Artemia v 1.1

Designers: Javad Alizadeh[1], Mohammad Reza Aref[1] and Nasour Bagheri[2]
[1]Information Systems and Security Lab. (ISSL),
Electrical Eng. Department, Sharif University of Technology, Iran,
alizadja@gmail.com, Aref@sharif.edu
[2]Electrical Engineering Department,
Shahid Rajaee Teacher Training University, Iran, NBagheri@srttu.edu

Submitter: Javad Alizadeh

2014.03.31

Abstract

This document specifies a family of the dedicated authenticated encryption Artemia. It is an online nonce-based authenticated encryption scheme which supports the associated data. Artemia uses the permutation based mode JHAE that is provably secure in the ideal permutation model. Artemia permutations, $Artemia : \{0,1\}^{2n} \rightarrow \{0,1\}^{2n}$, have the two variants in which $n = 256$ and $n = 128$ and are secure against the differential and linear cryptanalysis.

Contents

Chapter 1

Specification

This chapter defines the family of the dedicated authenticated encryption, namely Artemia. It has the two variants with the different security levels and resource's requirements. Artemia-256 uses a 512-bit permutation and Artemia-128 uses a 256-bit permutation in the JHAE mode.

1.1 Parameters

Artemia has the three parameters of the *key*, *nonce*, and *tag* and uses an integer n to denote the length of the parameters. The parameters and their length for Artemia-256 and Artemia-128 are summarized in Table 1.1.

1.2 Constants

The permutations $Artemia - 512$ and $Artemia - 256$ use the six constants of C_0 to C_5. These constants are represented in Table 1.2 and Table 1.2.

1.3 Conversions

In order To convert a string to another string of different lengths, one uses the little endian conversions.

Table 1.1: The parameters of Artemia

	The permutation lenght $(2n)$	*The key length* (n)	*The nonce length* (n)	*The tag length* (n)
Artemia-256	512	256	256	256
Artemia-128	256	128	128	128

Table 1.2: The constants of $Artemia - 512$ in the hexadecimal

C_0	00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
	00000000 00000000 00000000 00000000 00000000 00000000 00000000 0f1e2d3b
C_1	00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
	00000000 00000000 00000000 4b5a6978 00000000 00000000 00000000 00000000
C_2	00000000 00000000 00000000 00000000 00000000 00000000 00000000 8796a5b4
	00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
C_3	00000000 00000000 00000000 c3d2e1f0 00000000 00000000 00000000 00000000
	00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
C_4	00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
	00000000 00000000 00000000 00000000 00000000 00000000 2d3c4b5a 00000000
C_5	00000000 00000000 00000000 00000000 00000000 00000000 69788796 00000000
	00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000

Table 1.3: The constants of $Artemia - 256$ in the hexadecimal

C_0	00000000 00000000 00000000 00000000 00000000 00000000 00000000 0f1e2d3b
C_1	00000000 00000000 00000000 00000000 00000000 4b5a6978 00000000 00000000
C_2	00000000 00000000 00000000 8796a5b4 00000000 00000000 00000000 00000000
C_3	00000000 c3d2e1f0 00000000 00000000 00000000 00000000 00000000 00000000
C_4	00000000 00000000 00000000 00000000 00000000 00000000 2d3c4b5a 00000000
C_5	00000000 00000000 69788796 00000000 00000000 00000000 00000000 00000000

1.4 Specification of JHAE

JHAE was introduced in [1]. In this section, we describe the JHAE mode, depicted in Fig 1.1. JHAE is a developed mode from the JH hash function mode and iterates a fixed permutation $\pi : \{0,1\}^{2n} \to \{0,1\}^{2n}$. It is a nonce-based, single-pass, and an online dedicated AE mode that supports the AD.

1.4.1 Encryption and Authentication

JHAE accepts an n-bit key K, an n-bit nonce N, a message M and an optional AD (A), then it produces the ciphertext C and authentication tag T. The pseudo code of the JHAE's encryption-authentication is depicted in Table 1.4. We assume that the input message after padding, is a multiple of the block size n. The last block of the original message is concatenated by the padding data which is as follows:

1. Eight bits are used to represent the length of the nonce (N) in Artemia-256 and nine bits are used to represent the length of the nonce in Artemia-512.

2. 24 bits are used to represent the length of the associated data (A), e.g., it would be 0^{24} if there is no AD.

3. 64 bits are used to represent the length of the message (M).

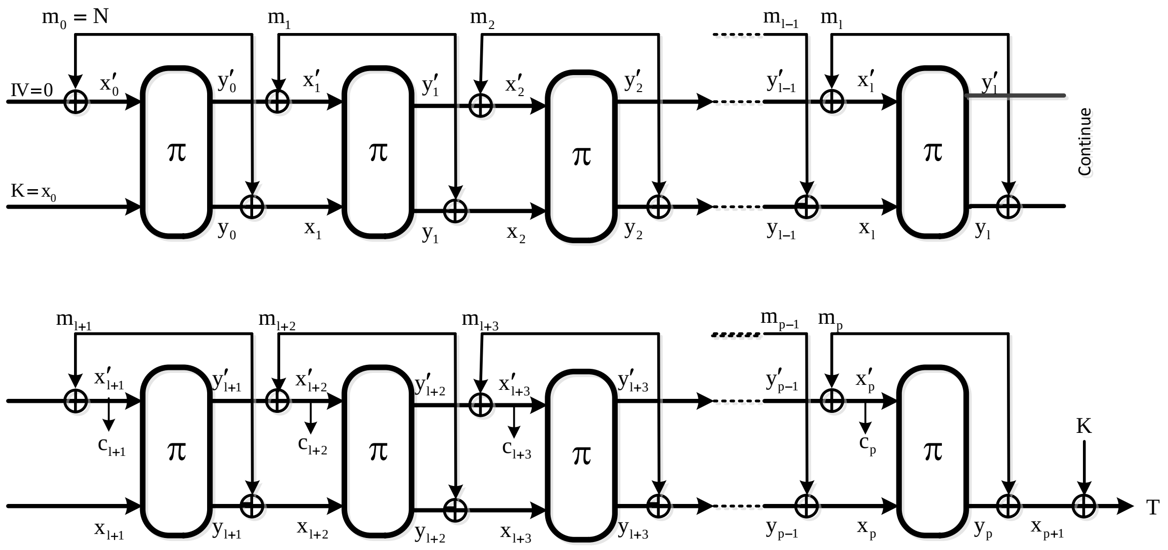


Figure 1.1: The JHAE mode of operation (the encryption and authentication), where $pad(A) = m_1 \| m_2 \| \ldots \| m_l$ and $pad(M) = m_{l+1} \parallel m_{l+2} \parallel \ldots \parallel m_p$

4. A bit '1' followed by a sequence of '0' is appended such that the padded message is a multiple of the block size n.

If there is the AD in the procedure, it is padded by a bit '1' followed by a sequence of '0' such that the padded AD would be a multiple of the block size n. The padded AD is processed in a way which is similar to the process of the message block with an exception that ciphertext blocks (c_i), are not produced for the AD blocks.

1.4.2 Decryption and Verification

JHAE decryption-verification procedure, depicted in Table 1.5, accepts an n-bit key K, an n-bit nonce N, a ciphertext C, a tag T, an optional AD (A), and it decrypts the ciphertext to get the message M and tag T'. If $T' = T$, it outputs M else it outputs $\perp$.

Table 1.4: The pseudo code of the encryption and authentication by JHAE

```
Algorithm1. $JHAE - E^{\pi}(K, N, M, A)$
Input: Key $K$ of $n$ bits, Nonce $N$ of $n$ bits, Associated data $A$
where $pad(A) = m_1 \| m_2 \| \ldots \| m_l$ and Message $M$
where $pad(M) = m_{l+1} \| m_{l+2} \| ... \| m_p$
Output: Ciphertext $C$, Tag $T$
$IV = 0; m_0 = N$
$x'_0 = IV \oplus m_0$; $x_0 = K$
$pad(A) \| pad(M) = m_1 \| m_2 \| ... \| m_p$
for $i = 0$ to $p - 1$ do:
    $y'_i \| y_i = \pi(x'_i \| x_i)$;
    $x'_{i+1} = y'_i \oplus m_{i+1}$;
    $x_{i+1} = y_i \oplus m_i$
end for
$y'_p \| y_p = \pi(x'_p \| x_p)$;
$x_{p+1} = y_p \oplus m_p$
$C = x'_{l+1} \| x'_{l+2} \| ... \| x'_p$
$T = x_{p+1} \oplus K$
Return $(C, T)$
```

1.5 Specification of the Permutation *Artemia*

In this section, we describe the permutations of $Artemia - 512$ and $Artemia - 256$.

1.5.1 $Artemia - 512$

$Artemia - 512$ is a 512-bit permutation ($Artemia - 512 : \{0,1\}^{512} \rightarrow \{0,1\}^{512}$) which includes the six rounds of $Artemia_{round} - 512 : \{0,1\}^{512} \rightarrow \{0,1\}^{512}$. Now, we explain the round function $Artemia_{round} - 512$.

Specification of $Artemia_{round} - 512$

$Artemia_{round} - 512$ is depicted in Fig 1.2. More precisely, at the beginning of the each round, the 512-bit input state is XORed by a round dependent constant value of the same length. The constant is introduced in Section 1.2. Next, the updated state is divided into four words of the 128-bit length. These 128-bit words are combined by a 4×4 recursive layer ($D1$), and other four words of the 128-bit length are produced. Then, each 128-bit value is passed an SBox layer ($S1$), which is 16 parallel 8×8-bit similar SBoxes and each SBox is applied to a byte of the internal state. Next, each 128-bit word is divided into four words of the 32-bit length and these 32-bit words are combined by a 4×4 recursive layer ($D2$), then other four words of the 32-bit length are produced. In this stage, there are four parallel recursive layers which one processes four words of the 32-bit length. Then, each 32-bit value is passed an SBox layer ($S2$), which

Table 1.5: The pseudo code of the decryption and verification by JHAE

```
Algorithm2. $JHAE - D^{\pi}(K, N, C, T, A)$
Input: Key $K$ of $n$ bits, Nonce $N$ of $n$ bits, Associated Data $A$
where $pad(A) = m_1 \| m_2 \| \ldots \| m_l$, ciphertext $C = c_1 \parallel c_2 \parallel ... \parallel c_p$ and Tag $T$
Output: Message $M$ or $\perp$
$IV = 0; m_0 = N$
$x'_0 = IV \oplus m_0$; $x_0 = K$
$x'_{l+1} \parallel x'_{l+2} \parallel ... \parallel x'_{l+p} = c_1 \parallel c_2 \parallel ... \parallel c_p$
for $i = 0$ to $l - 1$ do:
    $y'_i \parallel y_i = \pi(x'_i \parallel x_i)$;
    $x'_{i+1} = y'_i \oplus m_{i+1}$;
    $x_{i+1} = y_i \oplus m_i$
end for
for $i = l$ to $p - 1$ do:
    $y'_i \parallel y_i = \pi(x'_i \parallel x_i)$;
    $m_{i+1} = y'_i \oplus x'_{i+1}$;
    $x_{i+1} = y_i \oplus m_i$
end for
$y'_p \parallel y_p = \pi(x'_p \parallel x_p)$;
$x_{p+1} = y_p \oplus m_p$
$M = m_{l+1} \parallel m_{l+2} \parallel ... \parallel m_p$
$T' = x_{p+1} \oplus K$
if $T' = T$
  Return $M$
else
  Return $\perp$
```

Table 1.6: The pseudo Code of $Artemia_{round} - 512$

```
$Artemia_{round} - 512(X)$
Input:$X$ // a stream of 512-bit length
Output: $Y$ // a stream of 512-bit length
$C$ //a round constant of 512-bit;
$X \oplus C = W_3^1 \parallel W_2^1 \parallel W_1^1 \parallel W_0^1$; //$W_i^1 \in \{0,1\}^{128};\ 0 \leq i \leq 3$.
$W_3^2 \parallel W_2^2 \parallel W_1^2 \parallel W_0^2 = D1(W_3^1 \parallel W_2^1 \parallel W_1^1 \parallel W_0^1)$;
for $i = 0$ to 3 do:
    $W_i^3 = S1(W_i^2)$;
    $W_i^3 = W_{i,3}^3 \parallel W_{i,2}^3 \parallel W_{i,1}^3 \parallel W_{i,0}^3$; //$W_{i,j}^3 \in \{0,1\}^{32};\ 0 \leq j \leq 3$.
    $W_i^4 = W_{i,3}^4 \parallel W_{i,2}^4 \parallel W_{i,1}^4 \parallel W_{i,0}^4 = D2(W_{i,3}^3 \parallel W_{i,2}^3 \parallel W_{i,1}^3 \parallel W_{i,0}^3)$;
end for
for $i = 0$ to 3 do:
  for $j = 0$ to 3 do:
    $W_{i,j}^5 = S2(W_{i,j}^4)$;
    $W_{i,j}^5 = W_{i,j,3}^5 \parallel W_{i,j,2}^5 \parallel W_{i,j,1}^5 \parallel W_{i,j,0}^5$; //$W_{i,j,k}^5 \in \{0,1\}^8;\ 0 \leq k \leq 3$.
    $W_{i,j}^6 = W_{i,j,3}^6 \parallel W_{i,j,2}^6 \parallel W_{i,j,1}^6 \parallel W_{i,j,0}^6 = D3(W_{i,j,3}^5 \parallel W_{i,j,2}^5 \parallel W_{i,j,1}^5 \parallel W_{i,j,0}^5)$;
  end for
end for
for $i = 0$ to 3 do:
  for $j = 0$ to 3 do:
    for $k = 0$ to 3 do:
    $W_{i,j,k}^7 = S3(W_{i,j,k}^6)$;
    end for
  end for
end for
$Y = W_{3,3,3}^7 \parallel W_{3,3,2}^7 \parallel ... \parallel W_{0,0,0}^7$;
Return $Y$.
```

is four parallel 8×8-bit similar SBoxes and each SBox is applied to a byte of the internal state. Given the 16 words of the 32-bit length , each 32-bit word is divided into four bytes, the bytes are combined by a 4×4 recursive layer ($D3$), and other four bytes are produced. In this stage, there are 16 parallel recursive layers which one process four bytes of the internal state. Finally, each byte passes an SBox ($S3$). In the following, we explain the transformations $D1$, $S1$, $D2$, $S2$, $D3$ and $S3$. $S1$, $S2$ and $S3$ form the confusion layers of $Artemia_{round} - 512$, and $D1$, $D2$, and $D3$ form its diffusion layers.

The pseudo code of $Artemia_{round} - 512$ is represented in Table 1.6.

Transformations $S1$, $S2$ and $S3$

All the SBoxes used in the round function are the same and they are identical to the SBox of AES. The lookup table of the SBox is represented in Table 1.7. Fore example if $X = b2$ (a byte in the hexadecimal notation) is given as the

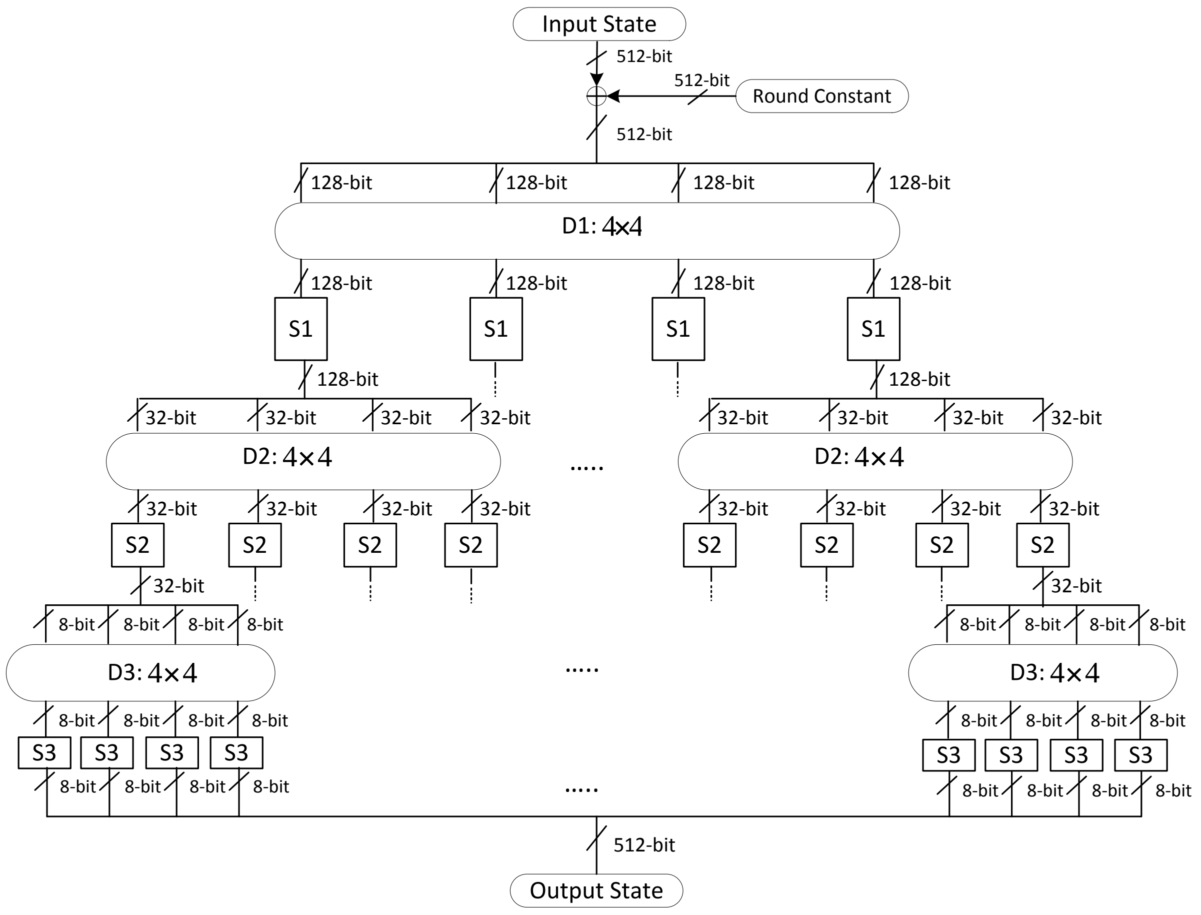


Figure 1.2: $Artemia_{round} - 512$

Table 1.7: The lookup table of the AES SBox.

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

input to the SBox, the output of the SBox would be $y = 37$ (in the hexadecimal notation).

Transformation $D1$

$D1$ is a MDS recursive diffusion layer given four words of the 128-bit length of X_0, X_1, X_2 and X_3, and it produces four words of the 128-bit length, Y_0, Y_1, Y_2 and Y_3. Such recursive (serial) MDS diffusion layers were first introduced in the designing of PHOTON [2] hash function and later also were used in the designing of LED [3]. The structure of the diffusion layer which is used in $D1$ was studied comprehensively in [4], and works as follows:

$$\left.\begin{array}{lll} Y_0 & = & X_0 \oplus X_2 \oplus X_3 \oplus L(X_1 \oplus X_3) \\ Y_1 & = & X_1 \oplus X_3 \oplus Y_0 \oplus L(X_2 \oplus Y_0) \\ Y_2 & = & X_2 \oplus Y_0 \oplus Y_1 \oplus L(X_3 \oplus Y_1) \\ Y_3 & = & X_3 \oplus Y_1 \oplus Y_2 \oplus L(Y_0 \oplus Y_2) \end{array}\right\} \tag{1.1}$$

where L is a linear function. If $L(X)$, $X \oplus L(X)$, $X \oplus L^3(X)$, and $X \oplus L^7(X)$ are invertible, the diffusion layer will be perfect [4] and provides the branch number 5. In addition, if L is an efficient linear function, the diffusion layer would be efficient. In $D1$, we use $L(X) = (X \ll 1) \oplus (X \gg 3)$ that satisfies the given conditions, i.e., $L(X)$, $X \oplus L(X)$, $X \oplus L^3(X)$, and $X \oplus L^7(X)$ are invertible. Hence, the diffusion layer $D1$ is perfect and efficient and its branch number is five.

Transformation D2

Similar to $D1$, $D2$ is also a recursive diffusion layer given four words of the 32-bit length produces other four words of 32-bit length. Its structure is identical to $D1$ with an exception that it works with the 32-bit words. In the case of $D2$, we have $L(X) = (X \ll 1) \oplus (X \gg 3)$. Since $L(X)$, $X \oplus L(X)$, $X \oplus L^3(X)$, and $X \oplus L^7(X)$ are invertible, $D2$ is a perfect diffusion layer and its branch number is five.

Transformation D3

Similar to $D1$ and $D2$, $D3$ is also a recursive diffusion layer given four bytes produces other four bytes. Its structure is identical to $D1$ and $D2$ with the two exceptions that it works with bytes and uses $L(X) = (X \oplus X \ll 1) \lll 1$. Since $L(X)$, $X \oplus L(X)$, $X \oplus L^3(X)$, and $X \oplus L^7(X)$ are invertible, $D3$ is a perfect diffusion layer and its branch number is five.

1.5.2 $Artemia - 256$

$Artemia - 256$ is a 256-bit permutation ($Artemia - 256 : \{0,1\}^{256} \to \{0,1\}^{256}$) which includes the six rounds of $Artemia_{round} - 256 : \{0,1\}^{256} \to \{0,1\}^{256}$. In the rest of this section we describe the round function $Artemia_{round} - 256$.

Specification of $Artemia_{round} - 256$

$Artemia_{round} - 256$ is depicted in Fig 1.3. More precisely, at the beginning of the each round, the 256-bit input state is XORed by a round dependent constant value of the same length. The constant is introduced in Section 1.2. Next, the updated state is divided into four words of the 64-bit length. These 64-bit words are combined by a 4×4 recursive layer ($D1$), and other four words of the 64-bit length are produced. Then, each 64-bit value is passed an SBox layer ($S1$), which is 8 parallel 8×8-bit similar SBoxes and each SBox is applied to a byte of the internal state. Next, each 64-bit word is divided into four words of the 16-bit length and these 16-bit words are combined by a 4×4 recursive layer ($D2$), then other four words of 16-bit length are produced. In this stage, there are four parallel recursive layers which one process four words of the 16-bit length. Then, each 16-bit value is passed an SBox layer ($S2$), which is two parallel 8×8-bit similar SBoxes and each SBox is applied to a byte of the internal state. Given 16 words of the 16-bit length, each 16-bit word is divided into two bytes, the bytes are combined by a 2×2 recursive layer ($D3$), and other two bytes are produced. In this stage, there are 16 parallel recursive layers which one processes two bytes of the internal state. Finally, each byte passes an SBox ($S3$). In the following we explain the transformations $D1$, $S1$, $D2$, $S2$, $D3$ and $S3$. $S1$, $S2$ and $S3$ form the confusion layers of $Artemia_{round} - 256$, and $D1$, $D2$ and $D3$ form its diffusion layers.

The pseudo code of $Artemia_{round} - 256$ is represented in Table 1.8.

Transformations $S1$, $S2$ and $S3$

Similar to $Artemia - 512$, any SBox used in the round function of $Artemia - 256$ is identical to the SBox of AES. The lookup table of the SBox is represented in

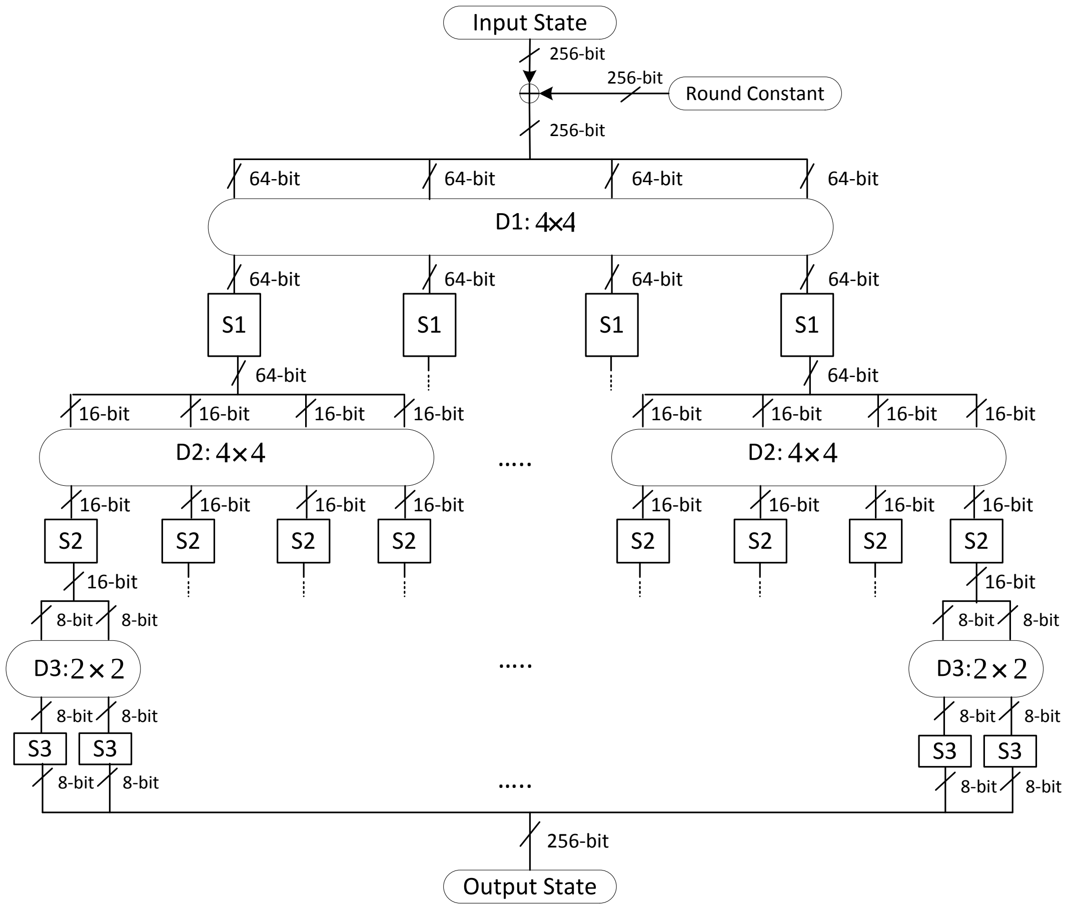


Figure 1.3: $Artemia_{round} - 256$

Table 1.8: The pseudo Code of $Artemia_{round} - 256$

```
$Artemia_{round} - 256(X)$
Input:$X$ // a stream of 256-bit length
Output: $Y$ // a stream of 256-bit length
$C$ //a round constant of 256-bit length
$X \oplus C = W_3^1 \parallel W_2^1 \parallel W_1^1 \parallel W_0^1$; $//W_i^1 \in \{0,1\}^{64};\ 0 \leq i \leq 3.$

$W_3^2 \parallel W_2^2 \parallel W_1^2 \parallel W_0^2 = D1(W_3^1 \parallel W_2^1 \parallel W_1^1 \parallel W_0^1)$;
for $i = 0$ to 3 do:
    $W_i^3 = S1(W_i^2)$;
    $W_i^3 = W_{i,3}^3 \parallel W_{i,2}^3 \parallel W_{i,1}^3 \parallel W_{i,0}^3$; $//W_{i,j}^3 \in \{0,1\}^{16};\ 0 \leq j \leq 3.$
    $W_i^4 = W_{i,3}^4 \parallel W_{i,2}^4 \parallel W_{i,1}^4 \parallel W_{i,0}^4 = D2(W_{i,3}^3 \parallel W_{i,2}^3 \parallel W_{i,1}^3 \parallel W_{i,0}^3)$;
end for
for $i = 0$ to 3 do:
  for $j = 0$ to 3 do:
    $W_{i,j}^5 = S2(W_{i,j}^4)$;
    $W_{i,j}^5 = W_{i,j,1}^5 \parallel W_{i,j,0}^5$; $//W_{i,j,k}^5 \in \{0,1\}^8;\ 0 \leq k \leq 1.$
    $W_{i,j}^6 = W_{i,j,1}^6 \parallel W_{i,j,0}^6 = D3(W_{i,j,1}^5 \parallel W_{i,j,0}^5)$;
  end for
end for
for $i = 0$ to 3 do:
  for $j = 0$ to 3 do:
    for $k = 0$ to 1 do:
    $W_{i,j,k}^7 = S3(W_{i,j,k}^6)$;
    end for
  end for
end for
$Y = W_{3,3,1}^7 \parallel W_{3,3,0}^7 \parallel ... \parallel W_{0,0,0}^7$;
Return $Y$.
```

Table 1.7.

Transformation $D1$

$D1$ is a recursive diffusion layer given four words of the 64-bit length of X_0, X_1, X_2 and X_3, and it produces other four words of the 64-bit length, Y_0, Y_1, Y_2 and Y_3 as shown in Equation 1.1. As we discussed about the recursive layers of $Artemia - 512$, if $L(X)$, $X \oplus L(X)$, $X \oplus L^3(X)$ and $X \oplus L^7(X)$ are invertible, the diffusion layer would be perfect [4] and provides the branch number five. In $D1$, $L(X) = (X \lll 1) \oplus (X \ggg 15)$ satisfying the given conditions is used. Hence, the diffusion layer $D1$ is perfect and its branch number is five.

Transformation D2

Similar to $D1$, $D2$ is also a recursive diffusion layer given four words of the 16-bit length produces other four words of the 16-bit length. Its structure is identical to $D1$ with two exceptions that it works with 16-bit words and uses a different L. In the case of $D2$, we have $L(X) = (X \lll 1) \oplus (X \ggg 1)$. Since $L(X)$, $X \oplus L(X)$, $X \oplus L^3(X)$, and $X \oplus L^7(X)$ are invertible, $D2$ is a perfect diffusion layer and its branch number is five.

Transformation D3

Similar to $D1$ and $D2$, $D3$ is also a recursive diffusion layer. However, it is a 2×2 recursive diffusion layer. It also is introduced in [4] and works as follows:

$$\left.\begin{array}{rcl} Y_0 & = & X_0 \oplus L(X_1) \\ Y_1 & = & X_1 \oplus L(Y_0) \end{array}\right\} \tag{1.2}$$

where L is a linear function. It is shown that if $L(X)$ and $X \oplus L(X)$ are invertible, the diffusion layer is perfect [4]. We use $L(X) = (X \lll 1) \oplus (X \ggg 3)$ (satisfying the conditions) in $D3$. Hence, $D3$ is a perfect diffusion layer and has the branch number three.

1.6 The Authenticated Encryption Artemia

We define Artemia-256 and Artemia-128 as the two variants of the family of the dedicated authenticated encryption which is named Artemia, as follows.

1.6.1 Artemia-256

Artemia-256 uses the permutation $Artemia - 512$ in the JHAE mode. Its key, nonce, AD blocks and message blocks have the length of 256-bit, and it produces the ciphertext blocks and a tag of the 256-bit length.

1.6.2 Artemia-128

Artemia-128 uses the permutation $Artemia-256$ in the JHAE mode. Its key, nonce, AD blocks and message blocks have the length of 128-bit, and it produces the ciphertext blocks and a tag of the 128-bit length.

Chapter 2

Security Goals

In this section, we clarify the security goals of Artemia. The padding process of Artemia uses the secret message length (64 bits) and the associated data length (24 bits) which determines the upper bound of the lenght of each field, *i.e.*, $2^{64}-1$ and $2^{24}-1$ respectively. It uses a nonce value as the public message, which is upper bounded by 256 bits for $Artemia-512$ and 128 bits for $Artemia-256$. The only restriction on the nonce value is that reuse of the padded-nonce value under a same key is not allowed. It is unnecessary that the nonce values have equal length (shorter values of the nonce value will be extended to maximum length by appending 0-bit to the left). Hence, the scheme does not provide any integrity or confidentiality if the legitimate user uses a same set (nonce, key) to encrypt two different sets of (plaintext, associated data). In addition, during the decryption, the scheme returns m if the received tag is correct and $\perp$ otherwise. Based on these assumptions, the security goals of Artemia are depicted in Table 2.1

Table 2.1: The security goals of Artemia

Goal	Artemia-256 bits of security	Artemia-128 bits of security
Confidentiality of the secret key	128	64
Confidentiality of the plaintext	128	64
Integrity of the plaintext	128	64
Integrity of the associated data	128	64
Integrity of the nonce	128	64

Chapter 3

Security Analysis

This chapter describes the security of Artemia in the two subsections: security analysis of JHAE and security analysis of the permutation *Artemia*.

3.1 Security Analysis of JHAE

In [1] it is shown that JHAE achieves the privacy (indistinguishability under chosen plaintext attack or IND-CPA) and integrity (integrity of ciphertext or INT-CTXT) up to $O(2^{n/2})$ queries, where the length of the used permutation is $2n$. One can summarize the security of JHAE in the two theorems as follows:

Theorem 1. *JHAE based on an ideal permutation $\pi : \{0,1\}^{2n} \rightarrow \{0,1\}^{2n}$ is (t_A, σ, ϵ)-indistinguishable from an ideal AE based on a random function RO and an ideal permutation π' with the same domain and range, for any t_A, then $\epsilon \leq \frac{\sigma(\sigma-1)}{2^{2n-1}} + \frac{\sigma^2}{2^{2n}} + \frac{\sigma^2}{2^n}$, where σ is the total number of blocks in queries to $JHAE-E$, π, and π^{-1}, by $\mathcal{A}$.*

Proof. [1]. □

Theorem 2. *For any adversary $\mathcal{A}$ that makes σ block queries to $JHAE-E$, π, or π^{-1} in total, JHAE based on an ideal permutation $\pi : \{0,1\}^{2n} \rightarrow \{0,1\}^{2n}$ is (t_A, σ, ϵ)-unforgeable, then $\epsilon \leq \frac{3\sigma^2}{2^n} + \frac{3q}{2^n}$.*

Proof. [1] □

3.2 Security Analysis of the Permutation *Artemia*

In this section, we investigate the security of *Artemia* against the differential and linear cryptanalysis. We show that any 2-round differential or linear characteristic has a minimum of 45 and 35 active SBoxes in $Artemia-512$ and $Artemia-256$ respectively. The numbers are a trivial lower bound for the

Table 3.1: The minimum number of active SBoxes and the differential and linear characteristic for *Artemia*

Artemia	# Rounds	# Minimum active SBoxes	Maximum probability of a differential characteristic	Maximum bias of a linear characteristic
$Artemia-512$	2	45	2^{-270}	2^{-180}
$Artemia-512$	4	90	2^{-540}	2^{-360}
$Artemia-256$	2	35	2^{-210}	2^{-140}
$Artemia-256$	4	70	2^{-420}	2^{-280}

minimum number of active SBoxes. The lower bound can be improved with respect to the diffusion layers of *Artemia* and the linear function that are used in the layers. On the other hand, the differential and linear characteristic of the SBox used in *Artemia* are 2^{-6} and 2^{-4} respectively. Hence, the probability of any 2-round differential characteristic for $Artemia-512$ and $Artemia-256$ are upper bounded by 2^{-270} and 2^{-210} respectively. Similarly, for any 2-round linear characteristic for $Artemia-512$ and $Artemia-256$, the biases are upper bounded by 2^{-180} and 2^{-140} respectively. By following a similar approach, any 4-round differential characteristic for $Artemia-512$ and $Artemia-256$ has a probability upper bounded by 2^{-540} and 2^{-420} respectively. And, any 4-round linear characteristic for $Artemia-512$ and $Artemia-256$ has a bias upper bounded by 2^{-360} and 2^{-280} respectively. These results are summarized in Table 3.2.

In the rest of this section, we show the correctness of our claims on the number of active SBoxes for $Artemia-512$ and $Artemia-256$.

3.2.1 $Artemia-512$

The Minimum Number of Active SBoxes in Two Rounds

In Fig 1.2, assume that a $D3$ recursive layer has been active. An active $D3$ guarantees at least five active SBoxes in $S2$ and $S3$. On the other hand, any active SBox in $S2$ comes from an active $D2$ which also guarantees five active SBoxes in $S1$ and $S2$. Hence, each active 128-bit word at the input of $D1$ in the $i-th$ round guarantees at least nine active SBoxes in the $i-th$ round and each active 128-bit word at the output of $D1$ in $i-th$ round guarantese at least nine active SBoxes in the $(i-1)-th$ round. Since the branch number of $D1$ is five, there are at least five active words in the input/output of any active $D1$. Hence, the minimum number of active SBoxes for two rounds of $Artemia-512$ is 45 (see also Fig. A.1 where the bold line is related to the lower bound). We summarize the minimum number of active SBoxes for two rounds of $Artemia-512$ in Table 3.2.

Table 3.2: The minimum number of active SBoxes for two rounds of $Artemia-512$.

# active words in the start of the round	# Minimum active SBoxes in the end of the round	# active words in the start of the next round	# Minimum active SBoxes in the end of the next round	# Minimum active SBoxes in two rounds of $Artemia-512$
1	36	4	9	45
2	27	3	18	45
3	18	2	27	45
4	9	1	36	45

Table 3.3: The minimum number of active SBoxes for two rounds of $Artemia-256$.

# active words in the start of the round	# Minimum active SBoxes in the end of the round	# active words in the start of the next round	# Minimum active SBoxes in the end of the next round	# Minimum active SBoxes in two rounds of $Artemia-256$
1	28	4	7	35
2	21	3	14	35
3	14	2	21	35
4	7	1	28	35

3.2.2 $Artemia-256$

The Minimum Number of Active SBoxes in Two Rounds

In Fig 1.3, assume that a $D3$ recursive layer has been active. An active $D3$ guarantees at least three active SBoxes in $S2$ and $S3$. On the other hand, any active SBox in $S2$ comes from an active $D2$ which also guarantees five active SBoxes in $S1$ and $S2$. Hence, each active 64-bit word at the input of $D1$ in the $i-th$ round guarantees at least nine active SBoxes in the $i-th$ round and each active 64-bit word at the output of $D1$ in $i-th$ round guarantees at least seven active SBoxes in the $(i-1)-th$ round. Since the branch number of $D1$ is five, there are at least five active words in the input/output of any active $D1$. Hence, the minimum number of active SBoxes for two rounds of $Artemia-256$ is 35 (see also Fig. A.2 where the bold line is related to the lower bound).

We summarize the minimum number of active SBoxes for two rounds of $Artemia-256$ in Table 3.3.

Chapter 4

Features

Artemia has provable security up to $O(2^{n/2})$ queries in the ideal permutation model where $2n$ is the length of the permutation. It is online, single-pass and supports the optional associated data. The Artemia security relies on the usage of nonces. However, it does not allow the reuse of a nonce under the same key. Artemia does not require the inverse of the permutation in the decryption function, this provides the resource efficiency.

The permutation *Artemia* has an efficient and a simple structure and is resistant to the differential and linear cryptanalysis. In order to design the permutation, we use the MDS recursive layers [2–4] that can be easily implemented in both software and hardware.

Compression to AES-GCM

AES-GCM is a block cipher based design while Artemia is a dedicated design. AES-GCM is a two-pass AE but Artemia is a single-pass one. Artemia does not contain key schedule and it does not use field multiplication. Artemia uses MDS recursive layers that can be easily implemented in both software and hardware. Hence we expect that Artemia provides a good performance in software and hardware.

Artemia-512 provides 128-bit integrity of the plaintext, associated data and nonce which is greater than the bounds provided by AES-GCM. The main disadvantage of Artemia is that it is serial.

Chapter 5

Design Rationale

Artemia has two main components: the JHAE mode and the permutation *Artemia*. In order to design each component, we use the publicly known elements to avoid any hidden weaknesses inside those components. In addition, the designers state that they have not hidden any weaknesses in this scheme. In the following, we give the rationale of the designing each component.

5.1 JHAE

JH [5] is a finalist of the SHA-3 competition and JHAE is a dedicated authenticated encryption mode based on the JH mode. JHAE is a sponge-like mode that uses a permutation and does not need any key schedule. On the other hand, in [1], it has been shown that JHAE is provably secure up to $O(2^{n/2})$. The important researches on JH hash mode have done in the duration of SHA-3 competition and show that there is not any significant vulnerability in the JH hash mode.

5.2 The Permutation *Artemia*

The permutation *Artemia* has the two main layers: the confusion and diffusion layer. In the confusion layer, the AES SBox having the appropriate characteristics is used. The diffusion layers are developed from the recently introduced recursive diffusion layers [2–4], that are simple and efficient. In [4], it is shown that these diffusion layers are perfect and provide the maximum branch number.

One can summarize the design rational of Artemia as follows:

- **Security**;
- **Simplicity**;
- **Using the known transformations as its components**;

- **Avoiding a Key Schedule**.

Chapter 6

Intellectual Property

Artemia itself is not covered by any patent and it is freely-available. On the other hand, as a building block, it uses the JH hash mode which is not covered by any patent as far as the designers of Artemia know but if there is any patent for JH then it would be applicable to the mode which is used in Artemia. If any of this information changes, the submitter will promptly (and within at most one month) announce these changes on the crypto-competitions mailing list.

Chapter 7

Consent

The submitter hereby consents to all decisions of the CAESAR selection committee regarding the selection or non-selection of this submission as a second-round candidate, a third-round candidate, a finalist, a member of the final portfolio, or any other designation provided by the committee. The submitter understands that the committee will not comment on the algorithms, except that for each selected algorithm the committee will simply cite the previously published analyses that led to the selection of the algorithm. The submitter understands that the selection of some algorithms is not a negative comment regarding other algorithms, and that an excellent algorithm might fail to be selected simply because not enough analysis was available at the time of the committee decision. The submitter acknowledges that the committee decisions reflect the collective expert judgments of the committee members and are not subject to appeal. The submitter understands that if he disagrees with published analyses then he is expected to promptly and publicly respond to those analyses, not to wait for subsequent committee decisions. The submitter understands that this statement is required as a condition of consideration of this submission by the CAESAR selection committee.

Bibliography

[1] J. Alizadeh, M. R. Aref, and N. Bagheri. JHAE: An Authenticated Encryption Mode Based on JH. Cryptology ePrint Archive, Report 2014/193, 2014. `http://eprint.iacr.org/`.

[2] J. Guo, T. Peyrin, and A. Poschmann. The PHOTON Family of Lightweight Hash Functions. In P. Rogaway, editor, *CRYPTO*, volume 6841 of *Lecture Notes in Computer Science*, pages 222–239. Springer, 2011.

[3] J. Guo, T. Peyrin, A. Poschmann, and M. J. B. Robshaw. The LED Block Cipher. In B. Preneel and T. Takagi, editors, *CHES*, volume 6917 of *Lecture Notes in Computer Science*, pages 326–341. Springer, 2011.

[4] M. Sajadieh, M. Dakhilalian, H. Mala, and P. Sepehrdad. Recursive diffusion layers for block ciphers and hash functions. In *FSE*, volume 7549 of *Lecture Notes in Computer Science*, pages 385–401. Springer, 2012.

[5] H. Wu. The Hash Function JH. Submission to NIST (round 3), 2011.

Appendix A

The Number of Active SBoxes

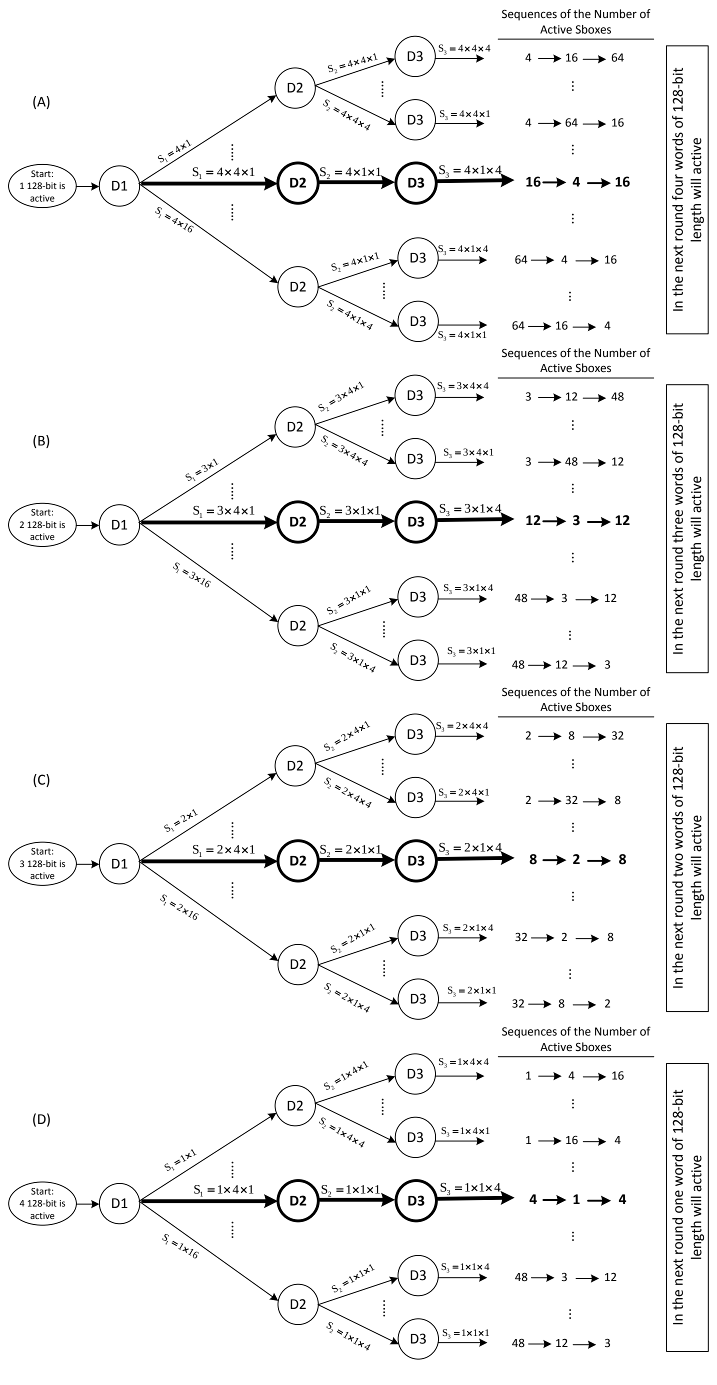
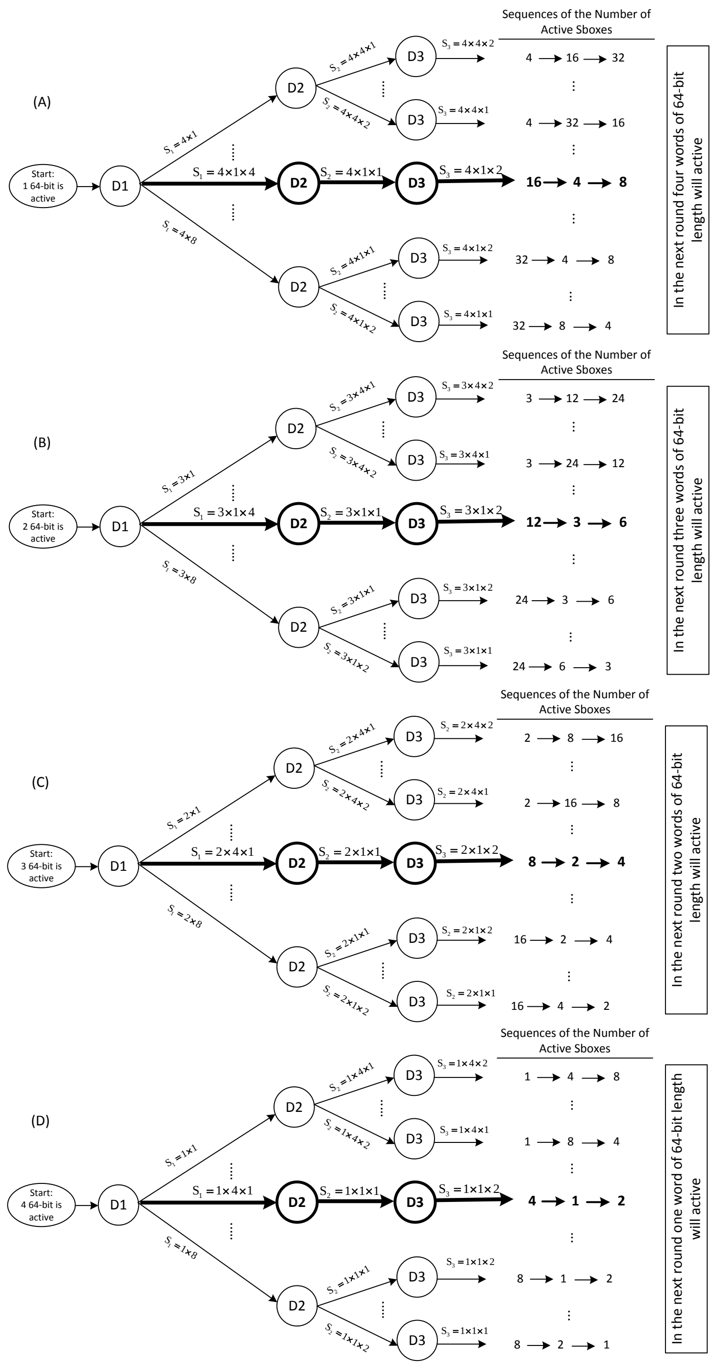


Figure A.1: The minimum number of SBoxes in $Artemia-512$. S_1, S_2, and S_3 are the minimum number of SBoxes in S1, S2, and S3 respectively.



(A)
Start: 1 64-bit is active
D1
D2
D3
S1 = 4×1×4
S2 = 4×1×1
S3 = 4×1×2
16 → 4 → 8
Sequences of the Number of Active Sboxes
In the next round four words of 64-bit length will active
(B)
Start: 2 64-bit is active
S1 = 3×1×4
S2 = 3×1×1
S3 = 3×1×2
12 → 3 → 6
In the next round three words of 64-bit length will active
(C)
Start: 3 64-bit is active
S1 = 2×4×1
S2 = 2×1×1
S3 = 2×1×2
8 → 2 → 4
In the next round two words of 64-bit length will active
(D)
Start: 4 64-bit is active
S1 = 1×4×1
S2 = 1×1×1
S3 = 1×1×2
4 → 1 → 2
In the next round one word of 64-bit length will active

Figure A.2: The minimum number of active SBoxes in $Artemia-256$. S_1, S_2, and S_3 are the minimum number of SBoxes in S1, S2, and S3 respectively.

Appendix B

Name

We named it *Artemia* because of:

> *Critical condition of Artemia Urmiana and possibility of extinction*[1].

[1] See http://saveurmia.com/main/2013/01/11/critical-condition-of-artemia-urmiana-and-possibility-of-extinction/