How to Construct an Ideal Cipher from a Small Set of Public Permutations

Rodolphe Lampe[1]* and Yannick Seurin[2]**

[1] University of Versailles, France
[2] ANSSI, Paris, France
rodolphe.lampe@gmail.com,yannick.seurin@m4x.org

Abstract. We show how to construct an ideal cipher with n-bit blocks and n-bit keys (*i.e.* a set of 2^n public n-bit permutations) from a small constant number of n-bit random public permutations. The construction that we consider is the *single-key iterated Even-Mansour cipher*, which encrypts a plaintext $x \in \{0,1\}^n$ under a key $k \in \{0,1\}^n$ by alternatively xoring the key k and applying independent random public n-bit permutations $P_1, \dots, P_r$ (this construction is also named a *key-alternating cipher*). We analyze this construction in the plain indifferentiability framework of Maurer, Renner, and Holenstein (TCC 2004), and show that twelve rounds are sufficient to achieve indifferentiability from an ideal cipher. We also show that four rounds are necessary by exhibiting attacks for three rounds or less.

Keywords: block cipher, ideal cipher, iterated Even-Mansour cipher, key-alternating cipher, indifferentiability.

1 Introduction

Block Ciphers. Block ciphers are one of the most important classes of primitives in cryptography. They are mainly used to provide confidentiality and authenticity to communication channels or local data storage means, but also to construct hash functions and in other more advanced cryptographic tasks. Syntactically, a block cipher E with message space $\{0,1\}^n$ and key space $\{0,1\}^m$ is a mapping from $\{0,1\}^m \times \{0,1\}^n$ to $\{0,1\}^n$ such that for each key $k \in \{0,1\}^m$, $E(k, \cdot)$ is an (efficiently invertible) permutation. Block cipher designs (virtually all of which rely on the iteration of some key-dependent round function) can be roughly split into two families (with some rare exceptions such as IDEA):

1) Feistel networks [23] and their generalizations, where the round function is given by $(x, y) \mapsto (y, x \oplus F(k_i, y))$, where x and y are the left and right $n/2$ bits of the state, and k_i is the round key; prominent examples include DES, Blowfish, KASUMI, and Camellia for "classical" Feistel networks, and CAST-256 and MARS for generalized Feistel networks;

* This author is partially supported by the French Direction Générale de l'Armement.
** This author is partially supported by the French National Agency of Research: ANR-11-INS-011.

2) substitution-permutation networks (SPNs), where one round generally consists of the composition of a round-key addition, a non-linear mixing layer, and a linear diffusion layer; notable examples include AES, SAFER, CRYPTON, SERPENT, PRESENT, and LED.

At an even higher design level, SPNs can be described (by collapsing the non-linear mixing layer and the linear diffusion layer at i-th round into a single n-bit permutation P_i) as successive applications of round-key additions and permutations P_i. Such a structure was named a *key-alternating cipher* by the designers of AES [17,18].

The traditional security notion for a block cipher is pseudorandomness, *i.e.* indistinguishability from a random permutation [41]: namely, no distinguisher with reasonable resources and having black-box access to a permutation (and also to its inverse in a more stringent variant of the security notion) should be able to distinguish whether it is interacting with the block cipher $E(k, \cdot)$ for a randomly chosen key k, or with a truly random permutation. In an asymptotic and more theoretical language, a family of block ciphers indexed by a security parameter meeting this security notion is called a pseudorandom permutation (PRP), or a strong pseudorandom permutation (SPRP) when the distinguisher has also access to the inverse permutation. The classical example of a construction for which we have some provable security results with respect to indistinguishability is the Feistel network. Starting from the seminal Luby-Rackoff paper [42] which showed that the Feistel construction with three rounds yields a PRP when its round functions are pseudorandom [28], and followed by a paper by Patarin [49] showing that four rounds yield a SPRP (which was stated in [42] without proof), a long series of works established refined results in the same vein, such as [43,44,59,50] to name a few.

The Ideal Cipher Model. Though there are numerous examples where the standard pseudorandomness assumption is sufficient to prove (in a reductionist sense) the security of a cryptographic scheme (*e.g.* for building a symmetric encryption scheme [3] or a MAC scheme [4]), there are also some settings where it might not be strong enough to derive a security proof. Indeed, in some situations, the adversary has more abilities than merely querying in a black-box way an encryption/decryption oracle. For example, there are some cases where the attacker might have access to a more powerful “related-key” oracle [9,5,1], *i.e.* it can ask encryption and decryption queries for keys that are related (in some limited and attack-dependent way) to the main key of the system.

Ideally, the ultimate security goal for a block cipher would be that it “behaves” as a random and independent permutation for each possible key. This naturally leads to the so-called *ideal cipher model* (ICM), the origin of which can be traced back to Shannon [56]. In the ICM, a block cipher E with n-bit blocks and m-bit keys is drawn at random from the set of $(2^n!)^{2^m}$ possible block ciphers of this form, and made available through oracle queries (for both encryption and decryption) to all parties (including the adversary). This is very similar in spirit to the random oracle model (ROM) [24,8] used to model a perfect hash function. To the best of our knowledge, this model was first formally used in a security proof by Winternitz [60] and later by Merkle [47] to show respectively the pre-image and collision resistance of the Davies-Meyer compression function. The ICM became increasingly popular after Black *et al.* [12] used it to extensively analyze the security of the PGV block cipher-based compression functions [51]. Since then, the ICM has been used to prove the security of a variety of other block cipher-based hash functions [30,31,58,40,46], of key length extension methods for block ciphers [35,21,7,25,26], of symmetric encryption schemes [33], and even of some public-key protocols such as signature schemes [29], ring signature schemes [53], public-key encryption [34], and key exchange protocols [6]. Despite these numerous successful applications, one must not lose from sight that the ICM only gives heuristic insurance just as the ROM [14]. In particular, Black [11] exhibited an (arguably artificial) block cipher-based hash function which is provably collision resistant in the ICM, but becomes insecure when the ideal cipher is instantiated with any concrete block cipher.

With the ICM at hand, the question now becomes: is it possible to argue that a given block cipher design is as close as possible to an ideal cipher? In the standard model, one immediately faces the problem that, unlike for pseudorandomness, it even seems hard to come with a satisfactory definition of what this formally means, without running into impossibility results (similarly to [14] and [11]) following from the fact that a concrete block cipher has a short description, whereas an ideal cipher does not. This unfortunate state of affairs has not prevented cryptanalysts from *disproving* that a concrete block cipher behaves as an ideal cipher by exhibiting some non-random behavior, *i.e.* some non-trivial[3] relation between inputs and outputs of the block cipher that can be found faster than for an ideal cipher, in a setting where the key is random and given to the attacker (known-key attacks), or when the attacker can freely choose the key(s) (chosen-key attacks). A classical example is the complementation property of DES which, despite being often viewed as a "benign" undesirable property, implies that DES does not behave as an ideal cipher. For AES, no such non-random properties were known until Biryukov *et al.* [10] showed that so-called q-multicollisions can be found faster for AES-256 than for an ideal cipher. Known-key and chosen-key attacks were first put forward as an important cryptanalysis goal by Knudsen ans Rijmen [36], and have since then become an active area of research [48,27,54].

Indifferentiability. Though we cannot hope to formalize (not to say prove) that a concrete block cipher behaves as an ideal cipher in any reasonable sense in the standard model, it is possible to obtain positive results in idealized models, *i.e.* by viewing some subcomponent of the block cipher as perfectly random. This perfect subcomponent is made available to all parties as a public oracle, which makes this setting formally distinct from classical indistinguishability. In order to assess whether a cryptographic construction based on an ideal subcomponent

[3] We stress that because of the lack of a rigorous definition, the meaning of non-trivial here is somehow subjective.

is secure, one has to employ the formalism of *indifferentiability*, introduced by Maurer *et al.* [45]. A construction $\mathcal{C}$ (*e.g.* a block cipher), based on some ideal primitive $\boldsymbol{F}$ (*e.g.* a random permutation), is said to be indifferentiable from some target ideal primitive $\boldsymbol{G}$ (*e.g.* an ideal cipher) if there exists an efficient simulator $\mathcal{S}$ (with black-box access to the primitive $\boldsymbol{G}$) such that the two systems $(\mathcal{C}^{\boldsymbol{F}}, \boldsymbol{F})$ and $(\boldsymbol{G}, \mathcal{S}^{\boldsymbol{G}})$ are indistinguishable. Informally, the goal of the simulator is to provide answers which are consistent with what a distinguisher can obtain from $\boldsymbol{G}$, without deviating too much from the distribution of answers of $\boldsymbol{F}$. An indifferentiability result can be interpreted as a way to make sure that the high-level design of the construction $\mathcal{C}$ has no structural defect. More importantly, a composition theorem [45] asserts that if $\mathcal{C}^{\boldsymbol{F}}$ is indifferentiable from $\boldsymbol{G}$, then any cryptosystem proved secure when used with $\boldsymbol{G}$ remains secure when used with $\mathcal{C}^{\boldsymbol{F}}$, therefore allowing modular proofs of security in idealized models.[4]

Soon after its introduction, Coron *et al.* [15] used the indifferentiability framework to revisit the design of a hash function from an ideal cipher: namely they showed that a number of variants of the Merkle-Damgård domain extension method [19,47], used with an ideal cipher in Davies-Meyer mode, are indifferentiable from a random oracle. The converse direction, *i.e.* proving that it is possible to construct an ideal cipher from a random oracle, turned out to be harder to achieve. A first attempt to prove that the Feistel construction with public random round functions is indifferentiable from a random permutation (and hence from an ideal cipher by prepending the key to each input to the random round functions) was made by Coron *et al.* for six rounds [16], and later by Seurin for ten rounds [55], but serious flaws were found in both proofs [37,32]. The situation was corrected with a proof by Holenstein *et al.* [32] that the 14-round Feistel construction with public random round functions is indifferentiable from a random permutation. This must be contrasted with the classical Luby-Rackoff result stating that the 4-round Feistel construction with pseudorandom round functions yield a SPRP.

Our Contribution. The indifferentiability result for the Feistel construction mentioned above is fundamentally about how to obtain a random permutation from a random (function) oracle. The step to obtain an ideal cipher (*i.e.* an exponential number of independent permutations) is trivially achieved through domain separation of the underlying primitive (namely by prepending the key to each call to the random function oracles). However, it does not tell us anything about how the key should be concretely mixed into the state. In a departure from this approach, we ask the following question: given a small number of objects with n-bit inputs (*e.g.* n-bit permutations $P_1, \ldots, P_r$), is there a way to "combine" them together with an m-bit key in order to obtain a construction which is close to an n-bit block and m-bit key ideal cipher, *i.e.* a set of 2^m independent permutations, without appealing to a trivial domain separation argument? This

[4] Care has to be taken with this composition result when the security definition for the cryptosystem puts some limitations on the adversary, such as an upper bound on its memory [52,20]

naturally prompts us to turn our attention towards the second class of designs, namely key-alternating ciphers.[5] More formally, we consider the construction of a block cipher with n-bit blocks and m-bit keys from r public n-bit permutations $P_1, \ldots, P_r$ defined as follows: derive $(r+1)$ n-bit round keys $(k_0, \ldots, k_r)$ from a master key K through some key derivation function, and encrypt the plaintext $x \in \{0,1\}^n$ by computing the ciphertext y defined as:

$$y = k_r \oplus P_r(k_{r-1} \oplus P_{r-1}(\cdots P_2(k_1 \oplus P_1(k_0 \oplus x)) \cdots)) \ .$$

When $r = 1$ and two independent n-bit keys (k_0, k_1) are used, so that the ciphertext is simply $y = k_1 \oplus P_1(k_0 \oplus x)$, one obtains the so-called Even-Mansour cipher [22]. When P_1 is modeled as a public random permutation (that the adversary can query in a black-box way), Even and Mansour [22] showed that the resulting block cipher is a SPRP, with security ensured up to $\mathcal{O}(2^{n/2})$ distinguisher queries. The indistinguishability of the general construction for $r > 1$ with independent keys $(k_0, \ldots, k_r)$ was later studied for two rounds by Bogdanov *et al.* [13], for three rounds by Steinberger [57], and for any number r of rounds (with non-tight security bounds) by Lampe *et al.* [38]. Unsurprisingly, the number of adversarial queries up to which the key-alternating cipher is indistinguishable from a random permutation increases with the number of rounds. Following [38], and to emphasize that we work in the random permutation model for $P_1, \ldots, P_r$, we will use the naming *r-round iterated Even-Mansour cipher* to designate the idealized key-alternating cipher where the permutations $P_1, \ldots, P_r$ are public and perfectly random permutations oracles.

In this paper, we consider the iterated Even-Mansour cipher from the point of view of indifferentiability, and ask whether this construction is indifferentiable from an ideal cipher for a sufficient number of rounds when the permutations $P_1, \ldots, P_r$ are public and random. A first simple observation is that the construction with $r+1$ independent n-bit keys $(k_0, \ldots, k_r)$ (resulting in a total key space $\{0,1\}^m = \{0,1\}^{(r+1)n}$) is never indifferentiable (for any r) from an ideal cipher with n-bit blocks and $(r+1)n$-bit keys (this had already been informally observed by [13]). In a sense, independent keys offer too much freedom to the attacker, enabling to easily find related-key relations. There are two possible approaches to solve this problem. The first one is to derive the round keys $(k_0, \ldots, k_r)$ from the master key using some cryptographic function (modeled as a random oracle for the indifferentiability proof). This was considered in an earlier and independent work by Andreeva *et al.* [2] (see below for a discussion of their result). The second possibility (not relying on any cryptographic assumption about the key derivation function) is to "correlate" the round keys. This is the approach we adopt: namely, we consider the iterated Even-Mansour cipher where the n-bit round keys $(k_0, \ldots, k_r)$ are obtained by applying efficiently invertible n-bit permutations $(\gamma_0, \ldots \gamma_r)$ to the n-bit master key k (see Figure 1 on page 10). As will appear clearly in view of its proof, the fact that the master key length is equal to the block length is crucial for our result. To insist on this

[5] One could certainly undertake the same study for Feistel-based block ciphers, but this seems more complicated.

particular point, we call this construction the *single-key* iterated Even-Mansour cipher. Our main result is the following one.

Theorem. *The 12-round single-key iterated Even-Mansour cipher with twelve independent random public n-bit permutations $(P_1, \dots, P_{12})$ and any efficiently invertible (public) n-bit permutations $(\gamma_0, \dots, \gamma_{12})$ for the key schedule is indifferentiable from an ideal cipher with n-bit blocks and n-bit keys.*

In fact, the key derivation permutations γ_i will not play any role in the proof, so that we will focus on the simple case where they are all equal to the identity. Additionally, we show that at least four rounds are necessary by describing attacks (using only a constant number of queries) for three rounds or less (see the full version of the paper [39]).

Together with the result of [2] discussed below, our main theorem validates the design strategy underlying SPNs and more generally key-alternating ciphers as a sound way to ensure security beyond pseudorandomness: it (theoretically) enables to achieve resistance against related-key, known-key and chosen-key attacks (that an ideal cipher can withstand). We stress that our result cannot be used as is to take *concrete* design decisions: first, our bounds (as is often the case with indifferentiability results) are extremely loose.[6] More importantly, the permutations P_i used in concrete block ciphers such as AES are often too simple to be deemed close to random permutations (not to say independent: they are often the same).

Our Techniques. The techniques used to prove our main theorem are very similar to the ones introduced in [16,55,32] for the Feistel construction (while the formalism we adopt is very close to [32]). We simply give a very cursory overview of the main ideas here (assuming all γ_i's are the identity). The simulator works by detecting and completing "partial chains" created by the queries of the distinguisher. Define the computation path for a plaintext x and a key k as the sequence of pairs (x_1, y_1), ..., (x_{12}, y_{12}) of corresponding input and output values for the simulated permutations $P_1, \dots, P_{12}$. It must hold that the value y obtained through this computation path matches the value $\boldsymbol{E}(k, x)$ obtained from the ideal cipher, otherwise one could straightforwardly distinguish the "simulated" world from the "real" world. Hence, simply answering the distinguisher queries randomly will not work: the simulator must somehow "adapt" the computation path to match the ideal cipher $\boldsymbol{E}$. Observe now the following important property of the single-key iterated Even-Mansour cipher: given only two consecutive values y_i and x_{i+1} of the computation path (*i.e.* the output value of permutation P_i and the input value to permutation P_{i+1}), it is possible to deduce the corresponding key $k = y_i \oplus x_{i+1}$, and hence to move forward and backward along the path. Note that this property essentially relies on the fact that the master key length is equal to the block length of the permutations (would the master key be larger, then it could not be uniquely determined by

[6] Since the proof is already quite involved, we favored simplicity rather than tightness, but the bounds can probably be improved at some places.

y_i and x_{i+1}). Note also that this is the exact analogue of the property of the Feistel network that the input and output values to two consecutive round functions enable to uniquely move forward and backward inside the construction. With this in mind, the strategy of the simulator will be to detect *partial chains* in computation paths created by queries of the distinguisher to two consecutive permutations, and "complete" them by moving forward and backward inside the iterated Even-Mansour construction (randomly setting undefined permutation values encountered along the way, and making a call to the ideal cipher to "wrap around") until the input x_ℓ and the output y_ℓ for one particular permutation P_ℓ are obtained (but still undefined inside P_ℓ history). This permutation is then "adapted" by setting $P_\ell(x_\ell) := y_\ell$ so that the corresponding input and output for the simulated Even-Mansour cipher and for the ideal cipher match. A moment of thinking should make clear that the simulator cannot complete each and every partial chain created in its history, since this would create a "chain reaction" leading to an exponential running time and an exponential number of ideal cipher queries from the simulator. Hence, one must make a careful and parsimonious choice of "detection zones" for deciding which partial chains to complete. In addition, one must ensure that the simulator never overwrites an entry when adapting permutation P_ℓ, thereby rendering a previously completed chain inconsistent. How exactly this is done is very similar to the case of the Feistel construction [55,32], and we refer to Section 3.1 for a more detailed overview.

As a retrospective afterthought, we note that the Feistel and the iterated Even-Mansour indifferentiability results are not that far apart: they both tell how to construct a "big object" (which in both cases has some specific syntactic constraints which are relevant only from a cryptographic perspective) taking $2n$ bits of input (the left and right n-bit halves of the input in the case of the Feistel network, and the key and the plaintext in the case of the iterated Even-Mansour cipher) from smaller objects with only n bits of input (fourteen n-bit to n-bit functions for the Feistel network, and twelve n-bit permutations for the iterated Even-Mansour cipher).

Related Work. In a prior and independent work [2], Andreeva *et al.* proved a result which is close and complementary to ours: they showed that the iterated Even-Mansour construction with *five* rounds and a key derivation function *modeled as a random oracle* is indifferentiable from an ideal cipher. Though significantly reducing the number of rounds required for the proof to go through, and lifting the restriction that the master key length be equal to the block length of the permutations, their technique puts a strong burden on the key derivation function, which can hardly be seen as close to a random oracle in most concrete block ciphers. In fact, most key schedules, such as the one of AES, are "lightweight" and invertible, which makes our result (where the key derivation function has no cryptographic role) more relevant to practice. On the other hand, the bounds obtained by [2] are better: the number of queries, the running time, and the indistinguishability bound achieved by their simulator are respectively $\mathcal{O}(q^2)$, $\mathcal{O}(q^3)$, and $\mathcal{O}(q^{10}/2^n)$, while for our simulator they are respectively $\mathcal{O}(q^4)$, $\mathcal{O}(q^6)$, and $\mathcal{O}(q^{12}/2^n)$.

Taken together, the two results indicate, not too surprisingly, that using a cryptographically strong key schedule, though not being necessary, enables to lower the number of rounds needed to obtain an ideal cipher (however this interpretation must be taken cautiously: it may well be that, say, the iterated Even-Mansour cipher with four rounds is indifferentiable from an ideal cipher, independently of the cryptographic strength of the key schedule).

Regarding the purely theoretical question of the minimal number of n-bit permutations needed to construct an n-bit block and n-bit key ideal cipher, it was additionally showed in [2] that six independent permutations is sufficient, by using a 5-round key-alternating cipher and an independent random permutation P_0 to build a key derivation function $k \mapsto P_0(k) \oplus k$.

2 Preliminaries

2.1 Notation and Definitions

Given a finite non-empty set S, we write $s \leftarrow_{\$} S$ to mean that a value is sampled uniformly at random from S and assigned to s. The security parameter will be denoted n and will be identified with the block length of permutations in the Even-Mansour construction. We will write $f \in \texttt{poly}(n)$ to denote a polynomially bounded function and $f \in \texttt{negl}(n)$ to denote a negligible function. For $\delta \in \{+,-\}$, we denote $\bar{\delta}$ the opposite of δ.

In the following, we will use calligraphic fonts $(\mathcal{A}, \mathcal{B}, \ldots)$ to denote interactive Turing machines, and typewriter fonts to denote `Procedures` attached to these machines. A distinguisher is an oracle Turing Machine $\mathcal{D}$ which takes as input a security parameter 1^n, has access to a set of oracles $O_1, \ldots, O_m$, and outputs a bit b, an experiment we denote $\mathcal{D}^{O_1,\ldots,O_m} = b$. We will always consider distinguishers that are deterministic and computationally unbounded, and restricted only with respect to the number of oracle queries they make.

An ideal primitive is a probability distribution on some set of functions, and will be denoted with bold fonts. In the corresponding *model*, a function is drawn at random from the corresponding distribution (say $\boldsymbol{F}$) and all parties (say $\mathcal{M}$) involved in the security experiment are given oracle access to the corresponding function, which we simply denote $\mathcal{M}^{\boldsymbol{F}}$. In the following we will consider the following two ideal primitives:

– a random permutation $\boldsymbol{P}_i$ on $\{0,1\}^n$, which is a permutation drawn at random from the set of all permutations on $\{0,1\}^n$, and which can be accessed in the two directions $\boldsymbol{P}_i(x)$ and $\boldsymbol{P}_i^{-1}(y)$; we will use the notation $\boldsymbol{P} = (\boldsymbol{P}_1, \ldots, \boldsymbol{P}_r)$ to denote a tuple of independent random permutations;
– an ideal cipher $\boldsymbol{E}$ with message space and key space $\{0,1\}^n$, which is drawn at random from the set of all block ciphers of this form, and which can be accessed in encryption, denoted $\boldsymbol{E}(k,x)$, and decryption, denoted $\boldsymbol{E}^{-1}(k,y)$.

2.2 Indifferentiability

We recall the usual definition of indifferentiability.

Definition 1. *Let $q, \sigma, t : \mathbb{N} \to \mathbb{N}$ and $\varepsilon : \mathbb{N} \to \mathbb{R}$ be four functions of the security parameter n. A Turing machine $\mathcal{C}$ with oracle access to an ideal primitive $\boldsymbol{F}$ is said to be statistically and strongly $(q, \sigma, t, \varepsilon)$-indifferentiable from an ideal primitive $\boldsymbol{G}$ if there exists an interactive Turing machine $\mathcal{S}$ with oracle access to $\boldsymbol{G}$ such that for any distinguisher $\mathcal{D}$ making at most q queries, $\mathcal{S}$ makes at most σ oracle queries, runs in time at most t, and the following holds:*

$$\left| \Pr\left[\mathcal{D}^{\boldsymbol{G}, \mathcal{S}^{\boldsymbol{G}}} = 1\right] - \Pr\left[\mathcal{D}^{\mathcal{C}^{\boldsymbol{F}}, \boldsymbol{F}} = 1\right] \right| \leq \varepsilon \ .$$

$\mathcal{C}^{\boldsymbol{F}}$ is simply said to be statistically and strongly indifferentiable from $\boldsymbol{G}$ if for any $q \in \mathtt{poly}(n)$, the above definition is fulfilled with $\sigma, t \in \mathtt{poly}(n)$ and $\varepsilon \in \mathtt{negl}(n)$.

This definition does not refer to the running time of $\mathcal{D}$. When only polynomial-time distinguishers are considered, indifferentiability is said to be *computational*. Weak indifferentiability is defined as above, but the order of quantifiers for the distinguisher and the simulator are switched (for all distinguisher, there is a simulator...).

In this paper, and similarly to [32], we will slightly tweak the definition of strong indifferentiability as follows: we will describe a simulator which, for any distinguisher $\mathcal{D}$ making a polynomial number q of queries, runs in time at most t and makes at most σ queries *with overwhelming probability* (rather than probability one) in system $\mathcal{D}^{\boldsymbol{G}, \mathcal{S}^{\boldsymbol{G}}}$. This is not a big concern since any such simulator $\mathcal{S}$ can be transformed into a simulator $\mathcal{S}'$ for weak indifferentiability (which is sufficient for the composition theorem of [45] to hold) which takes the maximal number of queries q of $\mathcal{D}$ as input, and aborts when its number of queries becomes larger than σ (computed as a function of q), hence making at most σ queries with probability one.

2.3 The Iterated Even-Mansour Cipher

Fix an integer $r \geq 1$. Let $P = (P_1, \ldots, P_r)$ be a tuple of permutations on $\{0,1\}^n$. The r-round iterated Even-Mansour construction associated with P, denoted $\bar{\mathcal{C}}_r^P$, is the block cipher with message space $\{0,1\}^n$ and key space $(\{0,1\}^n)^{r+1}$ which maps a message x and a key $(k_0, \ldots, k_r)$ to the ciphertext defined by:

$$\bar{\mathcal{C}}_r^P((k_0, \ldots, k_r), x) = k_r \oplus P_r(k_{r-1} \oplus P_{r-1}(\cdots P_2(k_1 \oplus P_1(k_0 \oplus x)) \cdots)) \ .$$

Let $\gamma = (\gamma_0, \ldots, \gamma_r)$ be a tuple of efficiently invertible permutations on $\{0,1\}^n$. The *single-key* r-round iterated Even-Mansour construction associated with P and γ, denoted $\mathcal{C}_r^{P,\gamma}$, is the block cipher with message space $\{0,1\}^n$ and key space $\{0,1\}^n$ which maps a message x and a key k to the ciphertext defined by (see Figure 1):

$$\mathcal{C}_r^{P,\gamma}(k, x) = \gamma_r(k) \oplus P_r(\gamma_{r-1}(k) \oplus P_{r-1}(\cdots P_2(\gamma_1(k) \oplus P_1(\gamma_0(k) \oplus x)) \cdots)) \ .$$

In all the following, we will focus on the case where all permutations γ_i are the identity, and simply denote $\mathcal{C}_r^P$ the resulting cipher, namely:

$$\mathcal{C}_r^P(k, x) = k \oplus P_r(k \oplus P_{r-1}(\cdots P_2(k \oplus P_1(k \oplus x)) \cdots)) \ .$$

We stress that our main result (Theorem 1) holds for arbitrary permutations γ_i as long as they are efficiently invertible.

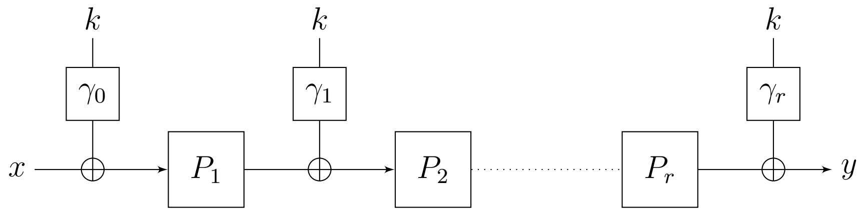

Fig. 1. The single-key iterated Even-Mansour cipher with r rounds $\mathcal{C}_r^{P,\gamma}$. We focus in this paper on the special case $\gamma_i = \mathrm{Id}$ for $i = 0, \dots, r$.

3 Indifferentiability for Twelve Rounds

In this section we prove the main result of this paper, which is the following theorem.

Theorem 1. *For any q, the 12-round single-key iterated Even-Mansour cipher $\mathcal{C}_{12}^{\boldsymbol{P},\gamma}$ with twelve independent random n-bit permutations $\boldsymbol{P} = (\boldsymbol{P}_1, \dots, \boldsymbol{P}_{12})$, and fixed, efficiently invertible n-bit permutations $\gamma = (\gamma_0, \dots, \gamma_{12})$ for the key schedule, is strongly and statistically $(q, \sigma, t, \varepsilon)$-indifferentiable from an ideal cipher $\boldsymbol{E}$ with n-bit blocks and n-bit keys, where:*

$$\sigma = 2^7 \times q^4, \quad t = \mathcal{O}(q^6), \quad \text{and} \quad \varepsilon = \frac{2^{91} \times q^{12}}{2^n} \;.$$

To prove this, we will describe an efficient simulator $\mathcal{S}$, and show that the two systems $(\mathcal{C}_{12}^{\boldsymbol{P},\gamma}, \boldsymbol{P})$ and $(\boldsymbol{E}, \mathcal{S}^{\boldsymbol{E}})$ are indistinguishable. For simplicity we focus on the case where all γ_i's are the identity, but the generalization is straightforward.

Notational convention. In all this section, we will use the following useful notational convention: we will interchangeably denote the input to the ideal cipher or the iterated Even-Mansour cipher x or y_0, and the output y or x_{13}.

3.1 Informal Description of the Simulator

We start with a high-level view of the simulator (see also Figure 2). It offers an interface to the distinguisher for querying the simulated permutations, which formally takes the form of a public procedure `Query`(i, δ, z), where $i \in \{1, \dots, 12\}$ names the permutation, $\delta \in \{+, -\}$ tells whether this is a direct or indirect query, and $z \in \{0, 1\}^n$ is the actual value queried. The simulator maintains an history for the simulated permutations under the form of hash tables $P_1, \dots, P_{12}$. Each such table maps entries $(\delta, z) \in \{+,-\} \times \{0,1\}^n$ to values $z' \in \{0,1\}^n$. We denote P_i^+, resp. P_i^-, the (time-dependent) sets of strings $z \in \{0,1\}^n$ such that $P_i(+,z)$, resp. $P_i(-,z)$, is defined. When the simulator receives a query (i, δ, z), it looks in hash table P_i to see whether the corresponding answer $P_i(\delta, z)$ is already defined. When this is the case, it outputs the answer and waits for the next query. Otherwise, it draws a uniformly random answer z' and defines in hash table $P_i(\delta, z) := z'$, as well as the answer to the opposite query $P_i(\bar{\delta}, z') := z$ (note that this last assignment may overwrite an entry in P_i).

Additionally, before outputting the answer z', and for some specific values of (i, δ), the simulator triggers a *chain detection* mechanism followed by a *chain completion* mechanism to ensure consistency of its answers with the ideal cipher $\boldsymbol{E}$. An essential point to notice about the iterated Even-Mansour cipher in order to understand these mechanisms is that given an output value y_i for permutation P_i and an input value x_{i+1} for permutation P_{i+1}, it is possible to compute the corresponding key $k = y_i \oplus x_{i+1}$, and therefore to move forward and backward in the construction up and down to the corresponding input x and output y to the cipher. Hence, any tuple (y_i, x_{i+1}, i) (a so-called *partial chain* later in the reasoning) defines a unique computation path inside the whole construction. This is the exact analogue of the property of the Feistel construction that the input values to two consecutive round functions uniquely define the computation path inside the Feistel network.

There are exactly six such values of (i, δ) for which the simulator performs additional steps: $(2,+)$, $(6,+)$, $(6,-)$, $(7,+)$, $(7,-)$, and $(11,-)$. The cases $(2,+)$ and $(11,-)$ are similar. When receiving a query $(2,+,x_2)$ for which the answer is still undefined, the simulator, after having drawn a random answer y_2 to this query, considers all values $y_1 \in P_1^-$, computes the corresponding key $k := y_1 \oplus x_2$, and moves backward in the iterated Even-Mansour cipher by computing $x_1 := P_1(-, y_1)$, $y_0 := x_1 \oplus k$, $x_{13} := \boldsymbol{E}(k, y_0)$ (hence making a query to the ideal cipher), and $y_{12} := x_{13} \oplus k$, and checks whether $y_{12} \in P_{12}^-$. When this is the case, it enqueues in a queue QUEUE the tuple $(y_0, x_1, 0, 4)$. The first three elements $(y_0, x_1, 0)$ specify the partial chain that must be completed, while the last element $\ell = 4$ specifies which permutation will be adapted during completion of the chain to ensure consistency with $\boldsymbol{E}$. The behavior of the simulator when receiving a query $(11, -, y_{11})$ is symmetric: after having drawn a random answer x_{11}, for all $x_{12} \in P_{12}^+$, it moves forward in the iterated Even-Mansour cipher to check whether the corresponding value x_1 is in P_1^+, and if so enqueues the corresponding tuple $(y_0, x_1, 0, 9)$ (note that in this case adaptation will take place at permutation P_9).

The four remaining cases $(i, \delta) = (6,+)$, $(6,-)$, $(7,+)$, and $(7,-)$ are similar, except that there is no check: the simulator enqueues a tuple $(y_6, x_7, 6, \ell)$ for each newly generated pair $(y_6, x_7) \in P_6^- \times P_7^+$. If this was a query with $i = 6$, then adaptation will take place at $\ell = 4$, while if this was a query with $i = 7$, adaptation will take place at $\ell = 9$. Assume for a concrete example that the simulator receives a query $(6, +, x_6)$ whose answer is undefined yet. Then it draws a random answer $y_6 \leftarrow_\$ \{0,1\}^n$, and enqueues $(y_6, x_7, 6, 4)$ for all $x_7 \in P_7^+$.

Immediately after having enqueued newly created chains (y_i, x_{i+1}, i, ℓ), the simulator starts completing the partial chains, by dequeuing tuples from Queue. For this, when dequeuing (y_i, x_{i+1}, i, ℓ), it computes the key $k := y_i \oplus x_{i+1}$, and moves forward and backward in the iterated Even-Mansour cipher, possibly defining missing permutations values $P_i(+, \cdot)$ or $P_i(-, \cdot)$, and making a query to $\boldsymbol{E}(k, \cdot)$ to "wrap around", until it reaches the input value x_ℓ for P_ℓ (when moving forward) and the corresponding output y_ℓ (when moving backward). It finally "adapts" permutation P_ℓ by setting $P_\ell(+, x_\ell) := y_\ell$ and $P_\ell(-, y_\ell) := x_\ell$ in order to ensure consistency of the entire chain with $\boldsymbol{E}$. It also adds chains that have been completed in a set Completed in order to avoid completing them twice. While completing a chain and adding possibly missing permutation values, the simulator uses the same chain detection mechanism as when receiving a direct query from the distinguisher. Hence new tuples may be enqueued while dequeuing and completing a chain, and the simulator keeps dequeuing tuples until the queue is empty. When this is the case, it returns the answer to the original query of the distinguisher.

As in the indifferentiability proof of the Feistel construction, there will be two crucial points to show: first, that the recursive chain completion mechanism terminates in polynomial time (except maybe with negligible probability); second, that the simulator can always adapt, *i.e.* that it never has (or only with negligible probability) to overwrite previously defined entries when adapting a chain, which would render previously completed chains inconsistent with the ideal cipher $\boldsymbol{E}$. Permutations P_3, P_5, P_8, and P_{10} (*i.e.* the permutations surrounding the two adaptation rounds P_4 and P_9) will play a key role while proving this last point: they will ensure that no bad collisions occur at the input or output of the two permutations used for adapting chains.

We defer the formal definition of the simulator to the full version of the paper [39].

3.2 Sketch of the Proof of Theorem 1

We sketch the main ideas of the proof of Theorem 1. The detailed proof is deferred to the full version of the paper [39].

We use intermediate systems that are depicted on Figure 3. System Σ_1 is the simulated world $(\boldsymbol{E}, \mathcal{S}^{\boldsymbol{E}})$, while Σ_4 is the real world $(\mathcal{C}_{12}^{\boldsymbol{P}}, \boldsymbol{P})$. In system Σ_2, the ideal cipher $\boldsymbol{E}$ is replaced with a so-called keyed two-sided random function $\mathcal{F}(\eta)$ which offers the same interface for encryption and decryption as the ideal cipher. However, when asked for an encryption query (k, x) or a decryption query (k, y), $\mathcal{F}$ first checks (by maintaining a hash table denoted E) whether this value appeared in a previous query, and if so answers consistently. Otherwise it draws a uniformly random answer (the randomness is made explicit through a uniformly random table η) and updates E. Besides, $\mathcal{F}$ has an additional interface $\mathcal{F}.\texttt{Check}(k, x, y)$ (only used by the simulator) which returns true if and only if $E(+, k, x) = y$ or $E(-, k, y) = x$ (in particular, if neither (k, x) was queried for encryption nor (k, y) for decryption, $\texttt{Check}(k, x, y)$ returns false). In Σ_2, the simulator $\mathcal{S}$ is slightly modified into a new simulator $\mathcal{T}$ which queries $\texttt{Check}$

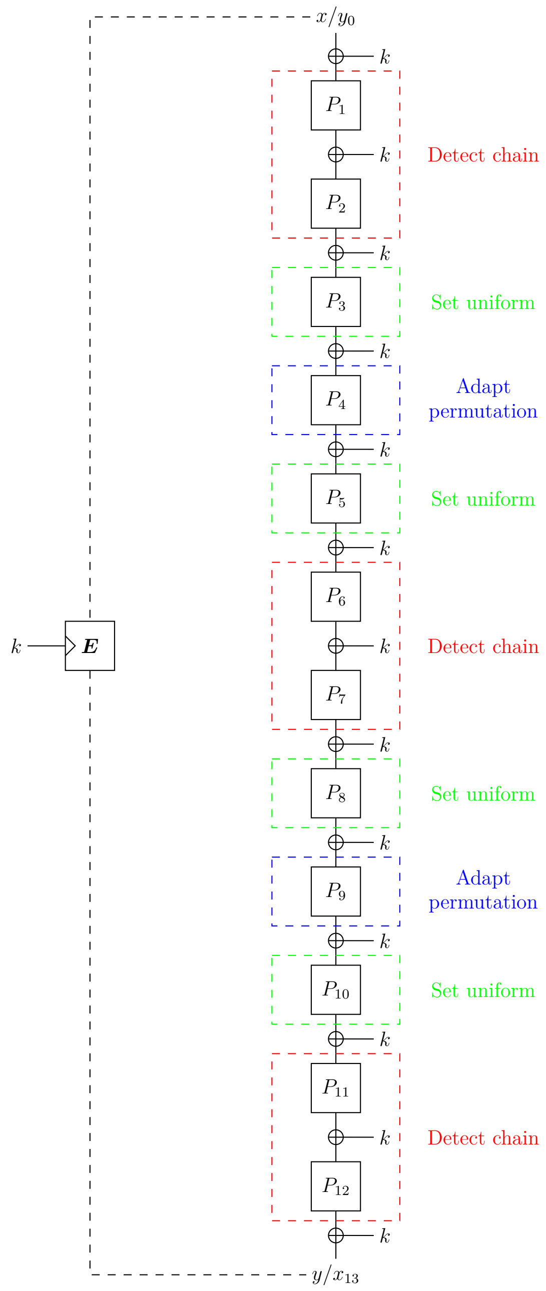

Fig. 2. Detection and adaptation zones used by the simulator.

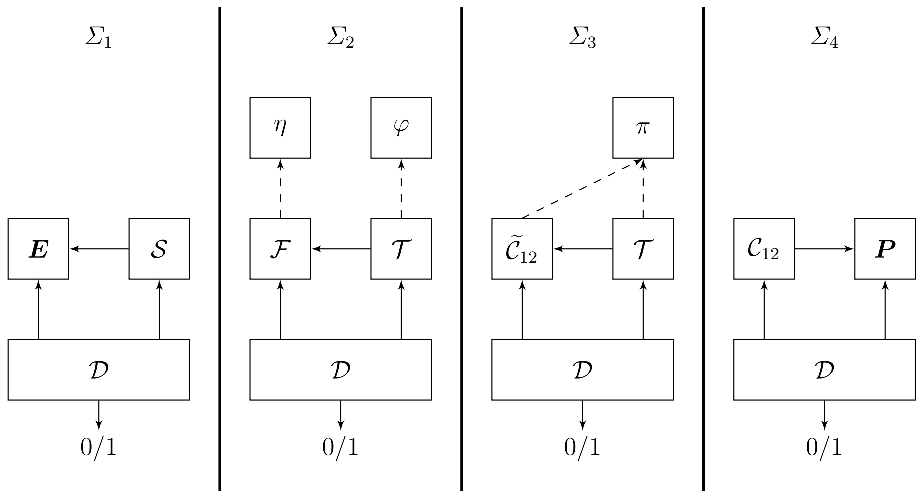

Fig. 3. Systems used in the indifferentiability proof.

rather than the encryption or decryption interface when deciding whether a tuple $(y_0, x_1, 0, \ell)$ must be enqueued. Moreover the randomness of $\mathcal{T}$ is made explicit with uniformly random tables $\varphi = (\varphi_1, \ldots, \varphi_{12})$. In system Σ_3, the keyed two-sided random function is replaced with an iterated Even-Mansour cipher using uniformly random permutations $\pi = (\pi_1, \ldots, \pi_{12})$, enhanced with a `Check` procedure similarly to $\mathcal{F}$. The simulator $\mathcal{T}$ now uses tables π as well for its random draws.

To prove Theorem 1, we will upper bound the statistical distance between successive worlds Σ_i. Additionally, we must show that $\mathcal{S}$ makes a polynomial number of oracle queries and runs in polynomial time in Σ_1 with overwhelming probability. We start the analysis in Σ_2: namely we show that in this system, $\mathcal{T}$ will always complete at most q chains of the form $(y_0, x_1, 0, \ell)$. The reason for this is quite simple: since $\mathcal{T}$ uses interface $\mathcal{F}$.`Check` to decide whether such a tuple must be enqueued, such a chain can be detected and enqueued only if (k, y_0) with $k = y_0 \oplus x_1$ appeared in the queries (or the answers) of the distinguisher to $\mathcal{F}$. Since by assumption the distinguisher makes at most q queries, this implies the result. Starting from this observation, one can then upper bound the size of the hash tables P_i maintained by the simulator as well as the number of queries of $\mathcal{T}$ to $\mathcal{F}$.

We then upper bound the statistical distance between Σ_1 and Σ_2. For this, we appeal to a previous result from [32] to obtain the following lemma.

Lemma 1. *For any distinguisher $\mathcal{D}$ which makes at most q queries in total, we have:*

$$\left|\Pr\left[\mathcal{D}^{\Sigma_1} = 1\right] - \Pr\left[\mathcal{D}^{\Sigma_2(\eta,\varphi)} = 1\right]\right| \leq \frac{2^{22} \times q^{12}}{2^n} \quad .$$

As a side result, this directly implies that with overwhelming probability, $\mathcal{S}$ runs in polynomial time and makes a polynomial number of queries to $\boldsymbol{E}$ in system Σ_1, as captured by the following lemma.

Lemma 2. *Assume that the distinguisher $\mathcal{D}$ makes at most q queries in total. Then with probability greater than $1 - 2^{21} \times q^{12}/2^n$ over an execution of $\mathcal{D}^{\Sigma_1}$, the simulator $\mathcal{S}$ makes at most $2^7 \times q^4$ queries to $\boldsymbol{E}$ or $\boldsymbol{E}^{-1}$ (assuming $\mathcal{S}$ never repeats a query), and runs in time at most $\mathcal{O}(q^6)$.*

We then move to the hard part of the proof, which is to upper bound the statistical distance between Σ_2 and Σ_3. For this, an important first step is to show that in Σ_2, the simulator never (more precisely only with negligible probability) overwrites an entry in hash tables P_i during a call to `ForceVal` (*i.e.* the procedure which adapts chains by forcing the value of permutation P_4 or P_9). To reason about the behavior of system Σ_2, we introduce the concept of *partial chain*, which is simply a tuple (y_i, x_{i+1}, i) for $i \in \{1, \ldots, 12\}$. Considering, at some point in the execution, hash tables $P_1, \ldots, P_{12}$ maintained by the distinguisher and the hash table E maintained by $\mathcal{F}$, we define for any partial chain $C = (y_i, x_{i+1}, i)$ and any $\ell \in \{1, \ldots, 12\}$ the functions $\mathtt{val}^+_\ell(C)$ and $\mathtt{val}^-_\ell(C)$ as follows: $\mathtt{val}^+_\ell(C)$ is defined as the direct input value x_ℓ to permutation P_ℓ obtained when moving forward in the Even-Mansour construction (possibly looking in hash table E to wrap around), or $\perp$ is at some point the computation stops because the necessary value was missing in some hash table (including E). Similarly $\mathtt{val}^-_\ell(C)$ is defined as the indirect input value y_ℓ to permutation P_ℓ obtained when moving backward in the Even-Mansour construction, or $\perp$ if the computation stops at some point.

As a preliminary step, we need to exclude some bad events that lead to a pathological behavior of Σ_2. These bad events correspond to the draw of bad values when the simulator randomly defines the value of some permutation P_i or when $\mathcal{F}$ draws a random answer. More precisely, the bad values are exactly those that can be written as the bitwise xor of up to five values in the history, where the history includes all n-bit strings appearing in hash tables P_i and E at the moment where the random answer is drawn. Since the size of the history remains polynomial, the probability of these bad events is negligible.

Then, the proof that the simulator never overwrites an entry in hash tables P_i during a call to `ForceVal` roughly consists of two steps. First, we show that just before the query which leads to some partial chain C being enqueued to be adapted at position ℓ, one has $\mathtt{val}^+_{\ell-1}(C) = \perp$ and $\mathtt{val}^-_{\ell+1}(C) = \perp$, unless an equivalent chain B (where equivalent means that one can obtain B from C by moving forward or backward in the Even-Mansour construction) has been previously enqueued. This crucially relies on fact that the two chain detection zones ("border" and "center") are "protecting" each other. For example, consider the case where some chain $C = (y_0, x_1, 0)$ is enqueued to be adapted at position $\ell = 4$ due to a query for $P_2(x_2)$. Then clearly, before $P_2(x_2)$ is defined, one has $\mathtt{val}^+_3(C) = \perp$. On the other side, if $\mathtt{val}^-_5(C) \neq \perp$, then this means that C is equivalent to some partial chain $B = (y_6, x_7, 6)$ with $y_6 \in P_6^-$ and $x_7 \in P_7^+$, so that D would have been enqueued previously due to some query to P_6 or P_7.

The second step is to show that between the moment where C is enqueued, and the moment where C is dequeued, the completion of other chains (possibly) in the queue will not lead to $\mathtt{val}^+_{\ell-1}(C) \in P^+_{\ell-1}$ or $\mathtt{val}^-_{\ell+1}(C) \in P^-_{\ell+1}$. In particular this requires to show that C cannot collision with another, previously enqueued chain D at round $\ell-1$ or $\ell+1$. This is carried out via a careful analysis of all the ways this could happen, which would all imply the occurrence of the bad event previously discussed. Once this is done, it is easy to show that no entry is overwritten during the call to `ForceVal` when adapting C. To finalize the reasoning, we use a randomness mapping argument similar to the one that was introduced in [32], and obtain the following lemma.

Lemma 3. *For any distinguisher $\mathcal{D}$ which makes at most q queries in total, we have:*

$$\left|\Pr\left[\mathcal{D}^{\Sigma_2(\eta,\varphi)} = 1\right] - \Pr\left[\mathcal{D}^{\Sigma_3(\pi)} = 1\right]\right| \leq \frac{2^{89} \times q^{12}}{2^n} \ .$$

Finally, upper bounding the statistical distance between Σ_3 and Σ_4 is easily handled, and yields the following lemma.

Lemma 4. *For any distinguisher $\mathcal{D}$ which makes at most q queries in total, we have:*

$$\left|\Pr\left[\mathcal{D}^{\Sigma_3(\pi)} = 1\right] - \Pr\left[\mathcal{D}^{\Sigma_4} = 1\right]\right| \leq \frac{2^{89} \times q^{12}}{2^n} \ .$$

Combining Lemmas 1, 2, 3, and 4 finally enables to prove Theorem 1.

Remark 1. Our choice to use a keyed two-sided random *function* and a simulator $\mathcal{T}$ accessing random *function* tables φ in system Σ_2 allows to handle uniformly random values, which slightly simplifies the computation of various bounds in the proof. It is however possible (and conceptually more satisfying) to use an ideal cipher enhanced with a `Check` procedure rather than a keyed two-sided random function, and random permutation tables rather than random function tables. This would have some nice effects in the analysis of system Σ_2, in particular this would exclude some bad events such as potential overwrites in the hash table E when $\mathcal{F}$ defines an answer by reading table η or in hash tables P_i when $\mathcal{T}$ defines an answer by reading tables φ_i. This kind of approach was taken in [2].

Remark 2. If one contents oneself with weak indifferentiability (where the simulator is allowed to depend on the distinguisher), one can slightly simplify the simulator by having it abort when it is about to complete more than q chains of the form $(y_0, x_1, 0)$; this allows to get rid of the intermediate system Σ_2 where the `Check` procedure is added to the keyed two-sided random function (or the ideal cipher) in order to ensure that the simulator makes a polynomial number of queries and runs in polynomial time with probability 1. Such a simplification does not seem to be possible if one wants to define a universal simulator which does not depend on q.

References

1. M. R. Albrecht, P. Farshim, K. G. Paterson, and G. J. Watson. On Cipher-Dependent Related-Key Attacks in the Ideal-Cipher Model. In A. Joux, editor, *Fast Software Encryption - FSE 2011*, volume 6733 of *Lecture Notes in Computer Science*, pages 128–145. Springer, 2011.
2. E. Andreeva, A. Bogdanov, Y. Dodis, B. Mennink, and J. P. Steinberger. On the Indifferentiability of Key-Alternating Ciphers. In R. Canetti and J. A. Garay, editors, *Advances in Cryptology - CRYPTO 2013 (Proceedings, Part I)*, volume 8042 of *Lecture Notes in Computer Science*, pages 531–550. Springer, 2013. Full version available at http://eprint.iacr.org/2013/061.
3. M. Bellare, A. Desai, E. Jokipii, and P. Rogaway. A Concrete Security Treatment of Symmetric Encryption. In *Symposium on Foundations of Computer Science - FOCS '97*, pages 394–403. IEEE Computer Society, 1997.
4. M. Bellare, J. Kilian, and P. Rogaway. The Security of the Cipher Block Chaining Message Authentication Code. *Journal of Computer and System Sciences*, 61(3):362–399, 2000.
5. M. Bellare and T. Kohno. A Theoretical Treatment of Related-Key Attacks: RKA-PRPs, RKA-PRFs, and Applications. In E. Biham, editor, *Advances in Cryptology - EUROCRYPT 2003*, volume 2656 of *Lecture Notes in Computer Science*, pages 491–506. Springer, 2003.
6. M. Bellare, D. Pointcheval, and P. Rogaway. Authenticated Key Exchange Secure against Dictionary Attacks. In B. Preneel, editor, *Advances in Cryptology - EUROCRYPT 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 139–155. Springer, 2000.
7. M. Bellare and T. Ristenpart. Multi-Property-Preserving Hash Domain Extension and the EMD Transform. In X. Lai and K. Chen, editors, *Advances in Cryptology - ASIACRYPT 2006*, volume 4284 of *Lecture Notes in Computer Science*, pages 299–314. Springer, 2006.
8. M. Bellare and P. Rogaway. Random Oracles are Practical: A Paradigm for Designing Efficient Protocols. In *ACM Conference on Computer and Communications Security*, pages 62–73, 1993.
9. E. Biham. New Types of Cryptanalytic Attacks Using Related Keys. *Journal of Cryptology*, 7(4):229–246, 1994.
10. A. Biryukov, D. Khovratovich, and I. Nikolić. Distinguisher and Related-Key Attack on the Full AES-256. In S. Halevi, editor, *Advances in Cryptology - CRYPTO 2009*, volume 5677 of *Lecture Notes in Computer Science*, pages 231–249. Springer, 2009.
11. J. Black. The Ideal-Cipher Model, Revisited: An Uninstantiable Blockcipher-Based Hash Function. In M. J. Robshaw, editor, *Fast Software Encryption - FSE '06*, volume 4047 of *Lecture Notes in Computer Science*, pages 328–340. Springer, 2006.
12. J. Black, P. Rogaway, and T. Shrimpton. Black-Box Analysis of the Block-Cipher-Based Hash-Function Constructions from PGV. In M. Yung, editor, *Advances in Cryptology - CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 320–335. Springer, 2002.
13. A. Bogdanov, L. R. Knudsen, G. Leander, F.-X. Standaert, J. P. Steinberger, and E. Tischhauser. Key-Alternating Ciphers in a Provable Setting: Encryption Using a Small Number of Public Permutations - (Extended Abstract). In D. Pointcheval and T. Johansson, editors, *Advances in Cryptology - EUROCRYPT 2012*, volume 7237 of *Lecture Notes in Computer Science*, pages 45–62. Springer, 2012.

14. R. Canetti, O. Goldreich, and S. Halevi. The Random Oracle Methodology, Revisited (Preliminary Version). In *Symposium on Theory of Computing - STOC '98*, pages 209–218. ACM, 1998. Full version available at http://arxiv.org/abs/cs.CR/0010019.
15. J.-S. Coron, Y. Dodis, C. Malinaud, and P. Puniya. Merkle-Damgård Revisited: How to Construct a Hash Function. In V. Shoup, editor, *Advances in Cryptology - CRYPTO 2005*, volume 3621 of *Lecture Notes in Computer Science*, pages 430–448. Springer, 2005.
16. J.-S. Coron, J. Patarin, and Y. Seurin. The Random Oracle Model and the Ideal Cipher Model Are Equivalent. In D. Wagner, editor, *Advances in Cryptology - CRYPTO 2008*, volume 5157 of *Lecture Notes in Computer Science*, pages 1–20. Springer, 2008.
17. J. Daemen and V. Rijmen. The Wide Trail Design Strategy. In B. Honary, editor, *Cryptography and Coding 2001*, volume 2260 of *Lecture Notes in Computer Science*, pages 222–238. Springer, 2001.
18. J. Daemen and V. Rijmen. *The Design of Rijndael: AES - The Advanced Encryption Standard*. Springer, 2002.
19. I. Damgård. A Design Principle for Hash Functions. In G. Brassard, editor, *Advances in Cryptology - CRYPTO '89*, volume 435 of *Lecture Notes in Computer Science*, pages 416–427. Springer, 1989.
20. G. Demay, P. Gazi, M. Hirt, and U. Maurer. Resource-Restricted Indifferentiability. In T. Johansson and P. Q. Nguyen, editors, *Advances in Cryptology - EUROCRYPT 2013*, volume 7881 of *Lecture Notes in Computer Science*, pages 664–683. Springer, 2013. Full version available at http://eprint.iacr.org/2012/613.
21. A. Desai. The Security of All-or-Nothing Encryption: Protecting against Exhaustive Key Search. In M. Bellare, editor, *Advances in Cryptology - CRYPTO 2000*, volume 1880 of *Lecture Notes in Computer Science*, pages 359–375. Springer, 2000.
22. S. Even and Y. Mansour. A Construction of a Cipher from a Single Pseudorandom Permutation. *Journal of Cryptology*, 10(3):151–162, 1997.
23. H. Feistel. Cryptography and computer privacy. *Scientific American*, 228(5):15–23, 1973.
24. A. Fiat and A. Shamir. How to Prove Yourself: Practical Solutions to Identification and Signature Problems. In A. M. Odlyzko, editor, *Advances in Cryptology - CRYPTO '86*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194. Springer, 1986.
25. P. Gazi and U. M. Maurer. Cascade Encryption Revisited. In M. Matsui, editor, *Advances in Cryptology - ASIACRYPT 2009*, volume 5912 of *Lecture Notes in Computer Science*, pages 37–51. Springer, 2009.
26. P. Gazi and S. Tessaro. Efficient and Optimally Secure Key-Length Extension for Block Ciphers via Randomized Cascading. In D. Pointcheval and T. Johansson, editors, *Advances in Cryptology - EUROCRYPT 2012*, volume 7237 of *Lecture Notes in Computer Science*, pages 63–80. Springer, 2012.
27. H. Gilbert and T. Peyrin. Super-Sbox Cryptanalysis: Improved Attacks for AES-Like Permutations. In S. Hong and T. Iwata, editors, *Fast Software Encryption - FSE 2010*, volume 6147 of *Lecture Notes in Computer Science*, pages 365–383. Springer, 2010.
28. O. Goldreich, S. Goldwasser, and S. Micali. How to construct random functions. *J. ACM*, 33(4):792–807, 1986.
29. L. Granboulan. Short Signatures in the Random Oracle Model. In Y. Zheng, editor, *Advances in Cryptology - ASIACRYPT 2002*, volume 2501 of *Lecture Notes in Computer Science*, pages 364–378. Springer, 2002.

30. S. Hirose. Provably Secure Double-Block-Length Hash Functions in a Black-Box Model. In C. Park and S. Chee, editors, *Information Security and Cryptology - ICISC 2004*, volume 3506 of *Lecture Notes in Computer Science*, pages 330–342. Springer, 2004.
31. S. Hirose. Some Plausible Constructions of Double-Block-Length Hash Functions. In M. J. Robshaw, editor, *Fast Software Encryption - FSE 2006*, volume 4047 of *Lecture Notes in Computer Science*, pages 210–225. Springer, 2006.
32. T. Holenstein, R. Künzler, and S. Tessaro. The Equivalence of the Random Oracle Model and the Ideal Cipher Model, Revisited. In L. Fortnow and S. P. Vadhan, editors, *Symposium on Theory of Computing - STOC 2011*, pages 89–98. ACM, 2011. Full version available at http://arxiv.org/abs/1011.1264.
33. É. Jaulmes, A. Joux, and F. Valette. On the Security of Randomized CBC-MAC Beyond the Birthday Paradox Limit: A New Construction. In J. Daemen and V. Rijmen, editors, *Fast Software Encryption - FSE 2002*, volume 2365 of *Lecture Notes in Computer Science*, pages 237–251. Springer, 2002.
34. J. Jonsson. An OAEP Variant With a Tight Security Proof. IACR Cryptology ePrint Archive Report 2002/034, 2002. Available at http://eprint.iacr.org/2002/034.
35. J. Kilian and P. Rogaway. How to Protect DES Against Exhaustive Key Search. In N. Koblitz, editor, *Advances in Cryptology - CRYPTO '96*, volume 1109 of *Lecture Notes in Computer Science*, pages 252–267. Springer, 1996.
36. L. R. Knudsen and V. Rijmen. Known-Key Distinguishers for Some Block Ciphers. In K. Kurosawa, editor, *Advances in Cryptology - ASIACRYPT 2007*, volume 4833 of *Lecture Notes in Computer Science*, pages 315–324. Springer, 2007.
37. R. Künzler. Are the random oracle and the ideal cipher models equivalent? Master's thesis, ETH Zurich, Switzerland, 2009.
38. R. Lampe, J. Patarin, and Y. Seurin. An Asymptotically Tight Security Analysis of the Iterated Even-Mansour Cipher. In X. Wang and K. Sako, editors, *Advances in Cryptology - ASIACRYPT 2012*, volume 7658 of *Lecture Notes in Computer Science*, pages 278–295. Springer, 2012.
39. R. Lampe and Y. Seurin. How to Construct an Ideal Cipher from a Small Set of Public Permutations. Full version of this paper. Available from http://eprint.iacr.org/2013/255.
40. J. Lee, M. Stam, and J. P. Steinberger. The Collision Security of Tandem-DM in the Ideal Cipher Model. In P. Rogaway, editor, *Advances in Cryptology - CRYPTO 2011*, volume 6841 of *Lecture Notes in Computer Science*, pages 561–577. Springer, 2011.
41. M. Luby and C. Rackoff. Pseudo-random Permutation Generators and Cryptographic Composition. In *Symposium on Theory of Computing - STOC '86*, pages 356–363. ACM, 1986.
42. M. Luby and C. Rackoff. How to Construct Pseudorandom Permutations from Pseudorandom Functions. *SIAM Journal on Computing*, 17(2):373–386, 1988.
43. U. M. Maurer. A Simplified and Generalized Treatment of Luby-Rackoff Pseudorandom Permutation Generator. In R. A. Rueppel, editor, *Advances in Cryptology - EUROCRYPT '92*, volume 658 of *Lecture Notes in Computer Science*, pages 239–255. Springer, 1992.
44. U. M. Maurer and K. Pietrzak. The Security of Many-Round Luby-Rackoff Pseudo-Random Permutations. In E. Biham, editor, *Advances in Cryptology - EUROCRYPT 2003*, volume 2656 of *Lecture Notes in Computer Science*, pages 544–561. Springer, 2003.

45. U. M. Maurer, R. Renner, and C. Holenstein. Indifferentiability, Impossibility Results on Reductions, and Applications to the Random Oracle Methodology. In M. Naor, editor, *Theory of Cryptography Conference- TCC 2004*, volume 2951 of *Lecture Notes in Computer Science*, pages 21–39. Springer, 2004.
46. B. Mennink. Optimal Collision Security in Double Block Length Hashing with Single Length Key. In X. Wang and K. Sako, editors, *Advances in Cryptology - ASIACRYPT 2012*, volume 7658 of *Lecture Notes in Computer Science*, pages 526–543. Springer, 2012.
47. R. C. Merkle. One Way Hash Functions and DES. In G. Brassard, editor, *Advances in Cryptology - CRYPTO '89*, volume 435 of *Lecture Notes in Computer Science*, pages 428–446. Springer, 1989.
48. M. Minier, R. C.-W. Phan, and B. Pousse. Distinguishers for Ciphers and Known Key Attack against Rijndael with Large Blocks. In B. Preneel, editor, *Progress in Cryptology - AFRICACRYPT 2009*, volume 5580 of *Lecture Notes in Computer Science*, pages 60–76. Springer, 2009.
49. J. Patarin. Pseudorandom Permutations Based on the DES Scheme. In G. D. Cohen and P. Charpin, editors, *EUROCODE '90*, volume 514 of *Lecture Notes in Computer Science*, pages 193–204. Springer, 1990.
50. J. Patarin. Security of Random Feistel Schemes with 5 or More Rounds. In M. K. Franklin, editor, *Advances in Cryptology - CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 106–122. Springer, 2004.
51. B. Preneel, R. Govaerts, and J. Vandewalle. Hash Functions Based on Block Ciphers: A Synthetic Approach. In D. R. Stinson, editor, *Advances in Cryptology - CRYPTO '93*, volume 773 of *Lecture Notes in Computer Science*, pages 368–378. Springer, 1993.
52. T. Ristenpart, H. Shacham, and T. Shrimpton. Careful with Composition: Limitations of the Indifferentiability Framework. In K. G. Paterson, editor, *Advances in Cryptology - EUROCRYPT 2011*, volume 6632 of *Lecture Notes in Computer Science*, pages 487–506. Springer, 2011.
53. R. L. Rivest, A. Shamir, and Y. Tauman. How to Leak a Secret. In C. Boyd, editor, *Advances in Cryptology - ASIACRYPT 2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 552–565. Springer, 2001.
54. Y. Sasaki and K. Yasuda. Known-Key Distinguishers on 11-Round Feistel and Collision Attacks on Its Hashing Modes. In A. Joux, editor, *Fast Software Encryption - FSE 2011*, volume 6733 of *Lecture Notes in Computer Science*, pages 397–415. Springer, 2011.
55. Y. Seurin. *Primitives et protocoles cryptographiques à sécurité prouvée*. PhD thesis, Université de Versailles Saint-Quentin-en-Yvelines, France, 2009.
56. C. Shannon. Communication Theory of Secrecy Systems. *Bell System Technical Journal*, 28(4):656–715, 1949.
57. J. Steinberger. Improved Security Bounds for Key-Alternating Ciphers via Hellinger Distance. IACR Cryptology ePrint Archive Report 2012/481, 2012. Available at http://eprint.iacr.org/2012/481.
58. J. P. Steinberger. The Collision Intractability of MDC-2 in the Ideal-Cipher Model. In M. Naor, editor, *Advances in Cryptology - EUROCRYPT 2007*, volume 4515 of *Lecture Notes in Computer Science*, pages 34–51. Springer, 2007.
59. S. Vaudenay. Decorrelation: A Theory for Block Cipher Security. *Journal of Cryptology*, 16(4):249–286, 2003.
60. R. S. Winternitz. A Secure One-Way Hash Function Built from DES. In *IEEE Symposium on Security and Privacy*, pages 88–90, 1984.