Indian Institute of Information Technology Kalyani

Department of Computer Science and Engineering
PINCODE-741235

Project Report

"FAULT ATTACK ON ROADRUNNER BLOCK CIPHER"

submitted by

Amit Anand **0000064/39/CSE/15009**
Tushar Nema **0000094/39/CSE/15039**

Supervised by:-

Dr.Sandip Karmakar

Assistant Professor

Dept of Computer Science and engineering

IIIT Kalyani

Introduction

TITLE: **Fault attack on RoadRunneR: A Small and Fast Bit slice Block Cipher for Low Cost 8-Bit Processors.**

Motivation:

With this growing field of computer Science , the prices of the small electronic devices decreases, leading to a very fine exposure to the notions like ubiquitous computing, Internet of things. The availability and programming nature of these CPU available in the market leads to acquainted economy and the reach to the loopholes of this technology becomes less vulnerable.

Thus the responsibility of the crypto-analysts increases to figure out the loopholes in the design before the attack begins.

Problem Statement:

Now with this insight we can do fault attack on RoadRunneR block cipher and after taking both cipher text and faulty cipher text (that comes after putting fault in the 9th round key)we try to figure out the key of block cipher.

Basically we get the equations from the cipher(i.e in figure 1) in which there are some unknown variables and we try to find that.

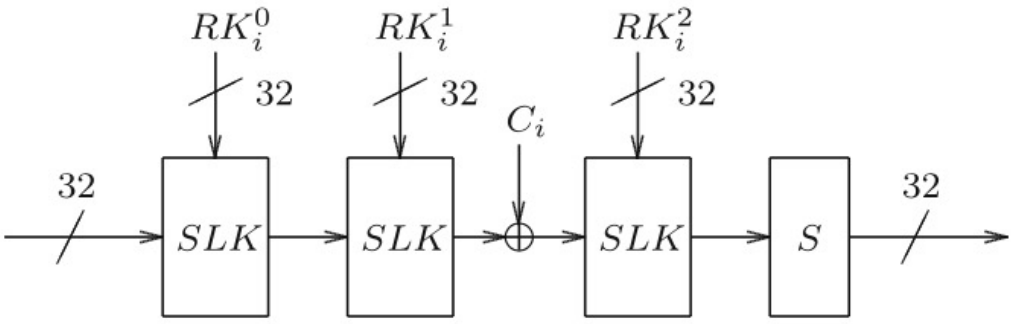
Literature Survey

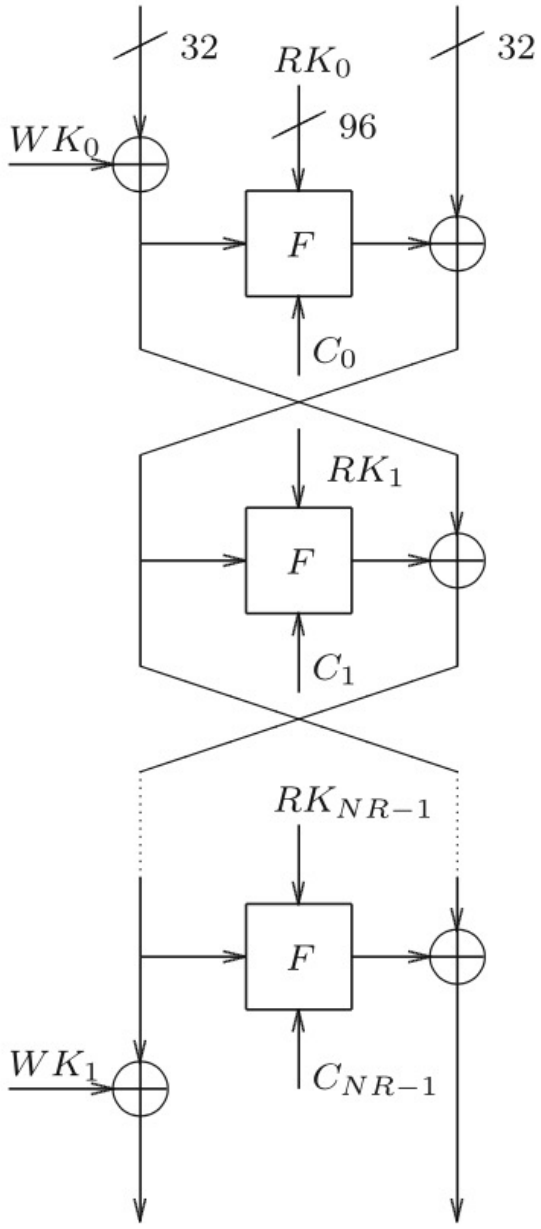
RoadRunneR

RoadRunneR,lightweight block cipher with the goal of efficiency (especially in 8-bit low cost CPUs) and provable security in terms of minimum number of active S-boxes in differential and linear trails. The cipher is especially designed to have a very low code size, while having high throughput.

Our pre- liminary cryptanalysis showed that RoadRunneR have a relatively high security margin in contrast to most lightweight ciphers. RoadRunneR has variable area- time-security trade-off characteristics with different implementation methods so that it can fit the needs of specific application it may be used in.

Moreover, we defined a new efficiency comparison metric for block ciphers which (to the best of our knowledge for the first time) takes into account the key size of the cipher. Using this metric, we could compare block ciphers with differ- ent key sizes in a fair way.





This is the internal structure of the function F in figure 1 that is basically a SLK function that is the consecutive application of S-box layer(S) ,Diffusion Layer(L) and Key addition (K).

Substitution and Diffusion:

When X (figure 1) that is basically a 32 bit when going into F function then first it will go to the S -box which is basically the following:

Input	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Output	0	8	6	D	5	F	7	C	4	E	2	3	9	1	B	A

Bit slice S-box structure has advantage in both hardware and software implementations. In software, the per- mutations before and after the S-boxes disappear. In hardware, on the other hand, they can be implemented by a simple wiring which consumes no extra area.

The bits(permutation) gets substituted according to the S-box and a new permutation is formed after coming out from the S-box which is followed by the diffusion in the L box as follows:

After the bitslice S-box layer, we used a linear function on each byte of the state to provide diffusion inside 8-bit words.

So we needed an efficient linear function operating on bytes. One classical solution for such a linear function for CPUs is using XOR of shifted and rotated values of the input word.

We try to build linear functions of the form :

$$L(x) = (x \ll i) \oplus (x \ll j) \oplus (x \ll k)$$

x ≪ i represents i-bit rotation of the CPU word x to the left.

Key Addition:

Table 1. Best L matrices under given constraints.

Matrix	i, j, k	# of instructions (for two matrix mult.)	Minimum # of active S-boxes in F
L_1	0,1,2	13	10
L_2	0,1,4	11	8
L_3	0,1,5	11	8
L_4	0,4,5	11	8
L_5	1,4,5	11	8

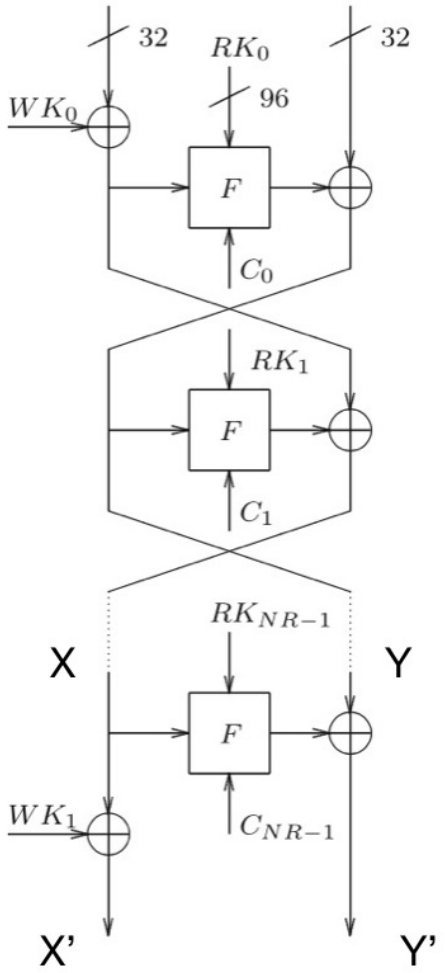
For example, if master key is 128-bit, it will be divided into 4 words of 32-bit as ABCD. Initial whitening key is A, first round key is B-C-D, second round key is A-B-C, etc.

We call it as Whitening Key because its basic function is to xor the initial bits in round 1 and the final bits after the last round.

What Ci Represents?

After the second SLK function, round constant is XORed to the least significant byte (rightmost byte, i.e., x3) of the state. For round i = 0,1,...,NR 1, the round constant is Ci = NR i, where NR is the number of rounds, and Ci is represented as 8-bit little endian integer, that is 12 = 00001100, 11 = 00001011, etc.

Contribution



The 64 bit block size is first divided in two parts which where comes out after the 9th round is referred to X(left side) and Y(right side).

Let the output coming after the 10th Round be X' and Y'.

$X' = WK_1 \oplus X$

$Y' = F(RK_{NR-1}, C_{NR-1}, X) \oplus Y$

Now the equations 1 and 2 came when we have not induced any fault at any step.

Now let us put a fault after the 9th round such that Y becomes Y1.

Now the perceived output show some change and let the new output be Y1.

Now the change in equation is :

$X' = WK_1 \oplus X$

$Y1' = F(RK_{NR-1}, C_{NR-1}, X) \oplus Y1$

Here the known variables are X ,Y and Y1 and unknown variables are X,Y,X,Y,WK1 Now we are working on to solve 5 unknown vari-

ables and We are exploring the function F and we are also trying to reduce the fair complexity in decrypting the function F. We have 3 equations and from this we will try to decipher the key using some mathematical operations on the equations.

Now we are doing Faults:

Fault in Y

$$Y'_f = F(RK_{NR-1}, C_{NR-1}, X) \oplus Y_f \quad \text{——— 4}$$

Fault in X

$$X'_f = WK_1 \oplus X_f \quad \text{——— 5}$$

$$Y''_f = F(RK_{NR-1}, C_{NR-1}, X_f) \oplus Y \quad \text{——— 6}$$

Xoring equation 1 and 4:

$$X' \oplus X'_f = X \oplus X_f \quad \text{——— 7}$$

Now we get 4 different values of 32 bit each

$$Y' \oplus Y'_f = Y \oplus Y_f \quad \text{——— 8}$$

Same is what we get from equation 7

We have 4 different values of Function $F(X,RK_{NR-1},C_{NR-1})$ as we get from equation 4

$$\mathbf{S(L(S(L(S(L(S(L(S(X))\oplus RK^0_i))\oplus RK^1 i))\oplus Ci))\oplus RK^2 i)=F(X,RK_{NR-1},C_{NR-1})}$$

$$\mathbf{L(S(L(S(L(S(L(S(X))\oplus RK^0_i))\oplus RK^1 i))\oplus Ci))\oplus RK^2 i=S^{-1}(F(X,RK_{NR-1},C_{NR-1}))}$$

$RK^2 i$ ⟶ 16 different values

$L(S(L(S(L(S(L(S(X))\oplus RK^0_i))\oplus RK^1 i))\oplus Ci))$ ⟶ 16 different values

$\mathbf{L(S(L(S(X))\oplus RK^0_i))}$ ⟶ 64 different values

$RK^1 i$ ⟶ 64 different values

$\mathbf{RK^0_i}$ ⟶ 256 different values

$RK^0 i$ ⟶ 256 different values

$RK^1 i$ ⟶ 64 different values

$RK^2 i$ ⟶ 16 different values

From equation 5 we can imply that WK_1 has 4 different values.

Total no. of different keys = $2^8 \times 2^6 \times 2^4 \times 2^2$

Now we reduce the exhaustive search space for possible keys of RoadRunneR:

$$2^{20} \times \text{No. of rounds} \approx 2^{23}$$

Conclusion

A very efficient Feistel type bitslice block cipher, RoadRunneR, with 64-bit block size and 80-bit or 128-bit key length was presented. RoadRunneR is a perfect choice for devices with very restricted memory resources and for applications requiring reasonable throughput expectations. Our cipher has a high security margin in contrast to most of other lightweight block ciphers.

We did a fault attack on the block cipher and found the related equations. Then we derived an algorithm to find the solutions to those equations. We have proposed all the possibiliy of the pair of keys which could have been found.

Future Work

We have derived an algorithm to find the key by doing Fault attack on RoadRunneR and thus we are on the way of implementing a program which can find out the key of the block cipher when it receive a cipher text.

Additionally we have also initialized writing our first research paper providing a relevant and substantial thesis of the Fault attack we did on the block cipher.

References

1. Adnan Baysal and Shap ahin
TBTAK BLGEM, 41470 Gebze, Kocaeli, Turkey, Department of Computer Engineering, Kocaeli University 41380 Umuttepe Yerlekesi, Turkey
adnan.baysal@tubitak.gov.tr

2. Differential Fault Analysis of AES-128 Key Schedule using a Single Multi-Byte Fault
Sk Subidh Ali and Debdeep Mukhopadhyay (Dept. of Computer Science and Engineering Indian Institute of Technology Kharagpur India).
debdeep@cse.iitkgp.ernet.in

3. Differential Fault Analysis of the Advanced Encryption Standard using a Single Fault
Michael Tunstall(Department of Computer Science, University of Bristol,Merchant Venturers Building
Woodland Road,Bristol BS8 1UB, United Kingdom. @cs.bris.ac.uk)
Debdeep Mukhopadhyay and Subidh Ali (Computer Sc. and Engg, IIT Kharagpur, India
debdeep,subidh@cse.iitkgp.ernet.in)
4. https://www.youtube.com/watch?v=2aHkqB2-46kt=28 (Cryptography by Christof Paar)