Feistel Networks made Public, and Applications

Yevgeniy Dodis[*] Prashant Puniya[†]

February 11, 2007

Abstract

Feistel Network, consisting of a repeated application of the Feistel Transform, gives a very convenient and popular method of designing "cryptographically strong" permutations from corresponding "cryptographically strong" functions. Up to now, all usages of the Feistel Network, including the celebrated Luby-Rackoff's result, critically rely on (a) *the (pseudo)randomness of round functions*; and (b) *the secrecy of (at least some of) the intermediate round values* appearing during the Feistel computation. Moreover, a small constant number of Feistel rounds was typically sufficient to guarantee security under assumptions (a) and (b). In this work we consider several natural scenarios where at least one of the above assumptions does not hold, and show that a constant, or even logarithmic number of rounds is *provably insufficient* to handle such applications, implying that a new method of analysis is needed.

On a positive side, we develop a new combinatorial understanding of Feistel networks, which makes them applicable to situations when the round functions are merely *unpredictable* rather than (pseudo)random and/or when the intermediate round values may be leaked to the adversary (either through an attack or because the application *requires* it). In essence, our results show that in any such scenario a super-logarithmic number of Feistel rounds is *necessary and sufficient* to guarantee security. This partially explains why practical block ciphers use significantly more than 3-6 rounds predicted by the previous theoretical results, and also gives the first theoretical justification regarding the usage of Feistel Networks not satisfying assumptions (a) or (b) above.

In particular, we show that super-logarithmic number of Feistel rounds is *necessary and sufficient* to yield

- a strong unpredictable permutation (UP) from any unpredictable function (UF).
- a strong pseudorandom permutation (PRP) from any pseudorandom function (PRF), which remains secure even if all the round values are made public.
- a strong verifiable unpredictable permutation (VUP) — a new notion we introduce here — from any verifiable unpredictable function (VUF).
- a strong verifiable random permutation (VRP) — a new notion we introduce here — from any verifiable random function (VRF, also known as unique signature scheme).

Of independent interest, our technique yields a novel domain extension method for messages authentication codes and other related primitives, settling the question studied by An and Bellare in CRYPTO 1999.

[*]Department of Computer Science, New York University, 251 Mercer Street, New York, NY 10012, USA. Email: dodis@cs.nyu.edu

[†]Department of Computer Science New York University, 251 Mercer Street, New York, NY 10012, USA. Email: puniya@cs.nyu.edu

1 Introduction

Feistel Networks are extremely popular tools in designing "cryptographically strong" permutations from corresponding "cryptographically strong" functions. Such networks consist of several iterative applications of a simple Feistel permutation $\Psi_f(x_L \parallel x_R) = x_R \parallel x_L \oplus f(x_R)$, with different (pseudo)independent round functions f used at each round. Among their applications, they are commonly used in the design of popular block ciphers, such as DES, as well as other constructs, such as popular padding schemes OAEP [3] or PSS-R [4]. In particular, the celebrated result of Luby and Rackoff [27] shows that three (resp. four) rounds of the Feistel transform are sufficient to turn a pseudorandom function (PRF) family into a pseudorandom permutation (PRP) family (resp. strong PRP family). There has been a lot of subsequent work (e.g., [35, 39, 30, 38]) on improving various aspects of the Luby-Rackoff's result (referred to as "LR" from now on). However, all these results crucially relied on:

(a) *the (pseudo)randomness of round functions*; and
(b) *the secrecy of (at least some of) the intermediate round values appearing during the Feistel computation*

In this work we consider several natural scenarios where at least one of the above assumptions does not hold, and show that a fundamentally new analysis technique is needed for such applications. But first let us motivate our study.

IS UNPREDICTABILITY ENOUGH? We start with the assumption regarding pseudorandomness of round functions. This assumption is quite strong, since practical block ciphers certainly do not use PRFs as their round functions. Instead, they heuristically use considerably more than the three-six rounds predicted by the LR and all the subsequent "theoretical justifications". Thus, a large disconnect still remains to be bridged. Clearly, though, we need to assume some security property of the round function, but can a weaker property be enough to guarantee security? In the context of domain extension of message authentication codes, An and Bellare [1] studied a natural question whether *unpredictability* — a much weaker property than pseudorandomness — can at least guarantee the unpredictability of the resulting Feistel permutation. Although not strong as pseudorandomness, this will at least guarantee some minimal security of block ciphers (see Section 6), is enough for basic message authentication, and anyway doubles the domain of the unpredictable function, which is useful (and non-trivial!) by itself. [1] gave a negative answer for the case of three rounds, and suggested that "even more rounds do not appear to help". This result indicates that previous "LR-type techniques" are insufficient to handle unpredictability (since in the case of PRFs three rounds are enough), and also leaves open the question whether more Feistel rounds will eventually be enough to preserve unpredictability. Our work will completely resolve this question. Along the way, it will prove that Feistel Networks could serve as domain extenders for message authentication codes.

IS IT SAFE TO LEAK INTERMEDIATE RESULTS? Another crucial reason for the validity of the LR result is the fact that all the intermediate round values are never leaked to the attacker. In fact, the *key* to the argument, and most of the subsequent results, is that the attacker effectively gets no information about most of these values in case a PRF is used for the round function, and simple attacks (which we later generalize to many more rounds) are possible to invalidate the LR result in case the intermediate values are leaked. Unfortunately, for many natural applications this assumption (or conclusion!) can not be enforced, and totally new argument is needed. We give several examples.

Starting with the simplest (but also least interesting) example, intermediate values might be inadvertently leaked through an attack. For example, one might imagine a smartcard implementing a block cipher via the Feistel network using a secure chip implementing a PRF. In this case the attacker might be able to observe the communication between the smartcard and the chip, although it is unable to break the security of the chip. More realistically, when the round functions are not PRFs, the attacker might get a lot of information about the intermediate values anyway, even without extra attack capabilities. For example, is the case of unpredictable functions (UFs) mentioned above, we will construct provably secure UFs such that the output of the Feistel Network completely leaks *all* the intermediate round values. Although artificial, this example illustrates that weaker assumptions on the round functions can no longer guarantee the secrecy of intermediate values. For yet another example, the round function might simply be public to begin with. This happens when one considers the question of implementing an ideal cipher from a random oracle, considered by the authors in TCC'06 [14]. In this case the round function is a publicly accessible random oracle, and is certainly freely available to the attacker. To see the difference with the usual block cipher setting where four round are enough, [14] showed that even five Feistel rounds are not sufficient to built an ideal cipher, although conjectured that a larger constant number of rounds is sufficient. The authors also showed a weaker positive implication in the so called "honest-but-curious model", although only for a super-logarithmic number of rounds (as they also showed, reducing the number of rounds in this model would imply the security in the usual, "malicious" model). As a final example (not considered in prior work), the attacker might get hold of the intermediate values because the *application requires to reveal such values*. This happens when one tries to add *verifiability* to PRFs and PRPs (or their unpredictable analogs), which we now describe in more detail.

VERIFIABLE RANDOM FUNCTIONS AND PERMUTATIONS. We consider the problem of constructing *verifiable random permutations* (VRPs) from *verifiable random functions* (VRFs). VRFs and VRPs are verifiable analogs of PRFs and PRPs, respectively. Let us concentrate on VRFs first. Intuitively, regular PRFs have a limitation that one must trust the owner of the secret key that a given PRF value is correctly computed. And even when done so, a party receiving a correct PRF value cannot later convince some other party that the value is indeed correct (i.e., PRF values are "non-transferable"). In fact, since the function values are supposed to be (pseudo)random, it seems that such verifiability of outputs of a PRP would contradict its pseudorandomness. The way out of this contradiction was provided by Micali, Rabin and Vadhan [32], who introduced the notion of a VRF. Unlike PRFs, a VRF owner must be able to provide a short proof that any given VRF output is computed correctly. This implies that the VRF owner must publish a public key allowing others to verify the validity of such proofs. However, every "unopened" VRF value (i.e., one for which no proof was given yet) should still look indistinguishable from random, even if many other values were "opened" (by giving their proofs). Additionally, the public key should commit the owner of the VRF to all its function values in a unique way, even if the owner tries to select an "improper" public key. Micali et al. [32] also gave a secure construction of a VRF based on the RSA assumption. Since then several more efficient constructions of VRFs have been proposed based on various cryptographic assumption; see [28, 13, 15].

The notion of a VRP, which we introduce in this paper, naturally adds verifiability to PRPs, in exactly the same natural way as VRFs do to PRFs. We will describe some applications of VRPs in Section 6, but here let us concentrate on the relation between VRFs and VRPs. On the one hand, it is easy to see that a VRP (on a "non-trivial domain") is also a VRF, just like in the PRF/PRP case. On a first look, we might hope that the converse implication holds as well, by simply applying the Luby-Rackoff result to VRFs in place of PRFs. However, a moment of reflection shows that this is not the case. Indeed, the proof for the iterated Feistel construction *must include all the VRF values for the intermediate rounds*, together with their proofs. Thus, the attacker can legally obtain all the intermediate round values for every input/output that he queries, except for the one on which he is being "challenged". This rules out the LR-type proof for this application. More critically, even the recent proof of [14] (implementing the ideal cipher from a random oracle in the "honest-but-curious" model) appears to be "fundamentally inapplicable" as well. Indeed, that proof crucially used the fact that truly random functions (in fact, random oracles) are used in all the intermediate rounds: for example, to derive various birthday bounds used to argue that certain "undesirable" events are unlikely to happen. One might then hope that a similar argument might be carried out by replacing all the VRFs by truly random function as well. However, such "wishful replacement" is prevented by the fact that we are required to prove the correctness of each intermediate round value, and we (provably) *cannot provide such proofs when we use a totally random function in place of a VRF* (which is "committed" to by its public key). To put it differently, with a random function we have no hope of simulating the VRF proofs that are "legally expected" by an adversary attacking the VRP construction. Thus, again, a new technique is needed.

VERIFIABLE UNPREDICTABLE FUNCTIONS AND PERMUTATIONS. We also consider the natural combination of the scenarios we considered so far, exemplified by the task of constructing *verifiable unpredictable permutations* (VUPs) from *verifiable unpredictable functions* (VUFs) [32] (also called *unique signature schemes* [23, 28]). A VUF is defined in essentially the same way as VRFs, except that the pseudorandomness requirement for VRFs is replaced by a weaker unpredictability requirement. Similarly, VUPs, introduced in this paper, are either the permutation analogs of VUFs, or, alternatively, unpredictable analogs of VRPs. Of course, as a VRP is also a VUP, we could attempt to build a VUP by actually building a VRP via the Feistel construction applied to a VRF, as suggested in the previous paragraph. However, this seems quite wasteful since VUFs appear to be much easier to construct than VRFs. Indeed, although in theory VUFs are equivalent to VRFs [32], the "Goldreich-Leven-type" reduction from VUFs to VRFs in [32] is *extremely* inefficient (it loses exponential security and forces the authors to combine it with another inefficient tree construction). Moreover, several previous papers [32, 28] constructed *efficient* VUFs based on relatively standard *computational* assumptions, while all the *efficient* VRF constructions [13, 15] are based on very ad hoc *decisional* assumptions. Thus, it is natural to study the security of the Feistel network when applied to VUFs. In this case, not only the round functions cannot be assumed pseudorandom, but also all the intermediate values must be leaked together with their proofs of correctness, making this setting the most challenging to analyze.

OTHER RELATED WORK. Several prior works tried to relax the security of some of the round functions. For example, Naor and Reingold showed that the first and the fourth round could use pairwise independent hash functions instead of PRFs. In a different vein, Maurer et al. [29] studied the case when the PRFs used are only non-adaptively secure. Already in this setting, the authors showed that it is unlikely that four Feistel rounds would yield a PRP (although this is true in the so called "information-theoretic" setting). However, in these results at least some of the round functions are still assumed random. In terms of leaking intermediate results, Reyzin and Ramzan [39] showed that in a four-round construction it is safe to give the attacker *oracle access* to the second and third (but not first and fourth) round functions. This is incomparable to our setting: we leak intermediate results actually happening during the Feistel computation, and for *all* the rounds. Finally, we already mentioned the paper by the authors [14], which showed how to deal with public intermediate results when *truly random* round functions are used. As we argued, however, this technique is insufficient to deal with unpredictability, and cannot even be applied to the case of VRFs (because one cannot simulate the proofs of correctness for a truly random function).

1.1 Our Results

In this work we develop a new understanding of the Feistel Network which allows us to analyze the situations when when the intermediate round values may be leaked to the adversary, and also handle cases when the round values are merely unpredictable rather than pseudorandom. In our modeling, a k-round Feistel Network is applied to k members $f_1 \dots f_k$ independently selected from some (not necessarily pseudorandom) function family C, resulting in a Feistel permutation π. Whenever an attacker makes a forward (resp. backward) query to π (resp. π^{-1}), we assume that it learns all the intermediate values (as we mentioned, this is either required by the application, or may anyway happen with unpredictable functions).

NEGATIVE RESULT. As our first result, we show a simple attack allowing an adversary to compute any value $\pi^{-1}(y)$ by making at most exponential in k number of *forward* queries to π. Since such an inversion should be unlikely (with polynomially many queries) even for an unpredictable permutation, this immediately means that at least a superlogarithmic number of Feistel rounds (in the security parameter λ) is *necessary* to guarantee security for *any* of the applications we consider. Aside from showing the *tightness of all our positive results* described below, this result partially explains *why practical block ciphers use significantly more than 3-6 rounds* predicted by all the previous "theoretical justifications" of the Feistel Network. Indeed, since all such ciphers heuristically use round functions which are not PRFs, and we just showed that even unpredictable round functions might leak a lot (or even *all*) of the intermediate results, the simple attack we present might have been quite applicable if a small constant number of rounds was used!

MATCHING POSITIVE RESULT. On a positive side, we show a general combinatorial property of the Feistel Network which makes essentially no assumptions (such as pseudorandomness) about the round functions used in the Feistel construction, and allows us to apply it to a wide variety of situations described above, where the previous techniques (including that of [14]) failed. In essence, for any $s \leq k/2$, we show that if an attacker, making a subexponential in s number of (forward or backward) queries to the construction and always learning all the intermediate round values, can cause a non-trivial collision somewhere between rounds s and $k - s$, then the attacker can also find a simple (and non-trivial) XOR condition on a constant (up to six) number of the round values of the queries he has made. This means that if a function family C is such that it is provably hard for an efficient attacker to find such a non-trivial XOR condition, — and we call such families *5-XOR resistant* (see Section 4), — then it is very unlikely that the attacker can cause any collisions between rounds s and $k - s$ (as long as s, and thus k, are super-logarithmic in the security parameter λ). And once no such collisions are possible, we show that is possible to directly argue the security of the Feistel Network for our applications. In particular, as even mere unpredictability is enough to establish 5-XOR resistance, we conclude that super-logarithmic number of Feistel rounds is *necessary and sufficient* to yield

- a (strong) unpredictable permutation (UP) from any unpredictable function (UF).
- a strong PRP from any PRF, which remains secure even if all the round values are made public.
- a strong VUP from any VUF.
- a strong VRP from any VRF.

These results are in sharp contrast with the "LR-type" results where a constant number of rounds was sufficient, but also give the first theoretical justification regarding the usage of Feistel Networks not satisfying assumptions (a) or (b) mentioned earlier. For the case of block ciphers, our justification seems to match more closely the number of rounds heuristically used in practical constructions.

IMPLICATIONS TO DOMAIN EXTENSION. Since the Feistel Network doubles the length of its input, our results could also be viewed in relation to the question of domain extension of UFs, VUFs and VRFs. In practice, the question of domain extension is typically handled by a collision-resistant hash function (CRHF): it uses only one call the the underlying n-bit primitive f and does not require the secret key to grow. However, the existence of a CRHF is a theoretically strong assumption, which does not seem to follow from the mere existence of UFs, VRFs or VUFs. This is especially true for UFs, whose existence follows from the existence of mere one-way functions and, hence, can even be "black-box separated" from CRHFs [40]. Thus, it makes sense to consider the question of domain extension *without introducing new assumptions*.

For PRFs, this question is easily solved by using (almost) universal hash functions (instead of CRHFs) to hash the message to n bits before applying the n-bit PRF. However, this technique fails for UFs, VUFs and VRFs: in the case of unpredictability because the output reveals information about the hash key, and for VRFs because it is unclear how to provide proofs of correctness without revealing the hash key. Another attempt (which works for digital signatures) is to use target collision-resistant hash functions [37] in place of CRHFs, but such functions have to be freshly chosen for each new input, which will break the unique provability of UFs, VUFs and VRFs. (Additionally, the hash key should also be authenticated, which further decreases the bandwidth). In case the underlying n-bit primitive f is shrinking (say, to $n - a$ bits), one can use some variant of the cascade (or Merkle-Damgård) construction. Indeed, this was formally analyzed for MACs by [1, 31]. However, the cost of this method is one evaluation of f per a input bits. In particular, in case the output of f is also equal to n, which is natural if one wants to extend the domain of a UF given by a block cipher, this method is either inapplicable or very inefficient.[1]

In contrast, our method builds a UP/VUP/VRP from $2n$ to $2n$ bits from the one from n to n bits, by using $k = \omega(\log \lambda)$ evaluations of f, albeit also at the price of increasing the secret key by the same amount. This answers the question left open by An and Bellare [1] (who only showed that three rounds are insufficient): *Feistel Network is a good domain extender for MACs if and only if it uses super-logarithmic number of rounds*!

Moreover, in the context of UFs (and VUFs), where one wants to minimize the output length as well, we notice that the output length can be easily reduced from $2n$ to n. This is done by simply dropping the "left half" of the

[1] In principle, such length-preserving f can be "truncated" by a bits, but this loses an exponential factor in a in terms of exact security. Thus, to double the input length, one would have to evaluate f at least $\Omega(n/\log \lambda)$ times.

Feistel permutation output! The justification for this optimization follows by noticing that in this case the attacker will only make forward queries to the Feistel construction. For such attackers, we can extend our main combinatorial lemma as follows. For any $s \leq k$, if a 5-XOR resistant family is used to implement the round functions and the attacker made less than exponential in s number of queries, then the attacker has a negligible chance to cause any collisions between rounds s and k (as opposed to $k - s$ we had when backward queries were allowed). From this, one can derive that $k = \omega(\log \lambda)$ Feistel rounds is enough to turn a UF (or VUF) from n to n bits into one from $2n$ to n bits. Moreover, in the case of UFs we expect that one would use a (possibly heuristic) pseudorandom generator to derive the k round keys (much like in the case of block ciphers), meaning that the only effective cost is k computations of the basic UF. Once the domain is doubled, however, one can use the cascade methods [1, 31] to increase it further without increasing the key or the output length.

OTHER APPLICATIONS. As a simple, but illustrative application, we notice that VRPs immediately yield non-interactive, setup-free, perfectly-binding commitments schemes. The sender chooses a random key pair (SK, PK) for a VRP π. To commit to m (in the domain of the VRP), the sender sends PK and the value $c = \pi_{SK}(m)$ to the receiver. To open m, the sender sends m and the proof that $c = \pi_{SK}(m)$, which the receiver can check using the public key PK. The hiding property of this construction trivially follows for the security of VRPs. As for binding, it follows from the fact that π is a permutation even for an *adversarial choice of* PK. As we can see, it is not clear how to achieve binding *directly* using plain VRFs. However, given our (non-trivial) equivalence between VRFs and VRPs, we get that VRFs are also sufficient for building non-interactive, perfectly binding commitment schemes without setup. Alternatively, to commit to a single bit b, one can use VUPs augmented with the Goldreich-Levin bit [22]. Here the sender would pick a random r and x, and send PK, r, $\pi_{SK}(x)$, and $(x \cdot r) \oplus b$, where $x \cdot r$ denotes the inner product modulo 2. Using our equivalence between VUPs and VUFs, we see that VUFs are sufficient as well.

We remark that the best general constructions of such commitments schemes was previously based on one-way permutations (using the hardcore bit) [8], since Naor's construction from one-way functions [34] is either interactive, or non-setup-free. Since the assumption of one-way permutations is incompatible with VUFs or VRFs, our new construction is not implied by prior work.

In Section 6 we illustrate many other applications of our results. For example, UPs are enough to argue weaker "fall-back" security properties for some applications of block ciphers, which is nice in case the PRP assumption on the block cipher turns out incorrect. VRPs, or sometimes even VUPs, can be useful in several applications where plain VRFs are insufficient. For example, to implement so called "invariant signatures" needed by Goldwasser and Ostrovsky [23] in constructing non-interactive zero-knowledge proofs, or to fix a subtle security flaw in the non-interactive lottery system of Micali and Rivest [33] (which can be extended into a minimally-interactive "reusable coin-flipping protocol"). Additionally, VRPs could be useful for adding verifiability to some application of PRPs (where, again, PRFs are not sufficient). For example, to construct verifiable CBC encryption or decryption, or to "truthfully", yet efficiently, sample certain verifiable huge (pseudo)random objects [21], such as random constant-degree expanders. Finally, our construction of VRPs from VRFs could lead to a "proof-transferable" implementation of the Ideal Cipher Model using a semi-trusted third party. We refer to Section 6 for more details, and hope that more applications of our constructs and techniques will be found.

2 Definitions and Preliminaries

Let λ denote the security parameter. We use $negl(\lambda)$ to denote a negligible function of λ. $\mathsf{Fibonacci}(k)$ denotes the k^{th} fibonacci number, and thus $\mathsf{Fibonacci}(k) = \mathcal{O}(1.618^k)$.

Now we give informal definitions of the various primitives that we use in this paper. For formal definitions, see appendix A. We start by defining the notion of *pseudorandom functions* (PRFs). We use a slightly non-standard definition of PRFs that is convenient to prove our results. However, this definition is equivalent to the usual definition.

In the new PRF attack game, the attacker A_f runs in three stages: (1) In the experimentation phase, it is allowed to query a PRF sampled from the PRF family. (2) In the challenge phase, it sends an unqueried PRF query and in response the challenger sends either the PRF output or a random output with equal probability. (3) In the analysis phase, the attacker again gets oracle access to the PRF, but cannot query it on the challenge query. At the end of the attack, A_f has to guess if the challenge response was random or pseudorandom. The attacker A_f wins if it guesses correctly. Similar to the notion of PRFs, we can define the notion of *(strong) pseudorandom permutations* (PRPs). Here the attacker has oracle access to both the forward as well as inverse PRP, but the attack game is otherwise similar to that for PRFs.

A slightly weaker notion than PRFs is that of *Unpredictable Functions* (UFs). Unpredictable functions are also popularly known as (deterministic) *Message Authentication Codes* (MACs). In this case, the UF attacker is allowed to query an unpredictable function from the UF family, and it needs to predict the output of the UF on an unqueried input at the end of the interaction. The advantage of the UF adversary is the maximum probability with which it predicts correctly. In an analogous fashion, we can also define the notion of *Unpredictable Permutations* (UPs), where the attacker has oracle access to both the forward and inverse permutation and has to predict an unqueried input/output pair.

We can define verifiable analogs of each of the above primitives. Thus, we get *verifiable random functions*, *verifiable random permutations*, *verifiable unpredictable functions* and *verifiable unpredictable permutations*. In each case, the primitive takes a public/private key pair, and consists of three algorithms $(\mathsf{Gen}, \mathrm{Prove}, \mathrm{Verify})$. The Gen algorithm outputs a public/private key pair. The Prove algorithm allows the private key owner to compute the function/permutation output as well as give a proof of correctness. Finally, the Verify algorithm allows anyone who knows the public key to verify the correctness of an input/output pair by observing the corresponding proof.

Each of these primitives satisfies two properties: (1) *Correctness*, i.e. one can verify correct input/output pairs, and (2) *Soundness*, i.e. there are no two output/proof pairs that verify correctly for the same input, even for an *adversarially chosen public key*. Additionally, these primitives satisfy the natural analogs of the pseudorandomness/unpredictability definition of the corresponding non-verifiable primitive (except the attacker also gets the proofs for all the values except for the challenge).

The *Feistel transformation* using $f : \{0,1\}^n \to \{0,1\}^n$ is a permutation Ψ_f on $2n$ bits defined as, $\Psi_f(x) \stackrel{def}{=} x_R \parallel x_L \oplus f(x_R)$. The symbols x_L and x_R denote the left and right halves of $2n$ bit string x. We will also call the construction based on k iterated applications of the Feistel transformation, a k-round LR construction, and denote it by $\Psi_{f_1 \ldots f_k}$ (or Ψ_k when $f_1 \ldots f_k$ are clear from context) where $f_1 \ldots f_k$ are the round functions used. On a $2n$ bit input, the construction Ψ_k generates $(k+2)$ n-bit round values, the last two of which form the output.

3 Insecurity of $\mathcal{O}(\log \lambda)$-round Feistel

We will demonstrate here that upto a logarithmic number of Feistel rounds do not suffice for any of our results. In order to make our proof precise, we show a simple adversary that is able to find the input corresponding to any permutation output $y \in \{0,1\}^{2n}$ by making polynomially many *forward* queries and observing the intermediate round values.

Theorem 3.1 *For the k round Feistel construction Ψ_k that uses $k = \mathcal{O}(\log \lambda)$ round functions, there exists a probabilistic polynomial time adversary A_π that takes oracle access to Ψ_k. The adversary A_π makes $\mathcal{O}(\mathsf{Fibonacci}(k)) =$ poly(λ) forward queries to Ψ_k and with high probability finds the input corresponding to an output y without actually making that query.*

Proof: The adversary A_π starts by choosing a permutation output y, that it will try to invert Ψ_k on. For concreteness, we assume that $y = 0^{2n}$ (anything else works just as well). We will describe the recursive subroutine that the attacker A_π is based on. Say the round functions of Ψ_k are $f_1 \ldots f_k$. The recursive function that we describe is $E(j, Y)$, where j is the number of rounds in the Feistel construction and Y is a $2n$ bit value, and the task of $E(j, Y)$ is to find the input such that the j^{th} and $(j+1)^{th}$ round values are Y_L and Y_R (the left and right halves of Y), respectively.

- $\mathbf{E(1,Y)}$: Choose a random $R'_0 \leftarrow \{0,1\}^n$. Make the forward query $R'_0 \parallel Y_L$ to Ψ_1, where the 2^{nd} round value is R'_2. Now the 1^{st} and 2^{nd} round values for the input $R'_2 \oplus R'_0 \oplus Y_R \parallel Y_L$ are Y_L and Y_R.
- $\mathbf{E(j,Y)}$, $\mathbf{j > 1}$: Perform the following steps,
  - Make a random query $R_0 \parallel R_1 \leftarrow \{0,1\}^{2n}$, and say the $2n$ bit value at the j^{th} round is is $R_j \parallel R_{j+1}$. Then, $f_j(R_j) = (R_{j-1} \oplus R_{j+1})$.
  - Run $E(j-2, (f_{j-1}(R_{j-1}) \oplus Y_L) \parallel R_{j-1})$ and the $2n$ bit value at the $(j-1)^{th}$ round is $R_{j-1} \parallel Y_L$. Hence $f_j(Y_L) = R_{j-1} \oplus R_{j+1}$.
  - Run $E((j-1), (f_j(Y_L) \oplus Y_R) \parallel Y_L)$, and the j^{th} and $(j+1)^{th}$ round values are Y_L and Y_R, respectively.

The adversary A_π essentially runs the algorithm $E(k, 0^{2n})$. Now we need to make sure that the adversary A_π does not query on the input corresponding to the output 0^{2n}. But since all the queries made in the recursive algorithm are essentially chosen at random, we know that the probability of this happening is $\frac{q}{2^{2n}}$. Hence, the probability that A_π succeeds is at least $\left(1 - \frac{q}{2^{2n}}\right)$. $\square$

We note that the above attacker works in a scenario where it can only make forward queries to the Feistel construction Ψ_k. In case, it can make inverse queries as well, it is possible to design a similar attacker that succeeds in $\mathcal{O}(\mathsf{Fibonacci}(k/2))$ queries. If the number of rounds $k = \mathcal{O}(\log \lambda)$, then the number of queries needed by either of these attackers is polynomial in the security parameter λ.

It is easy to see how such an attacker can be utilized in three of the four scenarios, if we use the Feistel construction for each of these cases.

- *PRP construction with public round values*: By definition, for a PRP we should not be able to invert an output without actually querying the construction on it.
- *VRP construction using VRFs*: In order to provide the proofs for the VRP, the VRP construction will need to reveal all intermediate VRF inputs/outputs and the corresponding proofs.
- *VUP construction using VUFs*: In this case, again the VUP construction will need to reveal all the intermediate VUF inputs/outputs and corresponding proofs.

On the first look, it seems that when we use a Feistel construction with *unpredictable functions* in each round to construct an *unpredictable permutation* (UP), the UP adversary cannot make use of the above attacker since it does not have access to all the intermediate round values. However, we will show that if certain pathological (but secure) unpredictable functions are used as round functions, then the UP adversary can infer *all* the round values simply by observing the output of the Feistel construction!

Lemma 3.2 *For any $k \leq \frac{n}{\omega(\log \lambda)}$ (in particular, if $k = \mathcal{O}(\log \lambda)$), there exist k secure unpredictable functions $f_1 \dots f_k$, such that by querying the k-round Feistel construction $\Psi_{f_1 \dots f_k}$ on any input an efficient attacker can always learn all intermediate round values.*

Proof: Let $\{g_i : \{0,1\}^n \to \{0,1\}^{n/k}\}_{i \in \{1 \dots k\}}$ be k secure unpredictable functions. For $i \in \{1, k\}$, we will define the functions $f_i : \{0,1\}^n \to \{0,1\}^n$ as $f_i(x) = 0^{(i-2)\cdot(n/k)} \parallel x_{i-1} \parallel g_i(x) \parallel 0^{(k-i)\cdot(n/k)}$, where x_{i-1} denotes the $(i-1)^{th}$ (n/k) bit block in the input x. Each of the functions f_i is a secure unpredictable function if the corresponding function g_i is a secure UF.

Consider a query $(R_0 \parallel R_1) \in \{0,1\}^{2n}$ made to the Feistel construction $\Psi_{f_1 \dots f_k}$. We will consider k blocks of (n/k) bits each in both R_0 and R_1, which we will denote by $R_0 = R_0^1 \parallel \dots \parallel R_0^k$ and $R_1 = R_1^1 \parallel \dots \parallel R_1^k$. Denote the round values generated in computing the output of this construction as $(R_0, R_1) \dots (R_k, R_{k+1})$, where

$R_k \parallel R_{k+1}$ is the output of this construction. If the number of rounds in the Feistel construction is even, then we note that the output of the construction is:

$$\begin{aligned} R_k &= (g_1(R_1) \oplus R_0^1 \oplus R_1^1) \parallel \ldots \parallel (g_{k-2}(R_{k-2}) \oplus R_0^{k-2} \oplus R_1^{k-2}) \parallel (g_{k-1}(R_{k-1}) \oplus R_0^{k-1}) \parallel R_0^k \\ R_{k+1} &= (g_1(R_1) \oplus R_0^1 \oplus R_1^1) \parallel \ldots \parallel (g_{k-1}(R_{k-1}) \oplus R_0^{k-1} \oplus R_1^{k-1}) \parallel (g_k(R_k) \oplus R_1^k) \end{aligned}$$

If number of rounds k is odd, then the output of the Feistel construction is,

$$\begin{aligned} R_k &= (g_1(R_1) \oplus R_0^1 \oplus R_1^1) \parallel \ldots \parallel (g_{k-2}(R_{k-2}) \oplus R_0^{k-2} \oplus R_1^{k-2}) \parallel (g_{k-1}(R_{k-1}) \oplus R_1^{k-1}) \parallel R_1^k \\ R_{k+1} &= (g_1(R_1) \oplus R_0^1 \oplus R_1^1) \parallel \ldots \parallel (g_{k-1}(R_{k-1}) \oplus R_0^{k-1} \oplus R_1^{k-1}) \parallel (g_k(R_k) \oplus R_0^k) \end{aligned}$$

Now it is easy to find each of the round function outputs (and hence the intermediate round values) by simply observing the right half of the output of the Feistel construction. ◻

Thus, we see that if the number of rounds in the Feistel construction (using UFs) used to construct *unpredictable permutations* is $k = \mathcal{O}(\log \lambda)$, then the resulting construction is insecure. Even if we attempt to shrink the output length of this MAC construction by chopping the left half of the output, it would be possible to retrieve all intermediate round values by simply observing the MAC output. In fact, even for $k = \omega(\log \lambda)$ (but less than $n/\omega(\log \lambda)$) rounds it might be possible to retrieve all intermediate round values, and hence a new proof technique is needed.

4 A Combinatorial Property of the Feistel Construction

In this section, we will prove a general combinatorial lemma about the k round LR-construction Ψ_k, that uses arbitrary round functions $f_1 \ldots f_k$. We will see in the following section that this lemma is crucial in deriving each of our results using the Feistel construction.

Consider an arbitrary ordered sequence of q forward/inverse permutation queries made to the construction Ψ_k, each of which is a $2n$ bit string. Denote the $(k+2)$ n-bit round values associated with the i^{th} query as $R_0^i, R_1^i \ldots R_k^i$, R_{k+1}^i, where $R_0^i \parallel R_1^i$ ($R_k^i \parallel R_{k+1}^i$) is the input if this is a forward (inverse) query. We say that such a sequence of queries produces an s^{th} round value collision, if the s^{th} round value collides for two different permutation queries from this query sequence. That is, we have that $R_s^i = R_s^j$ for $i, j \in \{1 \ldots q\}$ and $R_0^i \parallel R_1^i \neq R_0^j \parallel R_1^j$.

We essentially show that if any such sequence of q queries produces a r^{th} round value collision for any $r \in \{s \ldots (k-s)\}$ (where $s \leq (k/2)$), then one of the following must hold:

1. The number of queries q is exponential in s.
2. For this sequence of queries, there is at least one new round function evaluation such that the new round value generated can be represented as a bit-by-bit XOR of upto 5 previously existing round values.

We refer to the second condition above as the 5-*XOR condition*. We assume the natural order in which the queries are made, i.e. query i is made before query $i+1$ for $i = 1 \ldots q-1$. By a "new round function evaluation", we mean when a round function is evaluated on an input (i.e. the corresponding round value) to which it was not applied in an earlier query. If the i^{th} query is a forward (inverse) query and the round function evaluation $f_j(R_j^i)$ is a new one, then the new round value generated as a result is R_{j+1}^i (R_{j-1}^i resp.). The 5-XOR condition essentially states that for at least one such new round function evaluation, the new round value generated can be represented as the bit-by-bit XOR of upto 5 previously existing round values. This is formalized in the main lemma below (where, for future convenience, we denote R_j^i by $R[i,j]$).

Lemma 4.1 *Let Ψ_k be a k round LR construction that uses fixed and arbitrary round functions $f_1 \ldots f_k$. For any $s \leq \frac{k}{2}$, and any ordered sequence of $q = o(1.3803^{\frac{s}{2}})$ forward/inverse queries, with associated round values $R[i,0], \ldots, R[i,k+1]$ for $i = 1 \ldots q$, if the 5-XOR condition does not hold for this sequence of query then there is no r^{th} round value collision for these queries, for all $r \in \{s \ldots (k-s)\}$.*

We will describe here the basic intuition underlying lemma 4.1, leaving the formal proof for appendix B.

Proof Intuition: We will show that if for the given sequence of queries, the 5-XOR condition does not hold but it produces a r^{th} round value collision, then the number of queries $q = \Omega(1.3803^{\frac{\min(r,(k-r))}{2}})$, which will settle the lemma since $r \in \{s \ldots (k-s)\}$ will imply that $q = \Omega(1.3803^{\frac{s}{2}})$.

Without loss of generality, let one of the queries that are involved in the r^{th} round value collision be the last (or q^{th}) query. If not, then we can consider a smaller sequence of queries for which this holds. For simplicity of exposition, we assume here that all the queries in this sequence are forward queries. (In the formal proof given in appendix B, the query sequence may be comprised of both forward or inverse queries.) If this is the case, then we will show that the number of queries $q = \Omega(1.3803^{r/2})$. We denote by $\mathsf{p}(i,j)$ the query number where the round value $R[i,j]$ occurs for the first time as the j^{th} round value of a query.

Our main argument consists of four main steps which all rely on the fact that the 5-XOR condition does not hold for the given sequence of queries. We start by showing that if the round value $R[q,r]$ collides with the r^{th} round value in an earlier query, then all of the round values $R[q,1] \ldots R\,[q, r-1]$ also collide with corresponding round values in earlier queries. That is,

$$\mathsf{p}(q,r) < q \Rightarrow (\mathsf{p}(q,1) < q) \wedge \ldots \wedge (\mathsf{p}(q,(r-1)) < q)$$

Next, we show that not only were the queries $\mathsf{p}(q,1) \ldots \mathsf{p}(q,r)$ made before the q^{th} query, but these queries could have been made in only certain specific orders. In particular, we show that there is a $j \in \{1 \ldots r\}$ such that

$$\mathsf{p}(q,1) > \ldots > \mathsf{p}(q,j) < \ldots < \mathsf{p}(q,k/2)$$

In the third step, we choose one of the strictly descending/ascending query sequence, $\mathsf{p}(q,1) \ldots \mathsf{p}(q,j)$ or $\mathsf{p}(q,j) \ldots$ $\mathsf{p}(q,r)$, whichever consists of a greater number of queries. Without loss of generality, say the longer sequence is $\mathsf{p}(q,1) > \ldots > \mathsf{p}(q,j)$. We show that for each of the queries $\mathsf{p}(q,\ell)$ for $\ell \in \{1 \ldots (j-2)\}$, all the round values $R[\mathsf{p}(q,\ell),1] \ldots$
$R[\mathsf{p}(q,\ell),\ell-1]$ collide with the corresponding round value in an earlier query. This step is essentially the same as the first one, where we considered the q^{th} query. Thus, we show that

$$(\mathsf{p}(\mathsf{p}(q,\ell),1) < \mathsf{p}(q,\ell)) \wedge (\mathsf{p}(\mathsf{p}(q,\ell),\ell-1) < \mathsf{p}(q,\ell))$$

In the fourth step, we show that the queries $\mathsf{p}(\mathsf{p}(q,\ell),1) \ldots \mathsf{p}(\mathsf{p}(q,\ell),\ell-1)$ occur in a strictly descending order. Additionally, we also show that the first $\ell-2$ of these queries occur after query number $\mathsf{p}(q,\ell+1)$. Combining this all together, we show that for each $\ell = 1 \ldots j-2$,

$$\mathsf{p}(q,\ell+1) < \mathsf{p}(\mathsf{p}(q,\ell),\ell-2) < \ldots < \mathsf{p}(\mathsf{p}(q,\ell),1) < \mathsf{p}(q,\ell)$$

Finally, we notice that the last two steps can be applied recursively to the sequence of queries $\mathsf{p}(\mathsf{p}(q,\ell),\ell-2) <$ $\ldots < \mathsf{p}(\mathsf{p}(q,\ell),1)$. And in each such recursive analysis, we also show that the queries whose existence is proved in one step lie strictly in between two consecutive queries from the previous step, e.g. $\mathsf{p}(\mathsf{p}(q,\ell),\ell-2) \ldots \mathsf{p}(\mathsf{p}(q,\ell),1)$ occur between $\mathsf{p}(q,\ell+1)$ and $\mathsf{p}(q,\ell)$. Thus every query whose existence we show is distinct from all the queries that we have already shown to exist. Since all these queries occur before query number q, we get that

$$q \quad \geq \quad \mathcal{Q}\,(r/2)\,,\text{ where }\ \mathcal{Q}\,(i) = i + \sum_{\ell=2}^{i-2} \mathcal{Q}\,(\ell-2) = 2\cdot\mathcal{Q}(i-1) - \mathcal{Q}(i-2) + \mathcal{Q}(i-4) - \mathcal{Q}(i-5)$$

Upon solving this recurrence, we get that $q = \Omega(1.3803^{r/2})$. When both forward/inverse queries are permitted then as shown in appendix B, the number of queries $q = \Omega(1.3803^{\frac{\min(r,(k-r))}{2}})$. $\square$

Next we state here an alternative combinatorial lemma, when the adversary only makes forward queries to the Feistel construction. This lemma (whose proof is similar to the above and is omitted) will prove useful when we attempt *domain extension of MACs* in the next section.

Lemma 4.2 *Let Ψ_k be a k round LR construction that uses fixed and arbitrary round functions $f_1 \ldots f_k$. For any round number s, and any ordered sequence of $q = o(1.3803^{\frac{s}{2}})$ forward queries, with associated round values $R[i, 0], \ldots, R[i, k+1]$ for $i = 1 \ldots q$, if the 5-XOR condition does not hold for this sequence of forward queries then there is no r^{th} round value collision for these queries, for all $r \geq s$.*

In our applications, we will be interested in using the LR construction with round functions that resist the 5-XOR condition, when any adaptive adversary makes a polynomial number of queries to the construction while having access to all intermediate round values. We will specify this as a property of families of functions from which the round functions are independently derived. Hence, let us begin by describing a *function family*. A *function family* C is a set of functions along with a distribution defined on this set. For such a family, $f \leftarrow C$ denotes sampling a function according to the distribution specified by C. A function family is called a *5-XOR resistant function family* if the LR construction using independently sampled functions from this family resists the 5-XOR condition when queried a polynomial number of times by any adaptive adversary.

Definition 1 (5-XOR resistant function family) *A function family $C_{(k,n)}$, that consists of length preserving functions on n bits, is a 5-XOR resistant function family if for any adversary A,*

$$\Pr\left[\text{A 5-XOR condition holds in } (A \longleftrightarrow \Psi_{f_1 \ldots f_k}) \;\middle|\; f_1 \ldots f_k \longleftarrow C_{(k,n)}\right] \leq \epsilon_{xor} = negl(\lambda)$$

Here the advantage ϵ_{xor} of the adversary A depends on the running time of A and the security parameter λ. The running time of A, the input length n and number of Feistel rounds k are all polynomial functions of λ.

By applying lemma 4.1 to a LR construction using round functions independently sampled from a 5-XOR resistant function family, we can derive the following corollary.

Corollary 4.3 *Let Ψ_k be a k-round LR construction that uses round functions that are independently sampled from a 5-XOR resistant function family consisting of functions on n bits. For any adversary A that adaptively makes permutation queries to Ψ_k, while observing the intermediate round values, it holds that*

- *if A makes both forward/inverse queries, then for any round number $s \leq (k/2)$ with $s = \omega(\log \lambda)$,*
$$\Pr\left[\exists\, r^{th} \text{ round value collision during } A \leftrightarrow \Psi_k \text{ for some } r \in \{s \ldots (k-s)\}\right] \leq \epsilon_{xor}$$
- *if A makes only forward queries, then for any round number $s = \omega(\log \lambda)$,*
$$\Pr\left[\exists\, r^{th} \text{ round value collision during } A \leftrightarrow \Psi_k \text{ for some } r \in \{s \ldots k\}\right] \leq \epsilon_{xor}$$

Here the bound ϵ_{xor} denotes the maximum advantage of the XOR finding adversary that runs in time $\mathcal{O}(t_A + (q_A k)^5)$, where t_A is the running time of the adversary A and q_A denotes the number of queries made by it. Also, t_A, q_A and the input length n are all polynomial in λ.

This corollary is easily proved since the 5-XOR finding adversary simply runs the collision finding adversary, and performs a brute-force search for a 5-XOR condition when it finds a round value collision. From lemma 4.1, such a 5-XOR condition is guaranteed to exist. In fact, we will make use of this corollary in each of the results that we present in the next section, since each of these function families will turn out to be 5-XOR resistant (the proof of this will also be given in the appropriate subsection; here we just state the result).

Theorem 4.4 *For each of the primitives: (1)* unpredictable functions*, (2)* pseudorandom functions*, (2)* verifiable unpredictable functions*, and (4)* verifiable random functions*; a function family that yields an independent random sample of the appropriate primitive is a 5-XOR resistant function family.*

5 Implications

All the cryptographic applications of the Feistel construction until recently have relied on all or some of the round functions not being visible to the adversary. In the previous section, we proved a combinatorial property of the Feistel construction where the internal round function values were visible to the adversary. Now we will describe how this property can be applied to a variety of scenarios to yield new or improved cryptographic constructions than before.

We get the following constructions using this new technique: (1) secure construction of *unpredictable permutations* from *unpredictable functions*, (2) more resilient construction of *pseudorandom permutations* from *pseudorandom functions*, (3) construction of *verifiable unpredictable permutations* from *verifiable unpredictable functions*, and (4) construction of *verifiable random permutations* from *verifiable random functions*.

In each case, the proof consists of three parts: (1) showing that the function family under consideration is a 5-XOR function family (see Theorem 4.4); (2) using Corollary 4.3 to show that the corresponding permutation construction is unlikely to have collisions at "advanced" rounds; and (3) show that the lack of such collisions implies that the construction is secure.

5.1 Unpredictable Permutations

We saw in the section 3 that observing the output of a $k = n/\omega(\log \lambda)$ round Feistel construction with unpredictable round functions may leak all the intermediate round values. Even for realistic UFs, some partial information about the intermediate round values may be leaked through the output. As we discussed earlier, in such a case none of the previous proof techniques are applicable. We will prove a much stronger result here, by showing that if we use a super-logarithmic number of rounds in the Feistel construction then the resulting UP construction is secure even if the adversary gets all the intermediate round values along with the permutation output.

The UP construction $\Psi_{U,k}$ that we propose consists of $k = \omega(\log \lambda)$ rounds of the Feistel construction using independent *unpredictable functions* $f_1 \dots f_k \leftarrow F$. The following theorem essentially states that this construction is a secure UP construction. The proof of this theorem can be found in appendix C.

Theorem 5.1 *If there exists an efficient UP adversary A_π that has non-negligible advantage ϵ_π in the unpredictability game against $\Psi_{U,k}$ and which makes a polynomial number of queries to $\Psi_{U,k}$, then there also exists a UF adversary A_f that has non-negligible advantage in the unpredictability game against a UF sampled from the UF family F. From this, we get that the maximum advantage of the UP adversary A_π is $\epsilon_\pi = \mathcal{O}\left(\epsilon_f \cdot (qk)^6\right)$. Here ϵ_f denotes the maximum advantage of a UF adversary running in time $\mathcal{O}(t + (qk)^5)$ against a UF sampled from F, where t is the running time of the PRP adversary A_π and q is the number of queries made by it.*

DOMAIN EXTENSION OF MACS. The above result can also be viewed as a construction of MACs from $2n$ to $2n$ bits using MACs from n to n bits. We observe that it is possible to reduce the output length in the above construction to n by simply dropping the left half of the output. Using this technique, we get a MAC construction from $2n$ to n bits. To briefly justify it, in the usual MAC attack game the attacker can only make forward queries. From corollary 4.3, we get that for any $s = \omega(\log \lambda)$ no efficient attacker can cause a collision on any round value $r \in \{s \dots k\}$ with non-negligible probability. Thus, a proof of security for this MAC will proceed by plugging in the target n- to n-bit MAC in the last round function of the Feistel construction, and arguing that the attacker predicting the $2n$- to n-bit constructed MAC must also forge this last-round n- to n-bit MAC. This is done using a similar proof technique to that for theorem 5.1 (albeit using second part of corollary 4.3 to argue that no collision occurs at the last round).

5.2 More Resilient PRPs from PRFs

In this section, we give a construction of *pseudorandom permutations* from *pseudorandom functions*, that remains secure even if the PRF input/output pairs used in the intermediate rounds are visible to an attacker. Our proposed

PRP construction, $\Psi_{R,k}$, is a $k = \omega(\log \lambda)$-round Feistel construction, with independent PRFs $f_1 \ldots f_k \leftarrow F$ as round functions. The following theorem states that this construction $\Psi_{R,k}$ is a secure PRP. The proof of this theorem can be found in appendix D.

Theorem 5.2 *If there exists an efficient PRP adversary A_π that has a non-negligible advantage ϵ_π in the PRP attack game against the construction $\Psi_{R,k}$ (using round function from PRF family F), then there also exists a PRF adversary A_f that has non-negligible advantage ϵ_f in the PRF attack game against a PRF sampled from the PRF family F. From this, we get a bound $\epsilon_\pi = \mathcal{O}\left(qk\epsilon_f + \frac{(qk)^6}{2^n}\right)$, where ϵ_f denotes the maximum advantage of a PRF adversary running in time $\mathcal{O}(t + (qk)^5)$ against a PRF sampled from F, and t, q are the running time and number of queries made by A_π.*

5.3 Verifiable Unpredictable Permutations

Our VUP construction $\Psi_{VU,k}$ is a k-round Feistel construction using independent VUFs $f_1 \ldots f_k \leftarrow F$ as round functions. The public/private keys of $\Psi_{VU,k}$ are simply the concatenation of the public/private keys of the k VUFs. The Prove functionality for $\Psi_{VU,k}$ simply gives the permutation output, and as proof, it gives all intermediate round values along with the VUF proofs. The Verify functionality simply checks if all intermediate VUF proofs verify correctly.

Recall that a VUP construction needs to satisfy three security properties : *Completeness*, *Soundness* (or unique proofs) and *Unpredictability*. Completeness of the construction $\Psi_{VU,k}$ is a direct consequence of completeness of each of the VUFs used as round functions. The *soundness* of the construction is also obvious given the fact that all the intermediate VUFs are sound. If there are two output/proof pairs of $\Psi_{VU,k}$ that verify correctly, then we can find two VUF output/proof pairs that verify correctly for one of the round functions. The *unpredictability* property follows very similarly to Theorem 5.1, which we used to prove the UF to UP construction.

Theorem 5.3 *Let $\Psi_{VU,k} = (G_\pi, \Pi, V_\pi)$ be the VUP construction using k rounds of the Feistel construction using independent VUFs $f_1 \ldots f_k \leftarrow F$. For any probabilistic polynomial time oracle machine A_π that does not make a forward query on x or an inverse query on y, the advantage of A_π in winning the VUP game against $\Psi_{VU,k}$ is at most $\mathcal{O}(q^6 k^7 \cdot \epsilon_f)$, where ϵ_f denotes the maximum advantage of a VUF adversary running in time $\mathcal{O}(t + (qk)^5)$ against a VUF sampled from F, t is the running time of A_π and q is the number of queries made by A_π.*

5.4 Verifiable Random Permutations

The VRP construction $\Psi_{VR,k}$ that we use is identical to the VUP construction $\Psi_{VU,k}$ described above, except that we use independent VRFs instead of VUFs. The *completeness* and *soundness* properties of the VRP construction $\Psi_{VU,k}$ can be proven similar to that for the VUP construction $\Psi_{VR,k}$.

The *pseudorandomness* property of the VRP construction $\Psi_{VR,k}$ can be proven in a way similar to the proof for the PRP construction $\Psi_{R,k}$ in theorem 5.2. Thus, we get that

Theorem 5.4 *Let $\Psi_{VR,k} = (G_\pi, \Pi, V_\pi)$ be the VRP construction using a k-round Feistel construction using independent VRFs $f_1 \ldots f_k \leftarrow F$. For any probabilistic polynomial time oracle machine $A_\pi = (A_1, A_2)$ that does not query its oracle on x or try to invert the response to the challenge query, the advantage of A_π in winning the VRP game against $\Psi_{VR,k}$ is at most $\mathcal{O}\left(qk\epsilon_f + \frac{(qk)^6}{2^n}\right)$, where ϵ_f denotes the maximum advantage of a VRF adversary that runs in time $\mathcal{O}(t + (qk)^5)$ against a VRF sampled from F, and t and q are the running time and number of queries made by A_π.*

6 Other Applications

MORE RESILIENT BLOCK CIPHERS. Although not as strong as pseudorandomness, unpredictability is a meaningful property of block ciphers. First, we already mentioned that it is enough for message authentication, and our Feistel construction is also useful in the context of domain extension of MACs. We notice that it is also enough to argue certain weaker properties of popular modes of operation on block ciphers. For example, one can easily argue that the CBC mode with UPs (rather than PRPs) yield a "computationally ϵ-universal" hash function [2], which can then be used with an ordinary block cipher to get a secure "encrypted CBC-MAC". Even in the context of encryption, one can argue that CBC, OFB and CFB modes with UPs satisfy the following form of one-wayness against the usual chosen message attack. The attacker can ask encryptions or any messages. For the challenge, it specifies any message with one missing block. Then this block is chosen at random, and the encryption of the entire message (using the corresponding mode) is given to the attacker. Finally, the attacker has to recover this missing block, and using UPs guarantees that the attacker only has a negligible probability to succeeded in this game.

To summarize, the usage of UPs in place of PRPs still maintains weaker, but still meaningful security properties. Therefore, we see their primary utility as a way for providing a "graceful fall-back" property for the Feistel construction. If (nearly) pseudorandom round functions are used, then with $\omega(\log \lambda)$ rounds the resulting permutation is a PRP. As a bonus, it remains a PRP even if the intermediate round values could be leaked! Additionally, even if the round functions are only unpredictable, we still have some basic security left, so at the very least the system will not be "completely broken".

IDEAL CIPHER MODEL USING SEMI-HONEST TRUSTED PARTY. The *Ideal Cipher Model* (ICM) (also known as the "Shannon Model") assumes the existence of a publicly accessible Ideal Block Cipher, meaning that for every possible key s one has a fresh random permutation Π_s and its inverse Π_s^{-1}. Although the ICM is not as popular as the random oracle model, there are still several notable examples of schemes where this model has been used [7, 12, 17, 25, 26]. Unfortunately, just like the random oracle model, the ICM model cannot be provably realized without a trusted party T (see [6]). A naive implementation is easy, but inconvenient. First, T should keep track of all the queries already asked to ensure consistency, which quickly becomes very impractical. Second, the parties must trust that T has evaluated the value $\Pi_s(x)$ consistently across invocations. Third, once they get such a value, they cannot convince any other party of its validity: that party must independently go to T to check the correctness. Finally, they must trust that the answers of T are actually random.

It turns out that a VRP can considerably improve this naive implementation. First, we start with implementing a single truly random permutation Π (corresponding to an ideal cipher with a fixed key). Then T can publish the public key for a VRP π, and only keep the secret key as its state. When some party comes to T and asks a forward or backward query to Π, T simply evaluates π or π^{-1} on that query, and returns the result together with a proof of correctness. This way the parties are assured that: (a) they receive a correct and *consistent* value of Π; (b) they are really talking to T (or, if not, the value is correct anyway); (c) once T is committed to the public key, T cannot dynamically adjust the values of Π and Π^{-1}; (d) even if T selected a bad public key, T is committed to a *permutation*; in particular, the value of Π on a *random* point is guaranteed to be random. Finally, once somebody gets a value of Π or Π^{-1} from T, it can transfer this value on its own, without the need of other parties to come to T and verify it.

To extend this to a full blown Ideal Cipher, we face a problem that T must generate a new VRP for every key s of the Ideal Cipher. However, for our particular VRF-based construction we can do better. Instead of assuming the existence of a VRF from n to n bits, we assume the existence of a VRF from $n + a$ to n bits, where a is the length of the key s (if needed, such VRF can always be constructed from another VRF using the domain extension techniques we developed earlier). In this case, T will always prepend the key s to all the VRFs inputs when evaluating the Feistel Network for the value of Π_s. This way T still stores only $\omega(\log \lambda)$ keys for the VRFs, and can emulate 2^a possible random ciphers.

Next, we mention several examples how VRPs could be useful in scenarios where plain VRFs are not enough.

NON-INTERACTIVE ZERO-KNOWLEDGE (NIZK). We show that VRPs (and, thus, indirectly, VRFs), could be used to construct NIZK proofs (in the common reference string model). We remark, however, that Dwork and Naor [16] already gave a completely different construction of NIZK proofs from VRFs (and even a weaker primitive called *verifiable pseudorandom generator*). Thus, our construction only gives an alternative (and different) proof of an already known result by [16]. Nonetheless, we believe that it naturally illustrates the usefulness of VRPs in comparison to VRFs, and also solves a question left open by Goldwasser and Ostrovsky [23] (see below).

Feige et al. [18] reduced the question of constructing NIZK proofs (in the common reference string model) to the question of implementing the so called "hidden bits system" HBS, and showed how to implement HBS using trapdoor permutations. Later, Goldwasser and Ostrovsky [23] showed how to implement HBS using so called *invariant signatures*. In our modern terminology, invariant signatures are quite similar to VRFs, except for one additional requirement: they should induce a (pseudo)random distribution on the output when applied to a random input, *even if the public key for the VRF is adversarially chosen*. Thus, we can think of *invariant signatures* as "balanced" VRFs. Unfortunately, it is easy to see that regular VRFs are not enough to plug into the construction of [23]. Namely,

(a) Plain VRFs do not have to satisfy this property (and, as far as we can see, there is no trivial way to enforce it in VRFs; although, our results imply a non-trivial way to do so).
(b) More severely, there exist secure (and, of course, unbalanced) VRFs for which the transformation of [23] is completely insecure.

To briefly see point (a), imagine adding a new special public key PK^* to any secure VRF, for which the VRF is defined to be identically zero. It is clear that this still defines a VRF, since the prover is still committed to a unique function, even for the key PK^*. And pseudorandomness holds, since the chances PK^* will be selected are negligible. Yet, the new VRF is obviously unbalanced. In fact, if we use this new VRF in place of the invariant signature in the construction of [23], we will get a completely insecure HBS system (thus, showing (b)). Briefly, in the construction of [23] a VRF *selected by the prover* is applied to a bunch or random points to define the "hidden random string" (for which the prover can selectively open some part later). If the prover chooses PK^* as his public key, then the hidden random string is all zero as well, and it is easy to see that NIZK construction of [18] will completely fails with such non-random HRS.

On a positive side, VRPs trivially satisfy balancedness, since they are guaranteed to be permutations for any value of the public key. This means one can build NIZK proofs from VRPs. By our construction of VRPs from VRFs, we see that VRFs are also sufficient for NIZK proofs for NP. Also, even VUPs coupled with the Goldreich-Levin bit turn out to be sufficient for this application.

NON-INTERACTIVE LOTTERY FOR MICROPAYMENTS. Micali and Rivest [33] suggested the following elegant way to perform non-interactive lottery (with the main application in micropayments). The merchant published a public key PK for a VRF f, the user chooses a ticket x, and wins if some predicate about $f(x)$ is true (for example, if $f(x)$ is less than some threshold t). Since f looks random to the user, the user cannot significantly bias his odds no matter what x he chooses. Similarly, since the merchant is committed to f by the public key PK, they merchant cannot lie about the value $f(x)$. Unfortunately, this still leaves exactly the same problem we had for the NIZK application above. Nothing stops the merchant from publishing a "non-balanced" VRF. In the extreme case, a constant function $f(x) = c$, where c is selected so that the predicate does not hold. Once again, we need balancedness to ensure that the merchant not only cannot change the value of f after the commitment, but also guarantees that the value $f(x)$ is random at least for a *random* x. Once again, VRPs perfectly solve this problem.

Moreover, VRP have an extra advantage that one can *precisely* know the number of possible winners: it is exactly equal to the number of strings y satisfying the given predicate. Thus, one can always allocate a given number of prizes and never worry that with some small probability there will be more winners than prizes.

REUSABLE COIN-FLIPPING. We can extend the previous lottery example to the following coin flipping problem. Alice wants to publish some value PK (keeping the corresponding value SK secret) allowing other to noninteractively select a random number r as follows. Any party Bob can choose a random value x and send it to Alice. The value x (combined with PK) uniquely defines the final value of r. If needed, Alice can open the value of r and convince Bob that this value is correct. Additionally, we want the following properties.

(a) No matter how Bob selects x, the value r looks random to Bob (except if he "replays" some old r).
(b) For any x, Alice cannot produce two different r as the final value, even if she adversarially chooses the public key PK.
(c) Bob is sure that that if he selects x at random, the value r is random, even if Alice adversarially chooses the public key PK.
(d) Alice can reuse the same PK for many executions (and only has to worry about the replay attack from Bob).

It is clear that VRPs precisely solve this problem. In contrast, VRFs do not satisfy property (c), while other existing coin-flipping protocols are either inefficient or do not appear to have the reusability property (d).

Finally, we mention examples how VRPs could be useful to add verifiability to some application of PRPs (where, again, PRFs are not sufficient).

VERIFIABLE CBC ENCRYPTION. As the simplest example, using VRPs one can add verifiability to CBC encryption and decryption.

VERIFIABLE HUGE RANDOM OBJECTS. A bit less straightforwardly, we consider the question of "truthfully", yet efficiently, sampling huge (pseudo)random objects, initiated by Goldreich et al. [21]. In this work, the authors showed several applications where PRPs can be used to efficiently sample various exponential-sized objects (like random connected graphs). Using VRPs one can naturally add verifiability to these constructs, so that the sampler can compactly commit and selectively reveal small parts of the huge object (like an edge). However, there is a subtlety. Since the PRP is often used as only part of the sampling procedure, revealing the proofs might leak a lot of extra information which might be undesirable. For example, in the random connected graph example one first samples a (pseudo)random graph, and then uses the PRP to add a random Hamiltonian cycle to it (in order to make it connected). With VRPs in place of PRPs, revealing the VRP proof will reveal this a given edge is part of the "special" Hamiltonian cycle, which is probably undesirable.

Nevertheless, we can avoid this "privacy problem" in scenarios where only PRPs are used to sample the given object. We give one such example (not present in [21]). Specifically, we can use PRPs to sample a pseudorandom constant-degree graph or exponential size (which is very likely to be a great expander). In the case the graph should be bipartite, such sampling simply consists of choosing d independent PRPs, where d is the required degree. This allows one to easily find all the neighbors of a given node on either side of the graph. In case of regular graphs, we need to sample d random matchings, which can also be done using PRPs by using an elegant result of Naor and Reingold [36] allowing one to sample pseudorandom permutations with a prescribed cycle structure. In either case, by using VRPs in place of PRPs we get *verifiable* random, constant-degree graphs, which do not suffer from the problem we had for random connected graphs.

Notice also that PRFs/VRFs are not sufficient for this application, since with high probability they will not result in a truthful implementation. Additionally, such sampling is not "reversible" (i.e., if $f(x) = y$, then given x one can see that y is connected to it, but not vice versa).

We hope that more "verifiable" huge random objects could be "privately" sampled using our technique.

Acknowledgments

We would like to thank Rafail Ostrovsky and Shabsi Walfish for several helpful discussions.

References

[1] Jee Hea An and Mihir Bellare, *Constructing VIL-MACs from FIL-MACs: Message Authentication under Weakened Assumptions*, CRYPTO 1999: 252-269.

[2] M. Bellare, *New Proofs for NMAC and HMAC: Security without Collision-Resistance*, Advances in Cryptology - Crypto 2006 Proceedings, Lecture Notes in Computer Science Vol. 4117, C. Dwork ed, Springer-Verlag, 2006.

[3] M. Bellare and P. Rogaway, *Optimal Asymmetric Encryption*, Proceedings of Eurocrypt'94, LNCS vol. 950, Springer-Verlag, 1994, pp. 92–111.

[4] M. Bellare and P. Rogaway, *The exact security of digital signatures - How to sign with RSA and Rabin*. Proceedings of Eurocrypt'96, LNCS vol. 1070, Springer-Verlag, 1996, pp. 399-416.

[5] Mihir Bellare and Moti Yung, *Certifying Permutations: Noninteractive Zero-Knowledge Based on Any Trapdoor Permutation*, in *Journal of Cryptology* 9(3): 149-166 (1996).

[6] John Black, *The Ideal-Cipher Model, Revisited: An Uninstantiable Blockcipher-Based Hash Function*, Fast Software Encryption – FSE 2006, Graz, Austria, Mar 2006.

[7] John Black, Phillip Rogaway and Thomas Shrimpton, *Black-Box Analysis of the Block-Cipher-Based Hash-Function Constructions from PGV*, CRYPTO 2002: 320-335.

[8] Manuel Blum, *Coin Flipping by Telephone - A Protocol for Solving Impossible Problems*, COMPCON 1982: 133-137.

[9] M. Blum, A. De Santis, S. Micali, and G. Persiano, *NonInteractive Zero-Knowledge*, in *SIAM Journal on Computing*, 20(6):1084-1118, 1991.

[10] Manuel Blum, Paul Feldman and Silvio Micali, *Non-Interactive Zero-Knowledge and Its Applications (Extended Abstract)* in *STOC 1988*, 103-112.

[11] Ran Canetti, Shai Halevi and Jonathan Katz, *A forward-secure public-key encryption scheme*, in *Advances in Cryptology - EUROCRYPT'03*.

[12] Anand Desai, *The Security of All-or-Nothing Encryption: Protecting against Exhaustive Key Search*, CRYPTO 2000: 359-375.

[13] Y. Dodis, *Efficient construction of (distributed) verifiable random functions*, In *Proceedings of 6th International Workshop on Theory and Practice in Public Key Cryptography*, pp 1 -17, 2003.

[14] Y. Dodis and P. Puniya, *On the relation between Ideal Cipher and Random Oracle Models*, In *Theory of Cryptography Conference 2006*.

[15] Y. Dodis and A. Yampolskiy, *A Verifiable Random Function With Short Proofs and Keys*, In *Workshop on Public Key Cryptography (PKC)*, January 2005.

[16] C. Dwork and M. Naor, *Zaps and their applications*, In *Proceedings of the 41st Annual Symposium on Foundations of Computer Science*, pp 283, 2000.

[17] Shimon Even and Yishay Mansour, *A Construction of a Cipher From a Single Pseudorandom Permutation*, ASIACRYPT 1991: 210-224.

[18] Uriel Feige, Dror Lapidot and Adi Shamir, *Multiple NonInteractive Zero Knowledge Proofs Under General Assumptions*, in *SIAM Journal of Computing* 29(1): 1-28 (1999).

[19] Goldreich, O., *Foundations of Cryptography Basic Tools*, published by *Cambridge University Press* (2001).

[20] O. Goldreich, S. Goldwasser and S. Micali, *How to Construct Random Functions* in *Journal of the ACM*, Vol. 33, No. 4, October 1986.

[21] Oded Goldreich, Shafi Goldwasser and Asaf Nussboim, *On the Implementation of Huge Random Objects*, FOCS 2003: 68-79.

[22] Oded Goldreich and Leonid A. Levin, *A Hard-Core Predicate for all One-Way Functions*, STOC 1989: 25-32.

[23] Shafi Goldwasser and Rafail Ostrovsky, *Invariant Signatures and Non-Interactive Zero-Knowledge Proofs are Equivalent (Extended Abstract)*, in *CRYPTO 1992*: 228-245.

[24] Jens Groth, Rafail Ostrovsky and Amit Sahai, *Perfect Non-Interactive Zero Knowledge for NP*, in *Electronic Colloquium on Computational Complexity (097)*, 2005.

[25] E. Jaulmes, A. Joux, and F. Valette, *On the security of randomized CBC-MAC beyond the birthday paradox limit: A new construction*, In *Fast Software Encryption (FSE 2002)* (2002), vol. 2365 of Lecture Notes in Computer Science, Springer-Verlag, pp. 237 -251.

[26] J. Kilian, and P. Rogaway, *How to protect DES against exhaustive key search (An analysis of DESX)*, *Journal of Cryptology 14*, 1 (2001), 17 -35.

[27] M. Luby and C. Rackoff, *How to construct pseudo-random permutations from pseudo-random functions*, in *SIAM Journal on Computing*, Vol. 17, No. 2, April 1988.

[28] A. Lysyanskaya, *Unique Signatures and verifiable random functions from DH-DDH assumption*, in *Proceedings of the 22nd Annual International Conference on Advances in Cryptography (CRYPTO)*, pp. 597 612, 2002.

[29] Ueli M. Maurer, Yvonne Anne Oswald, Krzysztof Pietrzak and Johan Sjödin, *Luby-Rackoff Ciphers from Weak Round Functions?*, EUROCRYPT 2006: 391-408.

[30] Ueli M. Maurer and Krzysztof Pietrzak, *The Security of Many-Round Luby-Rackoff Pseudo-Random Permutations*, in *EUROCRYPT 2003*, 544-561.

[31] Ueli M. Maurer and Johan Sjödin, *Single-Key AIL-MACs from Any FIL-MAC*, ICALP 2005: 472-484.

[32] S. Micali, M. Rabin and S. Vadhan, *Verifiable Random functions*, In *Proceedings of the 40th IEEE Symposium on Foundations of Computer Science*, pp. 120 -130, 1999.

[33] Silvio Micali and Ronald L. Rivest, *Micropayments Revisited*, CT-RSA 2002, 149-163.

[34] Moni Naor, *Bit Commitment Using Pseudo-Randomness*, CRYPTO 1989: 128-136.

[35] Moni Naor and Omer Reingold, *On the construction of pseudo-random permutations: Luby-Rackoff revisited*, in *Journal of Cryptology*, vol 12, 1999, pp. 29-66.

[36] Moni Naor and Omer Reingold, *Constructing Pseudo-Random Permutations with a Prescribed Structure*, J. of Cryptology, vol 14, 2001.

[37] Moni Naor and Moti Yung, *Universal One-Way Hash Functions and their Cryptographic Applications*, STOC 1989: 33-43.

[38] Jacques Patarin, *Security of Random Feistel Schemes with 5 or More Rounds*, in *CRYPTO 2004*, 106-122.

[39] Z. Ramzan and L. Reyzin, *On the Round Security of Symmetric-Key Cryptographic Primitives*, in Advances in Cryptography - Crypto, LNCS vol. 1880, Springer-Verlag, 2000.

[40] Daniel R. Simon, *Finding Collisions on a One-Way Street: Can Secure Hash Functions Be Based on General Assumptions?*, EUROCRYPT 1998: 334-345.

A Formal Definitions

Let λ denote the security parameter. We use $negl(\lambda)$ to denote a negligible function of λ.

Let $a : \mathbb{N} \to \mathbb{N}$ and $b, c : \mathbb{N} \to \mathbb{N}$be polynomial time computable functions. We first define the notion of pseudorandom functions (PRFs). The definition given below is a seemingly different from the usual definition of PRFs, but is essentially equivalent.

Definition 2 (Pseudorandom Functions) *A* pseudorandom function family $F_{\{0,1\}^{c(\lambda)}} : \{0,1\}^{a(\lambda)} \to \{0,1\}^{b(\lambda)}$ *is an efficiently samplable distribution over the set of all functions from $a(\lambda)$ to $b(\lambda)$ bits such that, for any probabilistic polynomial time (PPT) adversary pair $A = (A_1, A_2)$, none of which query their oracles on the challenge query,it holds that,*

$$\Pr\left[b = b' \;\middle|\; \begin{array}{c} s \leftarrow \{0,1\}^{c(\lambda)}; (x, \alpha) \leftarrow A_1^{F_s}(1^\lambda); y_0 \leftarrow F_s(x); \\ y_1 \leftarrow \{0,1\}^{b(\lambda)}; b \leftarrow \{0,1\}; b' \leftarrow A_2^{F_s}(y_b, \alpha) \end{array} \right] = negl(\lambda)$$

Similar to the notion of PRFs, we can define the notion of a *(strong) pseudorandom permutations* (PRPs).

Definition 3 (Pseudorandom Permutations) *A* pseudorandom permutation family $\Pi_{\{0,1\}^{c(\lambda)}} : \{0,1\}^{a(\lambda)} \to \{0,1\}^{a(\lambda)}$ *is an efficiently samplable distribution of permutations on $a(\lambda)$ bits such that, for any probabilistic polynomial time (PPT) adversary pair $A = (A_1, A_2)$, none of which query their oracles on the challenge query or its inverse, it holds that,*

$$\Pr\left[b = b' \;\middle|\; \begin{array}{c} s \leftarrow \{0,1\}^{c(\lambda)}; (d_{\in\{-1,+1\}}, x, \alpha) \leftarrow A_1^{\Pi_s, \Pi_s^{-1}}(1^\lambda); y_0 \leftarrow \Pi_s^d(x); \\ y_1 \leftarrow \{0,1\}^{b(\lambda)}; b \leftarrow \{0,1\}; b' \leftarrow A_2^{\Pi_s, \Pi_s^{-1}}(y_b, \alpha) \end{array} \right] = negl(\lambda)$$

A slightly weaker notion than PRFs is that of *Unpredictable Functions* (UFs). Unpredictable functions are also popularly known as *Message Authentication Codes* (MACs).

Definition 4 (Unpredictable Functions) *An* unpredictable function family $F_{\{0,1\}^{c(\lambda)}} : \{0,1\}^{a(\lambda)} \to \{0,1\}^{b(\lambda)}$ *is an efficiently samplable distribution over the set of all functions from $a(\lambda)$ to $b(\lambda)$ bits such that, for any probabilistic polynomial time (PPT) adversary A, that does not query its oracle on the prediction query, it holds that,*

$$\Pr\left[y = F_s(x) \;\middle|\; s \leftarrow \{0,1\}^{c(\lambda)}; (x, y) \leftarrow A^{F_s}(1^\lambda) \right] = negl(\lambda)$$

Similarly, we can also define the notion of *Unpredictable Permutations* (UPs).

Definition 5 (Unpredictable Permutations) *An* unpredictable function family $F_{\{0,1\}^{c(\lambda)}} : \{0,1\}^{a(\lambda)} \rightarrow \{0,1\}^{a(\lambda)}$ *is an efficiently samplable distribution over the set of all permutations on $a(\lambda)$ bits such that, for any probabilistic polynomial time (PPT) adversary A, that does not query its oracle on the prediction query or its inverse, it holds that,*

$$\Pr\left[y = \Pi_s(x) \,\middle|\, s \leftarrow \{0,1\}^{c(\lambda)}; (x,y) \leftarrow A^{\Pi_s, \Pi_s^{-1}}(1^\lambda)\right] = negl(\lambda)$$

We can define verifiable analogs of each of the definitions above. Let us start by defining the notion of *Verifiable Pseudorandom Functions* (VRFs).

Definition 6 (Verifiable Random Functions) *A* Verifiable random function family $F_{\{0,1\}^{c(\lambda)}} : \{0,1\}^{a(\lambda)} \rightarrow \{0,1\}^{b(\lambda)}$ *consists of three algorithms* $(\mathsf{Gen}, \mathrm{Prove}, \mathrm{Verify})$ *such that* $\mathsf{Gen}(1^\lambda)$ *outputs a pair of keys (PK, SK);* $\mathrm{Prove}_{SK}(x)$ *outputs a pair $(F_{SK}(x), proof_{SK}(x))$, where $F_{SK}(x)$ is the function output and $proof_{SK}(x))$ is the corresponding proof of correctness; and* $\mathrm{Verify}_{PK}(x, y, prf)$ *verifies that $y = F_{SK}(x)$ using the proof prf (by outputting 1 if so). This VRF family should satisfy three requirements:*

- **Correctness***: if $(y, prf) \leftarrow \mathrm{Prove}_{SK}(x)$, then* $\mathrm{Verify}_{PK}(x, y, prf) = 1$.
- **Soundness***: no values $(PK, x, y_1, prf_1, y_2, prf_2)$, with $(y_1, prf_1) \neq (y_2, prf_2)$, can satisfy*

$$\mathrm{Verify}_{PK}(x, y_1, prf_1) = \mathrm{Verify}_{PK}(x, y_2, prf_2) = 1$$

- **Pseudorandomness***: For any PPT adversary pair $A = (A_1, A_2)$, neither of which query their oracle on the challenge input x, it holds that*

$$\Pr\left[b = b' \,\middle|\, \begin{array}{c} (PK, SK) \leftarrow \mathsf{Gen}(1^\lambda); (x, \alpha) \leftarrow A_1^{\mathrm{Prove}_{SK}}(1^\lambda); y_0 \leftarrow F_{SK}(x); \\ y_1 \leftarrow \{0,1\}^{b(\lambda)}; b \leftarrow \{0,1\}; b' \leftarrow A_2^{\mathrm{Prove}_{SK}}(y_b, \alpha) \end{array}\right] = negl(\lambda)$$

Along similar lines, we can define the notions of *Verifiable Pseudorandom Permutations* (VRPs), *Verifiable Unpredictable Functions* and *Verifiable Unpredictable Permutations* as verifiable analogs of PRPs, UFs and UPs respectively, each of which has three algorithms $(\mathsf{Gen}, \mathrm{Prove}, \mathrm{Verify})$, and satisfies the Completeness and Soundness properties as well.

Let $f : \{0,1\}^n \longrightarrow \{0,1\}^n$ be a function from n bits to n bits. The *Feistel transformation* using f is a permutation Ψ_f on $2n$ bits defined as, $\Psi_f(x) \stackrel{def}{=} x_R \parallel x_L \oplus f(x_R)$. The symbols x_L and x_R denote the left and right halves of $2n$ bit string x. We will also call the construction based on k iterated applications of the Feistel transformation, a k round LR construction, and denote it by $\Psi_{f_1 \dots f_k}$ (which we also call Ψ_k for brevity) where $f_1 \dots f_k$ are the round functions used. On a $2n$ bit input, the construction Ψ_k generates $(k+2)$ n-bit round values, the last two of which together form the output of the construction.

The seminal result of Luby and Rackoff [27] shows that a 4 round LR construction with independent and secure PRFs in each round is a secure (strong) pseudorandom permutation (PRP) construction.

B Proof of Lemma 4.1

Assume that the 5-XOR condition does not hold for the given sequence of queries. Without loss of generality, say one of the queries involved in the r^{th} round value collision is the last (q^{th}) query. If this is not the case, then we can consider a smaller sequence of queries for which this happens. We represent the j^{th} round value associated with the i^{th} query as $R[i,j]$. Thus we know that $\exists i < q \;:\; R[q,r] = R[i,r]$.

We maintain a q vector b that denotes the direction of each query. Thus if $\mathsf{b}[i] = 1$ then the i^{th} query was a forward one, else if $\mathsf{b}[i] = 0$ then the i^{th} query was an inverse query. We define a "first occurrence" function for each round value, i.e. $\mathsf{p} : \{1 \ldots q\} \times \{0 \ldots k+1\} \rightarrow \{1 \ldots q\}$. For any round value $R[i,j]$, $\mathsf{p}(i,j)$ is the *least input number* such that $R[\mathsf{p}(i,j), j] = R[i,j]$.

Assume that the colliding round number $r \leq k/2$. In this case, if the q^{th} query is a forward query then we get a worse lower bound on the number of queries. If $r > k/2$ then we get a worse bound if the q^{th} query is an inverse query. Thus, for now assume that the q^{th} query is a forward query, since this gives us the worst case bound if $r < k/2$.

As a first step, we prove that all the round values $R[q,1] \ldots R[q, r-1]$ collide with the corresponding round value in an earlier query.

Claim B.1 *If $\exists i < q \ : \ R[q,r] = R[i,r]$, then each of the round values $R[q,1] \ldots R[q,r-1]$ were defined before the q^{th} query was made. That is,*

$$\forall j \in \{1 \ldots r\} \ : \ \mathsf{p}(q,j) < q$$

proof of claim B.1: We will use induction on the round number j to show that $\mathsf{p}(q,j) < q$. However, we will start the induction with $j = r$ and go down to $j = 1$.

For $j = r$, we already know that $\mathsf{p}(q,r) = i$ from the statement of the claim. Now say the same holds for all $j = r \ldots c$ (for $c \leq r$), then we will show that the $(c-1)^{th}$ round value also collides with the corresponding round value in an earlier query. Say, for the sake of contradiction, that $R[q, c-1]$ is a new round value in input number q (i.e. $\mathsf{p}(q, c-1) = q$). Then the round function evaluation $f_{(c-1)}\left(R[q, c-1]\right)$ is a new round function evaluation, and hence $R[q,c]$ is a new round function value. But $R[q,c] = R[\mathsf{p}(q,c), c]$, and $\mathsf{p}(q,c) < q$ by induction hypothesis. But this is not possible since the 5-XOR condition does not hold for this sequence of queries. $\square$

Thus, we know that all the round values $R[q,1] \ldots R[q,r]$ were defined strictly before input number q. As our next step, we will show that the order in which the queries $\mathsf{p}(q,1) \ldots \mathsf{p}(q,r)$ occur is one of very few possible orders.

Claim B.2 *There is a round number $j \in \{1 \ldots r\}$, such that,*

$$\begin{array}{ccccccccc} \mathsf{p}(q,1) & > & \ldots & > & \mathsf{p}(q,(j-1)) & > & \mathsf{p}(q,j) & & & & \\ & & & & & & \mathsf{p}(q,j) & < & \ldots & < & \mathsf{p}(q,r-1) & < & \mathsf{p}(q,r) \end{array}$$

That is, the round value $R[q,j]$ was defined before any of the other round values $R[q,1] \ldots R[q,r]$. Moreover, the latter queries were made in the order $\mathsf{p}(q,j) \ldots \mathsf{p}(q,r)$, while the first j queries were made in the order $\mathsf{p}(q,j) \ldots \mathsf{p}(q,1)$.

proof of claim B.2: We will first prove that for any three consecutive round values $R[q,(i-1)], R[q,i]$ and $R[q,(i+1)]$ (where $i \in \{2 \ldots r-1\}$), it holds that,

$$[\mathsf{p}(q,(i-1)) > \mathsf{p}(q,i)] \vee [\mathsf{p}(q,i) < \mathsf{p}(q,(i+1))]$$

The claim will then follow as a straightforward consequence.

Assume to the contrary that $\mathsf{p}(q,(i-1)) \leq \mathsf{p}(q,i)$ and $\mathsf{p}(q,i) \geq \mathsf{p}(q,(i+1))$ for some $i \in \{2, r-1\}$. If $\mathsf{p}(q,(i-1)) = \mathsf{p}(q,i)$ (or $\mathsf{p}(q,i) = \mathsf{p}(q,(i+1))$) then it is easy to verify that queries $\mathsf{p}(q,i)$ and q are the same, which is not the case since q is the first query colliding with an earlier query in the r^{th} round value.

Thus, we have the case that $\mathsf{p}(q,(i-1)) < \mathsf{p}(q,i)$ and $\mathsf{p}(q,i) > \mathsf{p}(q,(i+1))$. But we know from the design of Ψ_k that,

$$f_i(R[\mathsf{p}(q,i), i]) = R[\mathsf{p}(q,i), (i-1)] \oplus R[\mathsf{p}(q,i), (i+1)]$$

It is also the case that,

$$\begin{array}{rlcl} & f_i(R[q,i]) & = & R[q,(i-1)] \oplus R[q,(i+1)] \\ \Rightarrow & f_i(R[\mathsf{p}(q,i),i]) & = & R[\mathsf{p}(q,(i-1)),(i-1)] \oplus R[\mathsf{p}(q,(i+1)),(i+1)] \\ \Rightarrow & R[\mathsf{p}(q,i),(i-1)] \oplus R[\mathsf{p}(q,i),(i+1)] & = & R[\mathsf{p}(q,(i-1)),(i-1)] \oplus R[\mathsf{p}(q,(i+1)),(i+1)] \end{array}$$

Thus, if $\mathsf{b}[\mathsf{p}(q,i)] = 0$ then $R[\mathsf{p}(q,i),(i-1)]$ can be represented as an XOR of three previously existing round values otherwise $R[\mathsf{p}(q,i),(i+1)]$ has such an XOR representation. In any case, this will give a 5-XOR condition which we know does not hold. Thus we can say that

$$\forall i \in \{2, r-1\} \;\; : \;\; [\mathsf{p}(q,(i-1)) > \mathsf{p}(q,i)] \vee [\mathsf{p}(q,i) < \mathsf{p}(q,(i+1))]$$

Now it is a straightforward task to verify that the query orders consistent with this constraint are exactly the ones in the statement of claim B.2. $\square$

From claim B.2, we can deduce that there exist at least $\frac{r}{2}$ consecutive round values in the q^{th} query, whose "first occurrence" queries are in strictly ascending/descending temporal order. Without loss of generality, we can assume that $q > \mathsf{p}(q,1) > \ldots > \mathsf{p}\left(q, \frac{r}{2}\right)$.

As our next step, we will prove a general property of such a strictly ordered sequence of "first occurrence" queries of consecutive round values. For this purpose, consider any three consecutive "first occurrence" queries out of such a sequence, say $i_j = \mathsf{p}(q,j)$, $i_{j+1} = \mathsf{p}(q,j+1)$ and $i_{j+2} = \mathsf{p}(q,j+2)$ (so that $i_j > i_{j+1} > i_{j+2}$). We will concentrate on proving properties of the "first occurrence" queries of the round values of the i_j^{th} query. Since $j \leq \frac{k}{4}$, we will assume that $\mathsf{b}(i_j) = 1$ or that the i_j^{th} query is a forward query. The reason for this choice is again the fact that it gives us the worse bound on the number of queries.

We will now show that all the round values $R[i_j,1] \ldots R[i_j,j-1]$ collide with the corresponding round values in queries before the i_j^{th} query. Moreover, we also show that the queries $\mathsf{p}(i_j,1) \ldots \mathsf{p}(i_j,j-2)$ were made after the i_{j+1}^{th} query, but before the i_j^{th} query. This is formally stated in the following claim.

Claim B.3 *Let the queries numbered i_j, i_{j+1} and i_{j+2} be the "first occurrence" queries of the round values $R[\ell,j]$,$R[\ell,j+1]$ and $R[\ell,j+2]$, respectively. Moreover, say that $i_j > i_{j+1} > i_{j+2}$. If the i_j^{th} query is a forward query (i.e. $\mathsf{b}[i_j] = 1$) then,*

$$i_j > \mathsf{p}(i_j,1) > \ldots > \mathsf{p}(i_j,j-2) > i_{j+1}$$

On the other hand, if $\mathsf{b}[i_j] = 0$ then,

$$i_j > \mathsf{p}(i_j,k) > \ldots > \mathsf{p}(i_j,j+2) > i_{j+1}$$

proof of claim B.3: Let us start by considering the case that $\mathsf{b}[i_j] = 1$, and the inverse query will turn out to be symmetric. We will analyze the round values $R[i_j,1] \ldots R[i_j,j-1]$. Consider the round value $R[i_j,j-1]$. If it is a new round value at the i_j^{th} query, then $f_{j-1}(R[i_j,j-1])$ is a new round function evaluation and $R[i_j,j]$ is the new round value generated. But we know that

$$\begin{array}{rlcl} & f_{j+1}(R[\ell,j+1]) & = & R[\ell,j] \oplus R[\ell,j+2] \\ \Rightarrow & f_{j+1}(R[i_{j+1},j+1]) & = & R[i_j,j] \oplus R[i_{j+2},j+2] \\ \Rightarrow & R[i_j,j] & = & R[i_{j+2},j+2] \oplus R[i_{j+1},j] \oplus R[i_{j+1},j+2] \end{array}$$

And since $i_j > i_{j+1} > i_{j+2}$, this would give us a 5-XOR condition, which we know does not hold. Thus, we know that $\mathsf{p}(i_j,j-1) < i_j$.

<u>**Conclusion 1:**</u> As in claim B.1, we can also deduce that $\forall j' \in \{1 \ldots j-1\} \; : \; \mathsf{p}(i_j,j') < i_j$.

Now we will try to find the order in which the queries $\mathsf{p}(i_j, j')$ could have been made. In addition, since we know that $i_{j+2} < i_{j+1} < i_j$, we will also be interested when the queries $\mathsf{p}(i_j, j')$ were made relative to i_{j+1} and i_{j+2}. Let us start by concentrating our attention on the queries i_{j+1}, $\mathsf{p}(i_j, (j-1))$ and $\mathsf{p}(i_j, (j-2))$.

Consider the case that $\mathsf{p}(i_j, (j-1)) < i_{j+1}$ and $\mathsf{p}(i_j, (j-2)) < i_{j+1}$. Then we know that,

$$\begin{array}{rlcl} & f_{j+1}(R[i_{j+1}, (j+1)]) & = & R[i_{j+1}, j] \oplus R[i_{j+1}, (j+2)] \\ \Rightarrow & f_{j+1}(R[i_{j+1}, (j+1)]) & = & R[i_{j+1}, (j+2)] \oplus R[i_j, (j-2)] \oplus f_{j-1}(R[i_j, (j-1)]) \\ \Rightarrow & R[i_{j+1}, (j+2)] \oplus R[i_{j+1}, j] & = & R[i_{j+1}, (j+2)] \oplus R[\mathsf{p}(i_j, (j-2)), (j-2)] \\ & & & \oplus R[\mathsf{p}(i_j, (j-1)), j] \oplus R[\mathsf{p}(i_j, (j-1)), j+1] \end{array}$$

Thus depending on whether i_{j+1} is a forward or inverse query, we get a representation of $R[i_{j+1}, j]$ or $R[i_{j+1}, (j+2)]$ as an XOR of five previous round values and since $R[i_{j+1}, (j+1)]$ is a new round value this contradicts the fact that 5-XOR condition does not hold for these queries. Thus $\mathsf{p}(i_j, (j-1)) > i_{j+1}$ or $\mathsf{p}(i_j, (j-2)) > i_{j+1}$. Along similar lines, we can also show that $\mathsf{p}(i_j, (j-2)) > \mathsf{p}(i_j, (j-1))$ or $i_{j+1} > \mathsf{p}(i_j, (j-1))$.

Conclusion 2: We can deduce that the only possible orders for these three queries are

$$\mathsf{p}(i_j, (j-2)) > \mathsf{p}(i_j, (j-1)) > i_{j+1} \text{ or } \mathsf{p}(i_j, (j-2)) > i_{j+1} > \mathsf{p}(i_j, (j-1))$$

We saw in claim B.2 that if we consider the "first occurrence" queries of three consecutive round values, the query corresponding to the middle round value cannot be the last one to be made. And since we know that $\mathsf{p}(i_j, (j-2)) > \mathsf{p}(i_j, (j-1))$ from *conclusion 1*, we can see that $\mathsf{p}(i_j, (j-3)) > \mathsf{p}(i_j, (j-2))$. Carrying this argument upto $R[i_j, 1]$, we thus deduce that,

$$i_j > \mathsf{p}(i_j, 1) > \ldots > \mathsf{p}(i_j, (j-2)) > i_{j+1}$$

If the query number i_j is an inverse query, then we can carry out the same argument for the round value $R[i_j, (j+1)] \ldots R[i_j, k]$ and deduce that

$$i_j > \mathsf{p}(i_j, k) > \ldots > \mathsf{p}(i_j, (j+2)) > i_{j+1}$$

□

Now we can apply claim B.3 to the sequence of first occurrence queries $\mathsf{p}(q, 1) > \ldots > \mathsf{p}\left(q, \frac{r}{2}\right)$. Considering query $\mathsf{p}(q, i)$ (for $i = 1 \ldots \frac{r}{2} - 2$), we have deduced that if this was a forward query, at least another $i - 2$ queries we made before this one but after the query $\mathsf{p}(q, (i+1))$. And since these $i-2$ queries are in a strictly descending temporal order as well from claim B.3, we can recursively apply our argument to each of these queries as well.

In order to bound from below the number of queries that produce a collision on the r^{th} round value, we will need to count the number of queries that are bound to exist by the argument above. Thus consider a strictly ascending/descending sequence of "first occurrence" queries of j consecutive round values, i.e. $\mathsf{p}(i, 1) > \ldots > \mathsf{p}(i, j)$ for some query i. We will denote by $\mathcal{Q}(j)$ the number of queries that were made before query number i in order to get such a sequence of queries. If we apply claim B.3 to the queries $\mathsf{p}(i, 1) \ldots \mathsf{p}(i, (j-2))$, we get a new sequence of $(\ell - 2)$ strictly ascending/descending "first occurrence" queries between each of the pairs of queries $\mathsf{p}(i, \ell)$ and $\mathsf{p}(i, (\ell+1))$ for $\ell = 1 \ldots (j-2)$. Hence we get the following expression for $\mathcal{Q}(j)$,

$$\begin{array}{rlcl} & \mathcal{Q}(j) & = & j + \sum_{\ell=3}^{j-2} \mathcal{Q}(\ell - 2) \\ \Rightarrow & \mathcal{Q}(j) & = & \mathcal{Q}(j-1) + \mathcal{Q}(j-4) + 1 \\ \Rightarrow & \mathcal{Q}(j) & = & 2 \cdot \mathcal{Q}(j-1) - \mathcal{Q}(j-2) + \mathcal{Q}(j-4) - \mathcal{Q}(j-5) \end{array}$$

The solution to the above homogeneous equation can be expressed in terms of the powers of the roots of the following algebraic equation:

$$x^5 - 2x^4 + x^3 - x^4 + 1 = 0$$

This equation has only one root greater than 1, which is 1.3803. Thus we can represent the solution of the above recurrence as:

$$\mathcal{Q}(j) = \Theta(1.3803^j)$$

And from claim B.2, we get that if any query collides with an earlier query in the r^{th} round value, we can find a strictly increasing/decreasing sequence of $\frac{r}{2}$ "first occurrence" queries. Thus, we get that

$$\begin{array}{rcl} q & \geq & \mathcal{Q}\left(\frac{r}{2}\right) \\ \Rightarrow \quad q & = & \Omega\left(1.3803^{r/2}\right) \\ \Rightarrow \quad q & \geq & \Omega\left(1.3803^{s/2}\right) \text{ , since } r \in \{s \ldots (k-s)\} \end{array}$$

The above proof only took into account the case that $r < k/2$. If $r > k/2$ then a similar argument can be carried out by assuming the intermediate queries to be inverse queries and we can derive that $q = \Omega\left(1.3803^{(k-r)/2}\right)$. In either case, we get the bound that $q = \Omega\left(1.3803^{s/2}\right)$, since $r \in \{s, (k-s)\}$ and $s \leq (k/2)$.

C Proof of theorem 5.1

The proof of this theorem consists of two main parts:

1. A UF family that yields secure and independent UFs on each sample is a 5-XOR resistant function family.
2. The construction $\Psi_{U,k}$ that uses secure and independent UFs in each round is a secure *unpredictable permutation*.

XOR-RESISTANCE OF UFS. Consider the k-round Feistel construction $\Psi_{U,k}$ using independent UFs $f_1 \ldots f_k \leftarrow F$ in each round. If there is an adversary A_{xor} that queries $\Psi_{U,k}$ and forces a 5-XOR condition through its queries with a non-negligible advantage ϵ_{xor}, then we can construction a UF adversary A_f that has non-negligible advantage in the unpredictability game against a UF sampled from the family F.

Claim C.1 *If there is an adversary A_{xor} that can force a 5-XOR condition in an interaction with $\Psi_{U,k}$ (that uses independent UFs sampled from a UF family $F_{(\cdot)} : \{0,1\}^n \rightarrow \{0,1\}^n$) with non-negligible probability ϵ_{xor} then there exists a VUF adversary A_f that has non-negligible success probability ϵ_f in the unpredictability against a UF sampled from the family F. In particular, we show that $\epsilon_f \geq \frac{\epsilon_{xor}}{(qk)^6}$.*

proof of claim C.1: On getting the challenge unpredictable function F_s, the UF adversary chooses a random round number i where it plugs in the challenge UF. Next, the UF adversary A_f generates $(k-1)$ independent UFs $f_1 \ldots f_{i-1}, f_{i+1} \ldots f_k$ from the same family and uses these as the remaining round functions to simulate the Feistel construction $\Psi_{U,k}$ for the XOR adversary A_{xor} to attack.

Then it lets the UF adversary run its attack on $\Psi_{U,k}$. Assuming a fixed and large enough polynomial upper bound q on the number of queries made by A_{xor}, the UF adversary A_f chooses a random query number $j \in \{1, q\}$. It guesses that the 5-XOR condition occurs after the i^{th} round function evaluation in the j^{th} query, i.e. R_i^j. Instead of querying this input to the UF oracle, it selects this as the challenge input and uses an XOR of upto 5 randomly chosen previously existing round values as its prediction of the output.

If all its guesses are correct, i.e. it chooses the correct round number i, the correct query number j and the correct XOR representation, then it succeeds in the UF game. The probability that all its guesses are correct is at least $\frac{1}{(qk)^6}$. Thus, we get that $\epsilon_f \geq \frac{\epsilon_{xor}}{(qk)^6}$. □

SECURITY OF THE UP CONSTRUCTION. We will now show that the UP construction $\Psi_{U,k}$, that uses round functions from the UF family F that gives secure and independent UFs on each sample, is a secure construction of a *unpredictable permutation*

Claim C.2 *If there exists a PPT UP adversary A_π that has non-negligible advantage ϵ_π in the unpredictability game against $\Psi_{U,k}$ and which makes a polynomial number of queries to $\Psi_{U,k}$, then there also exists a UF adversary A_f that has non-negligible advantage in the unpredictability game against a UF sampled from the UF family F. In particular, we get that the maximum advantage of the UP adversary A_π is $\epsilon_\pi = \mathcal{O}(\epsilon_f \cdot (qk)^6)$. Here ϵ_f is the maximum advantage of a UF adversary running in time $\mathcal{O}(t + (qk)^5)$ against a UF sampled from F, where t, q are the running time and number of queries made by A_π.*

proof of claim C.2: The UF adversary A_f gets oracle access to a challenge unpredictable function F_s. It samples $(k-1)$ independent UFs $f_1 \dots f_{(k/2)-1}, f_{(k/2)+1} \dots f_k$ from the same UF family F. It simulates the UP construction by plugging in the challenge UF as the $(k/2)^{th}$ round function, and using the self-generated UFs as the other round functions.

When the UP adversary sends its challenge query $R_0 \parallel R_1$ (or $R_k \parallel R_{k+1}$), and its predicted output $R_k \parallel R_{k+1}$ (resp. $R_0 \parallel R_1$), the UF adversary proceeds by using its self generated round functions to evaluate the intermediate round values $R_0, R_1, R_2 \dots R_{k/2}$ and from $R_{k+1}, R_k, R_{k-1} \dots R_{k/2+1}$. It then sends the challenge input/output pair $(R_{k/2}, R_{k/2-1} \oplus R_{k/2+1})$ as its prediction. It is easy to see that if the round value $R_{k/2}$ is a new round value and the UP adversary predicted correctly, then the UF adversary A_f succeeds. Thus, we can deduce that,

$$
\begin{aligned}
\Pr[A_f \text{ succeeds}] &= \Pr[(A_\pi \text{ succeeds}) \wedge (\text{no } (k/2)^{th} \text{ round collision})] && (1)\\
&= \Pr\left[(A_\pi \text{ succeeds}) \,\middle|\, \text{no } (k/2)^{th} \text{ round collision}\right] \cdot \Pr[\text{no collision}] && (2)\\
&\geq \left(\Pr[A_\pi \text{ succeeds}] - \Pr\left[(k/2)^{th} \text{ round collision}\right]\right) \cdot \Pr[\text{no collision}] && (3)\\
&\geq (\epsilon_\pi - \epsilon_{xor}) \cdot (1 - \epsilon_{xor}) && (4)\\
\Rightarrow \epsilon_\pi &\leq \frac{\epsilon_f}{1 - \epsilon_{xor}} + \epsilon_{xor} && (5)\\
\Rightarrow \epsilon_\pi &= \mathcal{O}(\epsilon_f \cdot (qk)^6) && (6)
\end{aligned}
$$

In the above argument, we have often bound the advantage of a UF adversary by ϵ_f. This is the maximum advantage of a UF adversary running in time $\mathcal{O}(t + (qk)^5)$, where t, q are the running time and number of queries made by A_π. The transition from step (3) to (4) is possible using corollary 4.3, that says that the advantage of an efficient collision finding adversary is same as that of an efficient XOR condition forcing adversary. The last step of the argument is possible through claim C.1. □

D Proof of theorem 5.2

We show that the PRP construction $\Psi_{R,k}$, using PRFs sampled from the PRF family $F_{(\cdot)} : \{0,1\}^n \to \{0,1\}^n$, is a secure PRP. The proof consists of two parts:

1. Showing that a PRF family that yields secure and independent PRFs upon each sample is a 5-XOR resistant function family.
2. Showing that no PRP adversary can succeed with non-negligible advantage in the PRP attack game against a $\omega(\log \lambda)$-round Feistel construction with independent and secure PRFs in each round.

XOR-RESISTANCE OF PRFS. Consider a k-round Feistel construction Ψ_k that uses k PRFs $f_1 \dots f_k$, independently sampled from a PRF family $F_{(\cdot)} : \{0,1\}^n \to \{0,1\}^n$, as round functions. Consider an XOR finding adversary A_{xor} that forces a 5-XOR condition through its queries with non-negligible advantage. We will show that using A_{xor}, we can design another attacker A_f that succeeds in the PRF attack game (see definition 2) against a PRF sampled from the family F.

Claim D.1 *If there is an PPT attacker A_{xor} that queries the k-round Feistel construction Ψ_k (that uses independent PRFs from a PRF family F), observes intermediate round values and forces the 5-XOR condition through its queries with probability ϵ_{xor}, then there exists a PRF adversary A_f that has advantage ϵ_f in the PRF attack game against a PRF sampled from F, where $\epsilon_f \geq \frac{1}{2qk} \cdot \left(\epsilon_{xor} - \frac{(qk)^6}{2^n}\right)$.*

proof of claim D.1: The PRF adversary A_f gets oracle access to the challenge PRF adversary F_s. It then needs to choose a challenge query, to which it either gets the PRF output or a random n-bit string, and its task is to distinguish between the two cases. The attacker A_f chooses a random round number $i \in \{1 \dots k\}$, and samples $(k-1)$ independent PRFs $f_1 \dots f_{i-1}, f_{i+1} \dots f_k$ from the family F. It then simulates the Feistel construction Ψ_k, with the challenge PRF as the i^{th} round function and the self-generated PRFs making up the remaining round functions. It then simulates the XOR attack game between A_{xor} and Ψ_k.

Assume a fixed, large enough, polynomial upper bound on the number of queries that the adversary A_{xor} makes to Ψ_k. The PRF adversary chooses a random query number $j \in \{1 \dots q\}$ where it chooses its challenge query. On getting the j^{th} query, it sends the i^{th} round value as the challenge PRF query, and uses the challenge response as the output of the i^{th} round function. It computes the remaining self-generated round functions as usual. If the i^{th} round function is applied to a new input, then it checks to see if the new round value generated has a 5-XOR representation in terms of previously existing round values. If so, then it guesses that the challenge response is the PRF output (say by outputting 1), otherwise it guesses the challenge response to be random (by outputting 0).

It is clear that if the attacker A_f makes all its guesses correctly, i.e. correct round number and correct query number, then it succeeds if a random response also does not have an 5-XOR representation. Hence, we get that

$$Adv(A_f) = \Pr[(A_f \text{ outputs } 1) \wedge (\text{PRF output})] + \Pr[(A_f \text{ outputs } 0) \wedge (\text{Random output})] - \frac{1}{2}$$

If ϵ_{xor} denotes the advantage of an XOR adversary then we get that,

$$\begin{aligned}
\Pr[(A_f \text{ outputs } 1) \wedge (\text{PRF output})] &= \Pr[(A_f \text{ outputs } 1) | (\text{PRF output})] \cdot \Pr[(\text{PRF output})] \\
&\geq \frac{\epsilon_{xor}}{qk} \cdot \frac{1}{2} \\
\Pr[(A_f \text{ outputs } 0) \wedge (\text{Random output})] &= \Pr[(A_f \text{ outputs } 0) | (\text{Random output})] \cdot \Pr[(\text{Random output})] \\
&\geq \left[1 - \frac{(qk)^5}{2^n}\right] \cdot \frac{1}{2} \\
\Rightarrow Adv(A_f) &\geq \frac{1}{2qk} \cdot \left[\epsilon_{xor} - \frac{(qk)^6}{2^n}\right]
\end{aligned}$$

□

SECURITY OF THE PRP CONSTRUCTION. We will now show that the construction $\Psi_{R,k}$, that is based on a k-round Feistel construction using independently sampled round functions from a PRF family $F_{(\cdot)} : \{0,1\}^n \to \{0,1\}^n$, is a secure PRP construction.

Claim D.2 *If there exists an efficient PRP adversary A_π that has a non-negligible advantage ϵ_π in the PRP attack game against the construction $\Psi_{R,k}$ (using round function from PRF family F), then there also exists a PRF adversary A_f that has non-negligible advantage ϵ_f in the PRF attack game against a PRF sampled from the PRF family F. In particular, we get that the maximum advantage of such a PRP adversary can be bounded by $\epsilon_\pi = \mathcal{O}\left(qk\epsilon_f + \frac{(qk)^6}{2^n}\right)$. Here ϵ_f is the maximum advantage of a PRF adversary running in time $\mathcal{O}(t + (qk)^5)$ against a PRF sampled from F, where t, q are the running time and number of queries made by A_π.*

proof of claim D.2: The PRF adversary A_f gets oracle access to a challenge PRF F_s. It samples $(k-1)$ independent PRFs $f_1 \dots f_{(k/2)-1}, f_{(k/2)+1} \dots f_k$ from the PRF family F. It simulates the PRP construction $\Psi_{R,k}$ by plugging in the challenge PRF as the $(k/2)^{th}$ round function and using the self-generated PRFs as the remaining round functions. It then simulates the PRP attack game between the attacker A_π and $\Psi_{R,k}$.

It computes the response to any query made by A_π by computing all the round values ($(k/2)^{th}$ one by querying the PRF oracle). When the attacker A_π sends its challenge query, then A_f computes all the self-generated round functions honestly, but sends the $(k/2)^{th}$ round value as its PRF challenge query and uses the challenge query response as the $(k/2)^{th}$ round function output. It then continues with the post-challenge phase as it did before the challenge query. Finally, it simply gives the same output as the PRP adversary A_π (i.e. guess PRF if A_π guesses PRP, else guess random).

We note that the PRF adversary succeeds if the following conditions all hold: (1) the PRP adversary A_π succeeds, (2) the $(k/2)^{th}$ round value in the challenge query is never required in any other query and, (3) the PRP challenge output looks random if the PRF challenge response is random. Thus, we have that,

$$Adv(A_f) \geq \Pr[(A_\pi \text{ succeeds}) \wedge (\text{no } (k/2)^{th} \text{ round collision}) \wedge (\text{PRF random} \Rightarrow \text{PRP random})] - \frac{1}{2}$$

Now we can estimate the probability in the above expression as,

$$\begin{aligned}
& \Pr[(A_\pi \text{ succeeds}) \wedge (\text{no } (k/2)^{th} \text{ round collision}) \wedge (\text{PRF random} \Rightarrow \text{PRP random})] \\
\geq\ & \Pr[(A_\pi \text{ succeeds}) \wedge (\text{PRF random} \Rightarrow \text{PRP random}) \,\big|(\text{no collision in } \{\tfrac{k}{2}-1, \tfrac{k}{2}+1\})] \\
& \cdot \Pr[\text{no collision in } \{\tfrac{k}{2}-1, \tfrac{k}{2}+1\}] \\
\geq\ & \Pr[(A_\pi \text{ succeeds}) \,\big|(\text{PRF random} \Rightarrow \text{PRP random}) \wedge ((\text{no collision in } \{\tfrac{k}{2}-1, \tfrac{k}{2}+1\})] \\
& \cdot \Pr[(\text{PRF random} \Rightarrow \text{PRP random}) \,\big|(\text{no collision in } \{\tfrac{k}{2}-1, \tfrac{k}{2}+1\})] \\
& \cdot \Pr[\text{no collision in } \{\tfrac{k}{2}-1, \tfrac{k}{2}+1\}] \\
\geq\ & \Pr[(A_\pi \text{ succeeds}) \,\big|(\text{PRF random} \Rightarrow \text{PRP random}) \wedge ((\text{no collision in } \{\tfrac{k}{2}-1, \tfrac{k}{2}+1\})] \\
& \cdot \Pr[(\text{PRF random} \Rightarrow \text{PRP random}) \,\big|(\text{no collision in } \{\tfrac{k}{2}-1, \tfrac{k}{2}+1\})] \cdot (1-\epsilon_{xor}) \\
\geq\ & \Pr[(A_\pi \text{ succeeds}) \,\big|(\text{PRF random} \Rightarrow \text{PRP random}) \wedge ((\text{no collision in } \{\tfrac{k}{2}-1, \tfrac{k}{2}+1\})] \\
& \cdot(1-2\epsilon_f) \cdot (1-\epsilon_{xor}) \\
\geq\ & \left(\epsilon_\pi - 2\epsilon_f \cdot \left(2qk\epsilon_f + \tfrac{(qk)^6}{2^n}\right)\right) \cdot (1-2\epsilon_f) \cdot \left(1 - 2qk\epsilon_f - \tfrac{(qk)^6}{2^n}\right) \\
\Rightarrow \epsilon_\pi =\ & \mathcal{O}\left(qk\epsilon_f + \tfrac{(qk)^6}{2^n}\right)
\end{aligned}$$

In the above argument, we have used ϵ_f to bound the advantage of all of out PRF adversaries (in L.H.S. as well as R.H.S.). This bound ϵ_f is the maximum advantage of a PRF adversary running in time $\mathcal{O}(t + (qk)^5)$, where t, q are the running time and number of queries made by the PRP attacker A_π. The initial two steps of the above argument can be derived as simple conditional probability manipulations. The third step can be derived as a result of corollary 4.3, that says that the advantage of the collision finding attacker is no more than that of a 5-XOR finding attacker.

In the fourth step, we use the fact that if a PRF family yields secure and independent PRFs, then the usual PRF attack definition is equivalent to a modified definition where the attacker has access to two independently sampled PRFs from the same family. In the challenge phase of this new attack scenario, either random or pseudorandom responses are given to challenge queries to both these functions. Since the attacker is not permitted to query the PRF oracles on these challenge queries, we need the property that no round value collision occur among round values in $\{(k/2) - 1 \ldots (k/2) + 1\}$. $\square$