Cryptanalysis of the Authentication in ACORN

Tairong Shi* and Jie Guan
Information Science and Technology Institute
Zhengzhou, Henan - China
[e-mail: strwanzi@163.com]
*Corresponding author: Tairong Shi

Received October 1, 2017; revised December 18, 2017; accepted September 28, 2018; published August 31, 2019

Abstract

ACORN is an authenticated encryption algorithm proposed as a candidate in the currently ongoing CAESAR competition. ACORN has a good performance on security and efficiency which has been a third-round candidate. This paper mainly concentrates on the security of ACORN under the forgery attack and the non-repudiation of ACORN. Firstly, we analyze the differential properties of the feedback function in ACRON are analyzed. By taking advantage of these properties, the forgery attacks on round-reduced ACORN are proposed with a success probability higher than 2^{-128} when the number of finalization rounds is less than 87. Moreover, the non-repudiation of ACRON in the nonce-reuse setting is analyzed. The known collision can be used to deny the authenticated message with probability 2^{-120}. This paper demonstrates that ACORN cannot generate the non-repudiation completely. We believe it is an undesirable property indeed.

Keywords: CAESAR competition, Authenticated encryption algorithm, ACORN, Forgery attack, Non-repudiation

This research was supported by the National Natural Science Foundation of China (Grant No.61572516, No.61602514).

http://doi.org/10.3837/tiis.2019.08.013 **ISSN : 1976-7277**

1. Introduction

Authenticated encryption (AE) [1] provides both confidentiality and authenticity. In order to select AE algorithms with strong security and superior performance exceeding the current standard AES-GCM, the CAESAR competition [2] was launched. CAESAR is similar with AES [3] and SHA-3 hash [4] algorithm competitions, which were supported by the National Institute of Science and Technology(NIST). Authenticated encryption algorithms can be used in many scenarios, such as Internet of Things (IoT) [5][6][7].There are firstly 57 proposals submitted to the first round of the CAESAR competition in 2014. In August 2016, fifteen successful candidates for the third round were announced, then the security of them attracts many researchers [8][9][10][11].

ACORN v3 [12], a lightweight authenticated stream cipher proposed by Wu, is one of the 15 third-round submissions. The structure of ACORN consists of six binary linear feedback shift registers (LFSRs) with lengths of 61, 46, 47, 39, 37 and 59 respectively, and an additional 4-bit register. ACORN uses a 128-bit key and a 128-bit initialization vector, and the state size of internal state is 293-bit. There are three Boolean functions in ACORN: one is to generate key stream bit, one is to compute the overall feedback bit, and the other one is to update the state. It should be noted that the following three requirements should be satisfied when using ACORN.

1. Each key should be generated in a random and secure way.

2. Nonce should not be reused, namely, each key and *IV* pair should not be used to protect more than one message; and each key and *IV* pair should not be used with two different tag sizes.

3. If verification fails, the decrypted plaintext and the wrong tag should not be released.

Related work. Liu et al. [13] analyzed the slid properties of keys and IVs in ACORN v1. Furthermore, they proposed state recovery attacks using guess-and-determine and differential algebraic techniques. The time complexities of the attacks are approximately 2^{180} and 2^{130} CPU cycles. Chaigneau et al. [14] explored key-recovery attack on ACORN v1 in nonce-reuse and decryption-misuse settings. Salam et al. [15] showed the existence of state collision attacks on ACORN v1 when the secret key is known. The same attacks in [15] can be extended to ACORN v2. Salam et al. [16] also investigated cube attacks on ACORN v1. Wang et al.[17] recover the state of ACORN in nonce-reuse attack.The cube attack on 477 initialization rounds of ACORN v1 can recover the 128-bit key with a complexity of 2^{35}. Jiao et al. [18] evaluated the security of ACORN v1 by using linear approximations, which needs about 2^{157} tests. Recently, a sat-based cryptanalysis of ACORN v3 is given by Lafitte [19]. Zhang et al.[20] studied the fault attack on ACORNv2. Siddhanti et al.[21] presented the implication of such results towards mounting TMDTO attack on ACORN v3.

In general, the majority of presented attacks that threaten ACORN are in nonce-reuse and misuse settings.There may be three reasons why we continue to study the security of ACORN in nonce-reuse setting.

First, as the users, the nonce should not be reused in the practical application when we need strong security. But when ACORN is used in various cryptosystems, we cannot assure every user use ACORN correctly in non-reuse setting. Then, as the designers, the security of the authentication encryption algorithm should be analyzed comprehensively. And in the drafts of other authentication encryption algorithms, the designers evaluated the security of these

algorithms in nonce-reuse setting. Although the nonce should not be reused, the designers do not ignore the threat of nonce-reuse. Last, as the other cryptographers, they want to break the authentication encryption algorithms and any information leakage would be the key to break these algorithms. So they would analyze the security of these algorithms in any setting.

In this paper, we focus on the latest version of ACORN, ACORN v3. Without specification, ACORN in the rest of paragraphs refers to ACORN v3. We discuss the properties of cipher against forgery attack in nonce-respect setting and non-repudiation in nonce-reuse setting respectively. In the former case, This part of work is based on our previous research on state collisions of ACORN [22], and forgery attacks can be mounted with a success probability higher than 2^{-128} when the number of finalization rounds less than 87. However, the full version of ACORN has 792 rounds, thus we detect that the authenticated process in ACORN has a good performance on resistance against forgery attack. In the latter case, we study the non-repudiation of ACORN by presenting how to repudiate concretely based on the available collision.

The known attacks on ACORN are shown in **Table 1**.

Table 1. The comparisons of attack results on ACORN

Version	Attack type	Attack complexity	Reference
ACORN v1	Guess-and-determine	2^{180} in single (K,IV)	[13]
ACORN v1	Differential-algebraic	2^{130} in two related (K,IV) pairs.	[13]
ACORN v1	Key-recovery attack in nonce-reuse setting	2^{109} for 4 chosen plaintexts	[14]
ACORN v1 and v2	Collision attack when secret key is known	An exhaustive search of 2^{286} over the internal state	[15]
ACORN v1 ACORN v2	Cube Attack in nonce-reuse setting	$2^{72.8}$	[16]
ACORN v1	State recovery attack in nonce-reuse setting	$2^{126.55}$	[17]
ACORN v1	State recovery attack	2^{157}	[18]
ACORN v2	Fault attack	$2^{179.19-1.76N}$, where N is the number of fault injections	[20]
ACORN v3	TMDTO attack	2^{171} and 2^{180} (without and with the help of a SAT solve)	[21]
ACORN v3	Forgery attack	Below 2^{128} when the number of finalization rounds is less than 87	This paper
ACORN v3	Repudiation	2^{120}	This paper

This paper is organized as follows. Section 2 provides a brief description of ACORN. In Section 3, we evaluate the security of ACORN against forgery attack in nonce-respect setting. We discuss the non-repudiation of ACORN in nonce-reuse setting in Section 5, followed by a conclusion in Section 6.

2. Description of ACORN

ACORN performs in five phases: initialization, processing the associated date, encryption, tag generation and decryption&verification. This section gives the three functions used in ACORN and then each phase. The cipher executes as **Fig. 1**.

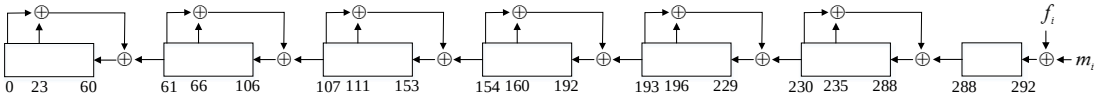

Fig. 1. Detailed representation of the internal state of ACORN

Output keystream generation function. Let $S_i = (s_{i,0}, s_{i,1}, \cdots, s_{i,292})$ indicate the internal state in the i-th step. ACORN generates the keystream ks_i from state S_i as follows $ks_i = KSG(S_i)$:

$$ks_i = S_{i,12} \oplus S_{i,154} \oplus maj(S_{i,235}, S_{i,61}, S_{i,193}) \oplus ch(S_{i,230}, S_{i,111}, S_{i,66}) \tag{1}$$

where the Boolean function maj is defined by

$$maj(x_1, x_2, x_3) = x_1x_2 \oplus x_1x_3 \oplus x_2x_3 \tag{2}$$

Nonlinear feedback function. The feedback bit f_i is computed as $f_i = FBK(S_i, ca_i, cb_i)$

$$f_i = S_{i,0} \oplus \overline{S_{i,107}} \oplus maj(S_{i,244}, S_{i,23}, S_{i,160}) \oplus ca_i \,\&\, S_{i,196} \oplus cb_i \,\&\, ks_i \tag{3}$$

where $\overline{S_{i,107}}$ denotes the complement value of $S_{i,107}$, ca_i and cb_i represent two control bits used in different progresses. The Boolean function ch is defined by

$$ch(x_1, x_2, x_3) = x_1x_2 \oplus \overline{x_1}x_3 \tag{4}$$

State update function. $S_{i+1} = Stateupdate(S_i, m_i, ca_i, cb_i)$. The state is updated as the following pseudo-code.

$S_{i,289} = S_{i,289} \oplus S_{i,235} \oplus S_{i,230}$;
$S_{i,230} = S_{i,230} \oplus S_{i,196} \oplus S_{i,193}$;
$S_{i,193} = S_{i,193} \oplus S_{i,160} \oplus S_{i,154}$;
$S_{i,154} = S_{i,154} \oplus S_{i,111} \oplus S_{i,107}$;
$S_{i,107} = S_{i,107} \oplus S_{i,66} \oplus S_{i,61}$;
$S_{i,61} = S_{i,61} \oplus S_{i,23} \oplus S_{i,0}$;
$f_i = FBK(S_i, ca_i, cb_i)$
For $j = 0$ to 291
    $S_{i+1,j} = S_{i,j+1}$;
End For
$S_{i+1,292} = f_i \oplus m_i$

2.1 The initialization of ACORN

During the initialization, ACORN loads the key and IV into the internal state, and runs for 1792 rounds. This progress is described by the following pseudo-code.

$S^0 = (0, 0, \cdots, 0)$
For $i = 0$ to 127
    $(ca_i, cb_i) = (1,1)$
    $m_i = k_i$
    $S_{i+1} = Stateupdate(S_i, m_i, ca_i, cb_i)$;
End for
For $i = 128$ to 255
    $(ca_i, cb_i) = (1,1)$
    $m_i = iv_i$
    $S_{i+1} = Stateupdate(S_i, m_i, ca_i, cb_i)$;
End for
$m_{256} = k_0 \oplus 1$;
$S_{i+1} = Stateupdate(S_i, m_i, ca_i, cb_i)$;
For $i = 257$ to 1791
    $(ca_i, cb_i) = (1,1)$
    $m_i = k_{i \bmod 128}$;
    $S_{i+1} = Stateupdate(S_i, m_i, ca_i, cb_i)$;
End for

2.2 Processing the associated data

After the initialization, the associated data $AD = (ad_0, \cdots, ad_{adlen-1})$ is used to update the state as shown in the following, where $adlen$ denotes the bit length of the associated data.

For $i = 0$ to $adlen - 1$
    $m_i = ad_i$
    $(ca_i, cb_i) = (1,1)$
    $S_{i+1} = Stateupdate(S_i, m_i, ca_i, cb_i)$;
End for
$m_{adlen} = 1$;
$S_{i+1} = Stateupdate(S_i, m_i, ca_i, cb_i)$;
For $i = adlen + 1$ to $adlen + 127$
    $m_{adlen} = 0$;
    $(ca_i, cb_i) = (1,0)$;
    $S_{i+1} = Stateupdate(S_i, m_i, ca_i, cb_i)$;
End for
For $i = adlen + 128$ to $adlen + 255$
    $m_{adlen} = 0$;
    $(ca_i, cb_i) = (0,0)$
    $S_{i+1} = Stateupdate(S_i, m_i, ca_i, cb_i)$;
End for

2.3 The Encryption

At each step of encryption, the plaintext bit p_i is used to update the cipher, and p_i is encrypted by XOR key stream bit ks_i. Let pl denote the bit length of plaintext.

$(m_0, m_1, \cdots, m_{pl+255}) = (p_0, p_1, \cdots, p_{pl-1}, 1, 0, \cdots 0)$

For $i = 0$ to $pl - 1$

$(ca_i, cb_i) = (1, 0)$

$S_{i+1} = Stateupdate(S_i, m_i, ca_i, cb_i);$

$c_i = p_i \oplus KSG(S_i)$

End for

For $i = pl$ to $pl + 127$

$(ca_i, cb_i) = (1, 0)$

$S_{i+1} = Stateupdate(S_i, m_i, ca_i, cb_i);$

End for

For $pl + 128$ to $pl + 255$

$(ca_i, cb_i) = (0, 0)$

$S_{i+1} = Stateupdate(S_i, m_i, ca_i, cb_i);$

End for

2.4 The Finalization

The authentication tag T is generated in the finalization, which is described by the following pseudo-code.

For $i = 0$ *to* 767

$m_i = 0;$

$(ca_i, cb_i) = (1, 1);$

$S_{i+1} = Stateupdate(S_i, m_i, ca_i, cb_i);$

End for

The tag is the last t key stream bits, as follows:

$$Tag = ks_{767-t+1} \| ks_{767-t+2} \| \cdots \| ks_{767} \tag{5}$$

The decryption and verification are very similar to the encryption and tag generation. So we leave out the description of them here.

3. Forgery Attack on round-reduced ACORN

This section provides a description of the forgery attack on the reduced-round ACORN. In nonce-respect setting, we are not allowed to modify the plaintext. Consequently, a common approach is to modify the ciphertext and corresponding tag, then forge in decryption and verification. Our attacks mainly depend on the differential technique.

The main issue affecting differential charateristic is nonlinear function. We first briefly show the differential properties of two nonlinear functions (*maj* and *ch*). Then we give the differential characteristics in the encryption and tag generation.

3.1 Differential properties of *maj* and *ch*

This part of work is based on our previous research on state collisions of ACORN [22]. The *maj* and *ch* are two nonlinear Boolean functions involved in the feedback function. Below, we present the differential properties of them.

For $maj(x_1, x_2, x_3)$,

$$maj(x_1, x_2, x_3) = x_1x_2 \oplus x_1x_3 \oplus x_2x_3$$

Let $(\Delta_1, \Delta_2, \Delta_3)$ denote the input difference of $maj(x_1, x_2, x_3)$, and the output difference of $maj(x_1, x_2, x_3)$ can be describe as

$$\begin{aligned} &maj(x_1 \oplus \Delta_1, x_2 \oplus \Delta_2, x_3 \oplus \Delta_3) \\ &= \Delta_1\Delta_2 \oplus \Delta_1x_2 \oplus \Delta_2x_1 \oplus \Delta_1\Delta_3 \oplus \Delta_1x_3 \oplus \Delta_3x_1 \oplus \Delta_2\Delta_3 \oplus \Delta_2x_3 \oplus \Delta_3x_2 \end{aligned} \tag{6}$$

If $(\Delta_1, \Delta_2, \Delta_3) = (0,0,0)$, the output difference of $maj(x_1, x_2, x_3)$ is 0 with probability 1;

If $(\Delta_1, \Delta_2, \Delta_3) = (1,1,1)$, the output difference of $maj(x_1, x_2, x_3)$ is 1 with probability 1;

Otherwise, if there is any other possible values of $(\Delta_1, \Delta_2, \Delta_3)$, due to some of the variables x_1, x_2, x_3 exist in the output difference and they are random and independent from each other, the output difference of $maj(x_1, x_2, x_3)$ is 0 or 1 with probability 0.5.

For $ch(x_1, x_2, x_3)$

$$ch(x_1, x_2, x_3) = x_1x_2 \oplus \overline{x_1}x_3$$

Let $(\Delta_1, \Delta_2, \Delta_3)$ be the input difference of $ch(x_1, x_2, x_3)$, and the output difference of $ch(x_1, x_2, x_3)$ can be describe as

$$ch(x_1 \oplus \Delta_1, x_2 \oplus \Delta_2, x_3 \oplus \Delta_3) = \Delta_1\Delta_2 \oplus \Delta_1x_2 \oplus \Delta_2x_1 \oplus \Delta_3 \oplus x_1\Delta_3 \oplus x_3\Delta_1 \oplus \Delta_1\Delta_3 \tag{7}$$

If $(\Delta_1, \Delta_2, \Delta_3) = (0,0,0)$, the output difference of $maj(x_1, x_2, x_3)$ is 0 with probability 1;

If $(\Delta_1, \Delta_2, \Delta_3) = (0,1,1)$, the output difference of $maj(x_1, x_2, x_3)$ is 1 with probability 1;

Otherwise, if there is any other possible values of $(\Delta_1, \Delta_2, \Delta_3)$, due to some of the variables x_1, x_2, x_3 exist in the output difference and they are random and independent from each other, the output difference of $ch(x_1, x_2, x_3)$ is 0 or 1 with probability 0.5.

The differential properties of two functions shown in **Table 2**.

Table 2. The differential properties of functions *maj* and *ch*

Function	$(\Delta_1, \Delta_2, \Delta_3)$	Output difference	Prpbability
maj(*x*, *y*, *z*)	(0,0,0)	0	1
	(1,1,1)	1	1
	Other values	0 or 1	0.5
ch(*x*, *y*, *z*)	(0,0,0)	0	1
	(0,1,1)	1	1
	Other values	0 or 1	0.5

Moreover, the differential characteristics in encryption and finalization will be presented as below.

3.2 Differential property of ACORN

As a valid forgery attack on ACORN, the success probability must be higher than 2^{-128}. Let $(\Delta_0, \Delta_{10}, \cdots, \Delta_{292})$ denote the input difference of internal state, Δf_i denote the output difference of f_i. Next, we propose the differential properties of state update function in decryption based on the differential properties of the nonlinear functions in ACORN. Because the values of the parameters (ca_i, cb_i) in decryption and verification are different, we set it into three phases to introduce the differential rules of f_i.

$$f_i = S_{i,0} \oplus \overline{S_{i,107}} \oplus maj(S_{i,244}, S_{i,23}, S_{i,160}) \oplus ca_i \,\&\, S_{i,196} \oplus cb_i \,\&\, ks_i \tag{8}$$

Phase 1: During the first 128 steps after encrypting the last ciphertext bit, the control parameters $(ca_i, cb_i) = (1,0)$. Thus the feedback function is described as $f_i = S_{i,0} \oplus \overline{S_{i,107}} \oplus maj(S_{i,244}, S_{i,23}, S_{i,160}) \oplus S_{i,196}$, and the rules used in differential deduction are given as follows:

If $(\Delta_{244}, \Delta_{23}, \Delta_{160}) = (0,0,0)$, then $\Delta f_i = \Delta_0 \oplus \Delta_{107} \oplus \Delta_{196}$;

If $(\Delta_{244}, \Delta_{23}, \Delta_{160}) = (1,1,1)$, then $\Delta f_i = \Delta_0 \oplus \Delta_{107} \oplus \Delta_{196} \oplus 1$;

Otherwise, $\Delta f_i = 0$.

Phase 2: During the second 128 steps after encrypting the last ciphertext bit, the control parameters $(ca_i, cb_i) = (0,0)$. Thus the feedback function is described as $f_i = S_{i,0} \oplus \overline{S_{i,107}} \oplus maj(S_{i,244}, S_{i,23}, S_{i,160})$, and the rules used in differential deduction are given as follows:

If $(\Delta_{244}, \Delta_{23}, \Delta_{160}) = (0,0,0)$, then $\Delta f_i = \Delta_0 \oplus \Delta_{107}$;

If $(\Delta_{244}, \Delta_{23}, \Delta_{160}) = (1,1,1)$, then $\Delta f_i = \Delta_0 \oplus \Delta_{107} \oplus 1$;

Otherwise, $\Delta f_i = 0$.

Phase 3: In the first 128 steps after encrypting the last ciphertext, the control parameters $(ca_i, cb_i) = (1,1)$. Thus the feedback function is described as

$$\begin{aligned} f_i = {} & S_{i,0} \oplus \overline{S_{i,107}} \oplus maj(S_{i,244}, S_{i,23}, S_{i,160}) \\ & \oplus S_{i,196} \oplus S_{i,12} \oplus S_{i,154} \oplus maj(S_{i,235}, S_{i,61}, S_{i,193}) \oplus ch(S_{i,230}, S_{i,111}, S_{i,66}) \end{aligned} \tag{9}$$

For *maj* and *ch*, there are two kinds of input difference leading to the output difference with probability 1. Otherwise, if there is any difference in the input, the output difference is 0/1 with probability 0.5. There are two *maj* functions and one *ch* function involved, so there are eight out of 2^9 input differences whose corresponding output differences are fixed with probability 1, others with probability 0.5. As a result, the rules used in differential deduction are given as follows:

If $(\Delta_{244}, \Delta_{23}, \Delta_{160}) = (0,0,0)$ or $(1,1,1)$, $(\Delta_{235}, \Delta_{61}, \Delta_{193}) = (0,0,0)$ or $(1,1,1)$, and $(\Delta_{230}, \Delta_{111}, \Delta_{66}) = (0,0,0)$ or $(0,1,1)$, then

$$\Delta f_i = \Delta_0 \oplus \Delta_{107} \oplus \Delta_{244} \oplus \Delta_{196} \oplus \Delta_{12} \oplus \Delta_{154} \oplus \Delta_{235} \oplus \Delta_{111};$$

Otherwise, $\Delta f_i = 0$.

After encrypting the last ciphertext bit, there is no message participate in the upstate function, so $\Delta_{292} = \Delta f_i \oplus \Delta m_i = \Delta f_i$. **Algorithm 1** provides a pseudo-code of the differential deduction.

Algorithm 1. Algorithm of differential deduction in decryption and verification

Set $counter = 0$;
For $i = 0$ to $N-1$ //N must greater than 256
$\Delta_{289} = \Delta_{289} \oplus \Delta_{235} \oplus \Delta_{230}$;
$\Delta_{230} = \Delta_{230} \oplus \Delta_{196} \oplus \Delta_{193}$;
$\Delta_{193} = \Delta_{193} \oplus \Delta_{160} \oplus \Delta_{154}$;
$\Delta_{154} = \Delta_{154} \oplus \Delta_{111} \oplus \Delta_{107}$;
$\Delta_{107} = \Delta_{107} \oplus \Delta_{66} \oplus \Delta_{61}$;
$\Delta_{61} = \Delta_{61} \oplus \Delta_{23} \oplus \Delta_0$;
When $i < 128$ //phase 1
If $(\Delta_{244}, \Delta_{23}, \Delta_{160}) = (0,0,0)$ $\Delta f = \Delta_0 \oplus \Delta_{107} \oplus \Delta_{196}$;
If $(\Delta_{244}, \Delta_{23}, \Delta_{160}) = (1,1,1)$ $\Delta f = \Delta_0 \oplus \Delta_{107} \oplus \Delta_{196} \oplus 1$;
Else $\Delta f = 0$ $counter++$.
When $128 \le i < 256$ //phase 2
If $(\Delta_{244}, \Delta_{23}, \Delta_{160}) = (0,0,0)$ $\Delta f = \Delta_0 \oplus \Delta_{107}$;
If $(\Delta_{244}, \Delta_{23}, \Delta_{160}) = (1,1,1)$ $\Delta f = \Delta_0 \oplus \Delta_{107} \oplus 1$;
Else $\Delta f = 0$ $counter++$.
When $256 \le i < N$ //phase 3
If $((\Delta_{244}, \Delta_{23}, \Delta_{160}) = (0,0,0) \,||\, (1,1,1)) \,\&\&\, ((\Delta_{235}, \Delta_{61}, \Delta_{193}) = (0,0,0) \,||\, (1,1,1))$
$\&\&\, ((\Delta_{230}, \Delta_{111}, \Delta_{66}) = (0,0,0) \,||\, (0,1,1))$
$\Delta f_i = \Delta_0 \oplus \Delta_{107} \oplus \Delta_{244} \oplus \Delta_{196} \oplus \Delta_{12} \oplus \Delta_{154} \oplus \Delta_{235} \oplus \Delta_{111}$;
Else $\Delta f = 0$, $counter++$.
For $j = 0$ to 291
$\Delta_j = \Delta_{j+1}$;
End for
$\Delta_{292} = \Delta f$;
End for
Set $n \leftarrow counter$.

From the above algorithm, we are allowed to launch attack on arbitrary rounds and gain the differential characteristics, where the probability is 2^{-n}.

Our attacks are searching for a differential chain as long as possible with the probability no less than 2^{-128}. We introduce various weights (Hamming weight) of input differences on ciphertext, including 1 bit, 2 bits, 3 bits, and calculate the probability respectively. The following **Table 3** lists various probabilities of a forgery in different weights of input differences.

Table 3. Probabilities for a forgery in various Hamming weights of input differences

Hamming weight	Round	Probability
1	86	2^{-127}
1	87	2^{-128}
2	70	2^{-127}
2	71	2^{-128}
3	59	2^{-127}
3	60	2^{-128}

Note that our attacks are searching for a differential chain as long as possible with the probability no less than 2^{-128}, so we only list the results whose probabilities around the boundary 2^{-128}. We illustrate **Table 3** using an example. When the weight of the input difference is 2 and the finalization round is less than 70, the probability of a forgery is higher than 2^{-127}. When the weight of the input difference is 2 and round is 70, the probability of a forgery is 2^{-127}.

Experimental results show that if a difference is in the last bit of ciphertext, a differential characteristic with probability 2^{-127} covering 256-round decryption and 86-round verification is available. The details of the results are given in **Appendix.** As a consequence, we can launch forgery attacks on cipher whose finalization rounds are less than 87 with a probability higher than 2^{-128}.

Meanwhile, we analyzed the cause why the round in our attack is limited. The reason mainly consists of 3 parts as below:

- The empty running rounds after each phase.

  After encryption, the cipher runs for another 256 steps. In fact, the empty rounds also exist after the initialization and the associated data loading process. It makes the difference must through more rounds.
- Alternate setting of control parameters (ca_i, cb_i) in encryption and finalization.

  It can be known from the former analysis that we set it into three phases to introduce the differential rules. In another hand, control parameters separate the finalization from the ciphertext means preventing using part of the keystream (derived from ciphertext) as the tag.
- Nonlinear transforms participating in each upstate round.

The nonlinear transforms lead the differential probability drop swiftly. Thus the cipher has a good performance on differential properties, and then resists forgery attack.

4. The non-repudiation of ACORN in nonce-reuse setting

Non-repudiation [23] of sender is mainly used to prevent the originator from denying the sent message. As an authentication encryption algorithm which may be used widely in future, the

practical security of ACORN should be evaluated carefully and it is necessary to analyze the non-repudiation of ACORN. In this section, we evaluate the non-repudiation of ACORN. Firstly, we introduce the details of an internal collision which is used to construct the repudiation. Next, by using the available collision, we can repudiate practically.

If the nonce is reused, an attacker can modify the plaintext during the encryption, and then always introduce difference to eliminate the difference in the internal state. As a result, two different plaintexts have an identical authenticated tag. Designers showed that when input difference (290 bits) Δd is as below, the internal collision occurs with differential probability 2^{-120}.

$$\begin{aligned}\Delta d = \{&1000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0110\ 0111\\ &0101\ 0011\ 0000\ 0010\ 0001\ 0100\ 0001\ 0011\ 1110\ 0010\\ &0011\ 1011\ 0101\ 0110\ 1000\ 1100\ 1011\ 1010\ 1100\ 1011\\ &1010\ 0010\ 1101\ 1111\ 0001\ 0111\ 1101\ 1111\ 0100\ 1011\\ &1001\ 1100\ 1000\ 1110\ 1110\ 0100\ 1110\ 1010\ 0010\ 0100\\ &1110\ 0110\ 0111\ 1111\ 1111\ 1100\ 1000\ 0101\ 1101\ 0000\\ &0011\ 0010\ 1100\ 0000\ 0101\ 1011\ 0000\ 0110\ 1101\ 0000\\ &1011\ 0110\ 01\}\end{aligned}$$

However, attackers could only control the message difference, so we let $\Delta m = \Delta d$. Thus the user has the access to collision.

Considering the limit that nonce should not be reused, this type of forgery attack is not a concern in the design of ACORN. Different from the status of designers, we investigate the non-repudiation of cipher in practical applications.

Data non-repudiation is very important in the secure communication. However, our results disclose a potential weakness of ACORN. Repudiation performs in the following 3 steps.

- Step1: Alice, a malicious user, who masters the secret K can encrypt the message m and send the corresponding (c, tag) to the receiver Bob.
- Step2: Soon after, Bob receives (c, tag) and decrypts it to get m then verities the authenticity of m using tag.
- Step3: In this case, the attacker Alice is able to deny the already authenticated message m, and claims that the untreated message $m' = m \oplus \Delta m$ is the message that has been sent. Thus ACORN algorithm cannot facilitate non-repudiation.

Fig. 2 illustrates the details of repudiation which is generated by Alice.

In fact, allowing the sender to access the secret key and create collisions is an undesirable property. For ACORN, the sender can find a collision for any given nonce and input plaintext, thereby the authenticity component of the authenticated encryption is compromising.

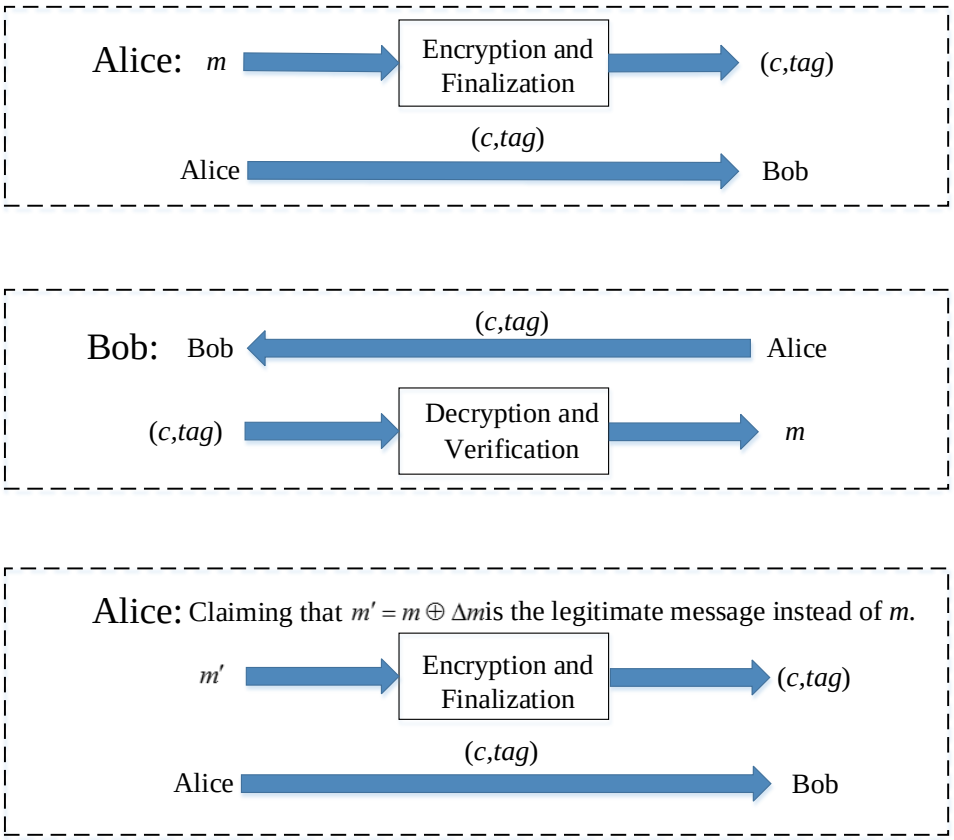

Fig. 2. Detialed representation of repudiation in ACORN

5. Conclusion

This paper discusses the security of authentication in ACORN. Forgery attack on 87-round ACORN is proposed, it shows that the finalization of ACORN has large security margin against forgery attack. Our results do not contradict the security claims by the designer. Furthermore, we present the repudiation existing in ACORN while there is no security claim for ACORN in these settings. However, it is difficult to evaluate the performance of the forgery attack for full version of ACORN since the large scale of algorithm. We cannot realize practical attack in the process of repudiation because of the requirement on data. As a consequence, it may be a potential weakness in practical applications. With the advancement of CAESAR competition, authenticated encryption receives more attentions and develops rapidly; meanwhile, it becomes much harder to break through the third-round candidates. So it is necessary to take the practical security into consideration.

In the future, we will use the similar method to analyze more candidates in the third-round CAESAR competition to find the potential weakness of these authenticated encryption algorithms.

References

[1] M. Bellare and C. Namprempre, "Authenticated encryption: Relations among notions and analysis of the generic composition paradigm," *Journal of Cryptology*, vol. 21, no. 4, pp. 469-491, Oct. 2008. Article (CrossRef Link).

[2] J. Daemen and V. Rijmen, "The Design of Rijndael: AES-the advanced encryption standard," *Springer, Berlin*, 2002. Article (CrossRef Link).

[3] "Caesar: Competition for authenticated encryption: Security, applicability, and robustness," Aug. 2016. Article (CrossRef Link).

[4] "SHA-3 Competition (2007-2012)," Feb. 2005.Article (CrossRef Link).

[5] P. Hu, H. Ning, T. Qiu et al., "Security and Privacy Preservation Scheme of Face Identification and Resolution Framework Using Fog Computing in Internet of Things," *IEEE Internet of Thingsf Journal*, vol. 4, no. 5, pp. 1143-1155, Jan. 2017. Article (CrossRef Link).

[6] Q. Xu, P. Ren, H. Song et al., "Security Enhancement for IoT Communications Exposed to Eavesdroppers with Uncertain Locations," *IEEE Access*, vol. 4, pp. 2840-2853, Jun. 2016. Article (CrossRef Link).

[7] S. Javanmardi, M. Shojafar, S. Shariatmadari et al., "FRTRUST: a fuzzy reputation based model for trust management in semantic P2P grids," *International Journal of Grid & Utility Computing*, vol. 6, no. 1, pp. 57-66, Apr. 2014. Article (CrossRef Link).

[8] C. Dobrauning, M. Eichlseder and F. Mendel, "Heuristic Tool for Linear Cryptanalysis with Applications to CAESAR Candidates," in *Proc. of ASIACRYPT 2014*, pp.490-509, December 7-11, 2014. Article (CrossRef Link).

[9] T. Peyrin, S. M. Sim, L. Wang et al., "Cryptanalysis of JAMBU," in *Proc. of FSE 2015*, pp. 264-281, March 8-11, 2015.

[10] V. T. Hoang, T. Krovetz, and P. Rogaway, "Robust authenticated-encryption: AEZ and the problem that it solves," in *Proc. of EUROCRYPT 2015*, pp. 15–44, April 26-30, 2015. Article (CrossRef Link).

[11] T. Fuhr, G. Leurent and V. Suder, "Collision Attacks against CAESAR Candidates Forgery and Key-Recovery against AEZ and Marble," in *Proc. of ASIACRYPT 2015*, pp. 510-532, November 29 – December 3, 2015. Article (CrossRef Link).

[12] H. J. Wu, "ACORN: A Lightweight Authenticated Cipher (v3)," Aug. 2015.

[13] M. C. Liu and D. D. Lin, "Cryptanalysis of Lightweight Authenticated Cipher ACORN," 2014. Article (CrossRef Link).

[14] C. Colin, F.Thomas, and G. Henri, "Full key-recovery on ACORN in nonce-reuse and decryption misuse settings," 2015. Article (CrossRef Link).

[15] M. I. Salam, K. K. Wong, H. Bartlett et al., "Finding state collisions in the authenticated encryption stream cipher ACORN," in *Proc. of Australasian Computer Science Week Multiconference*, pp. 36-56, February 2-5, 2016.

[16] M. I. Salam , H. Bartlett , E. Dawson et al., "Investigating Cube Attacks on the Authenticated Encryption Stream Cipher ACORN," in *Proc. of Applications and Techniques in Information Security*, pp.15-26, Oct 26-28, 2016.

[17] S. Wang, B. Hu, Y. Liu et al., "Nonce-reuse Attack on Authenticated Cipher ACORN". in *Proc. of AICS 2016*, pp.379-385, September 20-21 2016. Article (CrossRef Link)

[18] L. Jiao, B. Zhang and M. Wang, "Two Generic Methods of Analyzing Stream Ciphers," in *Proc. of ISC 2015*, pp. 379-396, Sep. 9-11, 2015.

[19] F. Lafitte, L. Lerman, O. Markowitch et al., "SAT-based cryptanalysis of ACORN,". Article (CrossRef Link).

[20] X Zhang, X Feng, D Lin, et al, "Fault Attack on the Authenticated Cipher ACORN v2," *Security & Communication Networks*, 2017, 1-16, 2017. Article (CrossRef Link).

[21] Siddhanti A A, Maitra S, Sinha N, "Certain Observations on ACORN v3 and the Implications to TMDTO Attacks," in *Proc. of Security, Privacy, and Applied Cryptography Engineering*, pp. 264-280, Dec. 13-17, 2017. Article (CrossRef Link).

[22] P. Zhang, J. Guan, J. Li et al., "Research on State Collisions of Authenticated Cipher ACORN," in *Proc. of ICSMIM 2015*, pp.459-465, July 23-24, 2016. Article (CrossRef Link).
[23] S. Steve, "Formal analysis of a non-repudiation protocol," in *Proc. of CSFW 1998*, pp. 54-65, June 9-11, 1998. Article (CrossRef Link).

Appendix A

In Section 3, we analyzed the forgery attack on round-reduced ACORN. Our analysis shows that the probability to attack 87-round cipher (256 rounds of encryption and 87 rounds of finalization) is 2^{-128}. **Table 4** gives the success probability at each round that an attacker can launch the forgery attack.

Table 4. Differential probability in each round of ACORN finalization

Round	Probability	Round	Probability
1	2^{-44}	45	2^{-87}
2	2^{-45}	46	2^{-88}
3	2^{-46}	47	2^{-89}
4	2^{-47}	48	2^{-90}
5	2^{-48}	49	2^{-91}
6	2^{-49}	50	2^{-91}
7	2^{-50}	51	2^{-92}
8	2^{-51}	52	2^{-93}
9	2^{-52}	53	2^{-94}
10	2^{-53}	54	2^{-95}
11	2^{-54}	55	2^{-96}
12	2^{-55}	56	2^{-97}
13	2^{-56}	57	2^{-98}
14	2^{-57}	58	2^{-99}
15	2^{-58}	59	2^{-100}
16	2^{-58}	60	2^{-101}
17	2^{-60}	61	2^{-102}
18	2^{-61}	62	2^{-103}
19	2^{-62}	63	2^{-104}
20	2^{-63}	64	2^{-105}
21	2^{-64}	65	2^{-106}
22	2^{-65}	66	2^{-107}
23	2^{-66}	67	2^{-108}

24	2^{-67}	68	2^{-109}
25	2^{-68}	69	2^{-110}
26	2^{-69}	70	2^{-111}
Round	**Probability**	**Round**	**Probability**
27	2^{-70}	71	2^{-112}
28	2^{-71}	72	2^{-113}
29	2^{-72}	73	2^{-114}
30	2^{-73}	74	2^{-115}
31	2^{-73}	75	2^{-116}
32	2^{-74}	76	2^{-117}
33	2^{-75}	77	2^{-118}
34	2^{-76}	78	2^{-119}
35	2^{-77}	79	2^{-120}
36	2^{-78}	80	2^{-121}
37	2^{-79}	81	2^{-122}
38	2^{-80}	82	2^{-123}
39	2^{-81}	83	2^{-124}
40	2^{-82}	84	2^{-125}
41	2^{-83}	85	2^{-126}
42	2^{-84}	85	2^{-127}
43	2^{-85}	87	2^{-128}
44	2^{-86}	≥ 88	$\leq 2^{-128}$

From this table, the forgery attack is invaild when the round of anthentication is more than 87.

Tai-Rong Shi received her master's degree in cryptology from the Information Science and Technology Institute, Zhengzhou, China, in 2018. Now, she is a PhD candidate at the Information Science and Technology Institute, Zhengzhou, China. Her main research interests include cryptology and information security.

Jie Guan received her PhD degree in cryptology from the Information Science and Technology Institute, Zhengzhou, China, in 2004. She is currently a professor at the Information Science and Technology Institute, Zhengzhou, China. Her main subject interest is cryptology, and she teaches information systems, the theory of cryptography, and quantum computation.