_1^j)$ if and only if $L^i \oplus L^j = Y_1^i \oplus Y_1^j$. However, we know Y_1^i and Y_1^j were sampled independently from each other (due to the distinct K'_1 values and Lemma 8) and are information-theoretically hidden from the adversary (again by Lemma 8). This means that the probability of this happening is upper-bounded by $1/|\mathsf{Dom}_\lambda|$.

2. Case $\phi^i = \phi^j$. In this case we surely have ${K'_2}^i = {K'_2}^j$ occurring in queries i and j, and it must be the case that $M^i = (L^i, R^i) \neq (L^j, R^j) = M^j$. Again we consider two sub-cases.
   (a) Case $R^i = R^j$.
   In this case we must have $L^i \neq L^j$ (since we know that the adversary does not place redundant queries) and, by construction, it follows that $R_1^j \neq R_1^i$ and therefore event F_4 cannot occur due to a collision with $({K'_2}^j, R_1^j)$.
   (b) Case $R^i \neq R^j$.
   In this case, event F_4 will occur due to a collision with $({K'_2}^j, R_1^j)$ if and only if $L^i \oplus L^j = Y_1^i \oplus Y_1^j$. However, we know Y_1^i and Y_1^j were sampled independently from each other (due to distinct R values and Lemma 8) and are information-theoretically hidden from the adversary (again by Lemma 8). This means that the probability of this happening is upper-bounded by $1/|\mathsf{Dom}_\lambda|$.

Putting together the results of our case analysis, and using the union bound over all query combinations, we get that

$$\Pr[F_4] \;\leq\; \sum_{i=1}^{Q(\lambda)} \sum_{j=1}^{i-1} \frac{2}{|\mathsf{Dom}_\lambda|} \leq \frac{Q(\lambda)^2}{|\mathsf{Dom}_\lambda|} \;.$$

The analysis for F_5 follows an identical structure to that presented for $\Pr[F_4]$, and we obtain precisely the same bound:

$$\Pr[F_5] \;\leq\; \frac{Q(\lambda)^2}{|\mathsf{Dom}_\lambda|} \;.$$

The lemma follows from the combination of the above results. □

We now use the previous results to complete the proof of Theorem 4.

Proof. We have defined Game_0 to be the Φ-RKCCA game and changed Game_0 to Game_1 by implementing the ideal keyed function via lazy sampling. The two games are identical and therefore

$$\Pr[S_1] = \Pr[S_0] \;.$$

From Game_1 to Game_2 we modify the way in which the ideal keyed function is simulated. Rather than keeping a single consistent list of answered calls, we use two separate lists List_{13} and List_{24}. We also add consistency check that may set a bad^2 flag when a query is placed by the adversary that would reveal an inconsistency in the random function simulation. Once this flag is set, the adversary receives only random values from the RKFN_1 and RKFN_1^{-1} oracles. However, if the bad^2 flag is not set, then the two games are identical, and thus

$$\Pr[S_1] - \Pr[S_2] \leq \Pr[\mathsf{Bad}_2^2] \;.$$

The analysis of the probability of event Bad_2^2 will be deferred until Game_7.

In Game_3 we introduce another conceptual change in the way we manage the values in List_{13} and List_{24}. We separate the values generated by all rounds into four separate lists, but we keep track of the values in exactly the same way as in Game_2. This change does not affect the game in any way, and so

$$\Pr[S_3] = \Pr[S_2] \quad \text{and} \quad \Pr[\mathsf{Bad}_2^2] = \Pr[\mathsf{Bad}_3^2] \;.$$

Combining the results so far, we have:

$$\Pr[S_0] - \Pr[S_3] \leq \Pr[\mathsf{Bad}_3^2] \ .$$

In Game_4 we make sure that round 1 in forward computations reuses values generated by this round alone, i.e., values in List_1. Similarly, we ensure that round 4 in backward computations reuses values from this value alone, i.e., values in List_4. The bad^4 flag signals situations in which this new form of simulating the round functions could introduce an inconsistency with Game_3. For this reason, Game_3 and Game_4 are identical until bad^4:

$$\Pr[S_3] - \Pr[S_4] \leq \Pr[\mathsf{Bad}_4^4] \ ,$$
$$\Pr[\mathsf{Bad}_3^2] - \Pr[\mathsf{Bad}_4^2] \leq \Pr[\mathsf{Bad}_4^4] \ .$$

The analysis of the probability of event Bad_4^4 will be deferred until Game_6. In any case, we obtain

$$\Pr[S_0] - \Pr[S_4] \leq \Pr[\mathsf{Bad}_4^2] + 2 \cdot \Pr[\mathsf{Bad}_4^4] \ .$$

In Game_5 we exclude the possibility that the adversary explores claws in the RKD function family. On every query to the $\mathrm{RKF}_{\mathrm{N}_1}$ or $\mathrm{RKF}_{\mathrm{N}_1}^{-1}$ oracles we check whether the adversary found a claw in the RKD function family. If so, we set a new bad^5 flag and return random values to the adversary. If not, we proceed as in Game_4. Hence

$$\Pr[S_4] - \Pr[S_5] \leq \Pr[\mathsf{Bad}_5^5] \ ,$$
$$\Pr[\mathsf{Bad}_4^4] - \Pr[\mathsf{Bad}_5^4] \leq \Pr[\mathsf{Bad}_5^5] \ ,$$
$$\Pr[\mathsf{Bad}_4^2] - \Pr[\mathsf{Bad}_5^2] \leq \Pr[\mathsf{Bad}_5^5] \ .$$

The analysis of the probability of setting the bad^5 flag is deferred until Game_7.

In Game_6 we change the simulation of the ideal keyed function once more, returning a fresh random value to all calls placed by rounds 2, 3 and 4 in forward computations and rounds 1, 2 and 3 in backward computations. The bad^6 flag signals situations in which this new form of simulating the round functions could introduce an inconsistency with Game_5. We have:

$$\Pr[S_5] - \Pr[S_6] \leq \Pr[\mathsf{Bad}_6^6] \ ,$$
$$\Pr[\mathsf{Bad}_5^5]] - \Pr[\mathsf{Bad}_6^5] \leq \Pr[\mathsf{Bad}_6^6] \ ,$$
$$\Pr[\mathsf{Bad}_5^4] - \Pr[\mathsf{Bad}_6^4] \leq \Pr[\mathsf{Bad}_5^5] \ ,$$
$$\Pr[\mathsf{Bad}_5^2] - \Pr[\mathsf{Bad}_6^2] \leq \Pr[\mathsf{Bad}_6^6] \ .$$

Combining these equations with the results above, we get:

$$\Pr[S_0] - \Pr[S_6] \leq \Pr[\mathsf{Bad}_6^2] + 2 \cdot \Pr[\mathsf{Bad}_6^4] + 4 \cdot \Pr[\mathsf{Bad}_6^5] + 8 \cdot \Pr[\mathsf{Bad}_6^6] \ .$$

Finally, we observe that in Game_6 the keys K_1 and K_2 are only used in the computation of Y_1 and Y_4 values, which by Lemma 8 are information-theoretically hidden from the adversary. Furthermore, due to the action of rounds 3 and 4 in the $\mathrm{RKF}_{\mathrm{N}_1}$ oracle and rounds 1 and 2 in the $\mathrm{RKF}_{\mathrm{N}_1}^{-1}$ oracle, the values returned by each query and uniformly and independently distributed from all previous answers provided to the adversary. Therefore, in Game_7 we make the conceptual change of simplifying Game_6 so that random answers are provided to the adversary. For convenience, we preserve the code related to the bad^2 and bad^5 flags. Since the view of the adversary in Game_7 is identical to that in Game_6,

$$\Pr[S_6] = \Pr[S_7], \quad \Pr[\mathsf{Bad}_6^2] = \Pr[\mathsf{Bad}_7^2] \quad \text{and} \quad \Pr[\mathsf{Bad}_6^5] = \Pr[\mathsf{Bad}_7^5] \ .$$

Consequently, we get

$$\Pr[S_0] - \Pr[S_7] \leq \Pr[\mathsf{Bad}_7^2] + 2 \cdot \Pr[\mathsf{Bad}_6^4] + 4 \cdot \Pr[\mathsf{Bad}_7^5] + 8 \cdot \Pr[\mathsf{Bad}_6^6] \ .$$

The probabilities of Bad_7^2 occurring and Bad_7^5 occurring are upper-bounded in Lemmas 6 and 7, respectively. The probabilities of Bad_6^4 occurring and Bad_6^6 occurring are upper-bounded in Lemmas 9 and 10, respectively.

Finally, in Game_7, and recalling that the adversary is assumed not to place redundant queries, we can see that the adversary is either interacting with the ideal keyed permutation, or with two ideal keyed functions implemented by oracles RKFN_1 and RKFN_1^{-1}. This means that, by construction, the adversary's probability of success in Game_7 is exactly that established in Lemma 1:

$$2 \cdot \Pr[S_7] - 1 = \mathbf{Adv}_{\mathcal{A},\Phi}^{\mathsf{rf/rp}}(\lambda) \ .$$

The theorem follows by combining the previous results:

$$2 \cdot \Pr[S_0] - 1 \leq \mathbf{Adv}_{\mathcal{A},\Phi}^{\mathsf{rf/rp}}(\lambda) + 2 \cdot \Pr[\mathsf{Bad}_7^2] + 4 \cdot \Pr[\mathsf{Bad}_6^4] + 8 \cdot \Pr[\mathsf{Bad}_7^5] + 16 \cdot \Pr[\mathsf{Bad}_6^6] \ .$$

□

H Security Analysis of $\mathbf{F}^\rho[1,2,3,4]$

Similarly to the CPA setting, we consider the question of which level of RKA security does the 4-round Luby–Rackoff construction achieve. We establish security with respect to all functions $\phi \in \Phi_4^\star$ of the form

$$\phi_{C_1,C_2,C_3,C_4} : (K_1, K_2, K_3, K_4) \mapsto (K_1 \oplus C_1, K_2 \oplus C_3, K_3 \oplus C_3, K_4 \oplus C_4)$$

such that, for any two $(C_1, C_2, C_3, C_4), (C_1', C_2', C_3', C_4') \in \Phi_4^\star$, if $(C_4, C_3) \neq (C_4', C_3')$ then $(C_1, C_2) \neq (C_1', C_2')$, and if $(C_1, C_2) \neq (C_1', C_2')$, then $(C_3, C_4) \neq (C_3', C_4')$. Our result is as follows.

Proposition 3. *Let $\Phi_4^\star$ be the RKD set above. Then the $\mathbf{F}^\rho[1,2,3,4]$ construction is a $\Phi_4^\star$-RKCCA-secure PRP. More precisely, for any* PPT *adversary $\mathcal{A}$ we have that*

$$\mathbf{Adv}_{\mathbf{F}^\rho[1,2,3,4],\mathcal{A},\Phi_4^\star}^{\mathsf{rkcca}}(\lambda) \leq \mathbf{Adv}_{\mathcal{A},\Phi_4^\star}^{\mathsf{rf/rp}}(\lambda) + \frac{2^9 \cdot Q(\lambda)^2}{|\mathsf{Dom}_\lambda|} \ ,$$

where $Q(\lambda)$ is the maximum number of queries that $\mathcal{A}$ to places to either of its oracles.

Proof (Sketch). The proof of this proposition uses a sequence of games similar to those in the proof of Theorem 4. Again, the main difference lies in the fact that the restrictions imposed on the RKD-set enforce exactly the same restrictions that we derived from the claw-free and switch-free conditions. More precisely, given that the XOR-based RKD set is intrinsically switch-free, the extra restriction imposed on the RKD-set are needed to obtain benefits analogous to the claw-free condition. In this case, for the case analysis to go through, we must ensure that different RKD functions imply a different key in at least one of rounds 1 and 2 in RKFN and at least one of rounds 3 and 4 in RKFN^{-1}. This justifies the $(C_4, C_3) \neq (C_4', C_3') \Rightarrow (C_1, C_2) \neq (C_1', C_2')$ and $(C_1, C_2) \neq (C_1', C_2') \Rightarrow (C_3, C_4) \neq (C_3', C')$ restrictions. The proposition then follows from the same information-theoretic arguments that conclude the proof of Theorem 4 together with an application of Lemma 1. □

We observe that, unlike the CPA case, we do not know if the restriction on $\Phi_4^\star$ is necessary. Indeed, it is an interesting open question to find an attack that shows that this is the case or, conversely, to strengthen the result by imposing a weaker restriction (see the conjecture in Section 8).