Differential Cryptanalysis of 24-Round CAST-256

Andrey Pestunov
Institute of Computational Technologies SB RAS
Academician M.A.Lavrentjev ave., 6, 630090, Novosibirsk, Russia
Email: pestunov@gmail.com

***Abstract*— A 48-round block cipher CAST-256 was a participant of the AES competition. There are two published attacks on this cipher. The first allows to break the cipher, consisted of 16 rounds. Another can break 36 rounds but only for some weak keys, in particulary, a 24-round version of CAST-256 can be broken for a 2^{-30} part of all possible keys. An attack described in this paper allows to break 24 rounds of CAST-256, but this attack works for all the keys and not only for the weak ones. Requirements of the attack are: 2^{24} chosen plaintexts, 2^{29} bytes of memory and 2^{244} encryptions. This complexity is less than the complexity of a brute-force attack for 256-bit keys. A success probability of the attack is over 90%.**

I. Introduction

An area of block ciphers usage is very wide, therefore a serious effort is devoted to their construction and examination. A typical block cipher is supplied by a secret key and encrypts plaintext in fixed length blocks. The secret key is a sequence of bits, used to derive an array of subkeys. An encryption procedure consists in several-time execution of a simple transformation which depends on the subkey(s) and called a *round.* The rounds of the cipher can be identical or different, for the decryption their reverse order is needed. The more rounds are implemented the more secure the cipher is, but the encryption becomes slower, therefore a designer of the cipher sets up a certain number of rounds which provides both security and speed.

Nowadays the only way to learn the strength of the cipher is trying to break it (to *cryptanalyse* it) [3]. Generally speaking *cryptanalysis* means looking for any weaknesses of the cipher but the main goal is recovering the secret key. Any cipher can be broken via an exhaustive key-search (a brute-force attack) that is why a cryptanalyst is supposed to invent an *attack,* i.e. a key-recovering algorithm, which works faster than the exhaustive key-search. Even if the attack is infeasible in practice, e.g. needs 2^{240} encryptions for 256-bit key, it is an important theoretical vulnerability of the cipher [3]. The most of the attacks focus on recovering the subkeys instead of the secret key but obviously the knowledge of the key or the knowledge of all the subkeys are equivalent.

A 48-round block cipher CAST-256 was a participant of the AES competition. There are two published attacks on this cipher. The first of them [5] allows to break the cipher, consisted of 16 rounds. Another [1] can break 36 rounds, but only for some weak keys, in particulary, a 24-round version of CAST-256 can be broken for a 2^{-30} part of all possible keys. An attack described in this paper allows to break 24 rounds of CAST-256, but this attack works for all the keys and not only for the weak ones. The attack requires 2^{24} chosen plaintexts, 2^{29} bytes of memory and 2^{244} encryptions. This complexity is less than the complexity of the brute-force attack for 256-bit keys. A success probability of the attack is over 90%.

In the remainder of the paper the CAST-256 algorithm is outlined (Section 1), a 18-round truncated differential characteristic is described (Section 2) and a differential attack based on this characteristic is shown (Section 3). See Table I for an indispensable notation.

TABLE I
Notation

$a := b$	a is inited with b
$a \oplus b$	bitwise exclusive or (xor)
$a \boxplus b$	addition modulo 2^{32}
$a \boxminus b$	substraction modulo 2^{32}
$a^{\lll n}$	left rotation of a by n bits
$a^{[0]}$	the least significant bit in the word
$a^{[31]}$	the most significant bit in the word
δ_n	a 32-bit value with 1 at n-th position and zeros everywhere else
(a_0, a_1, a_2, a_3)	a 128-bit block A divided into four 32-bit words
$(a_0, ?, ?, a_3)$	a block where some words are unknown
24_x	a number in hexadecimal
$IN^{[i,\cdots,j]}$	the bits of IN from i-th to j-th

II. The CAST-256 Algorithm

This section provides a brief description of CAST-256 (see [4] for a complete specification). The 128-bit block cipher CAST-256 comprises 48 rounds of two types: $\mathcal{A}$-rounds and $\mathcal{B}$-rounds. These rounds exploit three types a round function: F^1, F^2 and F^3. If a round contains the F^i-function it is denoted by $\mathcal{A}^i$ or $\mathcal{B}^i$, and if the type of the F-function is not important then the index i is omitted.

CAST-256 implements 24 $\mathcal{A}$-rounds and then 24 $\mathcal{B}$-rounds in a following order

$$\underbrace{\mathcal{A}^1,\ \mathcal{A}^2,\ \mathcal{A}^3,\ \mathcal{A}^1,\quad \mathcal{A}^1,\ \mathcal{A}^2,\ \mathcal{A}^3,\ \mathcal{A}^1, ...}_{24\ rounds} \qquad (1)$$

$$\underbrace{\mathcal{B}^1,\ \mathcal{B}^3,\ \mathcal{B}^2,\ \mathcal{B}^1,\quad \mathcal{B}^1,\ \mathcal{B}^3,\ \mathcal{B}^2,\ \mathcal{B}^1, ...}_{24\ rounds}$$

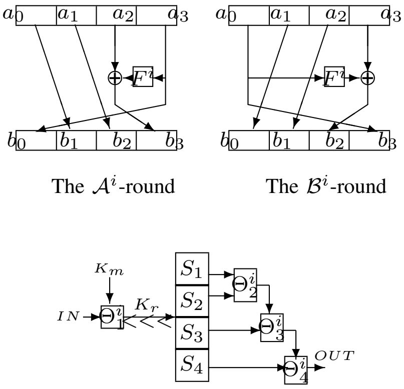
The F-functions (Fig. 1) have a similar structure but differ in an order of operations (Table II). Consider the F^2-function for instance. The function is injected with a 5-bit "rotation" subkey K_r and a 32-bit "masking" subkey K_m. The function

978-1-4244-2134-3/08/$25.00 ©2008 IEEE

has a 32-bit word IN as input and a 32-bit word OUT as output, they evolve during the F^2-function execution as follows:

$$\begin{aligned} \widetilde{IN} &:= (IN \oplus K_m)^{\lll K_r}; \\ OUT &:= ((S_1(\widetilde{IN}^{[31,\dots,24]}) \boxminus S_2(\widetilde{IN}^{[23,\dots,16]}))\boxplus \\ &\boxplus S_3(\widetilde{IN}^{[15,\dots,8]})) \oplus S_4(\widetilde{IN}^{[7,\dots,0]}). \end{aligned} \tag{2}$$

Assume that a block B is a result of the 1-round encryption of a block A. Then $(b_0, b_1, b_2, b_3) = (a_3, a_0, a_1, a_2 \oplus F(a_3))$ if $\mathcal{A}$-round is used, and $(b_0, b_1, b_2, b_3) = (a_1, a_2, a_3 \oplus F(a_0), a_0)$ if $\mathcal{B}$-round is used (Fig. 1).



The $\mathcal{A}^i$-round The $\mathcal{B}^i$-round

The F^i-function

$\oplus$	bitwise xor,
$\boxed{S_i}$	8 bits to 32 bits S-box,
Θ^i_j	binary operations (see Table II)
$\overset{K_r}{<<<}$	left rotation by K_r bits,
K_r	5-bit "rotation" subkey
$a_i,\ b_i,\ IN,\ OUT$	32-bit words
$\boxed{F^i}$	the F^i-function,
K_m	32-bit "masking" subkey

Fig. 1. The Structure of CAST-256

TABLE II
THE OPERATIONS IN THE F^i-FUNCTION

i	Θ^i_1	Θ^i_2	Θ^i_3	Θ^i_4
1	$\boxplus$	$\oplus$	$\boxminus$	$\boxplus$
2	$\oplus$	$\boxminus$	$\boxplus$	$\oplus$
3	$\boxminus$	$\boxplus$	$\oplus$	$\boxminus$

III. THE 18-ROUND DIFFERENTIAL CHARACTERISTIC

A. *Preliminaries of Differential Cryptanalysis*

Differential cryptanalysis was introduced by Eli Biham and Adi Shamir [6]. This method deals with an object called a *difference*. The difference between two values A and B is a result of their xor, i.e. $A \oplus B$. If there is no information about a relation between an input (plaintexts) and an output (ciphertexts) differences then all possible output differences are equiprobable. However, sometimes it can be established that a certain input difference Δ_{inp} causes a certain output difference Δ_{out} with a probability p greater than the others. The cipher which admits such relation can be vulnerable to a key-recovery (or a subkeys-recovery) attack.

The pair $(\Delta_{inp},\ \Delta_{out})$ is called a *differential* and a collection of all the differences on the internal rounds is called a *characteristic*. If some bits of Δ_{out} are unknown, for example $\Delta_{out} = (d_0, ?, ?, d_3)$, the differential is called a *truncated differential* [7]. Analogously the characteristic, which contains unknown bits is called a *truncated characteristic*.

Below differentials are denoted in the following way:

$$\Delta_{inp} \xrightarrow[p]{r\ rounds} \Delta_{out}.$$

B. *The 1-Round Characteristic*

Let a and z be randomly chosen 32-bit values and $b = a \oplus \delta_i$. Then an equation $a \oplus b = (a \boxplus z) \oplus (b \boxplus z)$ holds with probability 1/2 (or with probability 1 if $i = 31$) [8].

This idea can be extended. Assume that the previous conditions are held, but $b = a \oplus d$, where d has a hamming weight h. Then the named equation $a \oplus b = (a \boxplus z) \oplus (b \boxplus z)$ holds with probability 2^{-h} (or $2^{-(h-1)}$ if there is "1" is the most significant position).

Now using the preceding argumentation a characteristic which covers one $\mathcal{A}^2$-round will be constructed. Consider F^2-function, assume that K_r is known and let $IN_1 \oplus IN_2 = 29_x^{\lll 24-K_r}$. After xor of IN_1 and IN_2 with K_r the difference $29_x^{\lll 24-K_r}$ is preserved. A rotation by K_r makes it $29_x^{\lll 24}$. Hence, an input difference into S_1 is 29_x and the other S-boxes have zero input differences i.e. $\widetilde{IN}_1^{[23,\dots,0]} = \widetilde{IN}_2^{[23,\dots,0]}$.

Let

$$z = S_3(\widetilde{IN}_1^{[15,\dots,8]}) \boxminus S_2(\widetilde{IN}_1^{[23,\dots,16]}),$$

then (2) gives

$$OUT_i := (S_1(\widetilde{IN}_i^{[31,\dots,24]}) \boxplus z) \oplus S_4(\widetilde{IN}_1^{[7,\dots,0]});\ i = 1, 2.$$

It was estimated that two input pairs into S_1 (out of 256 possible) with difference 29_x : $(17_x, 3E_x)$ and $(3E_x, 17_x)$ cause an output difference $\beta = 60A40_x$. Regarding this case i.e. when $S_1(\widetilde{IN}_1^{[31,\dots,24]}) \oplus S_1(\widetilde{IN}_2^{[31,\dots,24]}) = \beta$ we obtain that

$$(S_1(\widetilde{IN}_1^{[31,\dots,24]}) \boxplus z) \oplus (S_1(\widetilde{IN}_2^{[31,\dots,24]}) \boxplus z) = \beta$$

with probability 2^{-5} because hamming weight of β is 5.

Xor with the output of S_4 preserves the difference. So assuming that K_r is known input difference $29_x^{\lll 24-K_r}$ provides output difference β with probability 2^{-12}. As K_r is not known it can be done the following: let n be chosen at random from $\{0, 1, 2, ..., 2^{-5} - 1\}$ and

$$\alpha = 29_x^{\lll n}, \tag{3}$$

then $\alpha = 29_x^{\lll 24-K_r}$ with probability 2^{-5} and output difference of F^2-function is β with probability 2^{-17}.

Now consider $\mathcal{A}^2$-round with input difference $(0,0,\beta,\alpha)$. The F^2-function output difference β is xored with the third word of input difference into $\mathcal{A}^2$-round and we obtain zero. So the characteristic which covers one $\mathcal{A}^2$-round is

$$(0,0,\beta,\alpha) \xrightarrow[p=2^{-17}]{\mathcal{A}^2-round} (\alpha,0,0,0). \qquad (4)$$

C. Concatenation of the Characteristics

Looking at the structure of the CAST-256 rounds it is easy to derive a 2-round characteristic

$$(\beta,\alpha,0,0) \xrightarrow[p=1]{2\ \mathcal{A}-rounds} (0,0,\beta,\alpha)$$

and another 15-round truncated characteristic, is shown in the Table III.

TABLE III
THE 15-ROUND TRUNCATED CHARACTERISTIC WITH PROBABILITY 1

	α	0	0	0
3 $\mathcal{A}$-rounds	0	α	0	0
	0	0	α	0
	0	0	0	α
12 $\mathcal{B}$-rounds	0	0	α	0
	0	α	0	0
	α	0	0	0
	0	0	?	α
	0	?	α	0
	?	α	0	0
	α	0	?	?
	0	?	?	α
	?	?	α	?
	?	α	?	?
	α	?	?	?
	?	?	?	α

Concatenation of this characteristics with (4) gives a 18-round characteristic

$$(\beta,\alpha,0,0) \xrightarrow[p=1]{2\ A-rounds} (0,0,\beta,\alpha) \xrightarrow[p=2^{-17}]{A^2-round} \qquad (5)$$

$$(\alpha,0,0,0) \xrightarrow[p=1]{15\ rounds} (?,?,?,\alpha).$$

In order to preserve the initial order of the rounds (1) in this characteristic we can suggest for example such sequence: 4-rd — 9-th $\mathcal{A}$-rounds plus 1-st — 12-th $\mathcal{B}$-rounds.

1-round characteristics analogous to (4) can be obtained for F^1 and F^3, but their probability depends on K_m because in these functions IN is not xored but added or subtracted from it. Nevertheless, they can be used to construct 18-round characteristics and to attack CAST-256 with 24 rounds but with greater complexity or the less success probability.

IV. The Key-Recovery Attack on 24 Rounds of CAST-256

A. The Main Part of the Attack

Let us regard the six 37-bit subkeys of the rounds 19–24 as one 222-bit subkey. The main part of the attack is devoted to recovering this subkey. Let $X = \mathcal{CAST}(A)$ be the encryption with 24 rounds of CAST-256. The attack works like a 19-layer filter. All possible 2^{222} subkeys are passed through this filter, which possesses a crucial feature: the right subkey can pass all the 19 layers and the wrong subkey can not. The attack proceeds in several steps (see Fig. 2 for a pseudocode):

1) Take 19 batches of 2^{20} different plaintext pairs A_i^b, B_i^b; $(b = 1,...,19;\ i = 1,...,2^{20})$ with differences $A_i^b \oplus B_i^b = (\beta, \alpha_i^b, 0, 0)$ (every α_i^b is chosen according to (3)).
2) For each pair A_i^b, B_i^b request the pair of ciphertexts $X_i^b = \mathcal{CAST}(A_i^b)$ and $Y_i^b = \mathcal{CAST}(B_i^b)$, the obtained $19 * 2^{20}$ ciphertext pairs and α_i^b store in the memory.
3) Guess all possible subkeys and for each subkey $sk = 0,...,2^{222}-1$ do the following:
   a) $b := 1$;
   b) partially (by six last rounds with the subkey sk) decrypt the stored ciphertext pairs from the group b and get pairs P_i^b and Q_i^b;
   c) if $P_i^b \oplus Q_i^b \neq (?,?,?,\alpha_i^b)$ for all $i = 1,...,2^{20}$ then sk is the wrong subkey, discard it and goto (3) taking the next subkey;
   if at least one pair provides $P_i^b \oplus Q_i^b = (?,?,?,\alpha_i^b)$ then goto (d);
   d) if $b < 19$ then $b := b+1$ and goto (b); else goto (e);
   e) it means that $b = 19$ and the subkey have passed 19 layers. It is the right subkey.

```
for b:=1 to 19 do
    for i:=1 to $2^{20}$ do
        $A = random(0,...,2^{128}-1)$;
        $n = random(0,...,2^5-1)$;
        $\alpha_i^b = 29_x^{\lll n}$;
        $X_i^b = \mathcal{CAST}(A)$;
        $Y_i^b = \mathcal{CAST}(A \oplus (\beta, \alpha_i^b, 0, 0))$;

for sk:=0 to $2^{222}-1$ do
    for b:=1 to 19 do
        flag:=false;
        for i:=1 to $2^{20}$ do
            $P \xleftarrow{6\ \mathcal{B}-rounds} X_i^b$;
            $Q \xleftarrow{6\ \mathcal{B}-rounds} Y_i^b$;
            if $P \oplus Q = (?,?,?,\alpha_i^b)$ then
                flag:=true; break;
        if flag=false then break;
        else if b=19 then $sk$ is the right key;

$Q \xleftarrow{6\ \mathcal{B}-rounds} Y_i^b$ — partial decryption of $Y_i^b$
                                  by 6 $\mathcal{B}$-rounds with $sk$
```

Fig. 2. Key-recovery attack

B. Complexity, Success Probability and the Full Attack

The attack is finished with a *success* if the right 222-bit subkey was found. It means that all the wrong subkey guesses were discarded and only the right guess survived. Let us estimate the success probability of the attack.

Since the pair X_i^b, Y_i^b partially decrypted with the wrong subkey doesn't satisfy the differential

$$(\beta, \alpha_i^b, 0, 0) \xrightarrow[p=2^{-17}]{18\ rounds} (\alpha_i^b, ?, ?, ?),$$

the difference $P_i^b \oplus Q_i^b$ can get any value (including $(\alpha_i^b, ?, ?, ?)$) uniformly, i.e. with probability 2^{-32}. Hence, the probability to obtain such difference among 2^{20} pairs is 2^{-12} and the probability to obtain such difference in all the 19 batches is 2^{-228}. Thus even one from $2^{222} - 1$ wrong subkeys will survive with probability about 2^{-6}.

The pair X_i^b, Y_i^b partially decrypted with the right subkey satisfies the named differential, so $P_i^b \oplus Q_i^b = (\alpha_i^b, ?, ?, ?)$ with probabilty 2^{-17} and the probability to get such difference among 2^{20} pairs is 0.999665 (here a well-known Poisson distribution is used). The probability to get such difference in all the 19 batches is 0.9936.

After the 222-bit subkey is established, the last 6 rounds are peeled off. The remaining 37-bit subkeys can be revealed analogously using the characteristic (5). The guesses in these cases are 2^{37} instead of 2^{222} so the complexity is negligible if compared with the described attack. The success probability of the whole attack which recovers all the subkeys surely is over than 0.9.

Looking at the Fig. 2 the complexity of the attack can be easily estimated. It needs $2^{222} * 19 * 2^{20} * 2 \approx 2^{247}$ partial 6-round decryptions it is equivalent to 2^{244} full CAST-256 decryptions (or encryptions). Also it needs about 2^{24} chosen plaintexts and 2^{29} bytes of memory to store them and α_i^b.

V. Conclusion

In this paper the differential attack which is capable to break 24 rounds of CAST-256 was introduced. The attack is based on the truncated differential characteristic where the starting difference is chosen with some randomization but not in a convenient way.

References

[1] Seki H., Kaneko T. Differential Cryptanalysis of CAST-256 Reduced to Nine Quad-Rounds // IEICE Trans. Fundam. Electron. Commun. Comput. Sci. – 2001. – Vol. E84-A, N. 4, pp. 913-918.

[2] "Advanced encryption standard (AES) project," 1997-2000, http://csrs.nist.gov/encryption/aes.

[3] B. Schneier, "A Self-study course in block-cipher cryptanalysis," *Cryptologia,* Vol. 24, No. 1, pp. 18-34, 2000.

[4] C. Adams, "The CAST-256 Encryption Algorithm," *AES submission,* 1998, http://www.mirrors.wiretapped.net/security/cryptography/algorithms/aes-testing/cast/cast-256.pdf.

[5] D. Wagner, "The boomerang Attack," *in Fast Software Encryption'99, Lecture Notes in Computer Science, Springer-Verlag,* vol. 1636, pp. 156-170, 1999.

[6] E. Biham and A. Shamir, "Differential cryptanalysis of DES-like cryptosystems," *Journal of Cryptology,* vol. 4, pp. 3-72, 1991.

[7] L. R. Knudsen, "Truncated and higher order differentials," *in Fast Software Encryption'94, Lecture Notes in Computer Science, Springer-Verlag,* vol. 1008, pp. 196-211, 1995.

[8] A. Biryukov and E. Kushilevitz, "Improved cryptanalysis of RC5," *in Eurocrypt'98, Lecture Notes in Computer Science, Springer-Verlag,* vol. 1403, pp. 85-99, 1998.