Related-key cryptanalysis of AES, Hierocrypt-3, and CipherUnicorn-A

Dmitry Khovratovich
Independent Researcher

for the CRYPTREC Project

November 17, 2012

Contents

Chapter 1

Introduction

1.1 History and relevance of related-key attacks

Block ciphers are a very important primitive in cryptography and are the primary tool to achieve confidentiality. Since the beginning of public cryptography in late 1970s, block ciphers have been extensively studied by both academic community and leading industry agents.

Originally, the security of block ciphers was informally defined as the (in)ability of adversary to read the encrypted information without the key that was used for encryption. Eventually, formal models with rigorous definitions appeared, making the use of block ciphers provably secure under reasonable assumptions.

However, with more and more applied systems in the need of security, the environments where block ciphers are used are regularly changed with sometimes dangerous consequences for the overall security. With the seemingly solid theory of block cipher design, cryptographers were tempted to use block ciphers and their elements as building blocks for other constructions: hash functions, stream ciphers, MACs, modes of operation, and some others. While some adaptations were shown to be secure, the other have been completely broken, some of them soon after introduction.

The notion of "security", however, is too vague. Naturally we assume that a block cipher is secure if it is hard to decrypt the ciphertext without a key. This assumption lacks an explicit formalization of hardness and means of adversary. Both have been formalized in the notion of *semantic security* (see [16] for comprehensive treatment of this and subsequent topics). We would like to avoid excessive formalization here, since the attacks we consider in the report are not treated in that framework. Less formally, semantic security means that the adversary, having submitted a couple of plaintexts to the encryption device, is unable to assign the outputs to the inputs with reasonable probability.

The simplest encryption with a block cipher assumes a message as long as the plaintext block (128 bits in AES). For this case semantic security follows from the properties of a secure *pseudo-random permutation* (PRP), which assumes that under randomly chosen key the permutation implemented with a block cipher is indistinguishable from a random one.

The property of secure PRP is not always enough, when a block cipher is used as a part of *message authentication code*, where it processes a longer message. To invalidate length-extension attacks, several code algorithms use a key derived from the master key in additional encryption calls. This situation might require the cipher to be resistant to attacks that exploit the relation between those keys (*related-key attack*). RMAC [15] and most of algorithms from ISO standard 9797-1 would require such property to constitute a secure MAC.

The situation becomes more difficult if a cipher is used as a building block for a hash function. The key input, earlier uncontrolled by an attacker, often becomes dependent on the message input and falls under a partial control of the adversary. In order to prove the hash function resistance to the message manipulation (natural requirement!), one has to prove the cipher resistance to the key manipulation. HMAC [1], which plugs a keyed hash function into MAC, also requires the underlying primitive to be resistant to related-key attacks.

There are also numerous ad-hoc modes of operation, hash function designs, and MACs, whose security depends on the resistance to some type of related-key attacks. If we include stream ciphers, which are rarely used for hashes and MACs, an infamous case of RC4 is a clear example. The key was modified predictably, so RC4, which is insecure to this type of attack, produced related ciphertexts under related keys, and these relations were easy to exploit in the practical setting [25].

As another environment, where related-key attacks are possible, we mention the so called fault attacks [6]. It is a type of physical attack, where an adversary is able to cause a fault in a block cipher circuit, which results in a variable or operation change. Applications of such attacks to key schedules lead to related-key attacks as we know them. One can argue that fault attacks are difficult to mount and the key schedule may not be an optimal place to apply them.

To summarize, the related-key attacks are a relevant, though arguably minor, threat to the security of a cryptosystem built on a block cipher. Since environments always change, understanding the ciphers' security to related-key attacks might help to make a right decision when designing a secure system.

1.2 Definitions and controversies

The key recovery attacks are clearly the most relevant among related-key attacks. They are easy to formalize and present the largest threat to the system. On the other hand, the so called distinguishing attacks may have far smaller, in some cases practical complexity. They might also demonstrate non-trivial properties of a cipher, which might prevent its use in some ad-hoc constructions.

The standard model of *single-key recovery* attacks on block ciphers assumes that an adversary has access to some data, encrypted or decrypted on an unknown key. Given D plaintexts/ciphertexts, the adversary has to recover a secret k-bit key K. An attack is considered *valid* if the adversary's costs (measured in time requirements T, data D, and/or computational power C) are lower compared to the exhaustive search for one of 2^k keys. The attacks are further classified by the requirements to the nature of data available to the attacker, ranging from ciphertext-only to adaptively-chosen-ciphertexts attacks.

The related-key setting assumes that an adversary is able to decrypt and encrypt not only under the key K_A, but also with keys $K_B = f_2(K), K_C = f_3(K), \ldots, f_r(K)$, which are called *related keys*. The *relation mappings* f_i are chosen by the adversary. The first related-key attacks dealt with simple mappings: rotations [3] and bit flips [18]. Recent attacks on AES [8] exploit dedicated relations, where the fixed difference is applied to expanded keys (subkeys), but not to original keys. These attacks were later called related-subkey attacks [7].

The total number r of related keys is also a parameter of the attack. While the first related-key attacks dealt with only two keys, some related-key attacks on AES [5, 19] worked with as many as 64 related keys in order to get a pair with a desirable relation. We notice that the generic time/memory/data/key tradeoff attack [11] recovers one of r keys with complexity $\frac{2^n}{r}$ for block cipher with an n-bit block.

Definition problems. A cipher is considered *secure* against regular key recovery attacks, if there is no valid attack on it (which we defined earlier). This definition, however, is only seemingly formal. In fact, given no restriction on how the texts are chosen, there are some sets of texts which contain the key material in their description, and hence allow for trivial attacks. Having these attacks ruled out, we are not still fine, as there are non-trivial and non-generic attacks that are valid by our definition and can be applied to virtually any cipher (biclique attacks [13]).

Related-key attacks are even more difficult. As there are trivial data sets, there are trivial key relations. Some of them admit trivial attacks with low complexity, which apply to any cipher [2]. Even worse, for any given cipher it is easy to define a relation that leads to a trivial attack on this particular cipher, e.g. the relation $f(K) = E_K(0)$. To rule out this class of attacks, one may try to restrict the set of relations that are "admissible". For example, we could forbid the relations that explicitly involve the internal operations of the cipher. The disadvantage of this approach is that such a restriction also kills already approved related-subkey attacks like [10].

Another idea is to consider related-key attacks only as an application to high-level constructions that use a cipher as a building block. Therefore, each related-key attack enlarges the set of constructions, which are insecure if instantiated with a particular cipher.

Chapter 2

Key recovery attacks in the related-key setting

Though little classification of cryptanalytic methods has been made so far, we can clearly distinguish statistical methods from deterministic ones. The statistical group works with property $\mathcal{P}$, which is preserved by a cipher for only a portion of plaintexts (or sets of plaintexts). For instance, differential cryptanalysis works with a pair of plaintext with fixed difference, which implies with high probability a specific difference in ciphertexts. A cryptanalyst shows that due to incomplete diffusion or weak confusion property $\mathcal{P}$ is preserved for a specific sets of plaintexts, which imply specific conditions on internal variables in the encryption algorithm. The attack then consists of finding those "right" inputs and deducing keys from those conditions. The related-key model is easily integrated into this framework, as the key relation becomes part of the property being observed in the cipher. The complexity of an attack of this kind as well as its success rate is a random variable.

The second group of methods employs a method that relies on deterministic properties such as invariants and infrequent use of key material during the encryption. For example, meet-in-the-middle attacks work for ciphers that use a half of the key in the first part of encryption, and the other half in the second part. This property leads to a divide-and-conquer method that tests the key parts independently. In contrast to statistical methods, deterministic attacks always output the key, and the attack complexity is easy to determine. However, the related-key setting has not been combined with deterministic methods. Hence we will concentrate on the statistical methods, with differential cryptanalysis being the most applicable tool.

2.1 Regular differential attacks

The idea of the differential cryptanalysis is to consider the bitwise *difference*

$$\Delta P = P_1 \oplus P_2 \tag{2.1}$$

between plaintexts P_1 and P_2, and its propagation through nonlinear and linear transformations of a primitive.

A differential over a transformation F maps input difference Δ_I to output difference Δ_O:

$$\Delta_I \xrightarrow{F} \Delta_O.$$

The *differential probability* (DP) is the number of ordered pairs with input difference Δ_I and output difference Δ_O divided by the total number of pairs with difference Δ_I:

$$\mathrm{DP}_F(\Delta_I, \Delta_O) = \#\{\{x, y\} \mid x \oplus y = \Delta_I \text{ and } F(x) \oplus F(y) = \Delta_O\}/2^n,$$

where n is the output length.

As most ciphers are iterative, differentials over ciphers are combinations of differentials over rounds. These combinations are called *characteristics*, or *trails*:

$$\begin{cases} \Delta_1 \overset{f_1}{\to} \Delta_2; \\ \Delta_2 \overset{f_2}{\to} \Delta_3; \\ \cdots \\ \Delta_{k-1} \overset{f_{k-1}}{\to} \Delta_k. \end{cases} \qquad \Longrightarrow \qquad \Delta_1 \overset{f_1}{\to} \Delta_2 \overset{f_2}{\to} \cdots \overset{f_{k-1}}{\to} \Delta_k.$$

A differential over a whole transformation $\Delta_1 \overset{F}{\to} \Delta_k$ is a cluster of trails with the same Δ_1, Δ_k.

The differential probability should be high enough to be exploitable. For a random function the number of pairs conforming to a differential is a random variable with the binomial distribution (in some cases can be approximated by the Poisson distribution) and with mean 2^{n-m}, where n is the input length and m is the output length. Since the number of ordered pairs with fixed difference is 2^n, a differential probability is expected to be around 2^{-m}. Therefore a differential with a higher probability is a potential weakness, because a random mapping is unlikely to have high probability for the same differential.

The key recovery attack based on a differential with probability 2^{-m} works as follows. An adversary asks for the encryption of $c \cdot 2^m$ plaintext pairs with difference Δ_1. Here c is some constant that determines the success rate and is ranging from 4 to 10 in most applications. The adversary obtains several pairs of ciphertexts with the difference Δ_k. Whenever a nonlinear operation has probability less than one in a differential, it yields a list of possible inputs that conform to the differential. Those inputs for the first and the last rounds are usually simple functions of plaintext, ciphertext, and key, which yields simple equations on key bits and restricts the key space (Figure 2.1). The actual key is found by further restrictions or by exhaustive search.

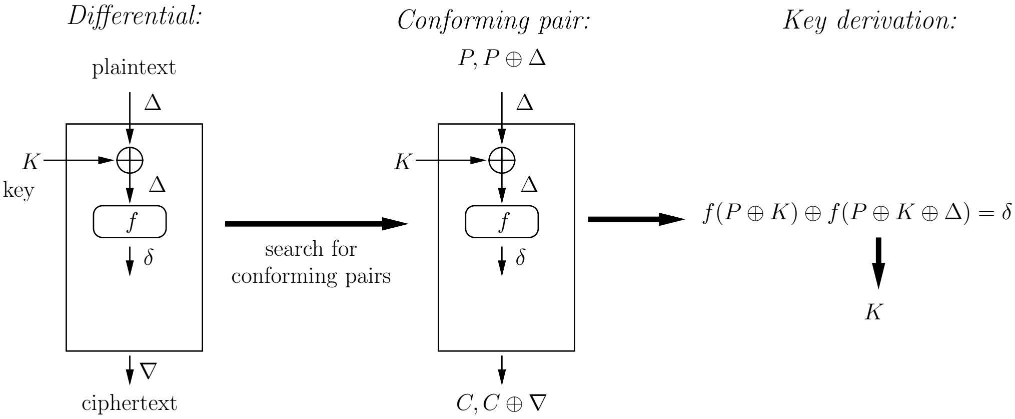

Figure 2.1: Key recovery in differential cryptanalysis.

The related-key setting expands differential attacks. Now the key schedule is part of the primitive over which a differential is constructed. Since there is usually no feedback from the internal state to the key schedule, a cryptanalyst assigns a difference to the master key hence creating related keys. In order to benefit from the related keys, the resulting differential should activate fewer nonlinear operations in the state than the best regular differential does. Therefore, the difference from subkeys should correct or cancel the difference in the internal state. However, this implies that the difference evolution in the key schedule should correspond to the one in the internal state. Moreover, the key schedule should be at least as vulnerable to differential attacks as the internal state part is. In fact, as we will see in the main body of the report, the related-key setting is more likely to give no benefit to key recovery attacks.

2.2 Boomerang attacks

Quite often the first rounds of a differential have much higher probability than the next ones thanks to the control that the adversary has over the plaintext. As a result, the probability of a long differential can be much lower than the multiplication of probabilities of its parts. The *boomerang attacks* are useful for this sort of ciphers, and they work as follows.

The basic boomerang attack [28] is applied to a cipher $E_K(\cdot)$ which is decomposed into $E_1 \circ E_0$. The first sub-cipher E_0 has a differential $\Delta \overset{E_0}{\to} \Delta'$, and E_1, the second one, has a differential $\nabla' \overset{E_1}{\to} \nabla$, with probabilities p and q, respectively.

We encrypt a pair of plaintexts (P, P') with the difference Δ and apply the difference ∇' to the ciphertexts (C, C') (Figure 2.2). Then a new pair of ciphertexts $(D = C \oplus \nabla, D' = C' \oplus \nabla)$ is decrypted to Q, Q'. With probability p the first pair has the difference Δ' in the middle: $E_0(P) = E_0(P') \oplus \Delta'$, and with the probability q^2 the pairs (C, D) and (C', D') have the difference ∇' in the middle:

$$E_1^{-1}(C) \oplus E_1^{-1}(D) = E_1^{-1}(C') \oplus E_1^{-1}(D') = \nabla'.$$

Then

$$E_1^{-1}(D) \oplus E_1^{-1}(D') = \Delta' = E_0(Q) \oplus E_0(Q').$$

Therefore, $Q \oplus Q' = \Delta$ holds with probability p. Finally, we get a *boomerang quartet* (P, P', Q, Q') with probability p^2q^2, while for a random permutation the probability of this event is 2^{-n}.

Real attacks, including attacks on AES, relax some of the differential properties. In particular, there might be a gap between E_1 and E_0, and some bytes of the plaintext difference might not be fixed, which increases the data complexity.

The *amplified boomerang attack* [17] (also called rectangle attack [4]) is a significant improvement, which runs in the chosen-plaintext scenario. The idea is to encrypt sufficiently many plaintext pairs so that the pairs in the middle with the difference ∇' appear due to the birthday paradox. Due to the quartet property we immediately get the second pair with the difference ∇'. These pairs produce two pairs of ciphertexts with the difference ∇ with total probability q^2. Therefore, we get a quartet. In total, we generate $N^2p^2q^2$ quartets out of N pairs, and the minimal data complexity is $2^{n/2}$.

Related-key boomerang attacks. Related-key boomerang attacks assume that the encryption of P, P' and decryption of D, D' is performed on four unknown but related keys K_A, K_B, K_C, K_D. The difference between keys propagate to subkeys and influence the difference propagation in the internal state. Evidently, the related-key attacks make sense only if the resulting differentials have higher probabilities, which usually requires the key schedule to be weaker compared to the internal state transformation.

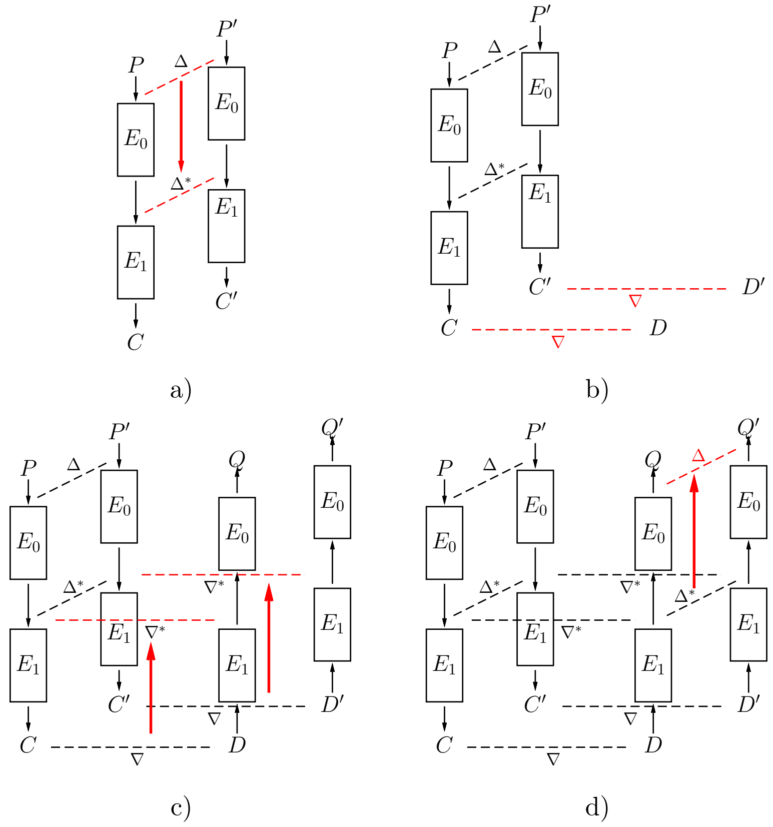

Figure 2.2: Outline of the boomerang attack.

Chapter 3

AES

AES, designed as Rijndael in 1997 and adopted as a standard in 2001 [21], has been the object of numerous cryptanalytic attacks in the last decade. If counting side-channel attacks, the total number of AES cryptanalysis papers is well over 100. There is also a dozen of related-key attacks on AES, which we discuss in this report.

To the large extent, the results presented in this chapter are not the original research. Resistance of AES to related-key attacks has been intensively studied in the crypto community.

The key schedule of AES is not that elegant as the internal round function. It lacks provable diffusion properties, and it is quite likely that resistance to related-key attacks has not been the primary target of the designers. The key schedule is almost linear with XOR being the only diffusion operator. The only nonlinear transformation is an S-box layer, which is applied to four (eight in 256-bit key variants) bytes of the key state per key schedule round.

Nevertheless, the AES designers claimed that the cipher is resistant to related-key attacks. In the first five years after the AES competition the related-key attacks advanced 1-2 rounds ahead of the attacks in the single-key model. Cryptanalysts used a traditional related-key model, where the key difference is selected for the master key. While for ciphers of early 1990s it was hardly a restriction due to simple key schedules in those ones, for AES it has been apparently a huge difference.

3.1 Description

AES is a block cipher with 128-bit plaintext and 128/192/256-bit key, whose internal state S and key K are treated as a 4×4 and $4 \times 4/4 \times 6/\ 4 \times 8$ byte matrices $S_{i,j}$ and $K_{i,j}$.

The encryption process starts with a bytewise XOR of the first 128 bits of the key to the plaintext. Then the internal state undergoes 10/12/14 rounds, depending on the key length. One round is depicted in Figure 3.1 and consists of the bytewise nonlinear transformation SubBytes (simply SB), the byte permutation ShiftRows (SR), the linear transformation MixColumns (MC), which operates columnwise, and the subkey addition AddRoundKey (AK). The last round omits the MixColumns transformation.

SubBytes applies an 8-bit S-box with maximum differential probability as low as 2^{-6} (for most cases 0 or 2^{-7}). The ShiftRows rotates bytes in row r by r positions to the left. The MixColumns is a linear transformation with branch number 5, i.e. in the column equation $(y_0, y_1, y_2, y_3) = MC(x_0, x_1, x_2, x_3)$ there can not be fewer than 5 non-zero variables.

Key schedule. The key schedule procedure is a bit different for each key length. Since each round invokes a 128-bit subkey, the number of key schedule rounds varies significantly depending on the key length. The 128-bit key undergoes 10 rounds of the key schedule and produces 11 expanded keys including the master key. The 192-bit key undergoes 8 rounds and produces 9 192-bit expanded keys, which are used as 13 128-bit subkeys. The 256-bit key undergoes 7 rounds and produces 8 256-bit expanded keys. The expanded keys are denoted as $K^0, K^1, \ldots, K^{10}$.

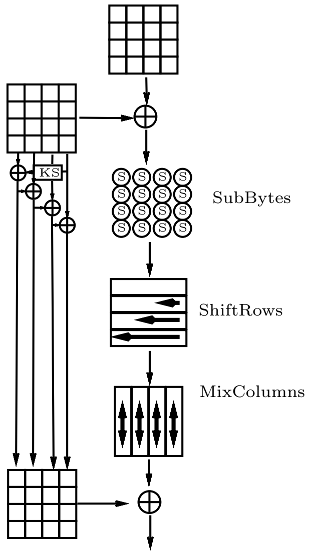

Figure 3.1: Round of AES-128

One key schedule round of AES-128 is mostly linear with only 4 S-boxes:

$$\begin{aligned} K^{l+1}[\text{Column 0}] &\leftarrow S(K^l[\text{Rotated Column 3}]) \oplus K^l[\text{Column 0}] \oplus C^l; \\ K^{l+1}[\text{Column j} > 0] &\leftarrow K^{l+1}[\text{Column } j-1] \oplus K^l[\text{Column j}], \end{aligned}$$

where $S()$ stands for the S-box, and C^l — for the round-dependent constant. Therefore, each round has 4 S-boxes.

One key schedule round of AES-192 is very similar:

$$\begin{aligned} K^{l+1}[\text{Column 0}] &\leftarrow S(K^l[\text{Rotated Column 5}]) \oplus K^l[\text{Column 0}] \oplus C^l; \\ K^{l+1}[\text{Column j} > 0] &\leftarrow K^{l+1}[\text{Column } j-1] \oplus K^l[\text{Column j}], \end{aligned}$$

One key schedule round of AES-256 has an additional S-box layer:

$$\begin{aligned} K^{l+1}[\text{Column 0}] &\leftarrow S(K^l[\text{Rotated Column 3}]) \oplus K^l[\text{Column 0}] \oplus C^l; \\ K^{l+1}[\text{Column 4}] &\leftarrow K^{l+1}[\text{Column } j-1] \oplus S(K^l[\text{Column j}]), \\ K^{l+1}[\text{Column j} = 1,2,3,5,6,7] &\leftarrow K^{l+1}[\text{Column } j-1] \oplus K^l[\text{Column j}]. \end{aligned}$$

The other details of the AES specification are irrelevant to our attack and we refer the reader to [21].

The expanded keys K^r have the following property: one byte in a subkey K^r affects only two bytes in the subkey K^{r-1}.

3.2 Local collision approach and long differentials

The notion of a local collision comes from the cryptanalysis of hash functions with one of the first applications by Chabaud and Joux [14]. The idea is as follows. The key/message schedule difference in the first subkey is injected to the internal state, causing a *disturbance*. The key difference in the second subkey must *correct* it with some reasonable probability.

Application of the inverse round copies both disturbance and the correction difference one column to the right, hence creating another local collision. Going back through the key schedule, we obtain more local collisions till the difference pass through the S-boxes in the key schedule. Since each local collision is probabilistic, the goal is to have as few collisions as possible in order to reduce the complexity of the attack.

The related-key scenario is very similar to the hash function collision search, as we assign arbitrary difference to the key similarly to what we do for the message. However the attacker can not control the actual value of the key, so the difference propagation in the key schedule must be deterministic.

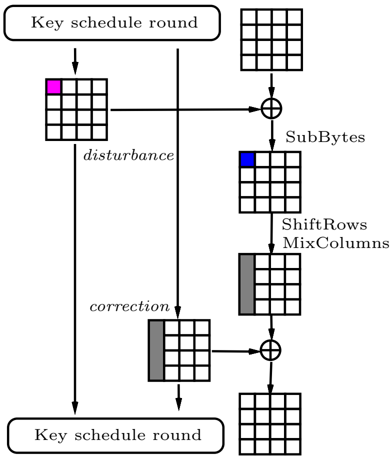

Figure 3.2: A local collision in AES-256.

Local collisions in AES are best understood on a one-round example in AES-256 (Fig. 3.2). The two subkeys that form a local collision pattern are independent part of the 256-bit expanded key, so the difference can be freely chosen. If we inject a one-byte difference in byte (0,0), it activates a single S-box, takes one of 2^7 possible values afterwards, and expands into a full column difference, which should be corrected by the second subkey. This differential holds with probability 2^{-6} if based on an optimal differential for an S-box:

$$\text{0x01} \overset{\text{SubBytes}}{\Longrightarrow} \text{0x1f}; \quad \begin{pmatrix}\text{0x1f}\\0\\0\\0\end{pmatrix} \overset{\text{MixColumns}}{\Longrightarrow} \begin{pmatrix}\text{0x3e}\\\text{0x1f}\\\text{0x1f}\\\text{0x21}\end{pmatrix}$$

The key schedule creates more local collisions in the previous key schedule rounds. Hence we obtain a set of local collisions, where the expansion of the disturbance (also called *disturbance vector*) and the correction differences compensate each other. The probability of the full differential trail is then determined by the number of active S-boxes in the key-schedule and in the internal state. The latter is just the number of the non-zero bytes in the disturbance vector.

Therefore, to construct an optimal trail we have to construct a minimal-weight disturbance expansion, which will become a part of the full key schedule difference. For the AES key schedule, which is mostly linear, this task can be viewed as building a low-weight codeword of a linear code. Simultaneously, correction differences also form a codeword, and the key schedule difference codeword is the sum of the disturbance and the correction codewords. In the simplest trail the correction codeword is constructed from the disturbance codeword by mere shifting four columns to the right and applying the S-box–MixColumns transformation.

An example of a good key-schedule pattern for AES-256 is depicted in Figure 3.3 as a 4.5-round codeword. In the first four rounds the disturbance codeword has only 9 active bytes (red cells in the picture), which is the lower bound. We want to avoid active S-boxes in the

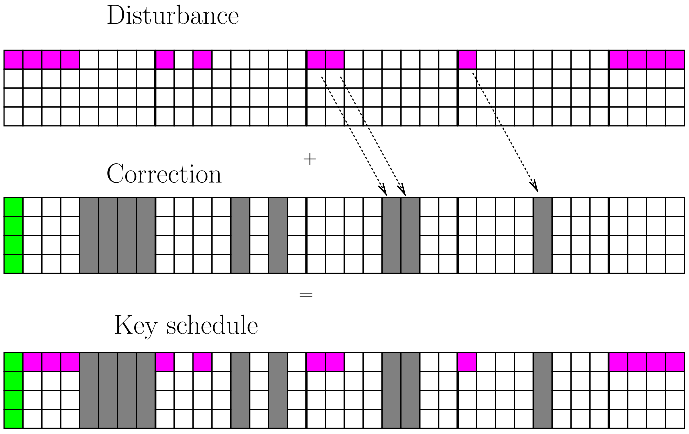

Figure 3.3: Full key schedule difference (4.5 key-schedule rounds) for AES-256.

key schedule as long as possible, so we start with a single-byte difference in byte $K^4_{0,0}$ and go backwards. Due to a slow diffusion in the AES key schedule the difference affects only one more byte per key schedule round. The correction (grey) column should be positioned four columns to the right, and propagates backwards in the same way. The last column in the first subkey is active, so all S-boxes of the first round are active as well, which causes an unknown difference in the first (green) column.

This pattern yields an 8-round differential in AES-256 with 9 local collisions. For some proportion of keys, the unknown leftmost column is the result of the MixColumn 1-to-4 expansion. Hence there is a set of keys, for which this pattern can be longer. This property has been used in the first attacks on the full AES-256.

AES-128 and AES-192 The AES versions with 128- and 192-bit keys are more difficult for a cryptanalyst. The length of the key schedule pattern is clearly limited by the key state width measured in columns. Indeed, as soon as the disturbance byte reaches the rightmost column and enters the nonlinear transformation with rotation, it is injected into improper position in the internal state. Hence the number of key schedule rounds should not exceed the column width. The following table gives evidence to the width/rounds ratio for different key sizes:

Key length	Key schedule rounds	Key state width (columns)
128	11	4
192	9	6
256	8	8

As a result, whereas good patterns exist for the full AES-256, only 60% of AES-192 and less than half of AES-128 can be analyzed with a single pattern. To penetrate more rounds, a more restrictive model of boomerang attacks is applied.

Optimality of related-key differentials. Biryukov and Nikolic described a tool that finds optimal related-key differentials [12]. They proved that AES-128 has no 6-round related-key differential trails with probability higher than 2^{-128}. The best differential trail for AES-192 covers 11 rounds and has 31 active S-boxes (20 in the state, 11 in the key). The best differential trail for AES-256 covers all 14 rounds. Similar properties are expected of differentials grouping several trails.

3.3 Distinguishing attacks

Though distinguishing attacks are not a direct threat, they demonstrate a potential vulnerability for high-level constructions that use the cipher. AES-256 has been the first and the only AES variant attacked in this manner; there have been other ciphers and hash functions later [20]. The following attack is a summary of [10], and is provided here also for its quite low complexity compared to other attacks.

The idea is to extend the key schedule pattern shown at Figure 3.3 so that it covers the full AES-256. The differential in the key schedule becomes probabilistic, as active bytes in fourth and eighth column of the key state activate a total of 5 S-boxes. It is required that the local collision idea still holds after the differences pass through S-boxes, which puts restriction on the number of possible actual values for differences.

Nevertheless, it is relatively easy to find particular values for the differences, and they are specified in [10]. The number of 5 active S-boxes implies that about 1 out of 2^{35} keys conforms to the key schedule differential. This yields a class of weak keys vulnerable to the attack of complexity 2^{131}.

A weaker model leads to attacks with lower complexity, later called distiguishing attacks. Keeping local collisions within the internal state, we obtain a related-key differential for the whole 14-round AES-256. To increase the probability, we lift the local collision requirement in the first rounds and get 16 active S-boxes in the first round. The plaintext has still the zero difference, and the ciphertext has a difference with only four active bits.

Since the differences have been chosen by a cryptanalyst, this raises questions on whether a tuple with similar properties can be found with a generic algorithm, and if so, what the complexity is. Indeed, one can choose two arbitrary plaintexts and encrypt them on two arbitrary keys, then obtaining a tuple which conforms to some trail for free. The situation is different if a cryptanalyst requires many tuples with the same difference to appear, even though the difference itself is not specified. This led to the notion of differential multicollision presented as follows.

Definition 1 *A set of two differences and q pairs*

$$\{\Delta_K, \Delta_P;\ (P_1, K_1),\ (P_2, K_2), \ldots, (P_q, K_q))\}$$

is called a differential q-multicollision *for a cipher $E_K(\cdot)$ if*

$$E_{K_1}(P_1) \oplus E_{K_1 \oplus \Delta_K}(P_1 \oplus \Delta_P) = E_{K_2}(P_2) \oplus E_{K_2 \oplus \Delta_K}(P_2 \oplus \Delta_P) = \\ = \cdots = E_{K_q}(P_q) \oplus E_{K_q \oplus \Delta_K}(P_q \oplus \Delta_P). \quad (3.1)$$

The following theorem says that if the cipher is viewed as a black box, then for sufficiently large q such a construction can be found with complexity 2^n or even larger, where n is the plaintext size.

Theorem 1 *To construct a differential q-multicollision for an ideal cipher with an n-bit block an adversary needs at least $O(q \cdot 2^{\frac{q-2}{q+2}n})$ queries on the average.*

On the other hand, it is easy to demonstrate that any q tuples conforming to a single related-key differential trail produce a differential q-multicollision. The authors of [10] show how to find such a tuple for the first five rounds of the trail for free. With 11 S-boxes left, the complexity of finding a single tuple grows to 2^{67}. Hence AES-256 is clearly far from being a good instantiation of an ideal cipher.

This attack is barely practical, as 2^{67} is beyond capabilities of academic institutions, putting aside the minority of the threat posed by the attack. Still it is possible to demonstrate some non-trivial properties of AES-256 in practice by relaxing the difference conditions for the plaintext. Then the complexity plummets to 2^{37}.

This techniques stops at 5 rounds of AES-128 and 11 rounds of AES-192. The latter case yields a key recovery attack with complexity 2^{186} [12].

3.4 Boomerang attacks on AES-128, AES-192, and AES-256

The presence of active S-boxes in the key schedule in the differential trail reduces the attack scope to a weak key class. To avoid this restriction a cryptanalyst has to work in an attack framework that allows for shorter differential trails. For all the versions of AES such a framework is yielded by the concept of boomerang attacks and their modifications (Section 2.2).

Boomerang attacks on AES-192 and AES-256 were proposed in [8] and later refined in [12]. A boomerang attack on AES-128 was published in [12].

AES-256. The boomerang attack on the full AES-256 is the simplest of the three. The subcipher E_0 covers rounds 1–9, and E_1 covers rounds 9–14. The E_0 differential is based on a local collision in round 7, where the difference is injected in byte (1,1). The key difference is specified in the expanded key K^3. Though the differential has 15 active S-boxes in the internal rounds, the most of them are in the first or in the last round and do not contribute to the amplified probability $\widehat{p}$, which is equal to 2^{-30}. The E_1 differential is based on a local collision in round 13 with the key difference specified in the expanded key K^6. It has a higher amplified probability of 2^{-18}, which yields 2^{-96} as the total probability $\mathbb{P}$ of the boomerang differnetials Due to a truncated differential in the first round, the total computational complexity of the attack slightly exceeds $1/\mathbb{P}$ and is equal to 2^{99}. The data requirements are 2^{99} adaptively-chosen plaintexts and ciphertexts. The memory complexity is estimated as 2^{77}.

It was proved that the relation between four keys is well defined, i.e. that any key uniquely determines the other three.

Though the computational complexity of this attack is high, it drops significantly as the number of rounds decreases. For instance, the 13-round version of AES-256 can be attacked with almost practical complexity of 2^{76} [9]. The reduction is achieved by removing the expensive first round from the differential characteristic.

AES-192. The boomerang attack on the full AES-192 is slightly more complicated. First, it is an amplified differential attack, because the ciphertext differential is not determined completely. The E_0 differential is based on a local collision in round 4, and the key difference is specified for the expanded key K^2. The E_1 differential is based on a local collision in round 8 with the key difference specified for the expanded key K^5. The total probability $\mathbb{P}$ of the boomerang is estimated as 2^{-48}. Due to the nature of the amplified boomerang attack, the total computational complexity is very large and is estimated as 2^{169} with additional requirements of 2^{116} chosen plaintexts [12].

It has been also demonstrated that the relation between four keys is well defined.

AES-128. Related-key attacks on AES are not much better than single-key differential attacks. The reason is that the relatively small key size provides sufficient diffusion to activate many key schedule S-boxes and render the local collision approach useless. As a result, the optimal boomerang attack covers 7 rounds only and involves only two related keys with no key difference in the E_0 differential [12]. The latter is a well-known 3-round boomerang differential. The E_1 differential is a 4-round related-key differential based on a local collision in round 6. The key difference is specified in the expanded key K^5. Due to faster diffusion, the E_1 differential has 8 active S-boxes, which brings the total boomerang probability to 2^{-96}. The computational complexity of the attack is 2^{97}, and it requires 2^{97} chosen plaintexts.

Chapter 4

Hierocrypt-3

Hierocrypt-3 [27] is an SPN-based cipher with each round containing two subrounds with distinct linear transformations. When investigating the resistance of Hierocrypt-3 to related-key attacks, we first checked the applicability of the local collision method earlier applied to AES.

The related-key attacks on AES are so powerful because the AES key schedule is mostly linear, and there are differentials with probability 1 in the key schedule which last for several rounds, depending on the key size. As we explain in the further text, this approach does not work for Hierocrypt-3.

Design

Here we discuss the internal rounds and the key schedule of Hierocrypt-3. We also introduce some new notation.

Hierocrypt-3 is a version of an SPN cipher, which operates on a 128-bit state. Each round consists of two sub-rounds. Odd sub-rounds starts with a subkey addition and proceed with a layer of 8-bit S-boxes and a diffusion layer mds_L, which transforms 32-bit words independently. Even sub-rounds have another diffusion layer MDS_H, which is applied to the entire state.

Each full round uses two 128-bit subkeys, which come from the key schedule procedure.

The key schedule, irrespectively of the key size, operates on a 256-bit state, which we call a *key state.* The 128-bit and 192-bit keys are padded to a 256-bit state $Z_1||Z_2||Z_3||Z_4$ as follows:

$$\begin{aligned} 128: &\quad K_1||K_2 \longrightarrow K_1||K_2||K_1||C_1; \\ 128: &\quad K_1||K_2||K_3 \longrightarrow K_1||K_2||K_3||C_2, \end{aligned}$$

where C_1, C_2 are predefined constants.

The very first key state is denoted by $Z_1^{(-1)}||Z_2^{(-1)}||Z_3^{(-1)}||Z_4^{(-1)}$.

The key state undergoes 8/9/10 rounds for 128/192/256-bit key length, respectively. The first round is special, as it omits a linear function. The rest are two groups of rounds, which we mark as 'type I' and 'type II'. The key state words Z_3 and Z_4 are updated linearly every round, while Z_1 and Z_2 follow the Feistel network with additional input from Z_3 and Z_4.

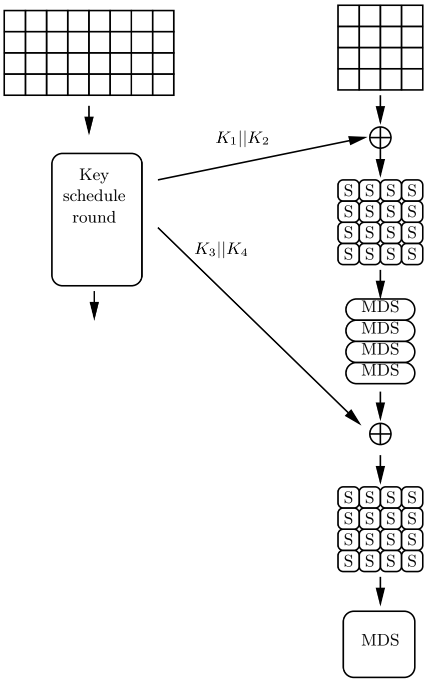

Figure 4.1: One round of Hierocrypt-3.

The key state variables in the round of the first type are modified as follows:

$$\begin{aligned}
(Z_3^{(r)}, Z_4^{(r)}) &\leftarrow L_{34}(Z_3^{(r-1)}, Z_4^{(r-1)});\\
Z_1^{(r)} &\leftarrow Z_2^{(r-1)};\\
F^{(r)} &\leftarrow F_\sigma(Z_1^{(r)} \oplus Z_3^{(r)});\\
Z_2^{(r)} &\leftarrow Z_1^{(r-1)} \oplus F_\sigma(Z_2^{(r-1)} \oplus Z_3^{(r)}), \quad r = 0, 1, 2, \ldots
\end{aligned}$$

where L_{34} is a linear function. We note that the round function is invertible. It is a also bitslice transformation, which splits its input into 8-bit words and process each slice of bits with an independent linear transformation. The other details are irrelevant for our analysis. The function F_σ consists of a level of S-boxes followed by the function $P^{(16)}$. The Hierocrypt S-box is an 8-bit nonlinear transformation and has the same differential-linear properties as the S-box of AES. The function $P^{(n)}$ splits its input into n-bit words and applies a bitwise invertible linear transformation to them.

Every round of type I generates a pair of 128-bit subkeys $K_1||K_2$ and $K_3||K_4$ as follows:

$$\begin{aligned}
K_1^{(r)} &\leftarrow Z_2^{(r)};\\
K_2^{(r)} &\leftarrow Z_3^{(r)} \oplus F^{(r)};\\
K_3^{(r)} &\leftarrow Z_4^{(r)} \oplus F^{(r)};\\
K_4^{(r)} &\leftarrow Z_1^{(r)} \oplus Z_4^{(r)}, \quad r = 1, 2, \ldots
\end{aligned}$$

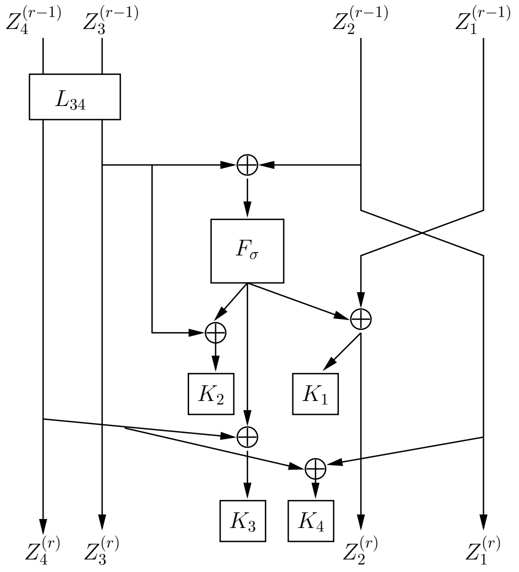

Figure 4.2: Round of Hierocrypt-3 key schedule

The round function of type II is quite similar. It is almost equivalent to the inversion of the type I round, but the linear function that operates on Z_3 and Z_4 is different.

$$\begin{aligned} Z_1^{(r)} &\leftarrow Z_2^{(r-1)}; \\ F^{(r)} &\leftarrow F_\sigma(Z_1^{(r)} \oplus Z_3^{(r-1)}); \\ (Z_3^{(r)}, Z_4^{(r)}) &\leftarrow L'_{34}(Z_3^{(r-1)}, Z_4^{(r-1)}); \\ F^{(r)} &\leftarrow F_\sigma(Z_1^{(r)} \oplus Z_3^{(r)}); \\ Z_2^{(r)} &\leftarrow Z_1^{(r-1)} \oplus F_\sigma(Z_2^{(r-1)} \oplus Z_3^{(r)}); \end{aligned}$$

In contrast to the most key schedules, subkeys of Hierocrypt-3 are not merely bits of the key state, but linear functions of them. Though it is possible to derive explicit equations that relate subkeys of distinct rounds, we found it difficult to exploit. Hence we proceed with the analysis of the key schedule rounds.

4.1 Key schedule properties

First, it is quite easy to express the key state as a function of the subkey pair and then express the subkeys of one round as a function of previous subkeys. For the round of the first group it is shown in Figure 4.3. However, a new round function uses two nonlinear operations F_σ instead of one. One can see that the resulting round function is also more complicated.

Secondly, we notice that rounds of type II have the same properties. If the subkeys are expressed via each other, the resulting scheme would be very similar to that in Figure 4.3 (we do not present the second scheme here).

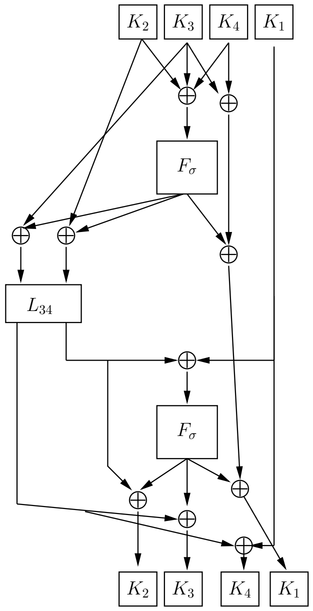

Figure 4.3: Round of Hierocrypt-3 key schedule: subkey view

Finally, we notice that the master key is a nonlinear function of subkeys. As a result, a simple difference in the first subkey may result a sophisticated and probabilistic difference in the initial 256-bit key state. Since the 128-bit and 192-bit keys are padded to 256 bits, this implies that a good subkey difference may not be a valid difference in a 128-bit and a 192-bit master key.

4.2 Differentials

Let us explore related-key differentials in Hierocrypt-3, and start with the key schedule. As we stressed earlier, the longest related-key differential in a cipher would not be longer than the longest key schedule differential. We found it difficult to construct a differential based on subkeys, and decided to operate on the key state instead.

First, we notice that the diffusion in the key state is good enough to spread a difference to the input of the nonlinear function F_σ within one or two rounds. Therefore, if the difference in the key state is not properly chosen, most of S-boxes are activated and the differential becomes key-dependent. Though there exist remarkable related-key attacks which work for a subclass of keys, we decided not to pursue this direction. The main reason is that the size of the resulting subclass decreases quickly as the number of rounds grows and most of S-boxes are active. Another reason is that the this type of attack is much less practical.

Therefore, we start with differentials which do not activate F_σ functions, and will subsequently relax this condition in the first and the last round of a differential. Let us denote the output difference of F_σ by ΔF. Then

$$\Delta F = 0 \Leftrightarrow \Delta Z_1 = \Delta Z_3.$$

This is a sufficient condition for a single round. However, such a short differential does not make much sense, since we can simply consider a difference in a single round key without referring to the key state.

Let us consider a longer high-probability differential. The condition $\Delta F = 0$ in two consecutive rounds r and $r+1$ yields the following equations:

$$\Delta Z_1^{(r)} = \Delta Z_3^{(r)};$$
$$\Delta Z_1^{(r+1)} = \Delta Z_3^{(r+1)} \Leftrightarrow \Delta Z_2^{(r)} = \Delta Z_3^{(r+1)}.$$

Let us denote $\Delta Z_1^{(r)}$ by δ and $\Delta Z_1^{(r+1)}$ by δ'. Then we get the following constraints for $r = 1$, i.e. for the first two rounds of the key schedule that output subkeys:

r	ΔZ_1	ΔZ_2	ΔZ_3	ΔZ_4	ΔF
1	δ	δ'	δ	–	0
2	δ'	δ	δ'	–	0

Since Z_3 and Z_4 update each other via an invertible linear transformation, these constraints (almost) uniquely define ΔZ_4:

r	ΔZ_1	ΔZ_2	ΔZ_3	ΔZ_4	ΔF
1	δ	δ'	δ	$L_5(\delta, \delta')$	0
2	δ'	δ	δ'	$L_6(\delta, \delta')$	0

where L_5 and L_6 are linear functions. We note, however, that the ΔF_σ is likely to be non-zero in round 0, so the master key difference may not be expressed as a simple function of δ and δ'.

We also notice that the functions L_5 and L_6 are bitslice in the same sense as the function L_{34}: it partitions the input into 8-bit words and processes each bit slice independently.

Hence the subkey differences can be expressed as functions of δ and δ':

r	ΔK_1	ΔK_2	ΔK_3	ΔK_4
1	δ'	δ	$L_5(\delta, \delta')$	$\delta \oplus L_5(\delta, \delta')$
2	δ	δ'	$L_6(\delta, \delta')$	$\delta' \oplus L_6(\delta, \delta')$

We could not find a way to use this class of differentials as is. First, we considered a local collision approach, where a subkey difference cancels the difference introduced by the previous subkey. However, we can not maintain a good difference in the key state. Though the difference propagation is deterministic, and the linear function L_5 and L_6 are bitslice, both functions have very good diffusion. As a result, either $K_1||K_2$ or $K_3||K_4$ have high weight in each non-zero slice. A local collision would be very expensive in terms of probability, as every S-box in the internal round would be active.

For the same reason a differential that does not cancel differences in the internal state has too low probability because of the internal diffusion. Then we attempted to construct a related-key differential that keeps zero difference in the internal state as long as possible. This would require some subkey be equal to zero. Since the full key can not have the zero difference, the zero difference in the internal state survives for one round only.

1-round differentials Let us consider differentials which determines three consecutive subkeys and sets the second subkey to zero.

The first case deals with $K_3||K_4 = 0$ and $K_1||K_2$ before and after this injection being non-zero. We have the following system of equations:

$$\begin{cases} \Delta Z_4^{(1)} = \Delta F^{(1)}; \\ \Delta Z_4^{(1)} = \Delta Z_1^{(1)}; \end{cases}$$

With $\Delta F^{(1)} = 0$ this implies

r	ΔZ_1	ΔZ_2	ΔZ_3	ΔZ_4	ΔF
1	0	δ'	0	0	0
2	$\Delta F^{(2)}$	$\Delta F^{(2)}$	0	0	$\Delta F^{(2)}$

(4.1)

and

r	ΔK_1	ΔK_2	ΔK_3	ΔK_4	ΔF
1	δ'	0	0	0	0
2	$\Delta F^{(2)}$	$\Delta F^{(2)}$	$\Delta F^{(2)}$	δ'	$\Delta F^{(2)}$

We note that the value $\Delta F^{(2)}$ depends on the actual key value.

Let us figure out the type of difference in the master key. We recall that the key schedule round is invertible, and Z_3, Z_4 are modified independently of Z_1, Z_2. As a result, the difference in $Z_3^{(-1)}||Z_4^{(-1)}$ is zero. This is valid option for a 192-bit key, but highly unlikely for a 128-bit key.

For the sake of completeness, we also consider the case with $K_1||K_2 = 0$ while the subkeys $K_3||K_4$ are non-zero. However, this condition together with $\Delta F = 0$ implies an all-zero difference, so this case proves worthless.

Key schedule rounds of type II. The key schedule rounds of type II are almost equivalent to the inversion of the rounds of type I. The distinction is linear transformation L_{34} and the subkey generation procedure. However, the main attack properties remain the same, so it equally difficult to construct a high-probability differential more than two rounds long. The two-round differential in the key schedule has the same property of high weight, which implies that it is not suitable for a good differential in the cipher. Since a version of the cipher that covers middle rounds is less attractive, we decided against the separate analysis of the rounds of type II. Instead, we claim that the following results and attacks can be translated with pretty much the same efficiency to any starting point within the cipher.

The middle two rounds where a round of one group follows a round of the other group appear to be as strong as any other pair of rounds.

4.3 Attacks

1-round attack. Let us describe a 1-round related-key attack for Hierocrypt-3-256. We work with the differential (4.1), which results in a valid difference in the 192/256-bit master key. We restrict here to 256-bit key.

Suppose we are able to encrypt and decrypt on two keys K_A and K_B, whose relation follows the differential (4.1) for some 64-bit constant δ'. Then we encrypt two plaintexts with difference $\Delta P = \delta'||0$ on different keys. Since the first subkey $K_1||K_2$ has difference $\delta'||0$, the two encryptions are identical until the last subkey injection.

The ciphertext difference ΔC is equal to $\Delta F^{(2)}||\Delta F^{(2)}$, where

$$\Delta F^{(2)} = F_\sigma(Z_1^{(1)} \oplus Z_3^{(1)}) \oplus F_\sigma(Z_1^{(1)} \oplus Z_3^{(1)} \oplus \delta').$$

Here Z_i stand for key state words of K_A.

Given $\Delta F^{(2)}$, we apply the inverse of the linear function in F_σ to it and obtain the output difference of the S-box layer. For each active S-box we obtain 2 or 4 candidates for the input value. If all S-boxes are active, we obtain at maximum 2^{16} candidates for the value $Z_1^{(1)} \oplus Z_3^{(1)}$. Hence we recover 48 bits of the key state.

Unfortunately, the other state bits are difficult to recover unless another key relation is considered. Due to few attacked rounds, we do not investigate this case. Nevertheless, for each remaining 208 bits of the key state we can reconstruct the master key and test if it the right key for given plaintext-ciphertext pairs. We will need one more plaintext-ciphertext pair for the filtering. Hence we obtain a simple 1-round attack, which requires 3 plaintexts and works in about 2^{208} time for the values δ' that activates all S-boxes of F_σ.

2-round boomerang attack. The 1-round attack can be extended into a 2-round attack in the boomerang framework. The boomerang attack has some notable properties in the related-key model. First, it usually involves four rather than two related keys, one for each computation line. Secondly, key relations within non-trivial key schedules can be quite complicated in the boomerang attack, because one of them is usually chosen to better suit a differential in the second part of the cipher, where subkeys can be a complex function of the master key.

		E_0		E_1	
		Key $K_A \oplus K_B$ $K_C \oplus K_D$	State	Key $K_A \oplus K_C$ $K_B \oplus K_D$	State
Plaintext			$\delta'\|\|0$		
Round 1	$K_1\|\|K_2$	$\delta'\|\|0 \to$			
			$0\|\|0$		
	$K_3\|\|K_4$	$0\|\|0 \to$			
			$0\|\|0$		$\delta\|\|0$
Middle state					
Round 2	$K_1\|\|K_2$	$\Delta F'\|\|\Delta F'$		$\delta\|\|0 \to$	
			$\Delta F'\|\|\Delta F'$		$0\|\|0$
	$K_3\|\|K_4$			$0\|\|0 \to$	
					$0\|\|0$
Whitening	$K_1\|\|K_2$			$\Delta F\|\|\Delta F$	
Ciphertext					$\Delta F\|\|\Delta F$

Table 4.1: Differentials and subkey relations in the boomerang attack on reduced Hierocrypt-3.

All these issues hold for Hierocrypt-3. Let E_0 and E_1 stand for two consecutive internal rounds. We consider two 1-round related key differentials, described in Section 4.2:

$$\delta||0 \to \Delta F||\Delta F \text{ and } \delta'||0 \to \Delta F'||\Delta F',$$

where δ and δ' are some constants. They can be chosen arbitrarily provided that they affect different S-boxes in the function F_σ. The value $F(\delta)$ depends on the key value, so it must be guessed in order to mount an attack. The attack recovers secret related keys K_A, K_B, K_C, K_D. The key relations are defined via subkeys in Table 4.1.

The brief attack description would be as follows:

- Prepare a pair of plaintexts P and $P' = P \oplus \delta||0$ and encrypt them on K_A and K_B, respectively. Denote the ciphertexts by C, C'.
- For each difference $\nabla = \Delta F'||\Delta F'$, which yields a valid differential for F_σ obtain new ciphertexts
$$D = C \oplus \nabla,\ D' = C' \oplus \nabla.$$

- Decrypt D, D' on K_C, K_D, respectively. Denote the new plaintexts by Q, Q'.
- Check if $Q = Q' \oplus \delta||0$, i.e. if P, P', Q, Q' form a boomerang quartet. If so, the guess of $F(\delta')$ is correct and we derive the input values of active S-boxes in the key schedule. This restricts the number of candidates for K_A, K_B, K_C, K_D.

The complexity of the attack depends on the hamming weight of δ' and the number of key relations we can exploit. In the natural case, where only a single key relation is available, the 64-bit difference δ' is optimal. Then we obtain 48 bits of the secret subkey, and the other bits we have to recover by exhaustive search. Still, partial key recovery is practical.

The full key recovery would require the exhaustive search for remaining 208 bits, which yields the total computational complexity of 2^{208} cipher calls. As we pointed out earlier, the difference in subkeys would not translate to the valid difference in the 128-bit key padded to the 256-bit state.

The 192-bit situation is more favorable. We guess $Z_3^{(-1)}$ and compute $Z_3^{(1)}, Z_4^{(1)}$ as $Z_4^{(-1)}$ is constant. It remains to exhaustively search for remaining $256 - 48 - 128 = 80$ bits, which yields the total complexity of the key recovery equal to 2^{144} cipher calls. Partial recovery of 48 key bits requires roughly 2^{48} calls.

We conclude that Hierocrypt-3 is resistant to related-key attacks. The most successful attack breaks only 2 rounds, and the total complexity is very high. We note that the best single key attacks break more rounds [26], because they do not have to control the difference propagation in the key schedule.

Chapter 5

CipherUnicorn-A

The cipher CipherUnicorn-A has been designed by NEC corporation [22] and submitted for the CRYPTREC project in early 2000s. It employs the Feistel scheme with 16 rounds of very heavy round function. The plaintext block is 128-bit long. The first round is preceded by and the last round is followed by the addition of a whitening 128-bit subkey.

Few cryptanalytic results have been published on CipherUnicorn-A, which we attribute to the strength of the round function. Existing self-evaluation report [23] is very brief on the related-key attacks and concentrates mainly on the statistical properties of the round function.

The round function F_r of CipherUnicorn-A consists of the subkey addition and two nonlinear functions with high diffusion (Figure 5.1). We refer to the specification for the details. We do not need a detailed explanation of the round function for related-key attacks, since they are efficiently prevented by the key schedule.

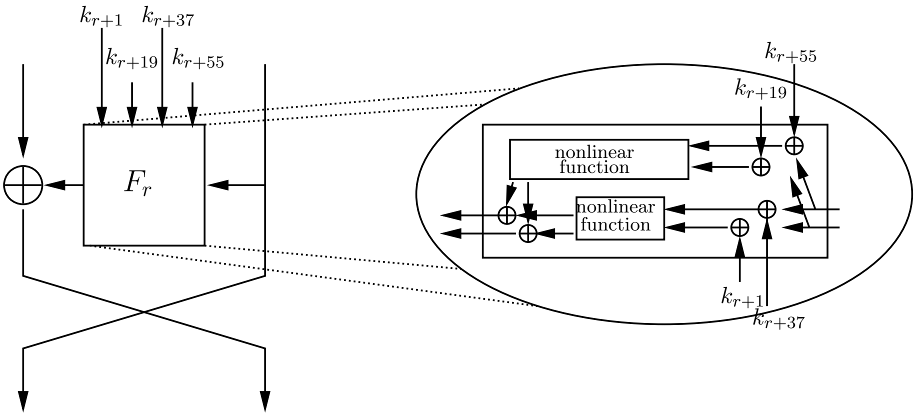

Figure 5.1: Round of CipherUnicorn-A and overview of the round function

The master key of 128/192/256 bits is splitted into 32-bit words. Let us describe the key schedule on the example of 128-bit key, as the other variants are similar.

The 128-bit key is divided into four 32-bit words M_0, M_1, M_2, M_3. Then the key undergoes a sequence of 156 identical rounds:

$$M_3^{\mathrm{new}}||M_0^{\mathrm{new}} = MT(M_0^{\mathrm{old}}, M_1^{\mathrm{old}}); \tag{5.1}$$

$$M_1^{\mathrm{new}} = M_2^{\mathrm{old}}; \quad M_2^{\mathrm{new}} = M_3^{\mathrm{old}}, \tag{5.2}$$

where $MT(X,Y)$ is a simple function (Figure 5.2):

$$X \leftarrow X \otimes 01010101 \pmod{2^{32}};$$
$$Y \leftarrow Y \oplus \big[S(X_3)||S(X_3)||S(X_3)||S(X_3)\big],$$

where $S()$ is the 8-bit S-box. The multiplication by 01010101 is a linear transformation on bytes of X:

$$X_3||X_2||X_1||X_0 \leftarrow (X_3 \oplus X_2 \oplus X_1 \oplus X_0)\,||\,(X_2 \oplus X_1 \oplus X_0)\,||\,(X_1 \oplus X_0)\,||X_0;$$

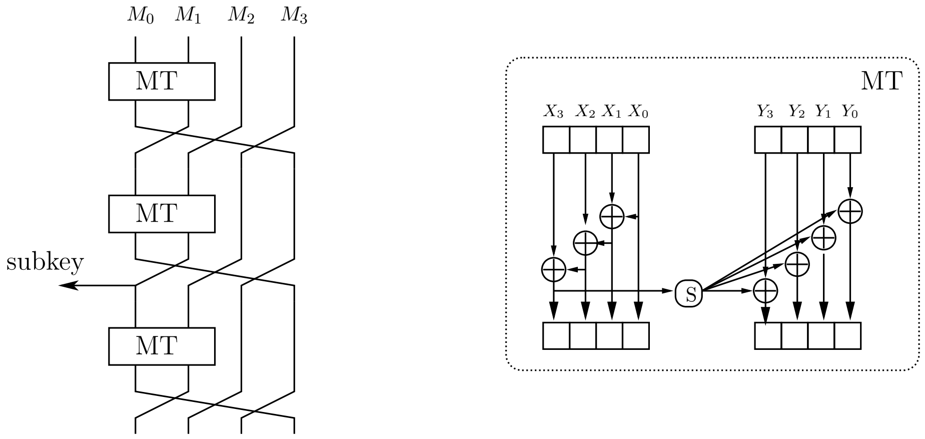

Figure 5.2: Three key schedule rounds of CipherUnicorn-A with a subkey output and the MT function.

Key schedule starts with 12 blank rounds. Then it alternates rounds that output subkeys with rounds that do not as follows: 8 blank rounds followed by 8 rounds with subkey output from M_0^{new} (Figure 5.2), then again 8 blank rounds followed by 8 rounds with the subkey production. As a result, subkeys are produced after key schedule rounds 21–28, 37–44, etc., if we enumerate them from 1.

The total of 72 32-bit subkeys are grouped into two 128-bit whitening keys and 16 groups of four subkeys for the Feistel function. The subkey order is very untrivial: for round r subkeys are $k_{r+1}, k_{r+19}, k_{r+37}, k_{r+55}$, i.e. not in the order they are produced.

CipherUnicorn-A-192 and CipherUnicorn-A-256. The key schedules for 192- and 256-bit key are very similar to the 128-bit version. The only difference is that the master key is divided into 6 (or 8) 32-bit words and undergoes more dummy rounds: 18 for CipherUnicorn-A-192 and 24 for CipherUnicorn-A-256.

Round function properties. The earlier differential analysis by Robshaw [24] fails to find a differential for the round function. Even the simplified version has the best differential of probability 2^{-14}. We have not found any improvement on this, nor we found a differential for the full round function. Thus we assumed that any related-key differential would likely to activate all the key additions.

5.1 Key schedule properties and (in)feasibility of related-key attacks

We emphasize two important properties of the key schedule of CipherUnicorn-A. On one hand, each round is a weak transformation, where the only nonlinear element is inactive as long as an 8-bit condition ($\Delta X_3 = 0$) is true. On the other hand, the number of key schedule rounds is huge, and the subkeys are used not in the order they are produced. The latter property has a tremendous effect on cryptanalysis.

We noticed earlier that a related-key differential in an internal round is highly likely to involve all the four subkeys due to the heavy round function. Secondly, to determine the subkeys in a single internal round, we have to compute as many as 55 subkeys k_i, which constitutes about 75% of the key schedule, or 112 rounds. Therefore, any related-key attack on even two rounds of CipherUnicorn-A must imply a weakness for the major part of the key schedule.

Let us investigate if the key schedule preserves any property for sufficiently many rounds. We have noticed that the nonlinear element of a round is active as long as $\Delta X_3^{new} = 0$, which is equivalent to

$$\Delta X_0 \oplus \Delta X_1 \oplus \Delta X_2 \oplus \Delta X_3 = 0. \tag{5.3}$$

Unfortunately, this property (let us denote it by $\Sigma_X = 0$) does not hold as invariant. Nevertheless, we have explored for how long a property of this kind can be sustained.

Let us make several observations. First, if $\Sigma_{M_0} = 0$, the function MT does not change Σ_{M_1} (nor, clearly, Σ_{M_2} nor Σ_{M_3}). Though Σ_{M_0} is modified, the modification is deterministic as long as ΔM_0 is known.

Secondly, we notice that as soon as $\Sigma_{M_0} \neq 0$, the difference activates the S-box and fills the other key words very quickly, one word by round. Hence we are interested in characteristics where $\Sigma_{M_0} = 0$ in all rounds, except for the last and the first ones.

Therefore, the longest characteristic is the one where $\Sigma_{M_0} = 0$ holds for the largest number of rounds. As long as it is the case, all Σ_{M_i} are updated independently. Hence the optimal case is yielded by all of them but one equalling zero. Thus we consider the case

$$\Delta M_1 = \Delta M_2 = \Delta M_3 = 0, \quad \Sigma_{M_0} = 0.$$

Since the property (5.3) is computed bitwise, without loss of generality we may assume all the bits but the least significant ones be equal to zero. Hence we consider only $2^4 = 16$ distinct differences for M_0, which we denote as $zzzz$, where z stands for 0 or 1. The 16 differences fall into six classes of equivalence:

- The difference 0000, or full zero, is a trivial case, as it renders the full key difference to zero.

- The difference 1000 is a fixed point, but Σ_{M_0} is always equal to 1. Hence it is useless for the attack.

- The difference 1100 is transformed by MT to 0100 and then back to 1100. Hence two MT functions have no active S-box, which yields 8 good rounds of the key schedule for 128-bit key, 12 rounds for 192-bit key, 16-rounds for 256-bit key.
- The difference 1001 starts a cycle of length 4:

  $$1001 \rightarrow 0111 \rightarrow 1101 \rightarrow 1011 \rightarrow 1001,$$

  where two consecutive entries 1001 and 0111 satisfy the condition $\Sigma_{M_0} = 0$. This yields 16 rounds of a deterministic differential in the 256-bit key schedule.
- The difference 1010 starts a cycle of length 4:

  $$1010 \rightarrow 0110 \rightarrow 0010 \rightarrow 1110 \rightarrow 1010,$$

  where three consecutive entries satisfy the condition $\Sigma_{M_0} = 0$. This yields 24 rounds of a deterministic differential for CipherUnicorn-A-256.
- The difference 1111 starts a cycle of length 4:

  $$1111 \rightarrow 0101 \rightarrow 0011 \rightarrow 0001 \rightarrow 1111,$$

  where four consecutive entries satisfy the condition $\Sigma_{M_0} = 0$. This yields 32 rounds of a deterministic differential for CipherUnicorn-A-256.

Therefore, the longest deterministic differential in the key schedule has 32 rounds, which is far below the required 112 rounds for the most reduced version. Thus it makes little sense to analyze the behaviour of this differential in the first and the last rounds, since even a careful analysis would add only a few rounds to the total.

We conclude that the nontrivial subkey selection of CipherUnicorn-A makes a great deal of preventing related-key attacks on this cipher. However, even if the subkey order were natural, we would still construct a property for only 5 (out of 16) rounds of the cipher. Hence we state that CipherUnicorn-A is resistant to related-key attacks with a large security margin.

5.2 Fixed point property and its implications

Robshaw noticed [24] that the key schedule transformation has a fixed point. This is natural for simple iterative transformations without round constants.

Let X_0 and Y_0 be 32-bit values such that

$$MT(X_0, Y_0) = Y_0||X_0. \tag{5.4}$$

Then the key schedule round (Equation (5.1)) has a fixed point:

$$X_0||Y_0||X_0||X_0 \longrightarrow X_0||Y_0||X_0||X_0.$$

If so, M_0^{new} is constant through the key schedule, which implies that all the subkeys are also constant equal to X_0. The property can be easily translated to 192- and 256-bit versions. Since these versions differ in the number of rounds and the number of words M_i, exactly the same property with the same values X_0, Y_0 holds for these versions too. Robshaw has found a pair X_0 and Y_0 conforming to (5.4), which yields a tuple of equivalent keys $(K_0^{128}, K_0^{192}, K_0^{256})$. Those keys yield, in turn, identical subkeys and hence identical encryption procedures.

This property, though certainly being a certificational weakness, has no apparent security implication. Nor it is relevant for related-key attacks per se. Let us check, however, if a similar property holds with respect to some relation between keys.

Differential fixed point. Let us reconsider Equation (5.4) from the differential point of view:

$$MT(\Delta X, \Delta Y) = \Delta Y || \Delta X. \tag{5.5}$$

Since MT is a nonlinear function, Equation (5.4) is probabilistic. Let us figure out if the probability is high for certain groups of keys.

Let us rewrite Equation (5.4) on the byte level:

$$\Delta Y = \Delta X \otimes 0x01010101; \tag{5.6}$$

$$\Delta X = \Delta Y \oplus S(\Delta_3 Y)||S(\Delta_3 Y)||S(\Delta_3 Y)||S(\Delta_3 Y). \tag{5.7}$$

Equation (5.6) implies that $\Delta_0 Y = \Delta_0 X$. Substituting this to (5.7), we get

$$S(\Delta_3 Y) = 0,$$

which is equivalent to $\Delta_3 Y = 0$. Therefore, we obtain that

$$\Delta X = \Delta Y.$$

However,

$$\begin{cases} \Delta Y = \Delta X \otimes 0x01010101; \\ \Delta X = \Delta Y. \end{cases} \quad \Leftrightarrow \quad \Delta X = \Delta Y = 0.$$

Therefore, there is no fixed point for differentials.

Rotational fixed point. Let us consider another type of key relation: the key K' is a rotation of K by r positions to the left:

$$K' = \overrightarrow{K}.$$

Let us figure out if the function MT preserves the rotational property for certain groups of keys.

Hence we reconsider Equation (5.5):

$$MT(\overrightarrow{X}, \overrightarrow{Y}) = \overrightarrow{Y} || \overrightarrow{X}. \tag{5.8}$$

However, this simply means that $\overrightarrow{X}$ and $\overrightarrow{Y}$ are yet another solution to Equation (5.4). We are not aware of any other solution but presented by Robshaw, whereas his analysis implicitly states that there is only one solution. Hence we conclude that there is no rotational fixed point in CipherUnicorn-A.

We conclude that the cipher CipherUnicorn-A is secure against related-key attacks thanks to its nontrivial key schedule. We did not consider 1-round attack, but demonstrated that any attacks on two or more rounds would require a property holding through many rounds of the key schedule. No property of differential or rotational nature that we considered has this effect.

Chapter 6

Conclusion

We have made a comprehensive investigation of resistance of Hierocrypt-3, CipherUnicorn-A, and AES to various kinds of related-key attacks. We have checked regular differential related-key attacks, boomerang attacks, rotational properties, and properties mentioned in previous research. An overview of related-key attacks on these designs is given in Table 6.

The cipher AES has been subject for numerous related-key attacks in various settings. Related-key attacks exist for both reduced and original versions of AES. Related-key distinguishers have practical complexity, the key recovery attacks have complexity far below the exhaustive key search. However, the key recovery attack use a relation between subkeys, which does not translate to simple relation in master keys, which render the attack model less practical.

The cipher Hierocrypt-3 has no related-key weakness, but it is possible to mount a related-key attack on a greatly reduced version. Nonlinearity of the key schedule and a nontrivial method of selecting subkeys makes longer attacks highly unlikely.

The cipher CipherUnicorn-A appears to be the most secure to related-key attacks. Due to its complicated key schedule and heavy round functions it is virtually invulnerable to related-key attacks.

We conclude that whenever the security to related-key attacks is required, e.g., when using a specific message authentication code, AES is not the best option, while Hierocrypt-3 and CipherUnicorn-A are a safe choice.

Type of attack	Rounds	Key relation	Complexity
CipherUnicorn-A(all keys, 16 rounds)			
N/A	N/A	N/A	N/A
Hierocrypt-3 (128-bit key)			
N/A	N/A	N/A	N/A
Hierocrypt-3 (192-bit key)			
Boomerang	1	Related subkey	2^{48} (partial) 2^{144} (full)
Hierocrypt-3 (256-bit key)			
Differential	1	Related subkeys	2^{48} (partial) 2^{208} (full)
Boomerang	2	Related subkeys	2^{48} (partial) 2^{208} (full)
AES-128, 10 rounds			
Boomerang	7	Related subkeys	2^{97}
AES-192, 12 rounds			
Differential	11	Related subkeys	2^{186}
Boomerang	12	Related subkeys	2^{169}
AES-256, 14 rounds			
Distinguisher	14	Chosen keys	2^{67} and 2^{37}
Boomerang	13	Related subkeys	2^{76}
Boomerang	14	Related subkeys	2^{99}
Differential	14	Related subkeys	2^{131}

Table 6.1: Related-key key recovery attacks on CipherUnicorn-A, Hierocrypt-3, and AES.

Bibliography

[1] Mihir Bellare, Ran Canetti, and Hugo Krawczyk. Keying hash functions for message authentication. In *CRYPTO'96*, volume 1109 of *Lecture Notes in Computer Science*, pages 1–15. Springer, 1996.

[2] Mihir Bellare and Tadayoshi Kohno. A theoretical treatment of related-key attacks: RKA-PRPs, RKA-PRFs, and applications. In *EUROCRYPT'03*, volume 2656 of *Lecture Notes in Computer Science*, pages 491–506. Springer, 2003.

[3] Eli Biham. New types of cryptanalytic attacks using related keys. *J. Cryptology*, 7(4):229–246, 1994.

[4] Eli Biham, Orr Dunkelman, and Nathan Keller. The rectangle attack – rectangling the Serpent. In *EUROCRYPT'01*, volume 2045 of *Lecture Notes in Computer Science*, pages 340–357. Springer, 2001.

[5] Eli Biham, Orr Dunkelman, and Nathan Keller. Related-key boomerang and rectangle attacks. In *EUROCRYPT'05*, volume 3494 of *Lecture Notes in Computer Science*, pages 507–525. Springer, 2005.

[6] Eli Biham and Adi Shamir. Differential fault analysis of secret key cryptosystems. In *CRYPTO'97*, volume 1294 of *Lecture Notes in Computer Science*, pages 513–525. Springer, 1997.

[7] Alex Biryukov, Orr Dunkelman, Nathan Keller, Dmitry Khovratovich, and Adi Shamir. Key recovery attacks of practical complexity on AES-256 variants with up to 10 rounds. In *EUROCRYPT'10*, volume 6110 of *Lecture Notes in Computer Science*, pages 299–319. Springer, 2010.

[8] Alex Biryukov and Dmitry Khovratovich. Related-key cryptanalysis of the full AES-192 and AES-256. In *ASIACRYPT'09*, volume 5912 of *Lecture Notes in Computer Science*, pages 1–18. Springer, 2009.

[9] Alex Biryukov and Dmitry Khovratovich. Feasible attack on the 13-round AES-256. *IACR Cryptology ePrint Archive*, 2010:257, 2010.

[10] Alex Biryukov, Dmitry Khovratovich, and Ivica Nikolić. Distinguisher and related-key attack on the full AES-256. In *CRYPTO'09*, volume 5677 of *Lecture Notes in Computer Science*, pages 231–249. Springer, 2009.

[11] Alex Biryukov, Sourav Mukhopadhyay, and Palash Sarkar. Improved time-memory tradeoffs with multiple data. In *Selected Areas in Cryptography'05*, volume 3897 of *Lecture Notes in Computer Science*, pages 110–127. Springer, 2005.

[12] Alex Biryukov and Ivica Nikolic. Automatic search for related-key differential characteristics in byte-oriented block ciphers: Application to AES, Camellia, Khazad and others. In *EUROCRYPT'10*, volume 6110 of *Lecture Notes in Computer Science*, pages 322–344. Springer, 2010.

[13] Andrey Bogdanov, Dmitry Khovratovich, and Christian Rechberger. Biclique cryptanalysis of the full AES. In *ASIACRYPT'11*, volume 7073 of *Lecture Notes in Computer Science*, pages 344–371. Springer, 2011.

[14] Florent Chabaud and Antoine Joux. Differential collisions in SHA-0. In *CRYPTO'98*, volume 1462 of *Lecture Notes in Computer Science*. Springer, 1998.

[15] Éliane Jaulmes, Antoine Joux, and Frédéric Valette. On the security of randomized CBC-MAC beyond the birthday paradox limit: A new construction. In *FSE'02*, volume 2365 of *Lecture Notes in Computer Science*. Springer, 2002.

[16] Jonathan Katz and Yehuda Lindell. *Introduction to Modern Cryptography*. Chapman and Hall/CRC Press, 2007.

[17] John Kelsey, Tadayoshi Kohno, and Bruce Schneier. Amplified boomerang attacks against reduced-round MARS and Serpent. In *FSE'00*, volume 1978 of *Lecture Notes in Computer Science*, pages 75–93. Springer, 2000.

[18] John Kelsey, Bruce Schneier, and David Wagner. Related-key cryptanalysis of 3-WAY, Biham-DES, CAST, DES-X, NewDES, RC2, and TEA. In *ICICS'97*, volume 1334 of *Lecture Notes in Computer Science*, pages 233–246. Springer, 1997.

[19] Jongsung Kim, Seokhie Hong, and Bart Preneel. Related-key rectangle attacks on reduced AES-192 and AES-256. In *FSE'07*, volume 4593 of *Lecture Notes in Computer Science*, pages 225–241. Springer, 2007.

[20] Florian Mendel and Tomislav Nad. Boomerang distinguisher for the SIMD-512 compression function. In *INDOCRYPT'11*, volume 7107 of *Lecture Notes in Computer Science*. Springer, 2011.

[21] National Institute of Standards and Technology (NIST), available at `http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf`. *FIPS-197: Advanced Encryption Standard*, November 2001.

[22] NEC Corporation. *Cryptographic Techniques Specifications: CipherUnicorn-A*.

[23] NEC Corporation. *Self Evaluation Report: CipherUnicorn-A*.

[24] Matt Robshaw. A cryptographic review of Cipherunicorn-A. Technical report, 2001.

[25] Erik Tews, Ralf-Philipp Weinmann, and Andrei Pyshkin. Breaking 104 bit WEP in less than 60 seconds. In *WISA'07*, volume 4867 of *Lecture Notes in Computer Science*, pages 188–202. Springer, 2007.

[26] Toshiba Corporation. *Self Evaluation: Hierocrypt-3*.

[27] Toshiba Corporation. *Specification on a Block Cipher : Hierocrypt3*, May 2002.

[28] David Wagner. The boomerang attack. In *FSE'99*, volume 1636 of *Lecture Notes in Computer Science*, pages 156–170. Springer, 1999.