Genetic Cryptanalysis of Two Rounds TEA

Julio César Hernández¹, José María Sierra¹, Pedro Isasi² and Arturo Ribagorda¹

¹ Computer Security Group, Computer Science Dept., Carlos III University,
28911 Leganés, Madrid, Spain
{jcesar, sierra, arturo}@inf.uc3m.es

² Artificial Intelligence Group, Computer Science Dept., Carlos III University,
28911 Leganés, Madrid, Spain
{isasi}@ia.uc3m.es

Abstract. Distinguishing the output of a cryptographic primitive such as a block cipher or a hash function from the output of a random mapping seriously affects the credibility of the primitive security, and defeats it for many cryptographic applications. However, this is usually a quite difficult task. In a previous work [1], a new cryptanalytic technique was presented and proved useful in distinguishing a block cipher from a random permutation in a completely automatic way. This technique is based in the selection of the worst input patterns for the block cipher with the aid of genetic algorithms. The objective is to find which input patterns generate a significant deviation of the observed output from the output we would expect from a random permutation. In [1], this technique was applied to the case of the block cipher TEA with 1 round. The much harder problem of breaking TEA with 2 rounds is successfully solved in this paper, where an efficient distinguisher is also presented.

1 Introduction

A cipher [2] is an algorithm characterised by a tuple $(\Pi, \Gamma, K, \mathcal{E}, \Delta)$ that verifies:

1. Π is the finite set of plaintexts
2. Γ is the finite set of ciphertexts
3. K is the finite set of possible keys
4. For every κ in K there is a ciphering function ϵ_κ in $\mathcal{E}$ and a deciphering function δ_κ in Δ that verifies $\delta_\kappa(\epsilon_\kappa(\pi)) = \pi$ for every plaintext π de Π .

The fundamental property here is number 4, that implies that the original plaintext can be recovered if the ciphering key is known. But this knowledge of the ciphering key must not only be sufficient for recovering the text, but also necessary. Any robust cipher must not allow the recovery of plaintext without the knowledge of the key used.

Block ciphers are a special class of ciphers, characterised for operating over bit blocks instead of over single bits as the stream ciphers do. This block length (typically 64, 128 or 256 bits) and the key length used to cipher (typically 64 or 128 bits) are fixed and fundamental for the block cipher's strength.

In an ideal block cipher, every key must define a random permutation of the input bits over the output bits. This is why the problem of deciding if a given block cipher is secure or not can be translated into the problem of deciding if the permutations it generates when fixing keys are or not random. Unfortunately, this problem is very hard. Randomness, although being a paramount matter in security, is quite hard to evaluate.

However, and independently of the techniques used, if we are able of distinguishing a cryptographic primitive from a random mapping in an efficient and statistically significant way, then this cryptographic primitive must be discarded for any cryptographic application, and immediately removed and replaced by a more robust algorithm. This is why the techniques for evaluating randomness have a paramount importance in cryptography.

This paper is organised as follows: Section I is a brief introduction to the concept of block ciphers and the importance of distinguishers in cryptanalysis. Section II briefly introduces the block cipher TEA, which will be the objective of our attacks. Section III highlights the main ideas behind what we call genetic or poisoned cryptanalysis. Section IV presents known results of this technique over TEA with 1 round, mostly for comparison purposes, and new results over the harder TEA with 2 rounds. Section V presents some conclusions and Section VI finally mentions some selected references.

2 The TEA Algorithm

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 [3].

TEA is a very fast block cipher that does not use predefined tables or s-boxes and does not need much initialisation time. It is a Feistel type algorithm, thus named because it divides its input in two halves and operates over them individually in each round and interchanges them at the end of every round. It works over 64 bit blocks and uses keys of 128 bits, which are large enough for today's security standards.

As they 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. The block cipher TEA has some additional advantages: it is very robust (only one published attack [4]

in more than 7 years of existence, and it is one *academic* attack, that is, a theoretical attack with little or no practical implications) and it is very portable, simple and efficient as its compact code below shows:

```
void code(long* v, long* w, long* k){
unsigned long y=v[0],z=v[1],sum=0,delta=0x9e3779b9,n=8;
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 ; }
```

3 Cryptoanalysis with genetic algorithms

Some classic statistical tests [5] are based in observing the output of a block cipher algorithm while *feeding* it with inputs as random as possible. This is because the idea behind these test is to verify if the execution of the block cipher algorithm maintains the good randomness properties of the input, so this must be as random as possible. Our proposal is based in a complete different idea. Instead of using a random input, we use what can be called *poisoned* inputs.

We try to check if the fixing of some of the input bits simplifies the process of discovering correlations between the input and the output bits. If significant correlations exist, we would be able to infer information about some bits of the plaintext just from the output bits of the ciphertext. These correlations could also be used to distinguish the block cipher algorithm from a random permutation in a simple and efficient way.

It is important to mention that this fixing of some bits in the input must be reasonable in length to allow for enough different inputs to mathematically verify the effect it has over the output. If a statistically significant deviation from randomness is found, then we would have an easy method for distinguishing the primitive from a random mapping and the algorithm must be discarded for any cryptographic use.

But how to decide which bits to fix in the input and to which values is a very complex problem. It is, essentially, a search in a huge space with 2^{I+K} elements. For TEA this space bitmask has 2^{192} possible values, so an exhaustive search is infeasible. For finding the best bitmasks in those huge spaces we propose the use of genetic algorithms in which individuals will codify bitmasks.

In our method, schematically represented in Figure 1, individuals codify bitmasks that will be used to perform a logical AND with every random input. In this way, by means of an AND with a given bitmask, we manage to fix some of the input bits to zero.

Figure 1: A schema of the procedure used for fixing some of the input bits to zero by performing an AND with a bitmask.

For every bitmask (individual) in a given population of the genetic algorithm, we will observe the deviation it produces in the observed output and we will decide if this deviation is or not statistically significant. Repeating this process, we will try to find the best individuals (bitmasks) that will be those which produce more significant deviations from the expected output.

We have used the implementation of the genetic algorithm of William M. Spears, from the Navy Center for Applied Research. After a number of preliminary tests, we determined that a 0.95 probability of *crossover* and a 0.05 probability of *mutation* were adequate for our problem and we decide to fix them to these values.

The *fitness* function we try to maximise is a chi-square statistic of the observed output. We decided to observe the distribution of the 10 least significant bits of the first output word of TEA because some authors, notably [6], have shown that block ciphers that use rotation as one of their round operations (as is the case of TEA) tend to show bad distributions in their least significant bits. So we will measure the fitness of an individual by how the bitmask that it codifies affects the probability distribution of the 10 rightmost bits of the first output word of TEA.

These 10 bits can be interpreted as the binary representation of the integers between 0 and 1023, and their distribution should uniformly take all these values. For checking if this is the case, we will perform a statistical chi-square test, which is one of the most extended test in cryptoanalysis due to its high sensibility to little deviations.

The statistic distribution should correspond to a chi-square distribution with $1024-1=1023$ degrees of freedom. The values for different percentiles of this distribution are shown in Table 1:

Table 1. Values of the Chi-Square distribution with 1023 D.o.F. for different percentils

p-value	0.5	0.75	0.90	0.95	0.99
χ^2 Statistic	1022.3	1053.1	1081.3	1098.5	1131.1

Our objective will be to find bitmasks for the TEA input (both the input block of 64 bits and the key block of 128 bits) that produce a value in the chi-square statistic as high as possible, which will imply a strong proof of nonuniformity in the output distribution and, hence, a strong proof of the block cipher nonrandomness.

4 Results

Every mask (codified as individuals in the genetic algorithm population) was evaluated by performing an AND with 2^{13} random inputs, different for every individual and every generation. This makes convergence harder, but improves the generality of the results obtained because it makes overfitting nearly impossible. An example of a bitmask for TEA1 obtained using this approach is:

{0101000010001010101101000000000,010101011100001110000000000000,010000011110100011001000000000,001010011011010101000000000000,0100001111110101011100010101011,0101110100011010001101111011001}

This bitmask has a length of 192 bits ($192=64+128$) and a weight of 73. This implies that, choosing input vectors at random and applying this mask over them could give us 2^{73} different inputs to the block cipher TEA1. This is a huge number, so this bitmask is useful and applicable. This would not occur if the weight of the bitmask were very low. It is clear that if two masks provoke the same deviation in the output we should prefer the heavier one because more ones in the bitmask imply more input bits that do not affect the behaviour of the observed output.

The chi-square statistic we are using as a *fitness* function can not increase indefinitely, but has a maximum, as we will show.

As we make 2^{13} tests for every individual and any of them can return a value between 0 and 1023 that we expect will be uniformly distributed, the number of occurrences of every value must be close to $2^{13}/2^{10}=2^3=8$. The maximum value for the chi-square statistic under these assumptions will occur when the observed distri-

bution is as far as uniform as possible, that is, when only one of these 1024 possible values really occurs and all the rest do not occur. In this case we say the distribution has collapsed in a single output value and the *fitness* will be

$$\chi^2 = \sum_{i=0}^{1023} (n_i - 8)^2 / 8 = \sum_{i=0}^{1022} (0 - 8)^2 / 8 + (2^{13} - 8)^2 / 8 = 8380416$$

This is exactly the case for the bitmask show before. It produces a collapse of all the 10 rightmost bits of the first output word of TEA1 into a single value.

For assuring the generality of this bitmask, it was tested with other sets of inputs not previously seen by the generic algorithm and in every case it produced a collapse of the output. So the bitmask shows that the 10 bits we observe do not depend uniformly of every input bit (those positions that have a value of 1 in the bitmask do not affect the output bits we are observing), which is a property that block ciphers must have.

So, apart from showing that there are high correlations between the input and the output of TEA1, we can also use this result for constructing a distinguisher able of deciding if a given function is TEA1 or a random mapping using as input a single random vector. This distinguisher algorithm can be described as:

INPUT: $\mathbf{F}: \mathbf{Z}_2^{192} \rightarrow \mathbf{Z}_2^{64}$, a random mapping or TEA1

ALGORITHM:

Generate a random vector $\mathbf{v}$ of $\mathbf{Z}_2^{192}$

Apply the mask $\mathbf{m}$ getting $\mathbf{v}' = \mathbf{v} \& \mathbf{m}$ that can take 2^{73} possible values

Compute $\mathbf{F}(\mathbf{v}') = \mathbf{w}[0]\mathbf{w}[1]$

Compute $\mathbf{r} = \mathbf{w}[0] \& 1023$

OUTPUT:

If $\mathbf{r}=441$ then $\mathbf{F}$ is TEA1 else $\mathbf{F}$ is not TEA1

It is important to note that the algorithm for distinguishing TEA1 from a random mapping only needs one plaintext (input vector). This is a very unusual fact in cryptanalysis, because most attacks and distinguishers need a large number of texts (numbers like 2^{47} are common) to work properly. This distinguisher is able of distinguishing TEA1 from a random mapping with an extremely low probability of false positives (only $1/1024=0.000977$, less than a 0.1%) and a zero probability of false negatives.

The case of TEA with 2 rounds is quite harder. An additional round significantly improves the strength of the algorithm, so the same approach does not produce interesting results, even after many generations and different iterations of the genetic algorithm. Different fitness functions were tested for extending this method to TEA2, and most of them were discarded. We finally got results using as a fitness function not the chi-square statistic used in TEA1, but its fourth power. This has the effect of amplifying differences in the chi square statistic that had little influence in the previous fitness function. It also makes the selection procedure of the genetic algorithm, that in our implementation is proportional to fitness, closer to a selection proportional to rank. Using this approximation, we managed to obtain the following bitmask

```
{00110101101000111000001010110010,00110110000100000011000110001000,000101000
11110111111000011100000,01100011010001001101000111011010,0000000010010000110
1010000100001,000111100011110011111100000000}
```

that produces chi-square values that vary around 1900. As this statistic value is extremely high (see Table 1) and the weight of the mask is 77, this bitmask can be used to design a distinguisher for TEA2. The construction of the distinguisher, once we have the bitmask, is trivial. The algorithm will be:

INPUT: $\mathbf{F}$: $Z_2^{192} \rightarrow Z_2^{64}$, a random mapping or TEA2

ALGORITHM:

Generate 2^{13} random vectors $\mathbf{v}_i$ of Z_2^{192}

Apply the mask $\mathbf{m}_2$ to every vector $\mathbf{v}_i$, getting $\mathbf{v}_i' = \mathbf{v}_i$

& $\mathbf{m}_2$ that can take 2^{77} possible values

Compute $\mathbf{F}(\mathbf{v}_i') = \mathbf{w}^i[0]\mathbf{w}_{[1]}^i$

Compute $\mathbf{r}_i = \mathbf{w}_{[0]}^i \& 1023$

Perform a chi-square test for checking if the observed distribution of $\mathbf{r}_i$ is consistent with the expected uniform distribution, calculating the corresponding chi-square statistic

OUTPUT: If $\chi^2 > 1291.44$ then $\mathbf{F}$ is TEA2 else $\mathbf{F}$ is not TEA2

This is a quite interesting distinguisher in the sense that it will produce a very low ratio of false positives (a value greater than 1291.44 will only occur in a chi-square statistic with 1023 degrees of freedom one in 10^{-8} times) and also a very low probability of false negatives (the average of the statistics produced by this bit mask is around 1900 so a value of less than 1291.442306 is extremely unlikely).

It is also worthy to mention that this distinguisher uses 2^{13} input vectors, not at all a huge number but many more than the corresponding distinguisher does for TEA1 does. This is because we must perform the chi-square test and we need enough inputs for expecting at least 5 occurrences of all the 1024 possible outputs. Slightly increasing this minimum of 5 to 8 leads to the actual number $2^{3+10=13}$ of input vectors proposed.

4 Conclusions

In this work over the use of genetic algorithms in cryptanalysis we have shown how a new technique which is useful to perform an automatic cryptanalysis of certain cryptographic primitives (named poisoned or genetic cryptanalysis) can be implemented with the aid of genetic algorithms. Although this fact was previously stated in [1], it was only proven to produce results over the very limited TEA1. By showing that this model is also able of finding strong correlations in variants of the block cipher TEA with more rounds (TEA2) we have finally provided enough evidence of the interest of this technique for cryptanalysis.

References

1. Hernández J.C., Isasi P., and Ribagorda A.: An application of genetic algorithms to the cryptanalysis of one round TEA. Proceedings of the 2002 Symposium on Artificial Intelligence and its Application. (To appear)
2. Douglas R. Stinson, Cryptography, Theory and Practice, CRC Press, 1995.
3. D. Wheeler, R. Needham: TEA, A Tiny Encryption Algorithm, Proceedings of the 1995 Fast Software Encryption Workshop. pp. 97-110 Springer-Verlag, 1995
4. John Kelsey, Bruce Schneier, David Wagner: Related-Key cryptanalysis of 3-WAY, Biham-DES, CAST, DES-X, NewDES, RC2 and TEA, Proceedings of the ICICS'97 Conference, pp. 233-246, Springer-Verlag, 1997.
5. Juan Soto et. al., NIST Randomness Testing for Round 1 AES Candidates Proceedings of the Round 1 AES Candidates Conference, 1999
6. John Kelsey, Bruce Schneier, David Wagner: Mod n cryptanalysis with applications against RC5P and M6, Proceedings of the 1999 Fast Software Encryption Workshop, pp. 139-155 Springer-Verlag, 1999.