Minimalism in Cryptography: The Even-Mansour Scheme Revisited

Orr Dunkelman[1,2], Nathan Keller[2], and Adi Shamir[2]

[1] Computer Science Department
University of Haifa
Haifa 31905, Israel
orrd@cs.haifa.ac.il
[2] Faculty of Mathematics and Computer Science
Weizmann Institute of Science
P.O. Box 26, Rehovot 76100, Israel
{nathan.keller,adi.shamir}@weizmann.ac.il

Abstract. In this paper we consider the following fundamental problem: What is the simplest possible construction of a block cipher which is provably secure in some formal sense? This problem motivated Even and Mansour to develop their scheme in 1991, but its exact security remained open for more than 20 years in the sense that the lower bound proof considered known plaintexts, whereas the best published attack (which was based on differential cryptanalysis) required chosen plaintexts. In this paper we solve this long standing open problem by describing the new *Slidex attack* which matches the $T = \Omega(2^n/D)$ lower bound on the time T for any number of known plaintexts D. Once we obtain this tight bound, we can show that the original two-key Even-Mansour scheme is not minimal in the sense that it can be simplified into a single key scheme with half as many key bits which provides exactly the same security, and which can be argued to be the simplest conceivable provably secure block cipher. We then show that there can be no comparable lower bound on the memory requirements of such attacks, by developing a new memoryless attack which can be applied with the same time complexity but only in the special case of $D = 2^{n/2}$. In the last part of the paper we analyze the security of several other variants of the Even-Mansour scheme, showing that some of them provide the same level of security while in others the lower bound proof fails for very delicate reasons.

Keywords: Even-Mansour block cipher, whitening keys, minimalism, provable security, tight security bounds, slide attacks, slidex attack.

1 Introduction

A major theme in cryptographic research over the last thirty years was the analysis of minimal constructions. For example, many papers were published on the minimal cryptographic assumptions which are necessary and sufficient in

order to construct various types of secure primitives. Other examples analyzed the smallest number of rounds required to make Feistel structures with truly random functions secure, the smallest possible size of shares in various types of secret sharing schemes, and the simplest way to transform one primitive into another by using an appropriate mode of operation. Since the vague notion of conceptual simplicity only partially orders all the possible schemes, in many cases we have to consider minimal schemes (which are local minima that become insecure when we eliminate any one of their elements) rather than minimum schemes (which are global minima among all the possible constructions).

In the case of stream ciphers, one can convincingly argue that the simplest possible secure scheme is the one-time pad, since any encryption algorithm requires a secret key, and XOR'ing is the simplest conceivable way to mix it with the plaintext bits. The natural question we address in this paper is its dual: What is the simplest possible construction of a block cipher which has a formal proof of security?

This problem was first addressed by Even and Mansour [7, 8] in 1991. They were motivated by the DESX construction proposed by Ron Rivest in 1984, in which he proposed to protect DES against exhaustive search attacks by XOR'ing two independent prewhitening and postwhitening keys to the plaintext and ciphertext (respectively). The resultant scheme increased the key size from 56 to 184 bits without changing the definition of DES and with almost no additional complexity. The Even-Mansour scheme used such whitening keys but eliminated the keyed block cipher in the middle, replacing it with a fixed random permutation that everyone can share. The resultant scheme is extremely simple: To encrypt a plaintext, XOR it with one key, apply to it a publicly known permutation, and XOR the result with a second key.

To argue that the Even-Mansour scheme is minimal, its designers noted in [8] that eliminating either one of the two XOR'ed keys makes it easy to invert the known effect of permutation on the plaintext or ciphertext, and thus to recover the other key from a single known plaintext/ciphertext pair. Eliminating the permutation is also disastrous, since it makes the scheme completely linear. However, as we show in this paper, the two-key EM block cipher is not minimal in the sense that it can be further simplified into a single-key variant with half as many key bits which has exactly the same provable security.

To compare various variants of the Even-Mansour scheme, we need tight bounds on the exact level of security they provide. Unfortunately, all the bounds published so far are not tight in the sense that the lower bound allows known message attacks whereas the best known upper bounds require either chosen plaintexts or an extremely large number of known plaintexts.

One of the main tools used in previous attacks was the slide attack [3]. Originally, slide attacks were developed in order to break iterated cryptosystems with an arbitrarily large number of rounds by exploiting their self similarity under small shifts. The attack searched the given data for a slid pair of encryptions which have identical values along their common part (see Section 3.2 for formal definitions). For each candidate pair, the attack uses the two known plaintexts

and two known ciphertexts to analyze the two short non-common parts in order to verify the assumption that the two encryptions are indeed a slid pair, and if so to derive some key material. A different variant of this attack, called *slide with a twist* [4], tries to find a slid pair consisting of one encryption and one decryption, which have identical values along their common parts (i.e., the attack considers both shifts and reversals of the encryption rounds). In both cases, the existence of slid pairs is a random event which is expected to have a sharp threshold: Regardless of whether we use known or chosen messages, we do not expect to find any slid pairs if we are given fewer than $2^{n/2}$ encryptions where n is the size of the internal state.[1] Consequently, we cannot apply the regular or twisted slide attack unless we are given a sufficiently large number of encryptions, even if we are willing to trade off the lower amount of data with higher time and space complexities.

In this paper we propose the *slidex attack*, which is a new extended version of the slide attack that can efficiently use any amount of given data, even when it is well below the $2^{n/2}$ threshold for the existence of slid pairs. Its main novelty is that we no longer require equality between the values along the common part, but only the existence of some known relationship between these values. By using this new attack, we can finally close the gap between the upper and lower bounds on the security of the Even-Mansour scheme, solving this long standing open problem.

To demonstrate the usefulness and versatility of the new slidex attack, we apply it to several additional schemes which are unrelated to Even-Mansour. In particular, we show how to break 20 rounds of GOST using 2^{33} known plaintexts in 2^{77} time, and how to use the complementation property of DES in order to attack it with a slide attack even when it is surrounded by Vaudenay's decorrelation modules.

The paper is organized as follows. In Section 2 we introduce the Even-Mansour scheme, describe its formal proof of security, and survey all the previously published attacks on the scheme. In Section 3 we describe the known types of slide attacks, and explain why they cannot efficiently exploit a small number of known plaintexts. We then introduce our new Slidex attack, and use it to develop a new upper bound for the security of the Even-Mansour scheme which matches the proven lower bound for any number of known plaintexts. In Section 4 we develop our new variant of the Even-Mansour scheme, which is strictly simpler but has the same level of provable security. In Section 5 we analyze the security of several other variants of the Even-Mansour scheme, demonstrating both the generality and the fragility of its formal proof of security. Another limitation of the proof technique is described in Section 6, where we show that no comparable lower bound on the memory complexity of our attacks can exist. Finally, in the Appendix we introduce some generalizations of the slidex attack (such as the *mirror slide* attack), and show how to use them in order to improve

[1] We note that for specific block cipher structures, e.g., Feistel networks, a dedicated slide attack can require fewer than $2^{n/2}$ plaintexts. However, there is no such method that works for general structures.

the best known attacks on several variants of well known block ciphers such as GOST and DES.

2 The Even-Mansour Scheme

In this section we present the Even-Mansour (EM) scheme, review its security proof given in [8] and describe previous attacks on it presented in [5] and [4].

2.1 Definition of the EM Scheme and its Notation

The Even-Mansour scheme is a block cipher which consists of a single publicly known permutation $\mathcal{F}$ over n-bit strings, preceded and followed by n-bit whitening keys K_1 and K_2, respectively, i.e.,

$$EM^{\mathcal{F}}_{K_1,K_2}(P) = \mathcal{F}(P \oplus K_1) \oplus K_2.$$

It is assumed that the adversary is allowed to perform two types of queries:

- Queries to a full encryption/decryption oracle, called an E-oracle, that computes either $E(P) = EM^{\mathcal{F}}_{K_1,K_2}(P)$ or $D(C) = (EM^{\mathcal{F}}_{K_1,K_2})^{-1}(C)$.
- Queries to an $\mathcal{F}$-oracle, that computes either $\mathcal{F}(x)$ or $\mathcal{F}^{-1}(y)$.

The designers of EM considered two types of attacks. In the first type, called *existential forgery attack*, the adversary tries to find a *new* pair (P, C) such that $E(P) = C$. The second type is the more standard security game, where the adversary tries to decrypt a message C, i.e., to find P for which $E(P) = C$. The data complexity of an attack on the scheme is determined by the number D of queries to the E-oracle and their type (i.e., known/chosen/adaptively chosen etc.), and the time complexity of the attack is lower bounded by the number T of queries to the $\mathcal{F}$-oracle.[2] The success probability of an attack is the probability that the single guess it produces (either a pair (P, C) for the first type of attack, or a plaintext P for the second type) is correct.

2.2 The Lower Bound Security Proof

The main rigorously proven result in [8] was an upper bound of $O(DT/2^n)$ on the success probability of any cryptanalytic attack (of either type) on EM that uses at most D queries to the E-oracle and T queries to the $\mathcal{F}$-oracle. This result implies that in order to attack EM with a constant probability of success, we must have $DT = \Omega(2^n)$. Since this security proof is crucial for some of our results, we briefly describe its main steps.

[2] In concrete implementations, this oracle is usually replaced by some publicly known program which the attacker can run on its own. In this case the type of query (e.g., whether the inputs are adaptively chosen or not) can determine whether the attack can be parallelized on multiple processors, but we ignore such low level details in our analysis.

The proof requires several definitions. Consider a cryptanalytic attack on EM, and assume that at some stage of the attack, the adversary already performed s queries to the E-oracle and t queries to the $\mathcal{F}$-oracle, and obtained sets S and T of E-pairs and $\mathcal{F}$-pairs, respectively, i.e.,

$$S = \{(P_i, C_i)\}_{i=1,\ldots,s}, \qquad \text{and} \qquad T = \{(X_j, Y_j)\}_{j=1,\ldots,t}.$$

We say that the key K_1 is *bad* with respect to the sets of queries S and T, if there exist i, j such that $P_i \oplus K_1 = X_j$. Otherwise, K_1 is *good* with respect to S, T. Intuitively, a good key is one whose feasibility can not be deduced from the available data, whereas a bad key is one whose feasibility has to be analyzed further (but not necessarily discarded). Similarly, K_2 is bad w.r.t. S, T if there exist i, j such that $Y_j \oplus K_2 = C_i$, and K_2 is good otherwise. The key $K = (K_1, K_2)$ is *good* with respect to S, T if both K_1 and K_2 are good. It is easy to show that the number of good keys w.r.t. S and T is at least $2^{2n} - 2st \cdot 2^n$. A pair $(K = (K_1, K_2), \mathcal{F})$ is *consistent* w.r.t. S and T if for any pair $(P_i, C_i) \in S$ we have $C_i = K_2 \oplus \mathcal{F}(P_i \oplus K_1)$, and for any pair $(X_j, Y_j) \in T$, we have $\mathcal{F}(X_j) = Y_j$.

The proof consists of two main steps.

1. The first step shows that all good keys are, in some sense, equally likely to be the correct key. Formally, if the probability over the keys and over the permutations is uniform, then for all S, T, the probability

$$\Pr_{K,\mathcal{F}}\Big[K = k \Big| (K, \mathcal{F}) \text{ is consistent with } S, T\Big]$$

   is the same for any key $k \in \{0,1\}^{2n}$ that is good with respect to S, T.
   We present the proof of this step, since it will be crucial in the sequel. It follows from Bayes' formula that it suffices to prove that the probability

$$p = \Pr_{K,\mathcal{F}}\Big[(K, \mathcal{F}) \text{ is consistent with } S, T \Big| K = k\Big] \tag{1}$$

   is the same for all good keys. Given a good key $k = (k_1, k_2)$, it is possible to transform the set S of E-pairs to an equivalent set S' of $\mathcal{F}$-pairs by transforming the E-pair (P_i, C_i) to the $\mathcal{F}$-pair $(P_i \oplus k_1, C_i \oplus k_2)$. Since the key k is good, the pairs in S' and T do not overlap, and hence p is simply the probability of consistency of a random permutation $\mathcal{F}$ with $s + t$ given distinct input/output pairs. This probability clearly does not depend on k, which proves the assertion.
2. The second step shows that the success probability of any attack is bounded by the sum of the probability that in some step of the attack, the right key becomes a bad key, and the probability that the adversary can successfully generate a "new" consistent E-pair (P, C) if the right key is still amongst the good keys. The first probability can be bounded by $4DT/(2^n - 2DT)$, and the second probability can be bounded by $1/(2^n - D - T)$. Hence, the total success probability of the attack is bounded by $O(DT/2^n)$. We omit the proof of this step since it is not used in the sequel.

We note that obtaining non-trivial information about the key (e.g., that the least significant bit of the K_1 is zero, or the value of $K_1 \oplus K_2$), is also covered by this proof. Hence, throughout the paper we treat such leakage of information as a "problem" in the security of the construction (even if the exact keys are not found).

2.3 Previous Attacks on the Even-Mansour Scheme

The first proposed attack on the Even-Mansour scheme was published by Joan Daemen at Asiacrypt 1991 [5]. Daemen used the framework of differential cryptanalysis [2] to develop a *chosen plaintext* attack which matched the Even-Mansour lower bound for any amount of given data. The approach is to pick D pairs of chosen plaintexts whose XOR difference is some nonzero constant Δ. This plaintext difference is preserved by the XOR with the prewhitening key K_1, and similarly, the ciphertext difference is preserved by the XOR with the postwhitening key K_2. For a known permutation $\mathcal{F}$, most combinations of input and output differences suggest only a small number of possible input and output values, but it is not easy to find them. To carry out the attack, all we have to do is to sample $2^n/D$ pairs of inputs to $\mathcal{F}$ whose difference is Δ, and with constant non-negligible probability we can find an output difference which already exists among the chosen data pairs. This equality suggests actual input and output values to/from $\mathcal{F}$ for that pair, and thus recovers the two keys.

This attack matches the time/data relationship of the lower bound, but it is not tight since it requires chosen plaintexts, whereas the lower bound allows known plaintexts. This discrepancy was handled ten years later by a new attack called *slide with a twist* which was developed by Alex Biryukov and David Wagner, and presented at Eurocrypt 2000 [4]. By taking two Even-Mansour encryptions, sliding one of them and reversing the other, they showed how to attack the scheme with known instead of chosen plaintexts.[3] However, in order to find at least one slid pair, their attack requires at least $\Omega(2^{n/2})$ known plaintext/ciphertext pairs, and thus it could not be applied with a reasonable probability of success given any smaller number of known pairs.

These two cryptanalytic attacks were thus complementary: One of them matched the full time/data tradeoff curve but required chosen plaintexts, while the other could use known plaintexts but only if at least $\Omega(2^{n/2})$ of them were given. In the next section we present the new slidex technique that closes this gap: it allows to use any number of known plaintexts with the same time/data tradeoff as in the lower bound proof, thus providing an optimal attack on the Even-Mansour scheme.

[3] The slide with a twist attack on EM is described in detail in Section 3.1.

3 The Slidex Attack and a Tight Bound on the Security of the Even-Mansour Scheme

In this section we present the new Slidex attack and use it to obtain a tight bound on the security of the Even-Mansour scheme. We start with a description of the slide with a twist attack on EM [4] which serves as a basis for our attack, and then we present the slidex technique and apply it to EM. For more information on slide attacks, we refer the reader to [1, 3, 4].

3.1 The Slide with a Twist Attack

The main idea of the slide with a twist attack on EM is as follows. Assume that two plaintexts P, P^* satisfy

$$P \oplus P^* = K_1.$$

In such a case, we have

$$E(P) = \mathcal{F}(P \oplus K_1) \oplus K_2 = \mathcal{F}(P^*) \oplus K_2,$$

and similarly,

$$E(P^*) = \mathcal{F}(P^* \oplus K_1) \oplus K_2 = \mathcal{F}(P) \oplus K_2$$

(see Figure 1(a)). Hence,

$$E(P) \oplus E(P^*) = \mathcal{F}(P) \oplus \mathcal{F}(P^*),$$

or equivalently,

$$E(P) \oplus \mathcal{F}(P) = E(P^*) \oplus \mathcal{F}(P^*).$$

This relation allows to mount the following attack:

1. Query both the E-oracle and the $\mathcal{F}$-oracle at the same $2^{(n+1)/2}$ known values $P_1, P_2, \ldots$.[4] Store in a hash table the pairs $(E(P_i) \oplus \mathcal{F}(P_i)), i)$, sorted by the first coordinate.
2. For each collision in the table, i.e., $E(P_i) \oplus \mathcal{F}(P_i) = E(P_j) \oplus \mathcal{F}(P_j)$, check the guess $K_1 = P_i \oplus P_j$ and $K_2 = E(P_i) \oplus \mathcal{F}(P_j)$.

By the birthday paradox, it is expected that the data set contains a slid pair, i.e., a pair satisfying $P_i \oplus P_j = K_1$, with a non-negligible constant probability. For a random pair (P_i, P_j), the probability that $E(P_i) \oplus \mathcal{F}(P_i) = E(P_j) \oplus \mathcal{F}(P_j)$ is 2^{-n}, and thus, only a few collisions are expected in the table. These collisions include the collision induced by the slid pair, which suggests the correct values of K_1 and K_2. The data complexity of the attack is $D = 2^{(n+1)/2}$ known plaintexts, and the number of queries to $\mathcal{F}$ it requires is $T = 2^{(n+1)/2}$. Thus, $DT = 2^{n+1}$, which matches the lower bound up to a constant factor of 2.

[4] Formally, the adversary obtains known plaintext/ciphertext pairs $(P_i, E(P_i))$ and queries the $\mathcal{F}$-oracle at the value P_i.

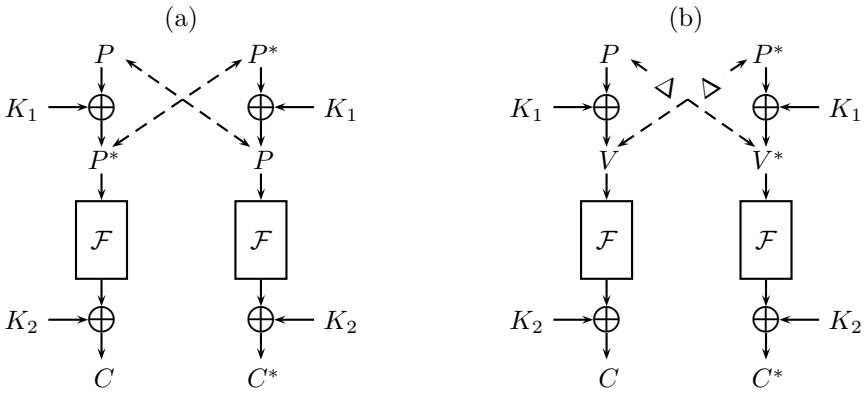

Fig. 1. (a) A twisted-slid pair; (b) A slidex pair

3.2 The New Slidex Attack

The *slidex* attack is an enhancement of the slide with a twist technique, which makes it possible to use a smaller number of known plaintexts (i.e., queries to the E-oracle), in exchange for a higher number of queries to the $\mathcal{F}$-oracle. The basic idea of the attack is as follows: Assume that a pair of plaintexts P, P^* satisfies

$$P \oplus P^* = K_1 \oplus \Delta,$$

for some $\Delta \in \{0,1\}^n$. In such a case,

$$E(P) = \mathcal{F}(P \oplus K_1) \oplus K_2 = \mathcal{F}(P^* \oplus \Delta) \oplus K_2,$$

and similarly,

$$E(P^*) = \mathcal{F}(P^* \oplus K_1) \oplus K_2 = \mathcal{F}(P \oplus \Delta) \oplus K_2$$

(see Figure 1(b)). Hence,

$$E(P) \oplus E(P^*) = \mathcal{F}(P^* \oplus \Delta) \oplus \mathcal{F}(P \oplus \Delta),$$

or equivalently,

$$E(P) \oplus \mathcal{F}(P \oplus \Delta) = E(P^*) \oplus \mathcal{F}(P^* \oplus \Delta).$$

This allows to mount the following attack, for any $d \leq n$:

1. Query the E-oracle at $2^{(d+1)/2}$ arbitrary values (i.e., known plaintexts) $P_1, P_2, \ldots$.
2. Choose 2^{n-d} arbitrary values $\Delta_1, \Delta_2, \ldots$ of Δ. For each Δ_ℓ, query the $\mathcal{F}$-oracle at the values $\{P_i \oplus \Delta_\ell\}_{i=1,2,\ldots,2^{(d+1)/2}}$, store in a hash table the pairs $(E(P_i) \oplus \mathcal{F}(P_i \oplus \Delta_\ell)), i)$, sorted by the first coordinate, and search for a collision.

Known Plaintext Attacks				
Attack	Data	Time	Memory	Tradeoff
Guess and determine [8]	2	2^n	2	—
Slide with a twist [4]	$2^{n/2}$	$2^{n/2}$	$2^{n/2}$	—
Slidex (Sect. 3.2)	2^d	2^{n-d}	2^d	$DT = 2^n$
Chosen Plaintext Attacks				
Attack	Data	Time	Memory	Tradeoff
Differential [5]	2^d	2^{n-d}	2^d	$DT = 2^n$
Adaptive Chosen Plaintext Attacks				
Attack	Data	Time	Memory	Tradeoff
Slidex (Sect. 6)	2^d	2^{n-d}	1	$DT = 2^n$ $(D \geq 2^{n/2})$

Table 1. Comparison of Results on the Even-Mansour scheme

3. For each collision in any of the hash tables, i.e., when P_i, P_j for which $E(P_i) \oplus \mathcal{F}(P_i \oplus \Delta_\ell) = E(P_j) \oplus \mathcal{F}(P_j \oplus \Delta_\ell)$ are detected, check the guess $K_1 = P_i \oplus P_j \oplus \Delta_\ell$ and $K_2 = E(P_i) \oplus \mathcal{F}(P_j \oplus \Delta_\ell)$.

For each triplet (P_i, P_j, Δ_ℓ), the probability that $P_i \oplus P_j \oplus \Delta_\ell = K_1$ is 2^{-n}. Since the data contains $2^d \cdot 2^{n-d} = 2^n$ such triplets, it is expected that with a non-negligible constant probability the data contains at least one *slidex triplet* (i.e., a triplet for which $P_i \oplus P_j \oplus \Delta_\ell = K_1$). On the other hand, since the probability of a collision in each hash table is 2^{d-n} and there are 2^{n-d} tables, it is expected that only a few collisions occur, and one of them suggests the correct key guess.

The number of queries to the E-oracle in the attack is $D = 2^{(d+1)/2}$, and the number of queries to the $\mathcal{F}$-oracle is $T = 2^{n-(d-1)/2}$. Thus, $DT = 2^{n+1}$, which matches the lower bound of [8] up to a constant factor of 2.

A summary of the complexities of all the old and new attacks on the Even-Mansour scheme appears in Table 1.

4 The New Single-Key Even-Mansour Scheme

In this section we present the single-key variant of the Even-Mansour scheme (abbreviated in the sequel as "SEM"), which has the same level of security while using only n secret key bits (compared to $2n$ bits in EM). First we define the scheme and show that the security proof of [8] can be adapted to yield a similar lower bound on its security, and then we present a simple attack on the new scheme which matches the lower bound, thus proving its optimality.

4.1 Definition of the Scheme and its Security Proof

Given a publicly known permutation $\mathcal{F}$ over n-bit strings and an n-bit secret key K, the Single-Key Even-Mansour (SEM) scheme is defined as follows:

$$SEM_K^{\mathcal{F}}(P) = \mathcal{F}(P \oplus K) \oplus K.$$

The attack model is the same as in the EM scheme. That is, the adversary can query an encryption/decryption E-oracle and an $\mathcal{F}$-oracle, and the complexity of an attack is determined by the number D of queries to the E-oracle and their type (known/chosen etc.), and the number T of queries to the $\mathcal{F}$-oracle.

Surprisingly, the security proof of the EM scheme [8] holds almost without a change when we apply it to the single-key SEM variant. The only modification we have to make is to define a key K as *bad* with respect to sets of oracle queries S and T if there exist i, j such that either $P_i \oplus K = X_j$ or $C_i \oplus K = Y_j$, and K as good otherwise. It is easy to see that if $|S| = s$ and $|T| = t$, then at least $2^n - 2st$ keys are still "good" keys. Exactly the same proof as for EM shows that all the good keys are equally likely to be the right key, and the bounds on the success probability of an attack apply without change for SEM.

Therefore, for any successful attack on SEM, we must have $DT = \Omega(2^n)$, which means that SEM provides the same security as EM, using only half as many key bits.

4.2 A Simple Optimal Attack on SEM

The slidex attack presented in Section 3 applies also to SEM, and is optimal since it uses only known plaintexts and matches everywhere the tradeoff curve of the security proof.

However, in the case of SEM, there is an even simpler attack (though, with the same complexity). Consider an encryption of a plaintext P through SEM, and denote the intermediate values in the encryption process by:

$$x = P, \qquad y = P \oplus K, \qquad z = \mathcal{F}(P \oplus K), \qquad w = E(P) = \mathcal{F}(P \oplus K) \oplus K.$$

Note that $x \oplus w = y \oplus z$. This allows to mount the following simple attack, applicable for any $D \leq 2^n$:

1. Query the E-oracle at D arbitrary values $P_1, P_2, \ldots, P_D$ and store in a hash table the values $(P_i \oplus E(P_i), i)$, sorted by the first coordinate.
2. Query the $\mathcal{F}$-oracle at $2^n/D$ arbitrary values $X_1, X_2, \ldots, X_{2^n/D}$, insert the values $X_j \oplus \mathcal{F}(X_j)$ to the hash table and search for a match.
3. If a match is found, i.e., $P_i \oplus E(P_i) = X_j \oplus \mathcal{F}(X_j)$, check the guess $K = P_i \oplus X_j$.

The analysis of the attack is exactly the same as that of the slide with a twist attack (see Section 3.1).

5 The Security of Other Variants of the Even-Mansour Scheme

In this section we consider two natural variants of the Even-Mansour scheme, and analyze their security.

The first variant replaces the XOR operations with modular additions, which are not involutions and are thus immune to standard slide-type attacks. However, we show that a new *addition slidex* attack can break it with the same complexity as that of the slidex attack on the original EM scheme.

The second variant considers the case in which the mapping $\mathcal{F}$ is chosen as an involution. This is motivated by the fact that in many "real-life" implementations of the EM scheme we would like to instantiate $\mathcal{F}$ by a keyless variant of a block cipher. Since in Feistel structures and many other schemes (e.g., KHAZAD, Anubis, Noekeon) the only difference between the encryption and decryption processes is the key schedule, such schemes become involutions when we make them keyless. In this section we show that this seemingly mild weakness of $\mathcal{F}$ can be used to mount a devastating attack on the EM scheme. In particular, we show that even when $\mathcal{F}$ is chosen uniformly at random among the set of all the possible involutions on n-bit strings, the adversary can recover the value $K_1 \oplus K_2$ with $O(2^{n/2})$ queries to the E-oracle and no queries at all (!) to the $\mathcal{F}$-oracle. This clearly violates the lower bound proof that no significant information about the key can be obtained unless $DT = \Omega(2^n)$ (which was proven for random permutations but seems to be equally applicable to random involutions), and is achieved by a new variant of the slide attack, which we call the *mirror slide* attack.

5.1 Even-Mansour with Addition

Consider the following scheme:

$$AEM^{\mathcal{F}}_{K_1,K_2}(P) = \mathcal{F}(P + K_1) + K_2,$$

where $\mathcal{F}$ is a publicly known permutation over n-bit strings, and '+' denotes modular addition in the additive group Z_{2^n}. In the sequel, we call it "Addition Even-Mansour" (AEM).

It is clear that the lower bound security proof of EM holds without any change for AEM. Similarly, it is easy to see that Daemen's differential attack on EM [5] can be easily adapted to AEM, by replacing XOR differences with modular differences.

It may seem that the new variant has better security with respect to slide-type attacks. As noted in [4], ordinary slide attacks can be applied only for ciphers in which the secret key is inserted through a *symmetric* operation such as XOR, and not through modular addition. In the specific case of EM, the slide with a twist attack relies on the observation that if for two plaintexts P, P^*, we have $P^* = P \oplus K_1$, then surely, $P = P^* \oplus K_1$ as well. This observation fails for AEM: If $P^* = P + K_1$, then $P^* + K_1 = P + 2K_1 \neq P$ (unless $K_1 = 0$ or

$K = 2^{n-1}$). The slidex attack presented in Section 3.2 fails against AEM for the same reason. Hence, it seems that none of the previously known attacks can break AEM in the *known plaintext* model.

We present an extension of the slidex attack, which we call *addition slidex*, which can break AEM with data complexity of D known plaintexts and time complexity of T $\mathcal{F}$-oracle queries, for any D, T such that $DT = 2^n$, hence showing that the security of AEM is identical to that of EM.

The basic idea of the attack is as follows: Assume that a pair of plaintexts P, P^* satisfies $P + P^* = -K_1 + \Delta$. (Note that somewhat counter intuitive, we consider the modular sum of the plaintexts rather than their modular difference!). In such a case,

$$E(P) = \mathcal{F}(P + K_1) + K_2 = \mathcal{F}(-P^* + \Delta) + K_2,$$

and similarly,

$$E(P^*) = \mathcal{F}(P^* + K_1) + K_2 = \mathcal{F}(-P + \Delta) + K_2.$$

Hence,

$$E(P) - E(P^*) = \mathcal{F}(-P^* + \Delta) - \mathcal{F}(-P + \Delta),$$

or equivalently,

$$E(P) + \mathcal{F}(-P + \Delta) = E(P^*) + \mathcal{F}(-P^* + \Delta). \tag{2}$$

Equation (2) allows us to mount an attack similar to the slidex attack, with the only change that instead of the values $(E(P_i) \oplus \mathcal{F}(P_i \oplus \Delta)), i)$, the adversary stores in the hash table the values $(E(P_i) + \mathcal{F}(-P_i + \Delta)), i)$.

We note that actually, the slidex attack can be considered as a special case of the addition slidex attack, since the addition slidex attack clearly applies to modular addition in any group, and the XOR operation corresponds to addition in the group Z_2.

5.2 Even-Mansour with a Random Involution as the Permutation

Let Involutional Even-Mansour (IEM) be the following scheme:

$$IEM^{\mathcal{I}}_{K_1,K_2}(P) = \mathcal{I}(P \oplus K_1) \oplus K_2,$$

where $\mathcal{I}$ is chosen uniformly at random amongst the set of involutions on n-bit strings. We present a new technique, which we call *mirror slide*, that allows to recover the value $K_1 \oplus K_2$ using $2^{n/2}$ queries to the E-oracle, and with no queries to the $\mathcal{I}$-oracle.

The idea of the technique is as follows. Consider two input/output pairs $(P, C), (P^*, C^*)$ for IEM. Assume that we have

$$P \oplus C^* = K_1 \oplus K_2. \tag{3}$$

In such case,

$$P \oplus K_1 = C^* \oplus K_2,$$

and hence, since $\mathcal{I}$ is an involution,

$$\mathcal{I}(P \oplus K_1) = \mathcal{I}^{-1}(C^* \oplus K_2).$$

However, by the construction we have

$$C = \mathcal{I}(P \oplus K_1) \oplus K_2, \qquad \text{and} \qquad P^* = \mathcal{I}^{-1}(C^* \oplus K_2) \oplus K_1,$$

and thus,

$$C \oplus K_2 = P^* \oplus K_1,$$

or equivalently,

$$P^* \oplus C = K_1 \oplus K_2 = P \oplus C^*,$$

where the last equality follows from Equation (3). Therefore, assuming that $P \oplus C^* = K_1 \oplus K_2$, we must have:

$$P \oplus C = P^* \oplus C^*.$$

This allows to mount a simple attack, similar to the slide with a twist attack. In the attack, the adversary queries the E-oracle at $2^{(n+1)/2}$ arbitrary values $P_1, P_2, \ldots$, and stores in a hash table the pairs $(E(P_i) \oplus P_i, i)$, sorted by the first coordinate. It is expected that only a few collisions exist, and that with a non-negligible probability, one of them results from a pair (P_i, P_j), for which $P_i \oplus E(P_j) = K_1 \oplus K_2$.

Therefore, the attack supplies the adversary with only a few possible values of $K_1 \oplus K_2$, after performing $2^{(n+1)/2}$ queries to the E-oracle and no queries at all to the $\mathcal{I}$-oracle. As we show later, the adversary cannot obtain K_1 or K_2 themselves (without additional effort or data), but at the same time, the adversary does learn a nontrivial information about the key, which contradicts the security proof of the original EM scheme.

Where the Security Proof Fails One may wonder, which part of the formal security proof fails when $\mathcal{F}$ is an involution. It turns out that the only part that fails is the argument in the first step of the proof showing that all good keys are equally likely to be the right key. Recall that in order to show this, one has to show that the probability

$$p = \Pr_{K,\mathcal{F}}[(K, \mathcal{F}) \text{ is consistent with } S, T | K = k]$$

is the same for all good keys. In the case of EM, p is shown to be the probability of consistence of a random permutation $\mathcal{F}$ with $s+t$ given distinct input/output pairs, which indeed does not depend on k (since such pairs are independent). In the case of IEM, the input/output pairs may be dependent, since it may occur that an encryption query to the E-oracle results in querying $\mathcal{I}$ at some value x,

while a decryption query to the E-oracle results in querying $\mathcal{I}^{-1}$ at the same value x. Since $\mathcal{I}$ is an involution, these queries are not independent and thus, the probability p depends on whether such dependency has occurred, and this event does depend on k. An examination of the mirror slide attack shows that this property is exactly the one exploited by the attack.

It is interesting to note that in the single-key case (i.e., for SEM where $\mathcal{F}$ is an involution, which we denote by ISEM), such event cannot occur, as in order to query $\mathcal{I}$ and $\mathcal{I}^{-1}$ at the same value, one must query E and E^{-1} at the same value. Since in the single-key case, the entire construction is an involution, such two queries result in the same answer for any value of the secret key, and hence, do not create dependence on the key. It can be shown, indeed, that the security proof does hold for ISEM and yields the same security bound, thus showing that in the case of involutions, the single-key variant is even stronger than the original two-key variant! Moreover, it can be noticed that in the case of EM, after the adversary recovers the value $K_1 \oplus K_2$, the encryption scheme becomes equivalent to a single-key Even-Mansour scheme with the key K_1, i.e., $E'(P) = \mathcal{I}(P \oplus K_1) \oplus K_1$. Thus, using two different keys in this case is totally obsolete, and also creates a security flaw which can be deployed by an adversary if the keys K_1 and K_2 are used also in other systems.

5.3 Addition Even-Mansour with an Involution as the Permutation

In this subsection we consider a combination of the two variants discussed in the previous subsections, i.e., AEM where $\mathcal{F}$ is a random involution. We abbreviate this variant as AIEM.

It can be easily shown that the mirror slide attack can be adapted to the case of AIEM, by modifying the assumption to $C^* - P = K_1 + K_2$, and the conclusion to $P + C = P^* + C^*$. The attack allows to recover the value $K_1 + K_2$, and then the scheme becomes equivalent to a *conjugation* EM scheme with a single key: $CISEM(P) = \mathcal{I}(P + K_1) - K_1$, and it can be shown that the security proof of EM applies also to CISEM. Thus, the security of AEM under the assumption that $\mathcal{F}$ is an involution is identical to that of the original EM.

An interesting phenomenon is that in the involution case, the security of single-key AEM (which we denote by AISEM) is much worse than that of AIEM. Indeed, the mirror slide attack allows to recover $K_1 + K_1 = 2K_1$, and hence to find K_1 (up to the value of the MSB), which breaks the scheme completely. This suggests that in the case of addition, the "natural" variant of single-key AEM is the conjugation variant, i.e., $CSEM(P) = \mathcal{F}(P + K_1) - K_1$, for which the security proof of EM indeed applies even if $\mathcal{F}$ is an involution, as mentioned above.

We list in Table 2 all 12 variants of Even-Mansour (single key/two keys, random permutation/random involution, and whether the keys are XORed, added, or conjugated). For each variant we list the obtainable security bound (if possible), and what attacks are applicable to match the bound.

	$\mathcal{F}$ is a Random Permutation		$\mathcal{F}$ is a Random Involution	
	Single Key	Two Keys	Single Key	Two Keys
Pre/Post-Whitening XOR Provable Security Bound Best Attack	SEM $DT \geq 2^n$ Slidex (or Sect. 4.2) (matches bound)	EM $DT \geq 2^n$ Slidex (matches bound)	SIEM $DT \geq 2^n$ Slidex (matches bound)	IEM $DT \geq 2^n$ Mirror slide Retrieves $K_1 \oplus K_2$ with $D = 2^{n/2}$
Pre/Post-Whitening Addition Provable Security Bound Best Attack	ASEM $DT \geq 2^n$ Addition Slidex (matches bound)	AEM $DT \geq 2^n$ Addition Slidex (matches bound)	ASIEM N/A Addition Slidex Complete break $D = 2^{n/2}$	AIEM $DT \geq 2^n$ Addition Slidex Retrieves $K_1 + K_2$ with $D = 2^{n/2}$
Conjugation Pre/Post-Whitening Provable Security Bound Best Attack	CSEM $DT \geq 2^n$ Addition Slidex (matches bound)	CEM $DT \geq 2^n$ Addition Slidex (matches bound)	CSIEM N/A Addition Slidex (matches bound)	CIEM $DT \geq 2^n$ Addition Slidex Retrieves $K_1 + K_2$ with $D = 2^{n/2}$

Table 2. Summary of the Security of the 12 Even-Mansour Variants

6 Memoryless Attacks on the Even-Mansour Scheme

All previous papers on the Even-Mansour scheme, including the lower bounds proved by the designers [8], Daemen's attack [5], and Biryukov-Wagner's slide attack [4], considered only the data and time complexities of attacks, but not the memory complexity. Analysis of the previously proposed attacks shows that in all of them, the memory complexity is $\min\{D, T\}$, where D is the data complexity (i.e., the number of E-queries) and T is the time complexity (i.e., the number of $\mathcal{F}$-queries). Thus, it is natural to ask whether the memory complexity can also be inserted into the lower bound security proofs, e.g., in the form $M \geq \min(D, T)$.

In this section we show that such a general lower bound can not exist, by constructing an attack with the particular data and time complexities of $O(2^{n/2})$, and with only a constant memory complexity. The attack is a memoryless variant of the slide with a twist attack described in Section 3.1. Recall that the main step of the slide with a twist attack is to find collisions of the form $E(P) \oplus \mathcal{F}(P) = E(P^*) \oplus \mathcal{F}(P^*)$.

We observe that such collisions can be found in a memoryless manner. We treat the function

$$\mathcal{G} : P \rightarrow E(P) \oplus \mathcal{F}(P)$$

as a random function, and apply Floyd's cycle finding algorithm [9] (or any of its variants, such as Nivasch's algorithm [12]) to find a collision in $\mathcal{G}$. The attack algorithm is as follows:

1. Query the E-oracle at a sequence of $O(2^{n/2})$ adaptively chosen values $P_1, P_2, \ldots$, such that P_1 is arbitrary and for $k > 1$, $P_k = E(P_{k-1}) \oplus \mathcal{F}(P_{k-1})$. (Here,

after each query to the E-oracle, the adversary queries the $\mathcal{F}$-oracle at the same value and uses its answer in choosing the next query to the E-oracle).
2. Use Floyd's cycle finding algorithm to find P_i, P_j such that $E(P_i) \oplus \mathcal{F}(P_i) = E(P_j) \oplus \mathcal{F}(P_j)$.
3. For each colliding pair, check the guess $K_1 = P_i \oplus P_j$ and $K_2 = E(P_i) \oplus \mathcal{F}(P_j)$.

The analysis of the attack is identical to the analysis of the slide with a twist attack. The memory complexity is negligible, and the data and time complexities remain $O(2^{n/2})$. Note that in this case we have to choose the queries to the E-oracle adaptively, whereas in the slide with a twist attack we could choose arbitrary queries to the E-oracle.

6.1 The Case $D < 2^{n/2}$

If the amount of available E-oracle queries is smaller than $2^{n/2}$, the adversary can still apply the slidex attack described in Section 3.2, but there seems to be no way to convert it into a memoryless attack by using the strategy described above. The main obstacle is that the adversary has to reuse the data many times in order to construct the hash tables for different values of Δ, which can be done only if the data is stored somewhere rather than used in an on-line manner which discards it after computing the next plaintext. This leads to the following open problem:

Problem 1. Does there exist a memoryless attack on the Even-Mansour scheme with D E-oracle queries and $2^n/D$ $\mathcal{F}$-oracle queries, where $D \ll 2^{n/2}$?

A similar question can be asked with respect to the Single-Key Even-Mansour scheme, where in addition to the slidex attack, the simple attack presented in Section 4.2 can also break the scheme when $D \ll 2^{n/2}$. The attack of Section 4.2 can also be transformed to a memoryless attack, by defining a random function:

$$\mathcal{H}(X) = \begin{cases} X \oplus E(X), & LSB(X) = 1 \\ X \oplus \mathcal{F}(X), & LSB(X) = 0, \end{cases}$$

and using Floyd's cycle finding algorithm to find a collision of $\mathcal{H}$. In the case when D and T are both close to $2^{n/2}$, with a constant probability such collision yields a pair (X_1, X_2) such that $X_1 \oplus E(X_1) = X_2 \oplus \mathcal{F}(X_2)$, concluding the attack. The problem is that if $D \ll 2^{n/2}$, then with overwhelming probability, a collision in $\mathcal{H}$ is of the form $X_1 \oplus \mathcal{F}(X_1) = X_2 \oplus \mathcal{F}(X_2)$, which is not useful to the adversary. Therefore, we state an additional open problem:

Problem 2. Does there exist a memoryless attack on the Single-Key Even-Mansour scheme with D E-oracle queries and $2^n/D$ $\mathcal{F}$-oracle queries, where $D \ll 2^{n/2}$?

If such memoryless attack can be found only for Single-Key EM and not for the ordinary EM, this will show that at least in some respect, the use of an additional key in EM does make the scheme stronger.

References

1. Biham, E., Dunkelman, O., Keller, N.: Improved Slide Attacks. In Biryukov, A., ed.: FSE. Volume 4593 of Lecture Notes in Computer Science., Springer (2007) 153–166
2. Biham, E., Shamir, A.: Differential Cryptanalysis of the Data Encryption Standard. Springer (1993)
3. Biryukov, A., Wagner, D.: Slide Attacks. In Knudsen, L.R., ed.: FSE. Volume 1636 of Lecture Notes in Computer Science., Springer (1999) 245–259
4. Biryukov, A., Wagner, D.: Advanced Slide Attacks. In Preneel, B., ed.: EUROCRYPT. Volume 1807 of Lecture Notes in Computer Science., Springer (2000) 589–606
5. Daemen, J.: Limitations of the Even-Mansour Construction. [10] 495–498
6. Dinur, I., Dunkelman, O., Shamir, A.: Improved Attacks on GOST. Technical report, to appear (2011)
7. Even, S., Mansour, Y.: A Construction of a Cipher From a Single Pseudorandom Permutation. [10] 210–224
8. Even, S., Mansour, Y.: A Construction of a Cipher from a Single Pseudorandom Permutation. J. Cryptology **10**(3) (1997) 151–162
9. Floyd, R.W.: Nondeterministic Algorithms. J. ACM **14**(4) (1967) 636–644
10. Imai, H., Rivest, R.L., Matsumoto, T., eds.: Advances in Cryptology - ASIACRYPT '91, International Conference on the Theory and Applications of Cryptology, Fujiyoshida, Japan, November 11-14, 1991, Proceedings. In Imai, H., Rivest, R.L., Matsumoto, T., eds.: ASIACRYPT. Volume 739 of Lecture Notes in Computer Science., Springer (1993)
11. Matsui, M.: The First Experimental Cryptanalysis of the Data Encryption Standard. In Desmedt, Y., ed.: CRYPTO. Volume 839 of Lecture Notes in Computer Science., Springer (1994) 1–11
12. Nivasch, G.: Cycle detection using a stack. Inf. Process. Lett. **90**(3) (2004) 135–140
13. Rivest, R.L.: DESX. Never published (1984)
14. Russian National Bureau of Standards: Federal Information Processing Standard-Cryptographic Protection - Cryptographic Algorithm. GOST 28147-89 (1989)
15. Vaudenay, S.: Provable Security for Block Ciphers by Decorrelation. In Morvan, M., Meinel, C., Krob, D., eds.: STACS. Volume 1373 of Lecture Notes in Computer Science., Springer (1998) 249–275
16. Vaudenay, S.: Decorrelation: A Theory for Block Cipher Security. J. Cryptology **16**(4) (2003) 249–286
17. Wagner, D.: A Generalized Birthday Problem. In Yung, M., ed.: CRYPTO. Volume 2442 of Lecture Notes in Computer Science., Springer (2002) 288–303

A The Mirror Slide Attack

In this section we present the general framework of the *mirror slide* attack, that was presented in Section 5.2 in the special case of the Even-Mansour scheme. We show that the mirror slide attack generalizes the *slide with a twist* attack [4] and can be combined with the *complementation slide* attack [4]. We apply the new technique to a 20-round variant of the block cipher GOST [14], and to variants of the DESX cryptosystem [13] in which the subkeys of the internal DES cipher are replaced by a 2-round or a 4-round self-similar sequence.

A.1 The General Framework

The mirror slide attack applies to block ciphers that can be decomposed as a cascade of three sub-ciphers: $E = E_2 \circ E_1 \circ E_0$, where the middle layer E_1 is an involution, i.e., $E_1 = (E_1)^{-1}$.[5]

Let E be such a cipher, and assume that for two plaintext/ciphertext pairs $(P, C), (P^*, C^*)$, we have

$$E_0(P) = E_2^{-1}(C^*). \tag{4}$$

In such case, since E_1 is an involution,

$$E_1(E_0(P)) = E_1^{-1}(E_2^{-1}(C^*)).$$

By the construction, this implies:

$$E_2^{-1}(C) = E_1(E_0(P)) = E_1^{-1}(E_2^{-1}(C^*)) = E_0(P^*). \tag{5}$$

If Equation (4) holds (and thus, Equation (5) also holds, the pair (P, P^*) is called a *mirror slid pair*.

The way to exploit mirror slid pairs in a cryptanalytic attack is similar to standard slide-type attacks [3, 4]: The adversary asks for the encryption of $2^{(n+1)/2}$ known plaintexts $P_1, P_2, \ldots$ (where n is the block size of E) and denotes the corresponding ciphertexts by $C_1, C_2, \ldots$. For each pair (P_i, P_j), the adversary assumes that it is a mirror slid pair and tries to solve the system of equations:

$$\begin{cases} C_j = E_2(E_0(P_i)), \\ C_i = E_2(E_0(P_j)) \end{cases}$$

(which is equivalent to Equations (4) and (5)). If E_0 and E_2 are "simple enough", the adversary can solve the system efficiently and recover the key material used in E_0 and E_2.

If the amount of subkey material used in E_0 and E_2 is at most n bits (in total), it is expected that at most a few of the systems of equations generated by the 2^n plaintext pairs are consistent (since the equation system is a $2n$-bit condition). One of them is the system generated by the mirror slid pair, which is expected to exist in the data with a constant probability since the probability of a random pair to be a mirror slid pair is 2^{-n}. Hence, the adversary obtains only a few suggestions for the key, which contain the right key with a constant probability. If the amount of key material used in E_0 and E_2 is bigger than n bits, the adversary can still find the right key, by enlarging the data set by a small factor and using key ranking techniques (exploiting the fact that the right key is suggested by all mirror slid pairs, while the other pairs suggest "random" keys).

The data complexity of the attack is $O(2^{n/2})$ known plaintexts, and its time complexity is $O(2^n)$ (assuming that the system of equations can be solved within constant time).

[5] We note that the attack can be applied also if E_1 has some other symmetry properties, as shown in Appendix A.4 below.

We note that the attack can be applied even when E_0 and E_2 are not "simple" ciphers using a meet-in-the-middle attack. If both E_0 and E_2 use $\kappa \leq n$ key bits at most, one can try and find the solutions to the above set of equations in time $\min\{O(2^{n+\kappa}), O(2^{n/2+2\kappa})\}$.[6]

A.2 The Slide with a Twist Attack and an Application to 20-Round GOST

The first special case of the mirror slide framework we consider is where in the subdivision of E, we have $E_2 = Identity$. In such case, the system of equations presented above is simplified to:

$$\begin{cases} C_j = E_0(P_i), \\ C_i = E_0(P_j). \end{cases} \tag{6}$$

It turns out that in this case, the attack is reduced exactly to the slide with a twist attack presented in [4]! (Though, in [4] the attack is described in a different way).

A concrete example of this case is a reduced-round variant of the block cipher GOST [14], that consists of the last 20 of its 32 rounds. It is well-known that the last 16 rounds of GOST compose an involution, and hence, this variant can be represented as $E = E_1 \circ E_0$, where E_0 is 4-round GOST, and E_1 (which is the last 16 rounds of GOST) is an involution.[7] As shown in [6], a 4-round variant of GOST can be broken with two plaintext/ciphertext pairs and time complexity of 2^{12} encryptions. Therefore, the mirror slide attack can break this 20-round variant of GOST with data complexity of 2^{33} known plaintexts (since the block size of GOST is 64 bits), and time complexity of $2^{65} \cdot 2^{12} = 2^{77}$ encryptions.

We note that a similar attack was described in [4] using the slide with a twist technique, but only on a 20-round version of a modified variant of GOST called GOST$\oplus$ in which the key addition is replaced by XOR.

A.3 Attacks on Block Ciphers Using Pre/Post Whitening Keys

The second special case we consider is when E_0 and E_2 consist of XOR with a subkey, as is the case in involutional block ciphers using pre- and post-whitening

[6] One can either take all plaintext/ciphertext pairs and partially encrypt the plaintext under all 2^κ keys for E_0 and partially decrypt the ciphertext under all 2^κ keys for E_2 to find the mirror pairs. Another option is to try for each pair of plaintexts (P_i, P_j) to solve the system

$$\begin{cases} E_2^{-1}(C_j) = E_0(P_i), \\ E_2^{-1}(C_i) = E_0(P_j) \end{cases}$$

which can be easily done in a meet-in-the-middle approach in time 2^κ for each (P_i, P_j).

[7] We note that due to the Feistel structure of GOST, we do not have $E_1 \circ E_1 = Id$, but rather $E_1 \circ swap \circ E_1 = Id$. This can be handled easily by inserting swap to the left hand side of Equation (6). The same correction can be performed in the other Feistel constructions discussed in the sequel.

keys. In this case, the system of equations is simplified to:

$$\begin{cases} C_j = P_i \oplus K_0 \oplus K_2, \\ C_i = P_j \oplus K_0 \oplus K_2, \end{cases}$$

where K_0 and K_2 are the subkeys used in E_0 and E_2, respectively. The system of equations can be further simplified (by XORing the equations) to $C_j \oplus C_i = P_i \oplus P_j$, or equivalently,

$$P_i \oplus C_i = P_j \oplus C_j.$$

As described in Section 5.2, this allows to mount an attack with data and time complexities of $O(2^{n/2})$ (compared to time complexity of $O(2^n)$ in the general mirror slide framework) that recovers the value $K_0 \oplus K_2$.

The simplest example of this case is the mirror slide attack on the IEM scheme (Even-Mansour with an involution as the permutation) described in Section 5.2. A bit more interesting example is the attack on AIEM (Addition Even-Mansour with involution as the permutation) presented in Section 5.3. As was shown in that attack, the technique can be easily modified to handle the case where the subkeys K_0 and K_2 are inserted through addition (instead of XOR).

In the next subsection we present even more interesting examples, in which the attack applies to ciphers with pre/post key whitening in which E_1 is not an involution (but rather has some weaker kind of symmetry).

A.4 Combination with the Complementation Slide Attack and Application to 2K-DESX

In this subsection we consider the case where E_1 is not an involution, but rather a Feistel cipher with 2-round self-similarity (see Figure 2). Such a cipher (but without the key whitening) was considered in [4], and it was shown that it can be broken with complexity of $O(2^{n/2})$, using a technique called *complementation slide*.[8] We show that the complementation slide technique can be combined with the mirror slide technique to yield an attack on the scheme including pre- and post- key whitening, with the same complexity.

A concrete example of such construction one may consider is a variant of DESX [13] in which the subkeys generated by the DES key schedule are replaced by the periodic sequence $(k_a, k_b, k_a, k_b, \ldots)$. Using the terminology of [3, 4], this variant can be called 2K-DESX. For the sake of simplicity, we demonstrate the attack on the example of 2K-DESX.

[8] We note that in [4], a Feistel cipher with 2-round self-similarity is also attacked using the slide with a twist technique (with even better results). In the attack, the cipher is represented as $E = E_1 \circ E_0$, where E_0 is a single round and E_1 is a $2m - 1$-round Feistel structure with 2-round self-similarity, which can be easily seen to be an involution. As described in Appendix A.2, such attack can be viewed as a special case of the mirror slide attack. We do not consider it in this subsection since the existence of pre- and post-whitening raises its time complexity to $\Theta(2^n)$, while the complexity of our attack on this cipher is $O(2^{n/2})$.

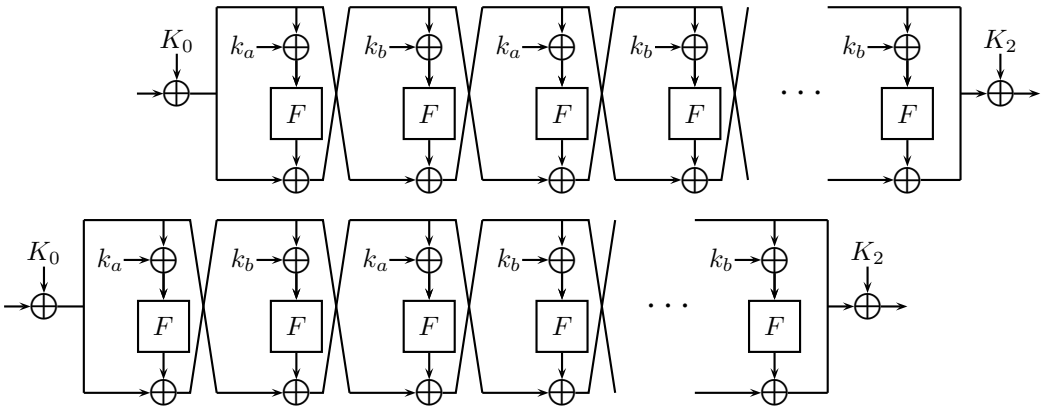

Fig. 2. Pre-/Post-Whitened Cipher with 2-Round Self Similarity

Consider two plaintext/ciphertext pairs $(P, C), (P^*, C^*)$ of 2K-DESX, and assume that

$$P \oplus C^* = K_0 \oplus K_2 \oplus (k_a \oplus k_b || k_a \oplus k_b),$$

where $||$ denotes concatenation of bit strings. In such case,

$$P \oplus K_0 = (C^* \oplus K_2) \oplus (k_a \oplus k_b || k_a \oplus k_b). \tag{7}$$

We would like to apply E_1 to the left hand side and E_1^{-1} to the right hand side, like in the standard mirror slide attack. In our case, E_1 is not an involution. However, this is compensated by the term $(k_a \oplus k_b || k_a \oplus k_b)$ in the right hand side of the equation. Indeed, in the first round of E_1, the subkey is k_a, and thus, the input to the F-function is $P_R \oplus K_{0R} \oplus k_a$ (where X_R denotes the right half of X). On the other side, the subkey in the first round of E_1^{-1}, which is the subkey in the last round of E_1, is k_b, and hence, the input to the F-function is $C^*_R \oplus K_{2R} \oplus k_b$. Therefore, by Equation (7), the two inputs are equal. A similar analysis shows that equality holds for the inputs of the F-functions in all rounds, and thus,

$$E_1(P \oplus K_0) = E_1^{-1}(C^* \oplus K_2) \oplus (k_a \oplus k_b || k_a \oplus k_b),$$

or equivalently,

$$C \oplus K_2 = P^* \oplus K_0 \oplus (k_a \oplus k_b || k_a \oplus k_b). \tag{8}$$

XORing Equations (7) and (8), we get

$$C \oplus C^* = P \oplus P^*.$$

This allows to apply an attack similar to the attack on IEM and recover the value $K_0 \oplus K_2 \oplus (k_a \oplus k_b || k_a \oplus k_b)$ with data and time complexities of $O(2^{n/2})$.

A.5 Application to a Variant of 4K-DESX

The last case we consider is a variant of DESX in which the number of rounds in DES is changed to $4m + 1$, and the subkeys are replaced by the sequence $(k_a, k_b, k_c, k_d)^m, k_a$. We show that another combination of the complementation slide technique with the mirror slide technique allows to break this variant with data and time complexity of $O(2^{n/2})$.

Consider two plaintext/ciphertext pairs $(P, C), (P^*, C^*)$, and assume that

$$P \oplus C^* = K_0 \oplus K_2 \oplus (k_b \oplus k_d||0),$$

where $||$ denotes concatenation of bit strings. In such a case,

$$P \oplus K_0 = (C^* \oplus K_2) \oplus (k_b \oplus k_d||0). \tag{9}$$

We apply E_1 to the left hand side of the equation, and E_1^{-1} to the right hand side of the equation. In the first round of E_1, the subkey is k_a, and thus, the input to the F-function is $P_R \oplus K_{0R} \oplus k_a$. The subkey in the first round of E_1^{-1} is also k_a, and hence, the input to the F-function in that round is $C^*_R \oplus K_{2R} \oplus k_a$. Therefore, by Equation (9), the two inputs are equal. In the second round of E_1 and E_1^{-1}, the subkey in E_1 is k_b, while the subkey in E_1^{-1} is k_d. However, this difference is cancelled with the term $k_b \oplus k_d$ in Equation (9). A similar analysis shows that equality holds for the inputs of the F-functions in all rounds, and thus,

$$E_1(P \oplus K_0) = E_1^{-1}(C^* \oplus K_2) \oplus (k_b \oplus k_d||0),$$

or equivalently,

$$C \oplus K_2 = P^* \oplus K_0 \oplus (k_b \oplus k_d||0). \tag{10}$$

XORing Equations (9) and (10), we get

$$C \oplus C^* = P \oplus P^*,$$

and the attack can be concluded as in the previous case and retrieve the value $K_0 \oplus K_2 \oplus (k_b \oplus k_d||0)$.

We note that this technique does not apply to the standard variant of 4K-DESX, in which the subkeys are $(k_a, k_b, k_c, k_d)^m$ (without an additional subkey k_a at the end). The reason is that the rate of symmetry between E_1 and E_1^{-1} is insufficient. While the asymmetry in the first two rounds can be compensated by adding the term $(k_a \oplus k_d||k_b \oplus k_c)$ to the equation, the inputs to the F-function in the third round will not be equal anymore.

B Addition Slide Attack – Application to Variants of DESX

In Section 5, we presented two new slide-type attacks that are applicable to ciphers in which the subkeys are inserted through modular addition (rather than XOR). The first was a variant of the *slidex* attack that was used in Section 5.1 to

attack AEM, i.e., an Even-Mansour scheme in which the key XOR is replaced by modular addition. The second was a variant of the *mirror slide* attack that was used in Section 5.3 to attack AIEM, i.e., AEM in which the internal permutation is an involution. These two attacks can be considered as special cases of a more general technique which we call the *addition slide* attack. The main feature of the technique (that appears in both special cases) is that the relation between the elements of a slid pair concerns their modular sum, rather than their difference (as one may expect in light of the standard slide-type attacks).

In this section we present another application of the *addition slide* technique. The attack targets Addition DESX, i.e., a variant of DESX [13] in which the whitening keys are inserted using modular addition (instead of XOR). We show that while this variant seems to be as secure as DESX, it can be broken using only two related keys and practical complexity of either 2^{34} in the chosen plaintext model, or 2^{43} in the known plaintext model. The attack exploits the well-known *complementation property* of DES, namely, that for any P, K,

$$DES_K(P) = \overline{DES_{\bar{K}}(\bar{P})},$$

where $\bar{X}$ denotes the bitwise complement of X (i.e., $\bar{X} = X \oplus FF \ldots FF_x = 2^{64} - 1 - X$). It is interesting to note that while in the cases of DES and DESX, this property can be used only either for a distinguishing attack or for speeding up exhaustive key search by a factor of 2, in our case it can be deployed to mount a key recovery attack.

After presenting the attack on Addition DESX, we show that a slightly modified variant of the attack applies (with the same complexities) to another variant of DESX in which the key pre/post whitenings are replaced by key-dependent decorrelation modules [16].

B.1 Attack on Addition DESX

The addition DESX block cipher is defined as:

$$E_{K_0,K_1,K_2}(P) = K_2 + DES_{K_1}(P + K_0),$$

where '+' denotes addition modulo 2^{64}. The basic idea of the attack is as follows. Let $(P, C), (P^*, C^*)$ be two plaintext/ciphertext pairs, such that P is encrypted under (K_0, K_1, K_2) and P^* is encrypted under $(K_0, \overline{K_1}, K_2)$. Assume that the pair (P, P^*) satisfies:

$$P + P^* + 2K_0 \equiv 2^{64} - 1 \pmod{2^{64}}. \tag{11}$$

In such a case, we have

$$P + K_0 = \overline{P^* + K_0}.$$

By the complementation property, this implies:

$$DES_{K_1}(P + K_0) = \overline{DES_{\overline{K_1}}(P^* + K_0)},$$

or equivalently,

$$DES_{K_1}(P + K_0) + DES_{\overline{K_1}}(P^* + K_0) \equiv 2^{64} - 1 \pmod{2^{64}}.$$

This, in turn, implies:

$$C + C^* = E_{K_0,K_1,K_2}(P) + E_{K_0,\overline{K_1},K_2}(P^*) \equiv 2^{64} - 1 + 2K_2 \pmod{2^{64}}. \quad (12)$$

Equation (12) cannot be exploited directly (like in all previous attacks) since the value of K_2 is not known to the adversary. However, we observe that since the right hand side of Equation (12) does not depend on P and P^*, it can be cancelled using another pair of plaintexts.

Let $(P, C), (P^*, C^*)$ be plaintext/ciphertext pairs such that the pair (P, P^*) satisfies Equation (11), and let $a \in Z_{2^{64}}$ be arbitrary. Consider the encryptions of $P + a$ and $P^* - a$ under the keys (K_0, K_1, K_2) and $(K_0, \overline{K_1}, K_2)$, respectively, and denote the corresponding ciphertexts by C' and C'^*. It is clear that the pair $(P + a, P^* - a)$ also satisfies Equation (11). Hence, we have

$$C' + C'^* \equiv 2^{64} - 1 + 2K_2 \pmod{2^{64}}. \quad (13)$$

Combining Equations (12) and (13), we get:

$$C + C^* = C' + C'^*,$$

or equivalently,

$$C - C' = C'^* - C^*.$$

This allows to mount the following attack:

1. Choose some arbitrary $a \in Z_{2^{64}}$.[9]
2. Ask for the encryption of 2^{32} arbitrary plaintexts $P_1, P_2, \ldots$ under the key (K_0, K_1, K_2), and denote the corresponding ciphertexts by $(C_1, C_2, \ldots)$. Ask for the encryption of the 2^{32} plaintexts $P_1 + a, P_2 + a, \ldots$ under the same key, and denote the corresponding ciphertexts by $(C'_1, C'_2, \ldots)$. Store in a hash table the pairs $((C_i - C'_i), i)$, sorted by the first coordinate.
3. Ask for the encryption of 2^{32} arbitrary plaintexts $P^*_1, P^*_2, \ldots$ under the key $(K_0, \overline{K_1}, K_2)$, and denote the corresponding ciphertexts by $(C^*_1, C^*_2, \ldots)$. Ask for the encryption of the 2^{32} plaintexts $P^*_1 - a, P^*_2 - a, \ldots$ under the same key, and denote the corresponding ciphertexts by $(C'^*_1, C'^*_2, \ldots)$. Insert the values $C'^*_j - C^*_j$ into the hash table and search for collisions.
4. For each collision in the table, i.e., $C_i - C'_i = C'^*_j - C^*_j$, check the guess $2K_0 = 2^{64} - 1 - P_i - P^*_j \pmod{2^{64}}$ and $2K_2 = C_i + C^*_j - (2^{64} - 1) \pmod{2^{64}}$.

As in the previous attacks, it is expected that only a few collisions occur, and that with a constant probability, one of them suggests the right key (K_0, K_2). A key guess suggested by the pair (P_i, P^*_j) can be checked by choosing another

[9] For example, if the encryption is performed in counter mode, it may be desirable to choose $a = 1$.

$a' \in Z_{2^{64}}$, asking for the encryption of $P_i + a'$ and $P_j^* - a'$ under the keys (K_0, K_1, K_2) and $(K_0, \overline{K_1}, K_2)$, respectively, and checking whether the corresponding ciphertexts (denoted by C_i'' and $C_j''^*$) satisfy:

$$C_i - C_i'' = C_j''^* - C_j^*.$$

If the equation is satisfied, then the pair (P_i, P_j^*) satisfies Equation (11) with overwhelming probability, and thus, the suggestion for (K_0, K_2) is correct (with the same probability). The value of K_1 can be found using auxiliary techniques (e.g., a differential or a linear attack on DES). The data complexity of the attack is 2^{34} chosen plaintexts encrypted under two keys, and its memory and time complexities are about 2^{34} (except for the part of recovering K_1). As in the previous cases, the attack can be transformed into a memoryless attack with the same time complexity, where the data complexity is 2^{34} adaptively chosen plaintexts.

A known-plaintext variant of the attack A variant of the attack can be performed in the known plaintext model without enlarging the number of examined plaintexts, at the expense of enlarging the time complexity. The attack uses the fact that the procedure described above succeeds for any value of a, and thus, the adversary can exploit many values of a simultaneously. The algorithm of the known plaintext attack is as follows:

1. Ask for the encryption of two pools of 2^{32} arbitrary plaintexts each under the key (K_0, K_1, K_2), and denote the plaintext/ciphertext pairs in the pools by $(P_1, C_1), (P_2, C_2), \ldots$, and $(P_1', C_1'), (P_2', C_2'), \ldots$, respectively.
2. Ask for the encryption of two pools of 2^{32} arbitrary plaintexts each under the key $(K_0, \overline{K_1}, K_2)$, and denote the plaintext/ciphertext pairs in the pools by $(P_1^*, C_1^*), (P_2^*, C_2^*), \ldots$, and $(P_1'^*, C_1'^*), (P_2'^*, C_2'^*), \ldots$, respectively.
3. Search for a four-collision of 128-bit values, of the form:

$$(P_i - P_j' + P_k^* - P_\ell'^*, C_i - C_j' + C_k^* - C_\ell'^*) = 0. \tag{14}$$

4. For each such collision, check the guess $2K_0 = 2^{64} - 1 - P_i - P_j^* \pmod{2^{64}}$ and $2K_2 = C_i + C_j^* - (2^{64} - 1) \pmod{2^{64}}$.

It is expected that among the 2^{128} examined plaintext quartets, about 2^{64} quartets satisfy the equation $P_i - P_j' + P_k^* - P_\ell'^* = 0$, and thus can be represented as $(P_i, P_i + a, P_k^*, P_k^* - a)$, for $a = P_j' - P_i$. Thus, with a constant probability, in at least one of these quartets, P_i and P_k^* satisfy Equation (11). For such a quartet, we must have $C_i - C_j' + C_k^* - C_\ell'^* = 0$, and thus, it generates a collision of the form needed for the attack. On the other hand, the probability that Equation (14) is satisfied for a random quartet is 2^{-128}, and hence, it is expected that only a few collisions exist, and at least one of them suggests the right key.

The data complexity of the attack is 2^{34} known plaintexts encrypted under two keys, and the memory and time complexities are about 2^{64}.

As the collision search performed in the attack is a solution of a standard *generalized birthday* problem, one can obtain a time/memory/data tradeoff using the improved algorithms for the generalized birthday problem presented by Wagner [17]. For example, if the data complexity is increased to $2^{42.6}$ known plaintexts, then the memory and time complexities can be reduced to $2^{42.6}$. As the key K_1 can be found with about 2^{43} known plaintexts using linear cryptanalysis [11], this allows to recover the full key (K_0, K_1, K_2) of Addition DESX with data complexity of about 2^{43} known plaintexts and time and memory complexities of 2^{43} in total.

B.2 Attack on DES Surrounded by Decorrelation Modules

Decorrelation modules, introduced by Vaudenay [16] in 1997, are tools to ensure security against statistical attacks such as differential and linear cryptanalysis. One of the basic decorrelation modules (used in COCONUT98 [15]), is the permutation: $DM_{K_1,K_2}(X) = (X \oplus K_1) \cdot K_2$, where the multiplication is done over the field $GF(2^n)$, and $K_2 \neq 0$.

One property of this decorrelation module is that once the key is set, the decorrelation module is linear, but when the key is random, the probability of any non-trivial differential going through the module equals $1/(2^n - 1)$ on average. A similar condition can be proved with respect to linear cryptanalysis as well. Thus, inserting decorrelation modules as an element in a block cipher is suggested in order to make it secure against differential and linear cryptanalysis.

It seems that surrounding a block cipher with key-dependent decorrelation modules is a stronger measure than adding pre/post key whitening. However, it turns out that in the case of DES, due to the complementation property, this leads to related-key attacks which are much stronger than the best known attacks on DESX in the related-key model.

Consider the block cipher *Decorrelation-DES*, defined as:

$$E_{(K_0,K_1),K_2,(K_3,K_4)}(P) = M_1(DES_{K_2}(M_0(P))),$$

where $M_0(X) = (X \oplus K_0) \cdot K_1$, $M_1(X) = (X \oplus K_3) \cdot K_4$, and $K_1 \neq 0, K_4 \neq 0$.

Consider two plaintext/ciphertext pairs (P, C) and (P^*, C^*), encrypted under the keys $(K_0, K_1, K_2, K_3, K_4)$ and $(K_0, K_1, \overline{K_2}, K_3, K_4)$, respectively. Assume that the plaintext pair (P, P^*) satisfies:

$$M_0(P) \oplus M_0(P^*) = (P \oplus P^*) \cdot K_1 = FF \ldots FF_x.$$

Then, by the complementation property of DES, we have

$$DES_{K_2}(M_0(P)) \oplus DES_{\overline{K_2}}(M_0(P^*)) = FF \ldots FF_x.$$

Since for a fixed key, the decorrelation module M_1 is linear, this implies:

$$C \oplus C^* = M_1(DES_{K_2}(M_0(P))) \oplus M_1(DES_{\overline{K_2}}(M_0(P^*))) = FF \ldots FF_x \cdot K_4. \tag{15}$$

As the right hand side of Equation (15) does not depend on the plaintexts, one can mount an attack similar to the attack on Addition DESX presented in Appendix B.1, with the pair $(P \oplus a, P^* \oplus a)$ considered instead of the pair $(P + a, P^* - a)$. The data and time complexities of the attack are exactly the same as the complexities of the attack on Addition DESX (including its known plaintext variant), and the attack allows to recover the subkeys K_1 and K_4.

Note that after recovering these subkeys, the cipher is equivalent (up to pre/post multiplication by known constants) to:

$$E_{K'_0,K_2,K'_3}(P) = DES_{K_2}(P \oplus K'_0) \oplus K'_3,$$

that is, to DESX![10] Hence, our attack shows that with respect to the related-key model, surrounding DES by decorrelation modules is weaker than adding pre/post key whitening, since it does not increase the security and on the other hand, it allows the adversary to retrieve part of the secret key efficiently.

[10] Note that actually, DESX is a special case of Decorrelation-DES, in which $K_1 = K_4 = 1$. Our attack is not effective against DESX since it allows only to recover the subkeys K_1 and K_4 which are known in the case of DESX to be equal to 1.