Transforming chosen IV attack into a key differential attack: how to break TRIVIUM and similar designs

Enes Pasalic
IMFM Ljubljana & University of Primorska, Koper
Slovenia
enespasalic@yahoo.se

Abstract

In this paper the applicability of differential cryptanalytic tool to stream ciphers is elaborated using the algebraic representation similar to early Shannon's postulates regarding the concept of confusion. In 2007, Biham and Dunkelman [3] have formally introduced the concept of differential cryptanalysis in stream ciphers by addressing the three different scenarios of interest. Here we mainly consider the first scenario where the key difference and/or IV difference influence the internal state of the cipher $(\Delta key, \Delta IV) \rightarrow \Delta S$. We then show that under certain circumstances a chosen IV attack may be transformed in the key chosen attack. Further we show that if there exist linear relations between the key and IV bits, then using the differentiation of internal state variables we are able to eliminate the presence of corresponding key variables. The method leads to an attack whose complexity may be well beyond the exhaustive search whenever the cipher admits exact algebraic description of internal state variables and the keystream computation is not complex. A successful application is especially noted in the context of stream ciphers whose keystream bits evolve relatively slow as a function of secret state bits. For instance, the attack is successfully applied to a full round TRIVIUM with the worst case complexity being 2^{68}, but combined with the key guessing it is significantly lower, about 2^{60}. The attack is also applicable to stream cipher DECIM-128 but without decimation mechanism.

1 Introduction

In 1990, Biham and Shamir [5] have proposed a new cryptanalysts' tool called differential cryptanalysis. This technique, along with linear cryptanalysis of Matsui [12], has been considered as one of the most powerful attacks on block ciphers to date. However, the same ideas have not been applied in the same extent in cryptanalysis of stream ciphers until recently. The use of concepts similar to differential cryptanalysis for block ciphers was frequently employed frequently during the eSTREAM project [9]. These attacks are commonly referred as chosen IV or key related attacks where the attacker is supposed to freely choose IV values and/or different keys in order to extract certain information about the secret state and/or key bits.

This approach has been successfully utilized in the cryptanalysis of the stream cipher LILI-128 [1], where different IV's used with a fixed secret key revealed a partial information about the state bits. In other direction the eSTREAM proposal Py [4] (and its tweaks PyPy and PyPy6) was successfully cryptanalyzed using the differential cryptanalysis [11], more precisely for any given key the subset of IV's was identified so that there was two IV's that would generate the identical initial state (and therefore identical keystream). The ideas of differential cryptanalysis have also been successfully applied in so-called fault analysis [6]. In such an attack the adversary introduce errors during the computation which leads to errors in the output. The difference between an unfaulty and a faulty computation would then reveal certain information about the computation itself.

This paper aims to give a thorough analysis of chosen IV and key related attacks in the view of Shannon's algebraic representation of encryption schemes. In difference to a general framework studied in [3], where the effect of IV/key difference is analyzed through the differential characteristics at the state variables ($(\Delta key, \Delta IV) \rightarrow \Delta S$), we take another approach and consider the differentiation of the key variables in the algebraic expressions of the initial state variables. Though the related key attack (differential key attack) is a more unrealistic sceanrio than a chosen IV attack, we show that this scenario is far more dangerous as we can completely eliminate the presence of certain key variables under the assumption that the IV is kept fixed. To satisfy the condition that IV is kept fixed we essentially consider a chosen IV attack and a fixed key (which is much more realistic scenario), and then in cases that a certain subset of IV and key bits are related in a linear manner after the initial loading, say $IV_i \oplus k_j$, the IV difference is transformed into the key difference.

Once we have successfully eliminated the presence of certain key variables, through the differentiation of the state variables, a simple key guessing attack may be combined to further reduce the complexity of the state variables. Obviously by guessing many key variables we can make the internal state bit equations to be arbitrary simple functions in remaining key variables. Then, if the keystream generation does not combine these state variables in a complicated manner (or at least a part of the keystream does not depend on the state variables in a complex way) we can efficiently solve the system for the remaining key bits. As an application of this approach we consider TRIVIUM [8], a hardware oriented stream cipher that reached the final third phase of the eSTREAM project [9], and propose an efficient attacking method for a full round TRIVIUM. The worst case complexity of the attack is 2^{68}, and is well beyond the exhaustive key search for the cipher of 80-bit security. This complexity is further reduced by combining a key guessing attack, and in the worst case complexity of the attack is 2^{63}. Another suitable design, due to its specific key and IV loading, is a stream cipher DECIM-128 [2]. But in this case the attack is only applicable to a modified cipher that omits the output decimation.

The paper is organized as follows. In Section 2 a convenient algebraic representation of the internal state variables is introduced. The main differential attack scenarios are discussed in Section 3. It is then shown that the key differentiation in general eliminates the presence of the key variables the differentiation is performed on. Here we also deduce the conditions when an IV chosen attack can be transformed into a key differential attack. In Section 4 we apply our method to certain stream cipher schemes. In particular, we show that a full round TRIVIUM is susceptible to a key differentiation attack with complexity well beyond the exhaustive search. Finally, Section 5 concludes the paper.

2 Algebraic representation of the initialization process

For the rest of this paper we only consider synchronous stream cipher as a dominant part of the family, though the similar analysis as given here can be easily extended to asynchronous stream ciphers. Technically, given a stream cipher, the cipher is fully specified by the key/IV setup algorithm (KSA) and encryption algorithm whose input parameters includes the key, IV, and state size. The key setup and IV setup are commonly performed in a single phase so that the secret state of the cipher is derived by processing key and IV bits in an iterative manner. The result of this phase is the so-called initial state $S = (s_0, s_1, \ldots, s_{L-1}) = F(IV_1, \ldots, IV_\kappa, k_1, \ldots, k_\kappa)$, for simplicity assuming the same length of the key and IV, and in addition denoting by $L \geq 2\kappa$ the state size; the state size being at least twice the key length to withstand time-memory-data trade-off attacks, see for instance [7, 10]. Now given a fixed key K and the publicly known $\mathrm{IV}^{(i)}$ the system of equations may be written as,

$$
\begin{aligned}
s_0^{(i)} &= f_0(IV_1^{(i)}, \ldots, IV_\kappa^{(i)}, k_1, \ldots, k_\kappa) = g_0(IV_1^{(i)}, \ldots, IV_\kappa^{(i)}, k_1, \ldots, k_\kappa) + h_0(k_1, \ldots, k_\kappa) \\
&\vdots \\
s_{L-1}^{(i)} &= f_{L-1}(IV_1^{(i)}, \ldots, IV_\kappa^{(i)}, k_1, \ldots, k_\kappa) = g_{L-1}(IV_1^{(i)}, \ldots, IV_\kappa^{(i)}, k_1, \ldots, k_\kappa) + h_{L-1}(k_1, \ldots, k_\kappa)
\end{aligned}
$$

The reason that we collect the terms that only contain the key bits in function g is that the differential of the form $s_r^{(i)} \oplus s_r^{(j)}$, $0 \leq r \leq L-1$, will not contain the key bit terms for arbitrary $\mathrm{IV}^{(i)}$ and $\mathrm{IV}^{(j)}$. This of course may be generalized for any set of IV's of even weight.

Following the early ideas of Shannon, each s_i should depend in a complicated manner on the key bits and involve many (preferably all) of them. This is also true for the IV bits, as otherwise the differential $s_r^{(i)} \oplus s_r^{(j)}$ may result in simple expressions that relate key and IV bits to the secret state bits. One should notice that there is a certain freedom in selecting these relations, the freedom that may additionally reduce the complexity of the system.

Let $\mathcal{I}, \mathcal{J} = \{1, 2, \ldots, \kappa\}$ denote the set of indices for the IV vector respectively the key bits. Then a convenient algebraic representation, used in [13], is to represent the secret bits as a collection of terms as follows,

$$
\begin{aligned}
s_r^{(i)} &= \bigoplus_{I=\emptyset}^{\{1,2,\ldots,\kappa\}} a_{r,I} IV_I \left[\oplus_{J=\emptyset}^{\mathcal{J}} b_{r,J} K_J\right] = \\
&= \underbrace{\bigoplus_{I \subset \mathcal{I}; I \neq \emptyset} a_{r,I} IV_I \left[\oplus_{J=\emptyset}^{\mathcal{J}} b_{r,J} K_J\right]}_{g_r^{(i)}} \oplus \underbrace{\left[\oplus_{J=\emptyset}^{\mathcal{J}} b_{r,J} K_J\right]}_{h_r^{(i)}}, \quad \text{for } 0 \leq r \leq L-1,
\end{aligned} \tag{1}
$$

where $a_I^{(i)}, b_J^{(i)}$ are algorithm specific binary coefficients, and for a given subset $I = \{i_1, \ldots, i_l\} \subset \mathcal{I}$ the term $IV_I = IV_{i_1} IV_{i_2} \cdots IV_{i_l}$ and similarly $K_J = k_{j_1} k_{j_2} \cdots k_{j_p}$ for some $J = \{j_1, \ldots, j_p\} \subset J$. We note that treating both the key and IV as variables the coefficients $a_{r,I}, b_{r,J}$ do not depend on the specific choice of IV's. The above expressions are valid for any stream cipher, the specific details being the size of the key, IV, state, and the particular KSA used.

If the secret state bits s_r had been generated randomly as the functions of the key and IV bits, then one would expect that the algebraic normal forms of each s_r contains approximately $2^{2\kappa-1}$ terms. In addition the high degree terms of order $2\kappa - 1$ should be present in such expressions as well, thus closely following the Shannon's idea of making complicated relations among the state bits and secret key bits. Given an algebraic description of the state of the cipher after running the KSA (this is commonly infeasible as the number of terms is excessively large), the attacker can try to reduce the complexities of these relations by either applying the chosen IV attack, related key attack or alternatively to combine these two.

3 Transforming chosen IV into key related attacks

In this section we consider the main attack scenarios in the realm of the differential cryptanalysis. Though similar in their essence, the cryptanalytic aspects significantly differ depending in which scenario the attack is performed.

3.1 Chosen IV attacks

A chosen IV attack is a very realistic scenario in which the attacker chooses certain subset of all possible IV's in order to combine the relations of resulting state bits and hopefully derive the simple relations between the state bits and secret key bits. Assume that for a fixed key, the attacker is able to trace a certain subset of IV'bits say I for which relatively simple relations may be derived. The easiest way to understand this approach is to specify IV values to be zero in all positions but those that correspond to I. Thus, given the $\mathrm{IV}^{(i)}$ and a vector $I = \{i_1, i_2, \ldots, i_p\}$ for which $\mathrm{IV}_l = 0, l \notin I$ the secret state bits could be expressed as,

$$\begin{aligned} s_r^{(i)} &= \bigoplus_{\xi \subset I} a_{r,I} IV_\xi \left[\oplus_{J=\emptyset}^{\mathcal{J}} b_{r,\xi,J} K_J\right] \oplus \left[\oplus_{J=\emptyset}^{\mathcal{J}} b_{r,I,J} K_J\right] = IV_{i_1} \left[\oplus_{J=\emptyset}^{\mathcal{J}} b_{r,i_1,J} K_J\right] \oplus \\ &\oplus \quad IV_{i_2} \left[\oplus_{J=\emptyset}^{\mathcal{J}} b_{r,i_2,J} K_J\right] \oplus \cdots \oplus IV_{i_1} IV_{i_2} \cdots IV_{i_p} [\oplus_{J=\emptyset}^{\mathcal{J}} b_{r,I,J} K_J] \oplus [\oplus_{J=\emptyset}^{\mathcal{J}} b_{r,I,J} K_J] \end{aligned}$$

Then keeping the key fixed we can further manipulate these expressions by specifying different IV's from this subset of indices. For instance, taking the $\mathrm{IV}^{(1)}$ and $\mathrm{IV}^{(2)}$ such that $\mathrm{IV}_{i_1}^{(1)} \neq \mathrm{IV}_{i_1}^{(2)}$ and $\mathrm{IV}_{i}^{(1)} = \mathrm{IV}_{i}^{(1)}$ for all $i \neq i_1$ would result in,

$$s_r^{(1)} \oplus s_r^{(2)} = \oplus_{J=\emptyset}^{\mathcal{J}} b_{r,i_1,J} K_J. \tag{2}$$

This is exactly the idea that has been used in [13], where an semi-exhaustive search was performed over the suitable set of indices I of weight 6 to derive linear expressions relating keystream bits of reduced KSA Trivium and the secret key bits. However, the authors derive such relations by setting the remaining IV bits (74 bits in the case of Trivium) to zero which makes the attack slightly unrealistic as the probability of selecting such IV's is negligibly small. The above expressions suggests that the type of relations similar to (2) may be derived for any fixed IV values making such an approach more realistic.

Thus, a pseudo algorithm for finding simple relations after the KSA procedure may be formulated as follows:

1. Locate some subset of indices I for which the particular KSA does not generate sufficiently complicated relations in secret key bits.

2. Based on this observation for any fixed combination of remaining IV bits not in I find relations of the form $s_r^{(1)} \oplus s_r^{(2)} \oplus \cdots s_r^{(m)}$ by varying the IV vectors $\mathrm{IV}^{(1)}, \mathrm{IV}^{(2)}, \ldots, \mathrm{IV}^{(m)}$ over the subset I, for all $r \in [0, L-1]$.

3. Once such relations are found these are used in the keystream generation process to derive low degree relations relating the keystream bits and the secret key bits (through the secret initial state bits).

3.2 Related key attacks

In this scenario the attacker can observe the operation of a cipher under several different keys whose values are initially unknown, but where some mathematical relationship connecting the keys is known to the attacker. This may be regarded as quite impractical scenario since a secure key management and derivation usually should not allow to locate the part of the bits which remain unchanged after the key derivation. Nevertheless, many KSA schemes allow a simple mixture of the IV and secret key bits and therefore there is a simple and elegant way to transform the chosen IV attack into the key related attack.

To illustrate this approach let the key K be fixed and for simplicity consider two different IV vectors $\mathrm{IV}^{(1)}$ and $\mathrm{IV}^{(2)}$ that differ in certain positions specified by some subset I, that is $\mathrm{IV}_i^{(1)} \neq \mathrm{IV}_i^{(2)}$ for all $i \in I$. Furthermore, we assume that the KSA procedure is such that a part of the cipher state of cardinality p is initialized by a simple XOR-ing of the IV and key bits, and then the KSA is run to generate the final initial state of the cipher. The initial loading of the IV and key bits is referred to as pre-setup phase, resulting in the pre-setup state $S^{ps} = (s_0^{ps}, s_1^{ps}, \ldots, s_{L-1}^{ps})$, where the XOR-ing is applied as follows,

$$s_{j_l}^{ps(1)} = \mathrm{IV}_i^{(1)} \oplus k_m; \;\; s_{j_l}^{ps(2)} = \mathrm{IV}_i^{(2)} \oplus k_m, \quad l \in \{1, 2, \ldots, p\}$$

for a suitable choice of indices sets say I, J, M that entirely depends on the specific loading mechanism. Neglecting the due details it is easy to realize that instead of considering the difference in IVs we may consider the relation so that $\mathrm{IV}^{(1)} = \mathrm{IV}^{(2)}$ but the key is no longer fixed and consequently one may define K',

$$K' = \begin{cases} k'_m = k_m; & m \notin M, i \notin I \\ k'_m = 1 \oplus k_m; & m \in M, i \in I \end{cases}$$

For simplicity, an IV differential in a single position i, i.e. $IV_i^{(2)} = IV_i^{(1)} \oplus 1$, would result in the key difference through,

$$s_r^{ps(1)} \;=\; IV_i^{(1)} \oplus k_m; \;\; s_r^{ps(2)} = IV_i^{(2)} \oplus k_m = IV_i^{(1)} + (k_m \oplus 1) = IV_i^{(1)} + k'_m.$$

Thus, we may analyze the effect of a single bit difference in IVs that is transformed in the key difference. As before, at the end of the initialization, this difference would give,

$$\begin{aligned}\Delta^{k_m} s_r \stackrel{\text{def}}{=} s_r^{k_m} \oplus s_r^{k'_m} &= \bigoplus_{I\subset\mathcal{I};I\neq\emptyset} a_{r,I} IV_I \left[\oplus_{J=\emptyset}^{\mathcal{J}} b_{r,I,J} K_J\right] \bigoplus_{J=\emptyset}^{\mathcal{J}} b_{r,J} K_J \\ &\oplus \bigoplus_{I\subset\mathcal{I};I\neq\emptyset} a_{r,I} IV_I \left[\oplus_{J=\emptyset}^{\mathcal{J}} b_{r,I,J} K'_J\right] \bigoplus_{J=\emptyset}^{\mathcal{J}} b_{r,J} K'_J, \qquad (3)\end{aligned}$$

where $k_j = k'_j$ for all j except for some $j = m$. This implies that in the above sum the most of the terms are canceled leaving only those K_J's for which $m \in J$. Then for any $I \subset \mathcal{I}$ and $J \subset \mathcal{J}$ such that $m \in J$ we can compute

$$\begin{aligned}&IV_I \left[\oplus_{J=\emptyset}^{\mathcal{J}} b_{r,J} K_J\right] \oplus IV_I \left[\oplus_{J=\emptyset}^{\mathcal{J}} b_{r,J} K'_J\right] = \\ &IV_I[k_m g(k_1,\ldots,k_{m-1},k_{m+1},\ldots,k_\kappa) \oplus h(k_1,\ldots,k_{m-1},k_{m+1},\ldots,k_\kappa)] \oplus \\ &IV_I[k'_{i_1} g(k_1,\ldots,k_{m-1},k_{m+1},\ldots,k_\kappa) \oplus h(k_1,\ldots,k_{m-1},k_{m+1},\ldots,k_\kappa)] = \\ &IV_I g(k_1,\ldots,k_{m-1},k_{m+1},\ldots,k_\kappa).\end{aligned}$$

It is straightforward to verify that the same arguments apply to the differential

$$\bigoplus_{J=\emptyset}^{\mathcal{J}} b_{r,J} K_J \oplus \bigoplus_{J=\emptyset}^{\mathcal{J}} b_{r,J} K'_J = h(k_1,\ldots,k_{m-1},k_{m+1},\ldots,k_\kappa),$$

thus, in general, lower degree expression (the decrease of the degree is at least one) in the secret state bits are derived for $\Delta^{k_m} s_r$.

Example 1 *Assume that the KSA for a given cipher enables a single IV difference to be translated into the key difference in* k_1*. Let the secret state bit* s_r *be evaluated as* $s_r = k_2k_3k_4k_5 \oplus k_1k_3k_4 \oplus k_2k_5k_6 \oplus k_2k_6 \oplus k_3 \oplus k_1$*; after the IV value has been specified. Then differentiating the first key bit* k_1 *we would compute,*

$$\Delta^{k_1} s_r = s_r^{(k_1)} \oplus s_r^{(k_1+1)} = k_3k_4 \oplus 1.$$

In this case we have decreased the degree of s_r *by two in a single step due to the cancellation of the highest degree term* $k_2k_3k_4k_5$ *for which the differentiation does not apply as it does not contain the variable* k_1*.*

A natural question that arise now is how to generalize this approach to include differentiation of as many key bits as possible. At the first sight this is not possible due to the following arguments. Let us define the key differential at positions $k_{j_1},\ldots,k_{j_m}$ with respect to some initial state bit s_r as,

$$\Delta^{k_{j_1},\ldots,k_{j_m}} s_r = s_r^{(k_{j_1},\ldots,k_{j_m})} \oplus s_r^{(k_{j_1}+1,\ldots,k_{j_m}+1)},$$

where we simply evaluate the differential by specifying the key difference simultaneously at given positions. Without loss of generality, to skip a complicated notation, we assume that the secret state bit s_r contains a single term i.e. $s_r = IV_I k_{j_1} k_{j_2} \cdots k_{j_{m-1}} k_{j_m} k_{j_{m+1}} \cdots k_{j_{m+v}}$. Then it is easy to compute,

$$\begin{aligned}
\Delta^{k_{j_1},\ldots,k_{j_m}} s_r &= IV_I k_{j_1} k_{j_2} \cdots k_{j_{m-1}} k_{j_m} k_{j_{m+1}} \cdots k_{j_{m+v}} \oplus \\
& IV_I (k_{j_1} \oplus 1)(k_{j_2} \oplus 1) \cdots (k_{j_{m-1}} \oplus 1)(k_{j_m} \oplus 1) k_{j_{m+1}} \cdots k_{j_{m+v}} = \\
& IV_I k_{j_{m+1}} \cdots k_{j_{m+v}} [1 + k_{j_1} \oplus \ldots + k_{j_m} \oplus k_{j_1} k_{j_2} \oplus \ldots \oplus k_{j_{m-1}} k_{j_m} \oplus \\
& \ldots \oplus k_{j_2} \cdots k_{j_{m-1}} k_{j_m}],
\end{aligned}$$

the expression containing all the monomials of degree less than m in the key bits $k_{j_1}, \ldots, k_{j_m}$. Consequently, simultaneous differentiation at m positions does only guarantee the degree decrease of order one but not m as desired. Nevertheless, one can get rid of the variables $k_{j_1}, k_{j_2} \cdots, k_{j_m}$ by simply considering the linear combinations of the weighted differentials of the form,

$$\bigoplus_{J \subset \{k_{j_1}, \cdots, k_{j_m}\}; J \neq \emptyset} \Delta^J s_r$$

For instance, to make the key bits k_{j_1} and k_{j_2} vanish from s_r we would compute the following differentials : $\Delta^{k_{j_1}} s_r, \Delta^{k_{j_2}} s_r$, and $\Delta^{k_{j_1},k_{j_2}} s_r$ and add them together. To confirm the validity of our claim for the above specified s_r we would compute,

$$\begin{aligned}
\bigoplus_{J \subset \{k_{j_1}, k_{j_2}\}} \Delta^J s_r &= \Delta^{k_{j_1}} s_r \oplus \Delta^{k_{j_2}} s_r \oplus \Delta^{k_{j_1},k_{j_2}} s_r = \\
&= IV_I k_{j_3} \cdots k_{j_m} k_{j_{m+1}} \cdots k_{j_{m+v}} [k_{j_2} \oplus k_{j_1} \oplus k_{j_2} \oplus k_{j_1} \oplus 1] = \\
&= IV_I k_{j_3} \cdots k_{j_m} k_{j_{m+1}} \cdots k_{j_{m+v}}
\end{aligned}$$

Of course the result is easily generalized for arbitrary number of terms and the degree order. For instance, if there is another term in the s_r, say $\alpha = IV_{I'} k_{j_1} k_{j_{m+u}} \cdots k_{j_{m+v}}$, thus containing only the variable k_1 (it is covered by the variables k_1 and k_2) then computing $\bigoplus_{J \subset \{k_{j_1}, k_{j_2}\}} \Delta^J \alpha$ would give,

$$\begin{aligned}
\bigoplus_{J \subset \{k_{j_1}, k_{j_2}\}} \Delta^J \alpha &= \Delta^{k_{j_1}} \alpha \oplus \Delta^{k_{j_2}} \alpha \oplus \Delta^{k_{j_1},k_{j_2}} \alpha = \\
&= IV_{I'} k_{j_{m+u}} \cdots k_{j_{m+v}} \oplus 0 \oplus IV_{I'} k_{j_{m+u}} \cdots k_{j_{m+v}} = 0,
\end{aligned}$$

and there is no influnce of the terms that are covered by the chosen subset of key variables. Thus, the only terms that remain after the differentiation are the complex expressions in the remaining key variables, the terms of the form $IV_I f_I(k_{j_{m+1}}, \ldots, k_{j_\kappa})$.

It should be emphasized that in difference to differential IV attacks the differential key attacks are far more dangerous as the differentiation in the latter case affects the key variables, whereas in case of IV differentials the decrease of the degree with respect to IV variables has no effect after the IV has been set to a given value. On the other hand, the condition that the IV and key variables are first XOR-ed and then processed via the KSA might be rather restrictive.

3.3 Combining a related key and key guessing attack

In the previous section we have shown that there is a possibility to differentiate the secret state equations at those key positions for which the loading procedure XORs the IV's and key bits. In this way we could eliminate the presence of the corresponding key variables making the equation(s) depend on less number of key variables, thereby reducing its complexity. An interesting approach that can be taken now is simply to guess a subset of remaining key variables leading to a further degree reduction. Under a reasonable assumption that the secret state bits (after the initialization) do not differ significantly in terms of degree and complexity of the expressions we propose the following pseudo algorithm for attacking the ciphers whose loading of a certain portion of the key and IV bits is done by simple XOR-ing.

1. Find the key positions k_j (subset of indices) for which the given KSA injects the key and IV bits via $IV_i \oplus k_j$. Denote this subset by $J = \{k_{j_1}, k_{j_2}, \dots, k_{j_m}\}$.

2. After loading the all key and IV bits, thus reaching the state $S^{ps} = (s_0^{ps}, s_1^{ps}, \dots, s_{L-1}^{ps})$, run the KSA to get the finial initial state of the cipher $S = (s_0, s_1, \dots, s_{L-1})$.

3. Depending on the complexity of the expressions $s_r = f_r(IV_1, \dots, IV_\kappa, k_1, \dots, k_\kappa)$ select a suitable subset of IV's for which these equations are of relatively low complexity.

4. Differentiate the key variables with respect to the subset J so that $\Delta^J s_r$ do not depend on the key variables from the indices set J for any $0 \leq r \leq L-1$.

5. Guess a suitable subset of key bits so that the expressions $\Delta^J s_r$ become of low degree (preferably linear).

6. Run the cipher in the output mode, i.e. producing keystream z_t, and analyze the relations $z_t = G(s_0, s_1, \dots, s_{L-1}) = H(k_{j_{m+1}}, k_{j_{m+2}}, \dots, k_{j_\kappa})$. Observe the suitable part of output sequence for which the function G does not introduce complicated (high degree) equations in secret state bits. Solve the system of equations and check its consistence (if necessary guess a few more key bits and solve the system). If the solution is inconsistent guess another key value in the previous step and repeat.

Note that the complexity of the above algorithm mainly depends on the specific KSA and the way the cipher generates the keystream (complicated relations on initial state bits or not). However, since the key bits $k_{j_1}, k_{j_2}, \dots, k_{j_m}$ have vanished from these equations the worst case complexity is estimated as,

$$C^{w.c.} = 2^{\kappa-m},$$

which correspond to guessing all the remaining bits. Once these bits are correctly set it remains to assign the unknown key bits $k_{j_1}, k_{j_2}, \dots, k_{j_m}$ which can be easily achieved by plugging in these bits in the equations and checking the consistency with keystream bits z_t. Thus, the total worst case complexity is estimated as,

$$C^{w.c.} = 2^{\kappa-m} + 2^m,$$

thus in the case $m \ll \kappa$ we approximately have $C^{w.c.} = 2^{\kappa - m}$. We illustrate the whole idea with an example, considering a hypothetical toy cipher satisfying all necessary conditions for the attack to work.

Example 2 *Let us consider a toy cipher that use the key $K = (k_1, k_2, k_3, k_4) = (0, 1, 1, 1)$, and assume that IV difference in two bits is translated in the key difference at positions k_1 and k_2. For some specified IV let the first three (secret) initial state bits be given as,*

$$\begin{aligned} s_0 &= k_1 k_2 k_3 + k_3 k_4 + k_3 + k_4 = 0, \\ s_1 &= k_1 k_3 k_4 + k_2 + k_4 = 0, \\ s_2 &= k_1 k_2 k_4 + k_2 k_3 k_4 + k_1 + k_2 + k_3 = 1. \end{aligned}$$

Obviously we cannot get rid of the degree 3 terms by considering linear combinations of the above equations. Let us compute the differentials for the 2 keystream bits $z_1 = s_0 + s_1$ and $z_2 = s_0 + s_2$,

$\mathbf{z}_1$	$\mathbf{z}_2$
$\Delta^{k_1} z_1 = \Delta^{k_1}(s_0 + s_1) = k_2 k_3$	$\Delta^{k_1} z_2 = \Delta^{k_1}(s_0 + s_2) = k_2 k_3 + k_2 k_4 + 1$
$\Delta^{k_2} z_1 = \Delta^{k_2}(s_0 + s_1) = k_1 k_3 + 1$	$\Delta^{k_2} z_2 = k_1 k_3 + k_3 k_4 + k_1 k_4 + 1$
$\Delta^{k_1,k_2} z_1 = k_3(1 + k_1 + k_2) + 1$	$\Delta^{k_1,k_2} z_2 = k_3(1 + k_1 + k_2) + k_3 k_4 + k_4(1 + k_1 + k_2)$
$\bigoplus_{J \subset \{k_1,k_2\}} \Delta^J z_1 = k_3$	$\bigoplus_{J \subset \{k_1,k_2\}} \Delta^J z_2 = k_3 + k_4$

The attacker observes outputs z_1 and z_2 under different IV's (translated into the key difference), thus he is able to compute the above differentials. For instance, to compute $\Delta^{k_1} z_1$ he would simply observe the first keystream bit for a specified IV and fixed key K and then the same keystream bit for a single change of one bit in the IV corresponding to the key position k_1. For the given values of K we may check that $\bigoplus_{J \subset \{k_1,k_2\}} \Delta^J z_1 = k_3 = 1$ and $\bigoplus_{J \subset \{k_1,k_2\}} \Delta^J z_2 = k_3 + k_4 = 0$. Hence, the correct values of the key bits $k_3 = 1$ and $k_4 = 1$ are found.

When applicable, the key related attack as discussed above is a powerful cryptanalytic tool. The complexity is always beyond the exhaustive key search if the necessary assumptions are valid. However, one needs to be careful when deriving the key differentials through the IV change. The reason is that the KSA of a given cipher starts to process the pre-setup state $S^{ps} = (s_0^{ps}, s_1^{ps}, \ldots, s_{L-1}^{ps})$ where the IV and key bits are initially loaded. Though some of these s_r^{ps} are expressible as $s_r^{ps} = IV_i \oplus k_j$ this may not be sufficient. For instance, if the loading procedure injects $s_0^{ps} = IV_1$ and $s_1^{ps} = IV_1 + k_1$ there is obviously ambiguity when translating the IV difference at position IV_1 into the key difference. Remember that the main idea to keep virtually IV fixed is not valid in this case as the assumption that the cipher is run using the same IV value and different keys is not true any longer. Nevertheless, there are many designs that satisfy all the above conditions thus being susceptible to the key related attacks.

4 Some practical applications of key related attacks

In this section we analyze the possible application of our technique to those designs that satisfy the necessary assumption. To successfully apply the differential analysis our main requirement was a linear mixing of certain subset of IV and key bits. It turns out that many ciphers actually emplyos such a linear mixing making them vulnerable to the key related attacks.

4.1 Case study: DECIM

DECIM-128 [2] was submitted as a hardware efficient cipher to the eSTREAM project. Though it was not successfully cryptanalyzed, as the originally proposed version called DECIM, it did not pass to the final phase of the project, but still it represents a strong design that is based on irregular decimation and nonlinear filtering. Nevertheless the initial loading of 128-bit key and 128-bit IV into the 288 bit register $S = (s_0, s_1, \ldots, s_{287})$ is performed as follows,

$$s_i = \begin{cases} k_i, & 0 \leq i \leq 127 \\ k_i \oplus IV_i, & 128 \leq i \leq 257. \end{cases}$$

Thus, DECIM-128 perfectly fits in the framework of the differential key cryptanalysis. For such an initial loading we are free to choose any subset of IV bits and to reduce the degree of equations by eliminating the dependency on the corresponding key variables. Since no decimation is involved during the initialization process which takes 1152 cycles the initial state of the cipher after the KSA can be determined (note that this might be totally infeasible as the state bits might contain as many as 2^{256} monomials - in such a case we would be forced to specify IV bits in advanced and even to guess some key bits). Then assuming that no decimation is involved in the keystream generation the attacker would simply observe the output keystream bits (in a known-plaintext scenario) recovering the secret key bits.

Since the keystream is generated by passing the output of the nonlinear filter through the ABSG decimation mechanism we cannot determine for sure which state bits constitute the output. Thus additional guessing would be necessary and this would lead to the complexity above the exhaustive search.

4.2 Case study: TRIVIUM

TRIVIUM [8] is a hardware oriented stream cipher that reached the final third phase of the eSTREAM project. Due to its simplicity and compact algebraic description it has been a popular target for cryptanalysis by a broad crypto community. However, TRIVIUM has resisted all the cryptanalytic attempts so far and the most efficient attack proposed recently [13] is a key recovery attack on a reduced round TRIVIUM in the chosen IV scenario, where the original KSA is halved, that is only 576 cycles instead of 1152 cycles are applied in the key/IV setup phase. Though the attack only considers 2 full cycles (corresponds to 576 steps) of key/IV setup, the fact that attack only employs a small subset of chosen IV's and certain combination of output keystream bits to derive linear equations involving the secret key bits seriously questions the security of the cipher. In the original paper [8] the designers claim that 2 full cycles for the key/IV setup are sufficient to properly mix the key and IV bits so that each state bit after 576 iterations depends on each key and IV bit in a highly nonlinear manner.

TRIVIUM cipher uses three LSFRs whose total length is 288, thus the state of the cipher is given by $S = (s_1, s_2, \ldots, s_{288})$. The state is contained in the three LFSRs so that the first register comprises the state bits $s_1, \ldots, s_{93}$, the second register stores $s_{94}, \ldots, s_{177}$, and $s_{178}, \ldots, s_{288}$ are kept in the third register. In the output mode the following computations are performed:
– Define $t_1 = s_{66} \oplus s_{93}$, $t_2 = s_{162} \oplus s_{177}$, and $t_3 = s_{243} \oplus s_{288}$.
– The output keystream is formed as $z = t_1 \oplus t_2 \oplus t_3$.
– Update the t_i's according to

$$t_1 = t_1 \oplus s_{91} \cdot s_{92} \oplus s_{171}; \quad t_2 = t_2 \oplus s_{175} \cdot s_{176} \oplus s_{264}; \quad t_3 = t_3 \oplus s_{286} \cdot s_{287} \oplus s_{69};$$

– Finally the registers are updated as follows:

$$(s_1, \ldots, s_{93}) = (t_3, s_1, \ldots, s_{92}); (s_{94}, \ldots, s_{177}) = (t_1, s_{94}, \ldots, s_{176});$$

$$(s_{178}, \ldots, s_{288}) = (t_2, s_{178}, \ldots, s_{287});$$

The initialization loads the 80-bit key to $s_1, \ldots, s_{80}$, and the 80-bit IV vector to the positions $s_{94}, \ldots, s_{173}$. The remaining state bits are filled with zeros apart from s_{286}, s_{287} and s_{288} which are set to one; this way the pre-setup state S^{ps} is completed. The key/IV setup is performed by clocking the cipher $4 \cdot 288 = 1152$ times without producing the output.

TRIVIUM seems to be a very suitable target for the application of key related attacks due to the following arguments.
– In the first place the first 66 output bits are linear combinations of six state bits and therefore the degree of these equations cannot be larger than the maximum degree of individual terms. Thus, if we can efficiently turn the state bit equations into linear by applying the key differentiation together with guessing some key bits, then 66 linear equations of the keystream are easily solved (below we show that only 68 key variables remain after the key differentiation).
– To apply the above technique we must ensure the existence of linear relations among certain IV and key bits. Obviously this is not the case for the pre-setup state S^{ps} as there is no mixture between the key and IV bits. But it can be easily verified that after running the key/IV setup a certain number of clocks the internal state will contain desired equations that mix IV and key bits in a linear manner. For convenience we denote the state at time i as $S^i = (s_1^{(i)}, \ldots, s_{288}^{(i)})$ where $S^{(0)} = S^{ps}$. Then after the very first clock the update variables are computed as $t_1 = k_{66} \oplus IV_{78}$, $t_2 = IV_{69} \oplus IV_{80}$, and $t_3 = k_{69}$. Hence, $s_{94}^{(1)} = k_{66} \oplus IV_{78}$ and this would give a rise to one linear relation. It is easy to verify that such a linear relations are obtained in the first 12 clocks,

$$s_{94}^{(i)} = k_{67-i} \oplus IV_{79-i}, \quad i = 1, \ldots, 12,$$

which leads to the following internal state after 12 clocks:

$$(s_1^{(12)}, \ldots, s_{93}^{(12)}, k_{55} \oplus IV_{67}, k_{56} \oplus IV_{68}, \ldots, k_{66} \oplus IV_{78}, s_{106}^{(12)}, \ldots, s_{288}^{(12)}).$$

It is important to notice that the key bits $k_{55}, \ldots, k_{66}$ and IV bits $IV_{67}, \ldots, IV_{78}$ are later mixed with remaining variables but their internal linear relation is always preserved.

Thus through the IV differential at positions $67, \dots, 78$ we get the key differential at positions $55, \dots, 66$. Technically, the IV bits at positions $1, \dots, 66$ and $79, 80$ are kept fixed, whereas the IV bits at positions $67, \dots, 78$ are varied so that the key differentiation is performed as explained before. Assuming we are able to handle the initial state equations after the KSA (which we can by specifying the IV and guessing certain number of key bits) we can get rid of the key bits $k_{55}, \dots, k_{66}$ therefore reducing the complexity of the exhaustive search to $C^{w.c.} = 2^{80-12} = 2^{68}$.

Nevertheless, we can successfully combine the key differentiation with the key guessing attack. Assume that we guess 60 key variables, thus leaving only 8 key variables in the algebraic expressions for the secret state bits (after the key variables $k_{55}, \dots, k_{66}$ has been eliminated) to be determined. Most likely the algebraic expressions are only quadratic in remaining key variables, and since we have 66 linear keystream equations we would get a system of 66 equations with the maximum number of terms equal to $\binom{8}{2} + \binom{8}{1} = 36$. The system is heavily overdefined and we easily find a correct solution. In the case there are cubic terms in the expressions we would then guess 62 bits and the number of terms would be $\binom{6}{3} + \binom{6}{2} + \binom{6}{1} = 41$ which again results in an overdetermined system. Finally, by guessing 63 bits of the key we are assured that the system is overdefined. In this case there are only 5 key variables left and the total number of terms is at most 32, thus 66 equations from the keystream generation should suffice.

4.2.1 Estimating the degree of equations

TRIVIUM is very efficiently designed with respect to the total number of gates. In particular, the update function and the keystream generation are fairly simple, though as the time evolves the equations derived become very complex due to the exponential expansion of the terms. Nevertheless, the degree of the equations does not increase exponentially in each clock of the initialization but rather a few times during one round that consists of 288 iterations. For instance, if we trace the first state bit $s_1^{(0)} = k_1$ then we notice that k_1 is not affected until it has been moved to position 66 (thus after 66 cycles) and then the update variable t_1 is computed as $t_1^{(66)} = k_1 \oplus k_{28} \oplus k_{26}k_{27} + IV_{12}$ so that $s_{94}^{(66)} = t_1^{(66)}$. The key bit k_1 is thereafter not involved in the update computation for the next 26 cycles. Then one can check that this bit is included as a linear term in the degree 4 equation for the $s_{94}^{(93)}$.

Computer simulations indicate that the degree of equations after the 4 full rounds (1152 cycles) is likely to reach its maximum value but it is unlikely that the initial state bits follow the ideal complexity reaching some 2^{159} number of terms as it would be the case with randomly generated functions. Thus there is a possibility for a further decrease in complexity after specifying the key and IV values. This however remains to be justified in a more exact manner, the main problem being that specifying the IV is not sufficient to handle enormously large expressions for the state bits that might contain as many as 2^{80} terms.

5 Conclusions

In this paper we have shown that a linear mixing of the IV and key bits might lead to an efficient key recovery attack. The application to other designs apart from TRIVIUM and DECIM-128 is currently in progress. To circumvent this type of attack the initial loading of the IV and key bits into the cipher should be performed in a nonlinear manner, thus slightly increasing the amount of clock cycles for the KSA procedure.

References

[1] S. Babbage. Cryptanalysis of LILI-128. Preproceedings of NESSIE 2-nd workshop, Egham, 2001.

[2] C. et al. Berbain. DECIM-128. Avaliable on ECRYPT Stream Cipher Project page, 2005. `http://www.ecrypt.eu.org/stream/decim.html`.

[3] E. Biham and O. Dunkelman. Differentialcryptanalysis in stream ciphers. Cryptology ePrint Archive, Report 2007/218, 2007. http://eprint.iacr.org/.

[4] E. Biham and J. Seberry. Py (Roo: A fast and secure stream cipher using rolling arrays. Avaliable on ECRYPT Stream Cipher Project page, 2005. `http://www.ecrypt.eu.org/stream/py/py.ps`.

[5] E. Biham and A. Shamir. Differential cryptanalysis of DES-like cryptosystems. *Journal of Cryptology*, vol. 4(1):3–72, 1991.

[6] E. Biham and A. Shamir. Differential fault analysis of secret key cryptosystems. In *Advances in Cryptology—CRYPTO'97*, volume LNCS 1294, pages 513–525. Springer-Verlag, 1997.

[7] A. Biryukov and A. Shamir. Cryptanalytic time/memory/data tradeoffs for stream ciphers. In *Advances in Cryptology—ASIACRYPT 2000*, volume LNCS 1976, pages 1–13. Springer-Verlag, 2000.

[8] C. Canniere and B. Preneel. TRIVIUM specifications. Avaliable on ECRYPT Stream Cipher Project page, 2005. `http://www.ecrypt.eu.org/stream/trivium.html`.

[9] ECRYPT. Call for stream cipher primitives. http://www.ecrypt.eu.org/stream/.

[10] J. Hong and P Sarkar. New applications of time memory data tradeoffs. In *ASIACRYPT*, volume LNCS 3788, pages 353–372. Springer-Verlag, 2005.

[11] W. Hongjun and B. Preneel. Differential cryptanalysis of the stream ciphers Py, Py6 and PyPy. In *Advances in Cryptology—EUROCRYPT 2007*, volume LNCS 4515, pages 276–290. Springer-Verlag, 2007.

[12] M. Matsui. Linear cryptanalysis method for DES cipher. In *Advances in Cryptology—EUROCRYPT'93*, volume LNCS 765, pages 386–397. Springer-Verlag, 1993.

[13] M. Vielhaber. Breaking ONE.FIVIUM by AIDA an Algebraic IV Differential Attack. Cryptology ePrint Archive, Report 2007/413, 2007. http://eprint.iacr.org/.