Security Analysis of KNOT-AEAD and KNOT-Hash

Wentao Zhang, Tianyou Ding, Chunning Zhou, Fulei Ji

State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
{*zhangwentao, dingtianyou, zhouchunning, jifulei*}*@iie.ac.cn*

Abstract. KNOT is one of the 32 second-round candidates in NIST's lightweight cryptography standardization process. To have a better understanding of the security of KNOT, in this paper, we concentrate on the search of the best differential and linear distinguishers with constraints that can be directly used to mount attacks on KNOT-AEAD and KNOT-Hash. Six attack models for KNOT-AEAD and two attack models for KNOT-Hash are considered. By studying differential/linear trails containing iterative sub-trails, we can efficiently obtain effective difference/linear propagations with constraints with respect to the 6 attack models for each KNOT-AEAD member and the 2 attack models for each KNOT-Hash member. Furthermore, we investigate the accuracy of our new method in two different ways. Firstly, we apply our new method to RECTANGLE, which is an ancestor of the KNOT permutations, and compare the results obtained by our new method with those provided by the designers of RECTANGLE. Secondly, we use MILP modelling method to compute the differential and linear clustering effect of the 256-bit KNOT permutation and compare the results obtained by MILP method with those obtained by our new method. According to these comparative results, we can reasonably infer that the results using our new approach provide a quite accurate security evaluation of KNOT-AEAD and KNOT-Hash. To sum up, based on our results in this paper, considering the data limit under one key, each KNOT-AEAD member has at least 50% security margin against the 6 attack models (especially, the initialization phase has at least 72% security margin); each KNOT-Hash member has at least 80% security margin against the 2 attack models.

Key words: lightweight cryptography, AEAD, hash, cryptanalysis, iterative trails

1 Introduction

The National Institute of Standards and Technology (NIST) is in the process of selecting one or more authenticated encryption and hashing schemes suitable for constrained environments through a public, competition-like process [14]. In February 2019, 57 candidate algorithms were submitted to NIST for consideration. Among these, 56 were accepted as first-round candidates in April 2019, marking the beginning of the first round of the NIST Lightweight Cryptography Standardization Process. In August 2019, NIST announced the 32 second-round candidates, which were selected based on public feedback and internal review of the first-round candidates [15]. The most important criterion of the process is the cryptographic security of the submissions. Therefore, it is very important to analyze the security of the second-round candidates.

KNOT [18] is a family of bit-slice lightweight AEAD (Authenticated Encryption with Associated Data) and hashing algorithms. It is currently one of the 32 second-Round candidates. KNOT uses permutation-based modes [4–6, 9, 1]. The primary member of KNOT-AEAD family and the primary member of KNOT-Hash family both have a state of 256 bits.

Due to small state size, bit-sliced design principle, careful selection of the 4-bit S-box and only 3 rotations as the diffusion layer, KNOT is friendly and efficient for both hardware and software implementations.

Differential cryptanalysis (DC) [3] and linear cryptanalysis (LC) [12] are among the most powerful approaches available for block ciphers, also for AEADs and hash functions. In the design document of KNOT, the designers present a detailed security evaluation of the KNOT permutations against DC and LC. Although distinguishers of the permutations can give insights in the resistance of the AEAD and hash primitives against various cryptanalytic attacks, they usually can not be directly used in an attack. To solve this problem and have a better understanding of the security of KNOT, in this paper, we concentrate on the search of the best differential and linear distinguishers with constraints which can be directly used to mount attacks on KNOT-AEAD and KNOT-Hash.

The security evaluation of the KNOT permutations against DC and LC is achieved by using Matsui's search algorithm and its improvements [13, 2]. However, when it comes to the search of the best differential/linear distinguishers with constraints, the efficiency is not satisfied. Take for example, searching for the best linear trail of the 256-bit KNOT permutation with constraints that all active bits of both the input mask and the output mask are only allowed in the rate part of the state, it takes about 98 hours to get the result for 11 rounds and additional 49 hours for 12 rounds, which shows that we can not obtain desirable results using the same method.

The design of the KNOT permutations inherits the design of RECTANGLE [17]. An observation on the block cipher RECTANGLE is that the best long-round differential and linear trail always contains iterative sub-trails, which motivates us to study differential/linear distinguishers containing iterative sub-trails.

Through repeated attempts, we have developed an efficient algorithm to search for differential/linear distinguishers containing iterative sub-trails. The main idea of our new search method are as follows:

1. By using the algorithm of finding elementary circuits [10] (a circuit is elementary if no vertex but the first and last appears twice), we can find all elementary iterative differential/linear trails for each of the KNOT permutations.
2. Then, by checking all the differential/linear iterative trail within the scope of consideration and connecting them repeatedly, we can efficiently compute the difference propagation probability (or linear propagation correlation) for a given round number and given input/output difference (or input/output mask).
3. Finally, by using Matsui's search algorithm, we can extend the difference (or linear) propagations obtained in the previous step both forward and backward for several rounds.

Using the above new method, we can efficiently obtain effective differential/linear distinguishers with constraints with respect to 6 attack models for each KNOT-AEAD member and 2 attack models for each KNOT-Hash member. Based on our experimental results, each KNOT-AEAD member has at least 50% security margin against the 6 attack models (especially, the initialization phase has at least 72% security margin); each KNOT-Hash member has at least 80% security margin against the 2 attack models.

Furthermore, to verify the accuracy of our results using this new method, we have conducted several comparative experiments. Firstly, we apply our new method to investigate the differential and linear clustering effect for 14-round RECTANGLE, and compare the results obtained by our new method with the RECTANGLE designers' results [17]. Secondly, for several of the best difference (or linear) propagations of the 256-bit KNOT permutation found using our new method, we use the MILP modelling method to compute their differential (or linear) clustering effect and compare the results obtained by the MILP method

with those obtained by our new method. We emphasize that the RECTANGLE designers' method (which uses an improved Matsui's search algorithm), MILP method and our new method are 3 different approaches. Based on these comparative studies, we can reasonably infer that the results using our new method provide a quite accurate security evaluation of KNOT-AEAD against the 6 attack models and KNOT-Hash against the 2 attack models.

Section 2 presents notations and terminologies used throughout the paper. Section 3 presents the KNOT specification. In section 4, we give 6 attack models for KNOT-AEAD and 2 attack models for KNOT-Hash. Section 5 gives a description of our new method for searching the best difference and linear propagations by using iterative trails. Section 6 reports our experimental results for each of the 4 KNOT-AEAD member w.r.t. the 6 attack models and each of the 4 KNOT-Hash member w.r.t. the 2 attack models. In section 7, we investigate the accuracy of our new method in two different ways. Section 8 is a discussion and conclusion. Finally, in Appendix, we present the best effective differential/linear distinguishers with regard to the longest number of rounds for the primary KNOT-AEAD member and the primary KNOT-Hash member.

2 Preliminaries

2.1 Notations

The following table summarizes the notation used throughout this paper.

Notation	Meaning
$y \parallel x$	Concatenation of two bitstrings x and y
0^l	All-zero bitstring of length l
$x \oplus y$	XOR of bitstrings x and y
$x_{m-1} \parallel \cdots \parallel x_1 \parallel x_0$	x_0 is the least significant bit (or block), x_{m-1} is the most significant bit(or block).
S	A b-bit state of the Sponge/Duplex construction
$S_r,\ S_c$	The r-bit rate and c-bit capacity part of a state S
nr (or nr_0, nr_f, nr_h)	The number of rounds for an underlying permutation
p_b	A round transformation with a width of b bits
$p_b[nr]$	A permutation consisting of nr-round p_b
ΔSI, ΔSO, $prob$	the input difference, output difference, differential probability of a difference propagation respectively
ΓSI, ΓSO, cor	the input mask, output mask, correlation of a linear propagation respectively
KNOT-AEAD(k, b, r)	A KNOT AE member with k-bit key, b-bit state and r-bit rate
KNOT-Hash(n, b, r, r')	A KNOT-Hash member with n-bit hash output, b-bit state, r-bit absorbing rate and r'-bit squeezing rate

2.2 (Truncated) Differential Trail, (Truncated) Difference Propagation, Linear Trail and Linear Propagation

Let β be a Boolean transformation operating on b-bit vectors that is a sequence of r transformations:

$$\beta = \rho^{(r)} \circ \rho^{(r-1)} \circ \cdots \circ \rho^{(2)} \circ \rho^{(1)}.$$

In this paper, β refers to a permutation, and a difference in differential cryptanalysis is referred to as an XOR.

A *differential trail* [7] Q over an iterative transformation consists of a sequence of $r+1$ difference patterns:

$$Q = (q^{(0)}, q^{(1)}, q^{(2)}, \cdots, q^{(r-1)}, q^{(r)}).$$

The probability of a differential step is defined as:

$$Prob(q^{(i-1)}, q^{(i)}) = 2^{-n} \times \sharp\{x \in F_2^n | \rho^{(i)}(x) \oplus \rho^{(i)}(x \oplus q^{(i-1)}) = q^{(i)}\}.$$

Assuming the independence of different steps, the probability of a differential trail Q can be approximated as:

$$Prob(Q) = \prod_i Prob(q^{(i-1)}, q^{(i)}).$$

A *difference propagation* [7] is composed of a set of differential trails, the probability of a difference propagation $(\Delta SI, \Delta SO)$ is the sum of the probabilities of all r-round differential trails Q with initial difference ΔSI and terminal difference ΔSO :

$$Prob(\Delta SI, \Delta SO) = \sum_{q^{(0)}=\Delta SI, q^{(r)}=\Delta SO} Prob(Q) \tag{1}$$

The *weight of a difference propagation* is the negative of the binary logarithm of the difference propagation probability.

Let the symbol $\star$ denote an unknown value, for a b-bit string $x_{b-1} \cdots x_1 x_0$, define

$$y_{b-1} \cdots y_1 y_0 = \text{TRUNC}(x_{b-1} \cdots x_1 x_0)$$

if and only if $y_i = x_i$ or $y_i = \star$ for all $0 \le i \le b-1$.

If we have a r-round differential trail $Q = (q^{(0)}, q^{(1)}, q^{(2)}, \cdots, q^{(r-1)}, q^{(r)})$, then

$$TQ = (tq^{(0)}, tq^{(1)}, tq^{(2)}, \cdots, tq^{(r-1)}, tq^{(r)})$$

is a *truncated differential trail* [11] if $tq^{(i)} = \text{TRUNC}(q^{(i)})$ for $0 \le i \le r$. The notion of truncated differential trail can be naturally extended to *truncated difference propagations*.

The correlation $C(f, g)$ between two binary Boolean functions $f(a)$ and $g(a)$ is defined as:

$$C(f, g) = 2 \times Prob(f(a) = g(a)) - 1.$$

A *linear trail* [7] U over an iterative transformation consists of a sequence of $r+1$ selection patterns:

$$U = (u^{(0)}, u^{(1)}, u^{(2)}, \cdots, u^{(r-1)}, u^{(r)}).$$

The *correlation contribution* [7] of a linear trail is the product of the correlation of all its steps:

$$Cor(U) = \prod_i C(u^{(i)} \cdot \rho^{(i)}(a), u^{(i-1)} \cdot a). \tag{2}$$

where "$\cdot$" denotes the inner product on F_2^b.

A *linear propagation* is composed of a set of linear trails, the correlation of a linear propagation $(\Gamma SO, \Gamma SI)$ is the sum of the correlation contributions of all r-round linear trails U with initial selection pattern ΓSI and final selection pattern ΓSO :

$$Cor(\Gamma SO, \Gamma SI) = \sum_{u^{(0)}=\Gamma SI,\, u^{(r)}=\Gamma SO} Cor(U). \tag{3}$$

Similarly, the *weight of a linear propagation* is the negative of the binary logarithm of the linear propagation correlation.

3 Specification of KNOT-AEAD and KNOT Hash

In this section, we give the specification of KNOT-AEAD and KNOT-Hash. For a more detailed description, we refer to the NIST submission document of KNOT [18].

3.1 KNOT Permutations

The underlying permutations of each KNOT member iteratively apply an SP-network round transformation. KNOT uses 3 different round transformations, which are defined by the width b (b=256, 384 or 512). Each of the rounds consists of the following 3 steps: $AddRoundConstant_b$, $SubColumn_b$, $ShiftRow_b$. Let p_b denote a round transformation, the following is a pseudo C code for p_b :

{ $AddRoundConstant_b(STATE, RC)$
$SubColumn_b(STATE)$
$ShiftRow_b(STATE)$
}
where RC denotes a round constant.

Let nr (or nr_0, nr_f, nr_h) denote the number of rounds for an underlying permutation. The concrete values of nr (or nr_0, nr_h) for each KNOT member are given afterwards.

The State A b-bit state is pictured as a $4 \times \frac{b}{4}$ rectangular array of bits. Let $W = w_{b-1} \parallel \cdots \parallel w_1 \parallel w_0$ denote a state, the first $\frac{b}{4}$ bits $w_{\frac{b}{4}-1} \parallel \cdots \parallel w_1 \parallel w_0$ are arranged in row 0, the next $\frac{b}{4}$ bits $w_{\frac{b}{2}-1} \parallel \cdots \parallel w_{\frac{b}{4}+1} \parallel w_{\frac{b}{4}}$ are arranged in row 1, and so on, as illustrated in Fig. 1, and Fig. 2 gives a two-dimensional description.

$$\begin{bmatrix} w_{\frac{b}{4}-1} & \cdots & w_1 & w_0 \\ w_{\frac{b}{2}-1} & \cdots & w_{\frac{b}{4}+1} & w_{\frac{b}{4}} \\ w_{\frac{3b}{4}-1} & \cdots & w_{\frac{b}{2}+1} & w_{\frac{b}{2}} \\ w_{b-1} & \cdots & w_{\frac{3b}{4}+1} & w_{\frac{3b}{4}} \end{bmatrix}$$

Fig. 1. A b-bit State

$$\begin{bmatrix} a_{0,\frac{b}{4}-1} & \cdots & a_{0,1} & a_{0,0} \\ a_{1,\frac{b}{4}-1} & \cdots & a_{1,1} & a_{1,0} \\ a_{2,\frac{b}{4}-1} & \cdots & a_{2,1} & a_{2,0} \\ a_{3,\frac{b}{4}-1} & \cdots & a_{3,1} & a_{3,0} \end{bmatrix}$$

Fig. 2. Two-dimensional Way

The $AddRoundConstant_b$ Transformation A simple bitwise XOR of a d-bit round constant to the first d bits of the intermediate state, with $d = 6$, 7 or 8.

Let $CONST_d$ denote the set of constants generated by the d-bit LFSR. For a KNOT member, the choice of d depends on the total number of rounds.

The $SubColumn_b$ Transformation Parallel application of S-boxes to the 4 bits in the same column. The operation of $SubColumn_b$ is illustrated in Fig. 3. The input of an S-box is $Col(j) = a_{3,j} \parallel a_{2,j} \parallel a_{1,j} \parallel a_{0,j}$ for $0 \leq j \leq \frac{b}{4} - 1$, and the output is $S(Col(j)) = b_{3,j} \parallel b_{2,j} \parallel b_{1,j} \parallel b_{0,j}$.

The S-box used in KNOT is a 4-bit to 4-bit S-box $S : F_2^4 \rightarrow F_2^4$. The action of this S-box in hexadecimal notation is given by the following table.

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$S(x)$	4	0	A	7	B	E	1	D	9	F	6	8	5	2	C	3

$$\begin{pmatrix} a_{0,\frac{b}{4}-1} \\ a_{1,\frac{b}{4}-1} \\ a_{2,\frac{b}{4}-1} \\ a_{3,\frac{b}{4}-1} \end{pmatrix} \cdots \begin{pmatrix} a_{0,1} \\ a_{1,1} \\ a_{2,1} \\ a_{3,1} \end{pmatrix} \begin{pmatrix} a_{0,0} \\ a_{1,0} \\ a_{2,0} \\ a_{3,0} \end{pmatrix}$$

$$\downarrow S \quad \cdots \quad \downarrow S \quad \downarrow S$$

$$\begin{pmatrix} b_{0,\frac{b}{4}-1} \\ b_{1,\frac{b}{4}-1} \\ b_{2,\frac{b}{4}-1} \\ b_{3,\frac{b}{4}-1} \end{pmatrix} \cdots \begin{pmatrix} b_{0,1} \\ b_{1,1} \\ b_{2,1} \\ b_{3,1} \end{pmatrix} \begin{pmatrix} b_{0,0} \\ b_{1,0} \\ b_{2,0} \\ b_{3,0} \end{pmatrix}$$

Fig. 3. $SubColumn_b$ **Operates on the Columns of the State**

The $ShiftRow_b$ Transformation A left rotation to each row over different offsets. Row 0 is not rotated, row 1 is left rotated over c_1 bit, row 2 is left rotated over c_2 bits, row 3 is left rotated over c_3 bits. The parameters (c_1, c_2, c_3) only depend on b, Table 1 gives the concrete values of (c_1, c_2, c_3) for the 3 different state width b.

Table 1. ShitRow offsets for the 3 state width

b	c_1	c_2	c_3
256	1	8	25
384	1	8	55
512	1	16	25

Let $\lll x$ denote left rotation over x bits, the operation $ShiftRow_b$ is illustrated in Fig.4.

$$\left(a_{0,\frac{b}{4}-1} \ \cdots \ a_{0,1} \ \ a_{0,0}\right) \xrightarrow{\lll 0} \left(a_{0,\frac{b}{4}-1} \ \cdots \ a_{0,1} \ \ a_{0,0}\right)$$

$$\left(a_{1,\frac{b}{4}-1} \ \cdots \ a_{1,1} \ \ a_{1,0}\right) \xrightarrow{\lll c_1} \left(a_{1,\frac{b}{4}-c_1-1} \ \cdots \ a_{1,\frac{b}{4}-c_1+1} \ \ a_{1,\frac{b}{4}-c_1}\right)$$

$$\left(a_{2,\frac{b}{4}-1} \ \cdots \ a_{2,1} \ \ a_{2,0}\right) \xrightarrow{\lll c_2} \left(a_{2,\frac{b}{4}-c_2-1} \ \cdots \ a_{2,\frac{b}{4}-c_2+1} \ \ a_{2,\frac{b}{4}-c_2}\right)$$

$$\left(a_{3,\frac{b}{4}-1} \ \cdots \ a_{3,1} \ \ a_{3,0}\right) \xrightarrow{\lll c_3} \left(a_{3,\frac{b}{4}-c_3-1} \ \cdots \ a_{3,\frac{b}{4}-c_3+1} \ \ a_{3,\frac{b}{4}-c_3}\right)$$

Fig. 4. $ShiftRow_b$ **Operates on the Rows of the State**

3.2 KNOT-AEAD

The KNOT-AEAD family has 4 members. For each member, the key length, the nonce length and the tag length are all equal to k bits. The mode of operation of KNOT is based on Duplex mode MonkeyDuplex [6]. Let S denote a b-bit state, S_r and S_c denote the rate and capacity parts of S. For the initialization, the number of rounds is nr_0; for the processing of the associated data and plaintext blocks, the number of rounds is nr; for the finalization, the number of rounds is nr_f. Let KNOT-AEAD(k, b, r) denote a KNOT-AEAD member with k-bit key, b-bit state and r-bit rate. Table 2 presents the parameter sets of the 4 AEAD members.

Table 2. Parameters for the 4 members of KNOT-AEAD(k, b, r) Family

Name	Bit Size				Constants	Rounds		
	k	b	r	c		nr_0	nr	nr_f
KNOT-AEAD(128, 256, 64)	128	256	64	192	$CONST_6$	52	28	32
KNOT-AEAD(128, 384, 192)	128	384	192	192	$CONST_7$	76	28	32
KNOT-AEAD(192, 384, 96)	192	384	96	288	$CONST_7$	76	40	44
KNOT-AEAD(256, 512, 128)	256	512	128	384	$CONST_7$	100	52	56

where KNOT-AEAD(128, 256, 64) is the primary AEAD member.

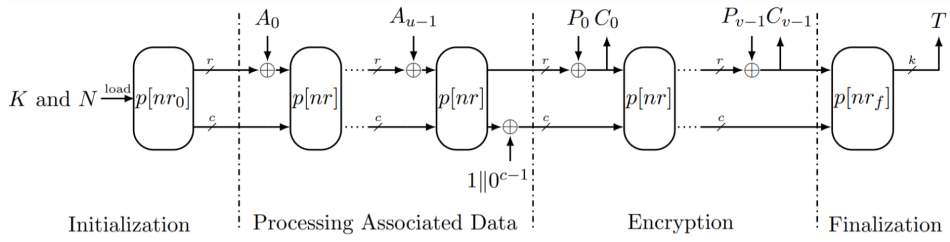

Fig. 1. Encryption of KNOT-AEAD

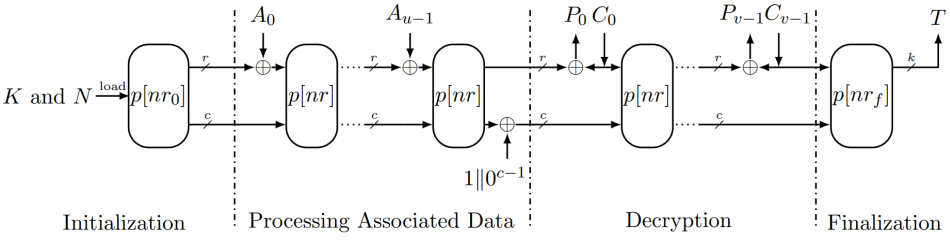

Fig. 2. Decryption of KNOT-AEAD

Assume the padded associated data have u blocks of r bits: $AD_{u-1} \parallel \cdots \parallel AD_0$, and the padded plaintext have v blocks of r bits: $P_{v-1} \parallel \cdots \parallel P_0$. Each member of KNOT-AEAD

has 4 phases: initialization, processing associated data, encryption and finalization. Figure 1 and Figure 2 illustrates the encryption and decryption of KNOT-AEAD respectively.

The authenticated encryption process is initialized by loading the key K and the nonce N (both k bits). The b-bit state is initialized as:

$$S = \begin{cases} (0^{128} \parallel K \parallel N) \oplus (1 \parallel 0^{383}) & \text{for KNOT-AEAD(128,384,192),} \\ K \parallel N & \text{for the other 3 AEAD members.} \end{cases} \quad (4)$$

Then nr_0 rounds of the round transformation $p_b[nr_0]$ are applied to the initial state.

In the finalization, the state is firstly updated by nr_f rounds of the permutation $p_b[nr_f]$. Then the tag consists the first k bits of the state.

Table 3 presents the security goals and data limit of KNOT-AEAD [18]. The security strength is indicated by the logarithm base 2 of the attack cost, and the unit is the underlying KNOT permutation $P_b[nr]$ used by the KNOT-AEAD member. The data limit M means that the number of processed plaintext and associated data blocks is limited to M blocks under one key.

Table 3. Claimed Security Strength and Data Limit for the 4 KNOT-AEAD Members

	Plaintext Confidentiality	Ciphertext Integrity	Data Limit
KNOT-AEAD(128, 256, 64)	125	125	2^{64}
KNOT-AEAD(128, 384, 192)	128	128	2^{64}
KNOT-AEAD(192, 384, 96)	189	189	2^{96}
KNOT-AEAD(256, 512, 128)	253	253	2^{128}

3.3 KNOT Hash

The KNOT-Hash family also has 4 members. The mode of operation of KNOT-Hash uses an extended Sponge construction with a different squeezing bitrate r'. Let KNOT-Hash(n, b, r, r') denote a KNOT-Hash member with n-bit hash output, b-bit state, r-bit absorbing rate and r'-bit squeezing rate. Table 4 presents the parameter sets of the 4 members of the KNOT-Hash family, KNOT-Hash(256, 256, 32, 128) is the primary hash member.

Table 4. Parameters for the 4 members of KNOT-Hash(n, b, r, r') Family

Name	Bit Size					Constants	Rounds
	n	b	c	r	r'		nr_h
KNOT-Hash(256,256,32,128)	256	256	224	32	128	$CONST_7$	68
KNOT-Hash(256,384,128,128)	256	384	256	128	128	$CONST_7$	80
KNOT-Hash(384,384,48,192)	384	384	336	48	192	$CONST_7$	104
KNOT-Hash(512,512,64,256)	512	512	448	64	256	$CONST_8$	140

where KNOT-Hash(256, 256, 32, 128) is the primary hash member.

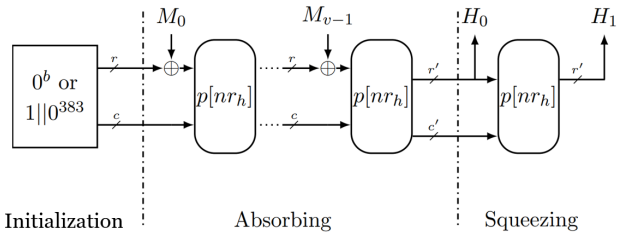

Fig. 3. KNOT Hash

Assume the padded message M (including the empty string) is divided into v blocks of r bits: $M_{v-1} \parallel \cdots \parallel M_0$. Each member of KNOT-Hash has 3 phases: initialization, absorbing and squeezing. Figure 3 illustrates KNOT-Hash.

The b-bit state is initialized as:

$$S = \begin{cases} 1 \parallel 0^{383} & \text{for KNOT-Hash(256, 384, 128, 128),} \\ 0^b & \text{for the other 3 hash members.} \end{cases} \tag{5}$$

For each member of KNOT-Hash, two extractions are needed to generate the n-bit tag, H_0 for the first extraction and H_1 for the second extraction.

Table 5 presents the security goals and data limit of KNOT-Hash [18]. The message length limit is in accordance with the corresponding KNOT-AEAD member.

Table 5. Claimed Security Strength and Data Limit for the 4 KNOT-Hash Members

Name	Security (bit)			Data limit
	pre.	2nd pre.	col.	
KNOT-Hash(256, 256, 32, 128)	128	112	112	2^{64}
KNOT-Hash(256, 384, 128, 128)	128	128	128	2^{64}
KNOT-Hash(384, 384, 48, 192)	192	168	168	2^{96}
KNOT-Hash(512, 512, 64, 256)	256	224	224	2^{128}

4 Attack Models

In this section, we describe 8 different attack models for KNOT. Section 4.1 presents differential distinguishing attack, linear key-recovery attack, linear distinguishing attack, all of which target the initialization phase of KNOT-AEAD. Section 4.2 presents forgery attack and another linear distinguishing attack, which both target the encryption phase of KNOT-AEAD. Section 4.3 presents another forgery attack, which targets the finalization phase of KNOT-AEAD. Section 4.4 presents two models of collision attacks on KNOT-Hash.

Note that attacks of AEAD are all in nonce-respecting and single-key scenarios, which means, the attacker never makes two queries to the encryption oracle with the same nonce, and the data complexity is limited to the data limit under one key (Table 3). Nevertheless, we need to note that, in a forgery attack, the attacker is allowed to repeat nonces in decryption queries.

4.1 Three Attacks Targeting Initialization Phase of KNOT-AEAD

The key is fixed. The scenario is a chosen-nonce and known-plaintext attack, that is to say, the attacker has the ability to choose the nonce, and collect plaintext-ciphertext pairs under the secret key and the chosen nonce. During the encryption phase of a Duplex-based AEAD algorithm, the attacker can get the rate part of the states by XORing the plaintext block with its corresponding ciphertext block, we call them key-stream blocks for simplicity.

The target is the initialization phase, hence we need to assume there is no associated data and the padded plaintext has only 1 block. In this case, each key-stream block is just the rate part of the state after the initialization. The goal of the attacker is either recover the secret key or distinguish the key-stream blocks from binary pseudorandom sequences.

For the initial b-bit state S, let S_N, S_K denote the nonce and the key part of S respectively.

Differential Distinguishing Attack Differential trails have the restriction that all active bits of the input difference are only allowed in the nonce part of the state, and the rate part of the output difference is some fixed value (note that the value in the capacity part of the output difference is not required, which means a truncated differential trail). That is to say, for the input difference ΔSI, the nonce part ΔSI_N is the only non-zero part; for the output difference ΔSO, the rate part ΔSO_r is a fixed value. In addition, we need to consider the clustering of such truncated differential trails.

By choosing pairs of nonces which have difference ΔSI_N, and calculating the difference of the key-stream blocks, the attacker can use such type of truncated difference propagation as key-stream distinguishers. Let $Prob$ denote the probability of such a truncated difference propagation. To be a successful distinguishing attack, the number of required key-stream blocks is proportional to $Prob^{-1}$, the constant of proportionality is typically a small integer.

Linear Key-recovery Attack Linear propagations have the restriction that the key part of the input mask is active, and all active bits of the output mask are only allowed in the rate part of the state. That is to say, for the input mask ΓSI, the key part ΓSI_K is non-zero; for the output mask ΓSO, the rate part ΓSO_r is the only non-zero part.

Such type of linear propagations can be used to construct linear equations that express bits of the key in terms of the nonce, the constant if any and key-stream block. One such equation recovers 1-bit information about the key. Let Cor denote the correlation of such a linear propagation. To be a successful key-recovery attack, the number of required key-stream blocks is proportional to Cor^{-2} (also the constant of proportionality is typically a small integer).

Linear Distinguishing Attack This attack is very similar to the above linear key-recovery attack. In such an attack, the key part of the input mask is non-active, hence the linear equation does not involve any key bits; however, it can still be used as a linear distinguishing attack. That is to say, for the input mask ΓSI, the key part ΓSI_K is zero; for the output mask ΓSO, the rate part ΓSO_r is the only non-zero part.

4.2 Two Attacks Targeting Encryption Phase of KNOT-AEAD

Forgery Attack The scenario is a chosen-nonce and adaptive chosen-plaintext-and-ciphertext attack, and the key is fixed. Let A be a nonce-respecting attacker. Note that the nonce-respecting condition applies only to encryption query, and the attacker is free to repeat

nonces in its decryption oracle. We say that A forges [16] if he can output a nonce-ciphertext pair (N, C) such that the ciphertext C was not the response to any encryption query and the decryption oracle returns a plaintext other than an error symbol.

Difference propagations have the restriction that all active bits of both the input difference and the output difference are only allowed in the rate part of the state. That is to say, for the input difference ΔSI, the rate part ΔSI_r is the only non-zero part; for the output difference ΔSO, the rate part ΔSO_r is the only non-zero part.

As an example, such type of difference propagations can be used in the associated data processing to mount a forgery attack. Note that the nonce can be repeated in the decryption oracle. The attacker can inject the difference ΔSI_r to the first block of the associated data and cancel the state difference by injecting the difference ΔSO_r to the second block of the associated data. If a full-state collision occurs, the attacker can certainly get a successful forgery. Let $Prob$ denote the probability of such a difference propagation. To be a successful forgery attack, the number of required encryption/decryption queries is proportional to $Prob^{-1}$.

Linear Distinguishing Attack The scenario is also a chosen-nonce and known-plaintext attack, and the key is fixed. Linear propagations have the restriction that all active bits of both the input mask and the output mask are only allowed in the rate part of the state. That is to say, for the input mask ΓSI, the rate part ΓSI_r is the only non-zero part; for the output mask ΓSO, the rate part ΓSO_r is the only non-zero part.

Such type of linear propagations can be used in a linear distinguishing attack, targeting the encryption phase. For each permutation during the encryption phase, the rate part of the permutation input is the previous ciphertext block, and the rate part of the permutation output can be calculated by XORing the current plaintext block with its corresponding ciphertext block. Similarly, let Cor denote the correlation of such a linear propagation. To be a successful distinguishing attack, the number of required key-stream blocks is proportional to Cor^{-2}.

4.3 Forgery Attack Targeting Finalization Phase of KNOT-AEAD

In this subsection, we introduce another forgery attack, which targets the finalization phase of KNOT-AEAD. For the final state S, let S_{tag} denote the tag part of S. Difference propagations have the restriction that all active bits of the input difference are only allowed in the rate part of the state, and the tag part of the output difference is a fixed value (also the value in the non-tag part of the output difference is not required, which means a truncated difference propagation). That is to say, for the input difference ΔSI, the rate part ΔSI_r is the only non-zero part; for the output difference ΔSO, the tag part ΔSO_{tag} is a fixed value.

Such type of truncated difference propagation can be used in the finalization permutation for a forgery attack. The attacker modifies the tag T to $T \oplus \Delta SO_{tag}$, and modifies the last ciphertext block C_{v-1} to $C_{v-1} \oplus \Delta SI_r$, then he calls the decryption oracle under the same nonce and checks if the returned plaintext block is equal to $P_{v-1} \oplus \Delta SI_r$. Let $Prob$ denote the probability of such a truncated difference propagation. To be a successful forgery attack, the number of required encryption/decryption queries is proportional to $Prob^{-1}$.

4.4 Collision Attack of KNOT-Hash

In this subsection, we present two different strategies for searching collisions of sponge-based hash functions. Both the strategies use difference propagations with certain constraints. Let

$Prob$ denote the differential probability, to be a successful (near) collision attack, the number of required queries is proportional to $Prob^{-1}$.

The first Strategy Difference propagations have the restriction that all active bits of both the input difference and the output difference are only allowed in the rate part of the state. That is to say, for the input difference ΔSI, the rate part ΔSI_r is the only non-zero part; for the output difference ΔSO, the rate part ΔSO_r is the only non-zero part.

Such type of difference propagations can be used for a collision attack. The attacker choose message pairs as follows. Each padded message in the pair has v ($v \geq 2$) blocks, $\Delta M_i = 0$ for $0 \leq i \leq v-3$, $\Delta M_{v-2} = \Delta SI_r$ and $\Delta M_{v-1} = \Delta SO_r$. Apply such a difference propagation in the absorbing permutation of M_{v-2}. If a message pair satisfies the difference propagation, then we have a full-state collision after absorbing the last message block M_{v-1}, which certainly results in a collision.

The Second Strategy For the initial state of the squeezing phase S, let S_h denote its hash value part.

Difference propagations have the restriction that all active bits of the input difference are only allowed in the rate part of the state, and the value in the hash value part of the output difference is zero (note that the value in the capacity part of the output difference is not required, which means a truncated difference propagation). That is to say, for the input difference ΔSI, the rate part ΔSI_r is the only non-zero part; for the output difference ΔSO, the hash part ΔSO_h is zero.

For each KNOT-Hash member, the squeezing phase needs 2 extractions. Hence, such type of difference propagations can be used for a near-collision attack. The attacker choose message pairs as follows. Each padded message in the pair has v ($v \geq 1$) blocks, $\Delta M_i = 0$ for $0 \leq i \leq v-2$, $\Delta M_{v-1} = \Delta SI_r$. Apply such a difference propagation in the absorbing permutation of M_{v-1}. If a message pair satisfies the difference propagation, then we have a r'-bit near collision.

5 An Efficient Search Method for the Best Difference and Linear Propagations for the KNOT Permutations

In this section, we give a description of our search method. More details are refer to [8]. The followings are the main steps:

1. In our first step, we use the algorithm of finding elementary circuits to find all elementary iterative trails for each of the KNOT permutations.
2. A difference(mask) value is associated to a vertex. According to the iterative trails we found in the previous step, the vertex set V contains all difference/mask values locating in at least one iterative trail. A 2-stage graph G_1 is generated where both the stages only contain vertices in V, that is $S_0 = S_1 = V$. Each edge $u \leftarrow v$ is associated with the differential probability (linear correlation) of the difference (linear) propagation from u to v. A $r+1$-stage graph G_r can be generated by concatenating G_1 to itself $r-1$ times. Given a round number n, input value $u \in V$ and output value $v \in V$, one can efficiently compute the difference probability or linear correlation of the propagation from u to v over n rounds in G_n through a stage by stage computation. We restore the difference probability or the linear correlation as $p_G(u, v, n)$ where $u, v \in V$.

3. For each vertex in V, we extend it both forward and backward for several rounds and limit the weight of the extension within a bound. Both the number of rounds and the bound limiting the weight can be manually tuned. Note that in order to avoid duplicate trails in the next step, the extension won't pass any values in V. The difference (mask) values reached by the backward extension over n rounds are kept in vertex set VB_n and $p_b(u, v, n)$ denotes the weight that the propagation from u to v costs according to the backward extension where $u \in VB_n, v \in V$. Similarly, the difference (mask) values reached by the forward extension over n rounds are kept in vertex set VF_n and $p_f(u, v, n)$ denotes the weight that the propagation from u to v costs according to the forward extension where $u \in V, v \in VF_n$.
4. Given the round number n, we can compute the difference probability or linear correlation from u to v over n rounds by

$$p(u, v, n) = \sum_{\substack{x_b, x_f, n_b, n_G, n_f \\ n_b+n_G+n_f=n \\ x_b \in V, x_f \in V \\ u \in VB_{n_b}, v \in VF_{n_f}}} p_b(u, x_b, n_b) \times p_G(x_b, x_f, n_G) \times p_f(x_f, v, n_f).$$

The weight of the best n-round differential or approximation is given by $-\log_2(\max p(u, v, n))$.

6 Experimental Results of KNOT

For convenience of description, we use the following notations in this section:

Notation	Meaning
Diff-Init-D	truncated difference propagation for distinguishing attack targeting the initialization of KNOT-AEAD
Linear-Init-KR	linear propagation for key-recovery attack targeting the initialization of KNOT-AEAD
Linear-Init-D	linear propagation for distinguishing attack targeting the initialization of KNOT-AEAD
Diff-Enc-F	difference propagation for forgery attack targeting the encryption of KNOT-AEAD
Linear-Enc-D	linear propagation for distinguishing attack targeting the encryption of KNOT-AEAD
Diff–Final-F	truncated difference propagation for forgery attack targeting the finalization of KNOT-AEAD
Diff-Col-I	difference propagation for collision attack of KNOT-Hash, using the first strategy
Diff-Col-II	difference propagation for near-collision attack of KNOT-Hash, using the second strategy
KNOT-AEAD vi	the i-th (i=1,2,3,4) version of KNOT-AEAD, where KNOT-AEAD v1 is the primary version
KNOT-Hash vi	the i-th (i=1,2,3,4) version of KNOT-Hash, where KNOT-Hash v1 is the primary version

Table 6- 9 gives the weight of the best distinguishers w.r.t. the 6 attack models for different number of rounds of the 4 KNOT-AEAD members respectively. Table 10 gives the weight of the best distinguishers w.r.t. the 2 attack models for different number of rounds of the 4 KNOT-Hash members. In these 5 tables, the numbers in red color denote the highest weight considering the data limit and the security strength of the corresponding attack model, and the corresponding number of rounds is the highest number of rounds of an effective distinguisher.

Take KNOT-AEAD v1 (the primary member) for example. For differential distinguishing attack targeting the initialization phase, an effective distinguisher can reach 14 rounds at most (the full number of rounds of the initialization is 52 rounds), with probability $2^{-62.2}$; for forgery attack targeting the encryption phase, an effective distinguisher can reach 12 rounds at most (the full number of rounds of the encryption is 28 rounds), with probability $2^{-62.4}$; for forgery attack targeting the finalization phase, an effective distinguisher can reach 13 rounds at most (the full number of rounds of the finalization is 32 rounds), with probability $2^{-61.4}$. In Appendix, we present examples of these best distinguishers, which can be used directly for attacks on reduced-round KNOT-AEAD v1 and KNOT-Hash v1.

Table 6. KNOT-AEAD v1 (primary version): weights of the best r-round distinguisher

r	Diff-Init-D	Linear-Init-KR	Linear-Init-D	Diff-Enc-F	Linear-Enc-D	Diff-Final-F
10	43	25	26	52.4	27	47.1
11	47.9	28	28	57.4	30	51.8
12	52.6	30	31	62.4	32	56.6
13	57.4	33	33	67.7	35	61.4
14	62.2	35	36	72.4	37	66.2
15	67	38	38	77.1	40	71.0

Table 7. KNOT-AEAD v2: weights of the best r-round distinguisher

r	Diff-Init-D	Linear-Init-KR	Linear-Init-D	Diff-Enc-F	Linear-Enc-D	Diff-Final-F
10	46.4	25	26	51.4	25.4	44.0
11	51.2	28	29	56.7	28	48.6
12	56	30	31	61.4	30.4	53.2
13	60.8	33	34	66.1	33	57.9
14	65.6	35	36	71.1	35.4	62.6
15	70.4	38	39	76.1	38	67.4

7 Verification of Our Search Method

7.1 Comparison with the Differential and Linear Clustering Results of RECTANGLE

In the design paper of RECTANGLE [17], the designers claim that: "For 14-round RECTANGLE, the probability of the best differential trail is 2^{-61}. We have searched for all

Table 8. KNOT-AEAD v3: weights of the best r-round distinguisher

r	Diff-Init-D	Linear-Init-KR	Linear-Init-D	Diff-Enc-F	Linear-Enc-D	Diff-Final-F
10	43	25	26	52.4	27	47.1
11	47.9	28	28	57.4	30	51.8
12	52.6	30	31	62.4	32	56.6
13	57.4	33	33	67.7	35	61.4
14	62.2	35	36	72.4	37	66.2
15	67	38	38	77.1	40	71.0
16	71.8	40	41	82.1	42	75.8
17	76.6	43	43	87.1	45	80.6
18	81.4	45	46	92.1	47	85.4
19	86.2	48	48	96.8	50	90.1
20	91	50	51	101.5	52	94.9
21	95.8	53	53	106.5	55	99.7
22	100.5	55	56	111.5	57	104.5

Table 9. KNOT-AEAD v4: weights of the best r-round distinguisher

r	Diff-Init-D	Linear-Init-KR	Linear-Init-D	Diff-Enc-F	Linear-Enc-D	Diff-Final-F
10	43	25	26	52.4	27	47.1
11	47.9	28	28	57.4	30	51.8
12	52.6	30	31	62.4	32	56.6
13	57.4	33	33	67.7	35	61.4
14	62.2	35	36	72.4	37	66.2
15	67	38	38	77.1	40	71.0
16	71.8	40	41	82.1	42	75.8
17	76.6	43	43	87.1	45	80.6
18	81.4	45	46	92.1	47	85.4
19	86.2	48	48	96.8	50	90.1
20	91	50	51	101.5	52	94.9
21	95.8	53	53	106.5	55	99.7
22	100.6	55	56	111.5	57	104.5
23	105.3	58	58	116.3	60	109.3
24	110.1	60	61	121.1	62	114.1
25	114.9	63	63	125.9	65	118.9
26	119.7	65	66	130.9	67	123.7
27	124.5	68	68	135.8	70	128.5
28	129.3	70	71	140.5	72	133.3

Table 10. KNOT-Hash: weights of the best r-round distinguisher

	KNOT-Hash v1		KNOT-Hash v2		KNOT-Hash v3		KNOT-Hash v4	
r	Diff-Col-I	Diff-Col-II	Diff-Col-I	Diff-Col-II	Diff-Col-I	Diff-Col-II	Diff-Col-I	Diff-Col-II
10	53.7	47.1	51.4	44.1	54.0	47.1	53.7	47.1
11	59.4	51.8	56.7	48.7	57.4	51.8	57.4	51.8
12	62.7	56.6	61.4	53.4	62.4	56.6	62.4	56.6
13	67.7	61.4	66.1	58.1	67.8	61.4	67.7	61.4
14	72.7	66.2	71.1	62.8	72.4	66.2	72.4	66.2
15	-	-	76.1	67.5	77.4	71.0	77.1	71.0
16	-	-	-	-	82.7	75.8	82.1	75.8
17	-	-	-	-	88.7	80.6	87.1	80.6
18	-	-	-	-	95.2	85.4	92.1	85.4
19	-	-	-	-	100.8	90.1	96.8	90.1
20	-	-	-	-	106.5	94.9	101.5	94.9
21	-	-	-	-	112.2	99.7	106.5	99.7
22	-	-	-	-	-	-	111.5	104.5
23	-	-	-	-	-	-	116.3	109.3
24	-	-	-	-	-	-	121.1	114.1
25	-	-	-	-	-	-	126.1	118.9
26	-	-	-	-	-	-	131.2	123.7
27	-	-	-	-	-	-	136.2	128.5

14-round differential trails with probability between 2^{-61} and 2^{-71} (up to a rotation equivalence), and examined all the difference propagations made up of these investigated trails." A 14-round difference propagation of RECTANGLE is given in [17] , the input difference is 0x0060000000000100, the output difference is 0x0000000000100861, and the probability is $2^{-60.66}$. Fixing the input and output difference and applying our method, we also obtain a probability of $2^{-60.66}$.

In the case of linear propagations, the designers of RECTANGLE claim that:"For 14-round RECTANGLE, we have searched for all the linear trails with correlation (absolute value) between 2^{-34} and 2^{37} (up to a rotation equivalence) and examined all the linear propagations made up of the investigated trails." The following is a 14-round linear propagation, the input mask is 0x0200000000003000, the output mask is 0x0000020000600004, the average correlation is $2^{-31.61}$ (averaged over over all possible keys, see Theorem 7.9.1 in [7] for details). Fixing both the input and output mask and applying our method, we obtain an average correlation of $2^{-32.32}$.

7.2 Another verification Using MILP Modelling Method

According to the experimental results in section 6, for the 256-bit KNOT permutation, we find a 10-round Diff-Col-I type difference propagation with input difference

$$0x0000000000000000000000000000000010000000000000001000000000000000,$$

and output difference

$$0x0000000000000000000000000000000000000010000000000000000000000001$$

with probability of $2^{-53.7}$. Fixing the input and output difference and using MILP modelling method, we have searched for all the differential trails with probability between 2^{-56} and 2^{-92}, and we get a total probability $2^{-53.5}$.

According to the experimental results in section 6, for the 256-bit KNOT permutation, we find a 10-round Linear-Enc-D type linear propagation with input mask:

$$0x1000000000000000000000001000000000000000000000000000000000000000$$

and output mask

$$0x0000000000000000000000001000000000000000010000000000000000000000$$

with correlation amplitude 2^{-27}. Fixing the input and output mask and using MILP method, we have searched for all the linear trails with probability between 2^{-30} and 2^{-40} , and get a correlation amplitude $2^{-25.4}$.

Moreover, according to the experimental results in section 6, we find a 11-round Linear-Enc-D type linear propagation with input mask:

$$0x1000000000000000000000000100000000000000000000000000000000000000$$

and output mask

$$0x0000000000000000000000001000000000000000010000000000000000000000$$

with correlation amplitude 2^{-30}. Fixing the input and output mask and using MILP method, we have searched for all the linear trails with probability between 2^{-34} and 2^{-46} , and get a correlation amplitude $2^{-29.7}$.

7.3 A Brief Summary

The KNOT permutations and RECTANGLE have a lot of similarities. The RECTANGLE designers' method (which uses an improved Matsui's search algorithm), MILP method and our new method are 3 different approaches. Based on the above comparative results, the differential probabilities (or linear correlations) are very close under the 2 different evaluation approaches. Hence, we can reasonably infer that the results in this paper provide a quite accurate security evaluation of KNOT-AEAD w.r.t the 6 attack models and KNOT-Hash w.r.t the 2 attack models.

8 Discussion and Conclusion

Based on the results in Table 6- 9, and considering the data limit under one key, all KNOT-AEAD members have sufficient security margin in single-key scenarios against the 6 attack models considered in this paper. Specifically, each KNOT-AEAD member has at least 50% security margin against the 6 attack models; especially, the initialization phase has at least 72% security margin.

Note that security decreases in a multi-key scenario. If taking multi-key attacks into account and assuming that the attacker has a power of 2^{125} data complexity and 2^{125} time complexity, the number of rounds in data processing and finalization phases of KNOT-AEAD (i.e, the values of nr and nr_f) need to be increased to have a more comfortable security margin. However, in practice, the data limit 2^{64} is sufficient for lightweight applications.

From Table 10 and considering the data limit, all KNOT-Hash members have sufficient security margin against the 2 attack models considered in this paper. Specifically, each KNOT-Hash member has at least 80% security margin against the 2 attack models.

Moreover, the high security margin suggests a possibility of reducing the number of rounds in KNOT-AE's initialization phase and KNOT-Hash, we are investigating this problem very carefully by considering other attacks of KNOT-AE and KNOT-Hash, including cube attack, differential-linear attack, rebound attack and so on.

References

1. Andreeva, E., Mennink, B., and Preneel, B.: The parazoa family: Generalizing the sponge hash functions. International Journal of Information Security 11(3), 149-165, 2012.
2. Bao, Z., Zhang, W., Lin, D.: Speeding up the Search Algorithm for the Best Differential and Best Linear Trails, Inscrypt'2014, LNCS, vol. 8957, 259–285, Springer (2014).
3. Biham, E., Shamir, A.: Differential Cryptanalysis of DES-like Cryptosystems. Journal of Cryptology 4(1), 3–72 (1991).
4. Bertoni, G., Daemen, J., Peeters, M., Van Assche, G.: Cryptographic sponge functions, 2011. https://keccak.team/files/SpongeFunctions.pdf.
5. Bertoni, G., Daemen, J., Peeters, M., Van Assche, G.: Duplexing the sponge: Single-pass authenticated encryption and other appli- cations, Selected Areas in Cryptography - SAC 2011, Revised Selected Papers, LNCS, vol. 7118, 320C337, Springer, 2011.
6. Daemen, J.: Permutation-based Encryption, Authentication and Authenticated Encryption. DIAC - Directions in Authenticated Ciphers, July 2012.
7. Daemen, J., Rijmen, V.: The Design of Rijndael: AES - The Advanced Encryption Standard. Springer (2002).
8. Ding T., Zhang W., Ji F., Zhou C., An Automatic Search Tool for Iterative Trails and its Application to KNOT, PRESENT, GIFT-64 and RECTANGLE, soon available at https://eprint.iacr.org/ (before 22th September, 2020).
9. Guo, J., Peyrin, T., and Poschmann, A.: The photon family of lightweight hash functions. In: P. Rogaway (Ed.), CRYPTO 2011. LNCS, vol. 6841, 222-239. Springer (2011).
10. Johnson, D. B. Finding all the elementary circuits of a directed graph. SIAM Journal on Computing, 4(1), 77-84, 1975.
11. Knudsen, L.R.: Truncated and higher order differentials. In B. Preneel, editor, Fast Software Encryption, FSE 1994, LNCS, vol. 1008, 196-C211, Springer (1995).
12. Matsui, M.: Linear Cryptanalysis Method for DES Cipher. In: Helleseth, T. (ed.) EUROCRYPT 1993. LNCS, vol. 765, 386–397. Springer (1994).
13. Matsui, M.: On Correlation between the Order of S-Boxes and the Strength of DES. In: De Santis, A. (ed.) EUROCRYPT 1994. LNCS, vol. 950, 366–375. Springer (1995).
14. National Institute of Standards and Technology. Lightweight Cryptography Standardization project, 2019. https://csrc.nist.gov/projects/ lightweight-cryptography.
15. National Institute of Standards and Technology. Status Report on the First Round of the NIST Lightweight Cryptography Standardization Process, 2019. https://nvlpubs.nist.gov/nistpubs/ir/2019/NIST.IR.8268.pdf.
16. Rogaway, P. Authenticated-encryption with associated-data. Proceedings of the 9th ACM Conference on Computer and Communications Security (CCS 02), ACM Press, 98C-107, 2002.
17. Zhang, W., Bao, Z., Lin, D., Rijmen, V., Yang, B., Verbauwhede, I.: RECTANGLE: a bit-slice lightweight block cipher suitable for multiple platforms. Sci. China Inf. Sci., 2015, 58: 122103(15).
18. Zhang, W., Ding, T., Yang, B., Bao. Z., Xiang Z., Ji F., Zhao X., KNOT - Submission to Round 2 of the NIST Lightweight Cryptography Standardization process, https://csrc.nist.gov/CSRC/media/Projects/lightweight-cryptography/documents/round-2/spec-doc-rnd2/knot-spec-round.pdf, 2019.

A Best Distinguishers w.r.t. the 6 Attack Models for the Primary KNOT-AEAD

In Appendix A, we present the best distinguishers w.r.t. the 6 attack models for the primary KNOT-AEAD member, which can be directly used to distinguish the reduced-round AEAD from a pseudorandom bit generator or even mount a key-recovery attack. The symbol * denotes an unknown bit.

Table 11. KNOT-AEAD v1: A 14-round Distinguisher of Type Diff-Init-D with Prob. $2^{-62.2}$

input difference
00000000000000000000000000000000000000000000000000000000000000000000
10000000000000000000000010000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000
output difference
00000000000000000000000000000000000000000000000000000000000000000000
****
****
****

Table 12. KNOT-AEAD v1: A 12-round Distinguisher of Type Linear-Init-KR with Cor. 2^{-30}

input mask
00000000000000000000000000000000000000000000000000000000000000000000
10000000000000000000000010000000000000000000000000000000000000000000
10000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000
output mask
00000000000000000000000010000000000000000000000010000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000

Table 13. KNOT-AEAD v1: A 12-round Distinguisher of Type Linear-Init-D with Cor. 2^{-31}

input mask
00000000000000000000000000000000000000000000000010000000000000000000
10000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000
output mask
00000000000000000100000000000000000000000000000000000000000000000001
00000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000

Table 14. KNOT-AEAD v1: A 12-round Distinguisher of Type Diff-Enc-F with Prob. $2^{-62.4}$

input difference
10000000000000000100000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000
output difference
00000000000001000000000000000000000001000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000

Table 15. KNOT-AEAD v1: A 11-round Distinguisher of Type Linear-Enc-D with Cor. 2^{-30}

input mask
10000000000000000000000001000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000
output mask
00000000000000000000000010000000000000001000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000

Table 16. KNOT-AEAD v1: A 13-round Distinguisher of Type Diff-Final-F with Prob. $2^{-61.4}$

input difference
10000000000000000100000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000
output difference
00000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000
****
****

B Best Distinguishers w.r.t. the 2 Attack Models for the Primary KNOT-Hash

In Appendix B, we present the best distinguishers w.r.t. the 2 attack models for the primary KNOT-Hash member, which can be used to mount (near) collision attacks on the reduced-round hash function.

Table 17. KNOT-Hash v1: A 12-round Distinguisher of Type Diff-Col-I with Prob. $2^{62.7}$

input difference
1000000000000000010000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
output difference
0000001000000000000000000000001000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000

Table 18. KNOT-Hash v1: A 13-round Distinguisher of Type Diff-Col-II with Prob. $2^{-61.4}$

input difference
1000000000000000010000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
output difference
0000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
****
****