FINDING EFFICIENT DISTINGUISHERS FOR CRYPTOGRAPHIC MAPPINGS, WITH AN APPLICATION TO THE BLOCK CIPHER TEA

JULIO C. HERNANDEZ

CAPS Team, INRIA-IRISA, Campus de Beaulieu, France

PEDRO ISASI

Artificial Intelligence Group, Department of Computer Science, Carlos III University, Madrid, Spain

A simple way of creating new and very efficient distinguishers for cryptographic primitives, such as block ciphers or hash functions, is introduced. This technique is then successfully applied over reduced round versions of the block cipher TEA, which is proven to be weak with less than five cycles.

Key words: cryptanalysis, block cipher, distinguisher, genetic algorithms, TEA.

1. INTRODUCTION

The construction of a distinguisher (Knudsen and Meier 2000) (i.e., an algorithm that is able of distinguishing a random permutation or random mapping from a given cryptographic primitive, such as a block cipher or hash function) is one of the main objectives of a cryptanalyst. Although a distinguisher may or may not be used to recover some of the plaintext or key bits, the existence of an efficient and effective distinguisher always means the cryptographic primitive in question is weak (Knudsen and Meier 2000; Shimoyama, Takeuchi, and Hayakawa 2000) and must be discarded for any cryptographic usage.

1.1. The Block Cipher TEA

TEA stands for Tiny Encryption Algorithm. It is the name of a block cipher invented by David Wheeler and Roger Needham, members of the Computer Security Laboratory of Cambridge University. It was presented in the 1994 Fast Software Encryption Workshop (Wheeler and Needham 1995).

TEA is a very fast block cipher that does not use predefined tables or S-boxes and does not need much initialization time. It is a Feistel type algorithm. It works over 64 bit blocks and uses keys of 128 bits. The authors say it has a security (with 8, 16, or 32 rounds) at least comparable with the data encryption standard (DES), and it is quite faster.

However, in the light of some recent results (Wagner, Kelsey, and Schneier 1997; Moon et al. 2002) this assertion about its security seems to be extremely optimistic. It is very portable, simple, and efficient as its compact code shows:

```
void code(long* v, long* w,long* k)
{ unsigned long y=v[0], z=v[1], sum=0,
   delta=0x9e3779b9, n=32;
    while (n-->0)
     { sum += delta ;
        y+=(z<<4)+k[0]^z+sum^(z>>5)+k[1];
        z+=(y<<4)+k[2]^y+sum^(y>>5)+k[3];
      }
  w[0]=y; w[1]=z;
}
```

Address correspondence to Julio C. Hernandez, CAPS Team, INRIA-IRISA, Campus de Beaulieu, France; e-mail: jcesar@irisa.fr

1.2. Overview

Our method to attack a given cryptographic primitive, is based in the search for subsets of the input space that produce a high (statistically significant) deviation of the distribution of a given subset of the output.

For this search we use a genetic-algorithm-based approach in which individuals of the population codify bit masks that are used to perform a logical AND with randomly generated inputs.

In this way, we get an extremely efficient representation of those input subsets characterized by having some of the input bits fixed to a zero value. Input subsets of this form are evaluated performing chi-square tests over the output distribution of the subset under observation.

These tests vary, because additional rounds of TEA exponentially increase the dispersion of the output, and thus the difficulty of finding significant deviations. This forces us to find tests that are increasingly more powerful, to be able to find bad statistical properties where others find no deviations from ideal behavior.

The genetic algorithm will evolve individuals and populations toward those that, by fixing the input bits that have a greater effect over the observed output, produce a higher deviation from the expected probability distribution.

2. RESULTS

In the following section, we will present the different approximations used in every case and the results obtained over them, proposing efficient distinguishers for all these versions of the algorithm. It is important to keep in mind that these results are of increasing importance and thus, results over TEA with $n + 1$ cycles generally imply results over TEA with n cycles.

2.1. One Cycle of TEA

Every input subset (or bitmask) is tested by generating 2^{11} random inputs and then performing a logical AND between each of these inputs and the bitmask. Then TEA1 is applied over thus generated vectors, and the values of the least significant eight bits of the first output word of TEA, that is $w[0]\&255$, are computed.

We have focused in this particular part of the output because there are some authors, notably (Kelsey, Schneier, and Wagner 1999) that have shown that block ciphers using rotation as one of their round operations (as is the case of many modern block ciphers, including TEA) tend to exhibit bad distributions in their least significant bits of their output words. The fitness function we propose for the genetic algorithm is highly related with the chi-square statistic χ^2, which in our case measures the deviation of the observed output distribution from a uniform, in this way:

$$\text{fitness} = \begin{cases} w^4 & \text{If } \chi^2 = 522{,}480 \\ \chi^2 & \text{in other case.} \end{cases}$$

The idea behind this fitness function is that the value of the chi-square statistic cannot increase indefinitely, but has a maximum. The maximum value of the chi-square statistic corresponding to a distribution with 255 degrees of freedom and 2^{11} observations is precisely 522,480, which is obtained when all the possible 256 outputs collapse in a single one.

Once we find bitmasks that produce this maximum deviation, our search must continue by looking for those bitmasks that are heavier (have more number of 1's), and this is the reason of including the weight w in the formula above, once the maximum deviation is obtained.

Heavier bitmasks are preferred because they allow for a larger set of different inputs, thus in this sense we can say they are more general, and also give more information about the input subset (the 0's in the bitmask, or inactive bits) that has a stronger influence over the observed output.

Obviously, we must try to maximize this fitness value. The code we used for testing was the implementation of the genetic algorithm of William M. Spears, from the Navy Center for Applied Research. Other parameters needed for running the genetic algorithm are the size of the population, the mutation rate, and the crossover probability. Those values were fixed, respectively, to 100, 0.005, and 0.85 after some trial and error testing, that showed they generally produced the best results.

The best bitmask we found after around 1,000 generations and 90,000 evaluations for the fitness function presented above was m_1:

```
{0xFFFFEF00,0xFFFFE000,0xFFFFFE00,
0xFFFFFF00,0xFFFFFFFF,0xFFFFFFFF}
```

which has a weight of 153 and produces a maximum chi-square value of 522,240.

This bitmask was then tested with different, previously unseen input subsets of size 2^{11} and in every case it produced maximal deviations (i.e., a collapse of the observed output). This bitmask can be used to construct an exceptionally efficient and simple distinguisher for TEA1, which pseudocode is presented below:

INPUT:
  $F : Z_2^{192} \longrightarrow Z_2^{64}$, a random mapping or TEA1
ALGORITHM:
  Generate a random vector $v \in Z_2^{192}$
  Apply the mask m_1, getting $v' = v\&m_1$ which can take any of 2^{153} values
  Compute $F(v') = w[0]w[1]$
  Compute $r = w[0]\&255$
OUTPUT:
  If $r = 185$ then F is TEA1
    else F is not TEA1

It is interesting to point out that this distinguisher is extremely efficient, given that with only one input text has a false positive probability of 1/256 (or around 0.4%) and a zero probability of false negatives.

2.2. Two Cycles TEA

TEA with two cycles (TEA2) is much harder than TEA1. The additional cycle significantly increases the strength of the algorithm and no usable bitmaks producing maximal deviations (full output collapses) were found, thus the fitness function used for TEA1 is not adequate here.

Although a fitness function consisting simply of the chi-square statistic can break TEA2, it needs some extra care with technical details (mainly a selection proportional to rank and not to fitness and quite different mutation and crossover probabilities) to produce good results. Another drawback of this fitness function is that it shows a very low convergence toward good

TABLE 1. Some p-Values for a Chi-Square Statistic with 255 Degrees of Freedom

p-value	Value
0.5	254.33
10^{-2}	310.45
10^{-3}	330.51
10^{-4}	347.65
10^{-5}	362.98

solutions (those which produce results statistically better than those that can be expected at random).

Furthermore, this simplistic approximation is not applicable to TEA3. Thus, after solving the case for TEA3, we turned back to TEA2 and observed that the same fitness function would be very adequate, thus we will present now a new fitness function that reflects an idea that is enough for breaking both TEA2 and TEA3. The fitness function used in this case is shown below:

$$\text{fitness} = \begin{cases} \frac{1}{w^3} + w^4 & \text{If } \chi^2 \geq 403.4579 \\ \frac{1}{w^3} & \text{in other case.} \end{cases}$$

The idea behind this fitness function is to divide the search for good and heavy bitmasks in two phases. In the first one, the chi-square values will be around the 0.5%, and the fitness function above will simply look for low-weighted bitmasks. When the bitmasks are sufficiently low to produce high values of the chi-square statistic (above the threshold of 403.4579), then the objective is to find heavier bitmasks between those which produce a very high statistical value (Table 1 shows some p-values of a chi-square distribution with 255 degrees of freedom).

In this way, we do not maximize the chi-square value but the weight of the masks that produce a statistic value over a threshold. In this case, the threshold is the corresponding value for a chi-square distribution with 255 degrees of freedom and a p-value of 5×10^{-9}, thus it clearly shows a very strong deviation from randomness.

Using this approximation, we got the following bitmask m_2 after around 800 generations and 70,000 evaluations:

```
{0xBFFFF0FA,0xFFFE7388,0xFFFFF7F8,
0xFFFFF3F8,0xFFFFEF85,0xFFFFFF8C},
```

which has a weight of 155.

After testing this bitmask with other previously unseen inputs, the average chi-square statistic obtained was 736.05. It is then feasible to construct an efficient distinguisher for TEA2, as shown in the following pseudocode:

INPUT:
$F : Z_2^{192} \longrightarrow Z_2^{64}$, a random mapping or TEA2
ALGORITHM:
Generate 2^{11} random vectors $v_i \in Z_2^{192}$
Apply mask m_2, getting $v_i' = v_i \& m_2$ which can take any of 2^{155} values
Compute $F(v_i') = w[0]_i w[1]_i$
Compute $r_i = w[0]_i \& 255$

Perform a chi-square test for checking if the observed distribution
of r_i is consistent with the expected uniform distribution,
calculating the corresponding chi-square statistic χ^2
OUTPUT:
If $\chi^2 > 390.0315$ then F is TEA2
else F is not TEA2

The 390.0315 is the value corresponding to a p-value of 10^{-7}, thus the distinguisher described before will have a false positive probability of 10^{-7} and a false negative probability even closer to zero.

2.3. Three Cycles TEA

Using essentially the same approximation described before, we get the following bitmask m_3 for TEA with three cycles after around 750 generations and 63,000 evaluations

```
{0xFFE1F040,0xFCE70446,0xFFEFF06E,
0xFFE7F42A,0xFFBF1825,0xFFFA0064}
```

which has a weight of 116 and produces an average chi-square statistic of 393.6 over previously unseen cases.

The distinguisher will be, then:

INPUT:
$F : Z_2^{192} \longrightarrow Z_2^{64}$, a random mapping or TEA3
ALGORITHM:
Generate 2^{11} random vectors $v_i \in Z_2^{192}$
Apply mask m_3, getting $v_i' = v_i \& m_3$ which can take any of 2^{116} values
Compute $F(v_i') = w[0]_i w[1]_i$
Compute $r_i = w[0]_i \& 255$
Perform a chi-square test for checking if the observed distribution
of r_i is consistent with the expected uniform distribution,
calculating the corresponding chi-square statistic χ^2
OUTPUT:
If $\chi^2 > 390.5197$ then F is TEA3
else F is not TEA3

The 330.5197 is the value corresponding to a p-value of 10^{-3}, thus this distinguisher will have a false positive probability of 10^{-3} and a false negative probability of around 0.5%.

2.4. Four Cycles TEA

This is a considerably harder case. When using the same approximation that was successful over the prior two cases, we only managed to obtain bitmasks of relatively low weight, not more than 47.

This is interesting and can be useful for different cryptanalytic purposes, for example, for starting a search of impossible differentials, but it is not enough in this case, as it does not allow for 2^{64} different inputs. If not having at least 2^{64} different elements in the input subset, we cannot ask to obtain a good distribution of the output, as it must somehow reflect the low entropy of the input (although it does not need to be precisely in $w[0]\&255$, the bits we observe).

TABLE 2. Some p-Values for a Chi-Square Statistic with 64 Degrees of Freedom

p-value	Value
0.5	63.33
10^{-2}	93.21
10^{-3}	104.71
10^{-4}	114.83
10^{-5}	124.10

Thus we need a different approximation, a subtler one, able of distinguishing from randomness behaviors that may have past undetected by other, less sensible, tests.

Our proposal is based on a test used to measure the Strict Avalanche Criterion or SAC (Forre 1988). The SAC will be measured just by flipping at random a bit that is at a position where there is an active bit (i.e., one that has a 1 in the corresponding input mask), then measuring the Hamming distance of the two outputs.

A mapping $Z_2^m \longrightarrow Z_2^n$ has the SAC over a certain input subset $S \in Z_2^m$ iff the Hamming distance of the output of inputs x and x′ that only differ in a position (i.e., $w(x \oplus x') = 1$) and belong to the input subset S follow a Binomial distribution with parameters $\frac{1}{2}$ and n.

In the case of TEA, we should have a $B(\frac{1}{2}, 64)$. It is important to note that the satisfactibily of this criterion implies the avalanche effect (changes in the input of only one bit should produce a change of around half of the output), because the average of the distribution $B(\frac{1}{2}, n)$ is $\frac{n}{2}$.

For testing if a given bitmask represents an input subset which elements meet the SAC, we propose to perform a chi-square test for the goodness of fit of the observed output distribution of the Hamming values with respect to the theoretical Binomial distribution.

In this case, we have a chi-square statistic with 64 degrees of freedom. Table 2 shows some p-values of this distribution.

Using this approximation, we got the following bitmask m_4:

```
{0x96922A0C,0x42C06402,0x35B11001,0x97000000,
0xF0000001,0xBEB00001}
```

with a weight of 50 (the weight is not a limit here, as far as sufficiently many input elements exist to perform the test), a fitness of 420.95 and an average chi-square value over previously unseen examples of 294.86.

In this case, we did not introduce any preference for heavier bitmasks. The corresponding distinguisher pseudocode will then be:

INPUT:
$F : Z_2^{192} \longrightarrow Z_2^{64}$, a random mapping or TEA4
ALGORITHM:
a) Generate 2^{11} random vectors $v_i \in Z_2^{192}$
b) Apply mask m_4, getting $v_i' = v_i \& m_4$ which can take any of 2^{50} values
c) For every v_i'
i) Select a position p at random from those that
have a 1 in the bitmask (active positions)
ii) Generate v_i'' by changing the value of v_i' at position p
d) Compute $r_i = H(F(v_i'), F(v_i''))$, the Hamming distance
between $F(v_i')$ and $F(v_i'')$

e) Perform a chi-square test for checking if the observed
distribution of r_i is consistent with the expected distribution,
calculating the corresponding chi-square statistic χ^2

OUTPUT:

If $\chi^2 > 148.3564$ then F is TEA4
else F is not TEA4

where 148.3564 is the threshold value corresponding to a p-value of 10^{-8}. The distinguisher will, then, have a false positive probability of 10^{-8} and a negligible false negative probability.

3. CONCLUSIONS

We have presented a radically new method of constructing distinguishers for cryptographic mappings such as block ciphers or hash functions, which has been proven useful.

This method represents a completely new way of applying heuristic search algorithms to the cryptanalysis of cryptographic primitives. Until now, all proposed methods were focused in directly searching for the key, but this approach seriously limited the scope of those attacks to only classical, old, and completely broken ciphers. This is because only in ancient cryptosystems one could find the property that, deciphering with a value very close to the key produces a text very close to the original text. This linearity was then used to measure the distance of a certain key to the right key by measuring somehow the distance of the decripted text to a normal English-readable text, thus guiding the search for the correct key. A classical example of this strategy is Bagnall, Mckeown, and Rayward-Smith (1997) and Spillman et al. (1993). This property is not present at all in nowadays block ciphers, where even if just by chance during the search one arrives to get 127 out of 128 correct bits in the key, the corresponding plaintext will be garbage, a collection of pseudorandom values, and thus this classical approach will be of no value at all.

This new method is applied over reduced round variants of the modern block cipher TEA, which is proved to be weak with four cycles, that is, eight rounds.

Furthermore, the distinguishers obtained are very efficient, needing very few inputs to obtain high distinguishing probabilities. In the case of TEA1 we only need one text to obtain a high distinguishing probability, and even with TEA4 we manage to construct a distinguisher that only needs 2^{11} texts.

Moreover, these output distributions discovered with the aid of genetic algorithms and used to construct the distinguishers, could also be useful in future attacks for recovering part of the plaintext or key bits, as in Knudsen and Meier (2000).

Additionally, this attack could help in avoiding pitfalls while designing new cryptographic primitives. For example, if we substitute in the TEA code all operations by additions,

```
void TEA_SUMs(long* v, long* w,long* k)
{ unsigned long y=v[0], z=v[1], sum=0,
   delta=0x9e3779b9, n=32;
    while (n-->0)
     { sum += delta ;
        y+=(z<<4)+k[0]+z+sum+(z>>5)+k[1];
        z+=(y<<4)+k[2]+y+sum+(y>>5)+k[3];
      }
  w[0]=y; w[1]=z;
}
```

the resulting cipher is only slightly weaker than the original, as proved by the possibility of extending this same attack to five cycles.

If, on the other hand, we change all operations by XOR's,

```
void TEA_XORs(long* v, long* w,long* k)
{ unsigned long y=v[0], z=v[1], sum=0,
   delta=0x9e3779b9, n=32;
    while (n-->0)
     { sum += delta ;
        y^=(z<<4)^k[0]^z^sum^(z>>5)^k[1];
        z^=(y<<4)^k[2]^y^sum^(y>>5)^k[3];
      }
  w[0]=y; w[1]=z;
}
```

the final cipher is much weaker than the original, because this attack scheme could find efficient distinguishers even with 128 cycles.

Thus, in a way, and by using this attack technique, we can give reasons to support the exact combination of additions and XOR's proposed by TEA designers and avoid serious pitfalls during the development of new cryptographic algorithms.

Finally, these results clearly show that the block cipher TEA with less than five cycles is not robust enough as to being useful for any cryptographic purposes.

Although we acknowledge that previous works, specially Moon et al. (2002) have shown that TEA cannot be considered a secure block cipher, and have presented stronger attacks on it, we still think the proposed approach remains very valuable because it is the first published attack using AI techniques that is able of producing worthy cryptanalytic results when confronted against modern ciphers.

ACKNOWLEDGMENT

This research was supported by project TIC2002-04498- C05-4 of the Spanish Ministerio de Ciencia y Tecnologia.

REFERENCES

BAGNALL, A. J., G. P. MCKEOWN, and V. J. RAYWARD-SMITH. 1997. The cryptanalysis of a three rotor machine using a genetic algorithm. *In* Proceedings of the Seventh International Conference on Genetic Algorithms, ICGA 1997, Morgan Kaufmann, San Mateo, CA.

FORRE, R. 1988. The strict avalanche criterion: Special properties of Boolean functions and extended definition. *In* Advances in Cryptology—CRYPTO'88, LNCS, Vol. 403, Springer-Verlag, Berlin, pp. 450–468.

KELSEY, J., B. SCHNEIER, and D. WAGNER. 1999. Mod n cryptoanalysis with applications against RC5P and M6. *In* Proceedings of the 1999 Fast Software Encryption Workshop. Springer-Verlag, Berlin, pp. 139–155.

KNUDSEN, L., and W. MEIER. 2000. Correlations in RC6 with a reduced number of rounds. *In* Proceedings of the Seventh Fast Software Encryption Workshop. Springer-Verlag, Berlin.

MOON, D., K. HWANG, W. LEE, S. LEE, and J. LIM. 2002. Impossible differential cryptanalysis of reduced round XTEA and TEA. *In* Fast Software Encryption, FSE 2002, LNCS, Vol. 2365, Leuven, Belgium, Springer, Berlin, pp. 49–60.

SHIMOYAMA, T., K. TAKEUCHI, and J. HAYAKAWA. 2000. Correlation attack to the block cipher RC5 and the simplified variants of RC6. *In* Third AES Candidate Conference AES3.

SPILLMAN, R. et al. 1993. Use of a genetic algorithm in the cryptanalysis of simple substitution ciphers. Cryptologia, **17**(1).

WAGNER, D., J. KELSEY, and B. SCHNEIER. 1997. Related-key cryptoanalysis of 3-WAY, Biham-DES, CAST, DES-X, NewDES, RC2 and TEA. *In* Proceedings of the ICICS'97 Conference. Springer-Verlag, Berlin, pp. 233–246.

WHEELER, D., and R. NEEDHAM. 1995. TEA, A tiny encryption algorithm. *In* Proceedings of the 1995 Fast Software Encryption Workshop. Springer-Verlag, Berlin, pp. 97–110.