Analysis of the Authenticated Cipher MORUS (v1)

Aleksandra Mileva
Vesna Dimitrova
Vesselin Velichkov

BalkanCryptSec 2015
3–4 September 2015, Koper, Slovenia

Outline

In this paper is presented the first external analysis of the authenticated cipher MORUS.

The following new observations are reported:

- (1) distinguisher with probability 1 in a nonce-reuse scenario,
- (2) differential biases in the words of the state after initialization and
- (3) collision on the StateUpdate function of MORUS.

This results do not threaten the security of the scheme.

Introduction

- CAESAR - Competition for Authenticated Encryption: Security, Applicability, and Robustness
  - Announced in 2013
  - Initiated by the University of Illinois at Chicago, USA
  - Supported by the US National Institute of Standards and Technology (NIST)
- The goal of CAESAR - to select algorithm/s that can ensure both confidentiality and integrity within a single primitive
- 56 First-round candidates
- 29 Second-round candidates

Introduction

- MORUS - one of the Second-round candidates
  - a very promising design - both efficient and secure
- Idea:
  - WG4 Meeting on Authenticated Encryption, COST CryptoAction IC1306 "Cryptography for Secure Digital Interaction", co-located with Eurocrypt 2015

Description of MORUS

Description of MORUS

Table: Notation.

Symbol	Meaning
$\oplus$	Bit-wise exclusive OR
$\wedge$	Bit-wise AND
$\|\|$	Concatenation
$\lll$	Bit rotation to the left
$\ggg$	Bit rotation to the right
$b^{(n)}$	A sequence of n binary digits $b \in \{0, 1\}$
$\|X\|$	Length of the bit string X (in bits)
$\overline{x}$	Negation of all bits of x i.e. $\overline{x} = x \oplus 1^{(n)}$
Rotl_xxx_yy(x, b)	Divide the xxx-bit block x into 4 yy-bit words and rotate each word left by b bits. Example: Rotl_128_32(x, b) is used in MORUS-640 and Rotl_256_64(x, b) is used in MORUS-1280.
Rotr_xxx_yy(x, b)	Analogous to Rotl_xxx_yy(x, b) with a right rotation
LSB, MSB	Least Significant Bit, Most Significant Bit
IV	Initialization Vector (Nonce)

Description of MORUS

- Bitwise operations:
  bit shift, AND and XOR
- The size of the internal state:
  640 for MORUS-640 or 1280 bits for MORUS-1280
- Supported keys:
  128 and 256 bit keys loaded into the input state together with a public 128 bit IV and 3 specified constants
- The main building block:
  - The state update function StateUpdate(S, M), where S is the state, and M is a message block with length $|S|/5$
  - This function consists of 5 rounds with similar operations
  - In each round, only two state elements are modified:
    - one with left rotation with coefficients w_i $(i \in \{0, 1, 2, 3, 4\})$
    - other with Rotl_xxx_yy operation with coefficients b_i, $(i \in \{0, 1, 2, 3, 4\})$.

Description of MORUS

- Operates in four phases:
  - (1) Initialization
    - five state elements are initialized (IV, K, $1^{(128)}$, const_0 and const_1)
    - the state is updated by 16 applications of StateUpdate
    - the result is XOR-ed with the key K
  - (2) Processing of associated data
    - the associated data AD is processed using StateUpdate
  - (3) Encryption
    - the plaintext P is encrypted in blocks of 128 bits
  - (4) Finalization
    - the authentication tag is generated by 8 applications of StateUpdate
    - the output is a ciphertext together with an authentication tag of size at most 128 bits

Distinguisher

Distinguisher on MORUS-640 in a nonce-reuse scenario

- $|\text{AD}| = 128$, P consist of a single 128-bit block M
- $S^0 = (s_0, s_1, s_2, s_3, s_4)$ - the output of the init. phase
- the output after the second phase is expressed as:

$$S^1 = (x_0, x_1, x_2, x_3, x_4) = \text{StateUpdate}(S^0, \text{AD}) \ , \qquad (1)$$

- the output of the third phase is C and the state S^2:

$$C = M \oplus x_0 \oplus (x_1 \lll 96) \oplus (x_2 \wedge x_3) \ , \qquad (2)$$

$$S^2 = (z_0, z_1, z_2, z_3, z_4) = \text{StateUpdate}(S^1, M) \qquad (3)$$

- in the final phase T is obtained as:
  1. $\text{tmp} = z_3 \oplus (\text{adlen} \,||\, \text{msglen})$
  2. $z_4 = z_4 \oplus z_0$
  3. For $i = 2$ to 9 do $S^{i+1} = \text{StateUpdate}(S^i, \text{tmp})$
  4. $T = \oplus_{i=1}^{4} S_i^{10}$

Distinguisher on MORUS-640 in a nonce-reuse scenario

Theorem

Let $S^0 = (s_0, s_1, s_2, s_3, s_4)$ *be the state of MORUS-640 after initialization under* 128 *bit secret key* K *and* 128 *bit public* IV. *Let* $\mathrm{AD}_1 = 0^{(128)}$ *and* $\mathrm{AD}_2 = 0^{(127)}||1$ *be two* 128 *bit blocks of authenticated data that differ only in their least significant bit. Finally, let* X *and* Y *be the internal states of MORUS after the second phase (processing of associated data) under* AD_1 *and* AD_2 *respectively:*

$$X = (x_0, x_1, x_2, x_3, x_4) = \mathrm{StateUpdate}(S^0, \mathrm{AD}_1) \ , \tag{4}$$

$$Y = (y_0, y_1, y_2, y_3, y_4) = \mathrm{StateUpdate}(S^0, \mathrm{AD}_2) \ . \tag{5}$$

Then the following statements are true:

1. $x_0 = y_0$.
2. x_1 *and* y_1 *differ only in the 33-th bit.*
3. x_2 *and* y_2 *differ only in the 89-th bit.*
4. x_3 *and* y_3 *differ only in the 106-th and 107-th bit.*
5. x_4 *and* y_4 *differ only in the 108-th and 115-th bit (with probability* 1*) and in the 33-th bit with probability* $1/2$.

Note: all bits within the 128 *bit words* X *and* Y *are counted starting from MSB (bit* 1*) down to LSB (bit* 128*).*

Construction of the distinguisher

- Let AD_1, AD_2, $X = (x_0, x_1, x_2, x_3, x_4)$ and $Y = (y_0, y_1, y_2, y_3, y_4)$ be as in Theorem
- Denote $x_j = (x_{j0}, x_{j1}, x_{j2}, x_{j3})$ and $y_j = (y_{j0}, y_{j1}, y_{j2}, y_{j3})$, where $|x_{ji}| = |y_{ji}| = 32$ bits and $0 \leq j \leq 4$, $1 \leq i \leq 3$.
- Let M be a 128 bit message block.
- According to the enc. function the two ciphertexts C_1 and C_2 resp. under AD_1 and AD_2 are expressed as:

$$C_1 = M \oplus x_0 \oplus (x_1 \lll 96) \oplus (x_2 \wedge x_3) \ , \tag{6}$$

$$C_2 = M \oplus y_0 \oplus (y_1 \lll 96) \oplus (y_2 \wedge y_3) \ . \tag{7}$$

Distinguisher - analysis

Our analysis (using the statements from the Theorem) shows that C_1 and C_2 differ in 4 bits in total.

- they differ in the 65-th bit with probability 1 and
- in the 89-th, 106-th and 107-th bit with probability 1/2 (for each of the three bits).

Conclusions:

- Given the ciphertext C_1 under message (AD_1, M), an attacker can predict 125 bits of an (unknown) ciphertext C_2 under a different message (AD_2, M) with probability 1.
- The same probability for a random oracle is 2^{-125}.

Distinguisher - analysis

- The same technique can also trivially be extended to distinguish pairs of plaintexts (AD_1, M_1) and (AD_2, M_2) that contain also differences in the message words i.e. $AD_1 \neq AD_2$ and $M_1 \neq M_2$ and such that the messages M_1 and M_2 have the same length as a single block: $|M_1| = |M_2| = 128$.
- In this case a difference in any k bits of M_1 and M_2 (other than bits 89, 106 and 107) results in a difference in the corresponding k bits of the ciphertexts C_1 and C_2 with probability 1.

Differential Biases After Initialization

Differential Biases After Initialization

- We describe differential biases in the words of the state after initialization in reduced word versions of MORUS.
  - MORUS-160, version with 8 bit words (160 bit state),
  - MORUS-200, version with 10 bit words (200 bit state)
- The rotation constants are the same as in the original version modulo the word size.
- For both small versions we initialize the input state to a fixed random value.
- We try all differences in one word of the IV and we measure the probabilities with which differences in the words of the state after the initialization appear.

Differential Biases After Initialization - Example

Let X be an input state to MORUS-160 initialized as:

$$X = (\texttt{3B}, \texttt{77}, \texttt{E3}, \texttt{DE}, \texttt{F0}, \texttt{EC}, \texttt{D9}, \texttt{DD}, \texttt{2F}, \texttt{D0}, \texttt{F5}, \texttt{C7}, \texttt{DF}, \texttt{7E}, \texttt{C8}, \texttt{DD}, \texttt{E7}, \texttt{1D}, \texttt{3A}, \texttt{73})$$

where IV $= \texttt{3B77E3DE}$ and $K = \texttt{F0ECD9DD}$.
Let $X' = X \oplus \Delta X$ be a second input state that differs from X only in the first (i.e. leftmost) word of the IV by the difference:

$$\Delta X = (\texttt{AB}, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$$

Then the sixth word of the output difference ΔY between the corresponding output states Y and Y' after initialization is expected to be $\texttt{4}$ with probability $2^{-4.19}$.

$$P(\Delta X = (\texttt{AB}, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \xrightarrow{\text{Init.}}$$
$$\Delta Y = (*, *, *, *, *, \texttt{4}, *, *, *, *, *, *, *, *, *, *, *, *, *, *)) = 2^{-4.19}$$

where $*$ denotes an unknown difference.

Differential Biases After Initialization

Conclusions:

- For MORUS-160 the best observed probability is $2^{-4.19}$, while for MORUS-200 it is 2^{-6}.
- These values are higher than what is expected in the random case resp. 2^{-8} and 2^{-10}.
- Due to the bitwise nature of the cipher, these results also suggest that for the original 32 and 64 bit versions of MORUS resp. MORUS-640 and MORUS-1280 we can expect probabilities significantly higher than resp. 2^{-32} and 2^{-64}.

Collisions in the StateUpdate(S, M) Function

Collisions in the StateUpdate(S, M) Function

Proposition

Let $M,\ x_i \in \mathbb{Z}_2^n,\ \ i \geq 0$ *and* $n \in \{128, 256\}$, *and let* $w_i \leq n$ *and* $b_i \leq n/4$ *be some rotation constants. Then the following function* $F_M : (\mathbb{Z}_2^n)^5 \rightarrow (\mathbb{Z}_2^n)^5$ *is a permutation on* $(\mathbb{Z}_2^n)^5$*:*

$$
\begin{aligned}
&F_M(x_i, x_{i+1}, x_{i+2}, x_{i+3}, x_{i+4}) \\
&= (\text{Rotl\_xxx\_yy}(x_i \oplus (x_{i+1} \wedge x_{i+2}) \oplus x_{i+3} \oplus M, b_i), x_{i+1}, x_{i+2}, (x_{i+3} \lll w_i), x_{i+4})
\end{aligned}
$$

Each round of StateUpdate(S, M) can be represented as five applications of the function F_{m_i}, where $m_i = M$ for $i = \{1, 2, 3, 4\}$ and $m_i = 0^{(n)}$ for $i = 0$:

$$(s_i, s_{(i+1)}, s_{(i+2)}, s_{(i+3)}, s_{(i+4)}) = F_{m_i}(s_i, s_{(i+1)}, s_{(i+2)}, s_{(i+3)}, s_{(i+4)})$$

The function StateUpdate(S, M) is a permutation on $(\mathbb{Z}_2^n)^5$.

Collisions in the StateUpdate(S, M) Function

Proposition

Let $w_i \leq n$ *and* $b_i \leq n/4$, *where* $i \geq 0$ *and* $n = \{128, 256\}$, *be some rotation constants. For all* $M_1, M_2 \in \mathbb{Z}_2^n$ *and each vector* $(x_0, x_1, x_2, x_3, x_4) \in (\mathbb{Z}_2^n)^5$, *the following holds:*

$$F_{M_1}(x_0, x_1, x_2, x_3, x_4) = F_{M_2}(M_1 \oplus M_2 \oplus x_0, x_1, x_2, x_3, x_4)$$

Corollary

For all $M_1, M_2 \in \mathbb{Z}_2^n, n = \{128, 256\}$ *and each vector* $(x_0, x_1, x_2, x_3, x_4) \in (\mathbb{Z}_2^n)^5$, *the following holds:*

$$\begin{aligned}&\text{StateUpdate}((x_0, x_1, x_2, x_3, x_4), M_1) = \\ &\text{StateUpdate}((x_0, M_1 \oplus M_2 \oplus x_1, M_1 \oplus M_2 \oplus x_2, M_1 \oplus M_2 \oplus x_3, M_1 \oplus M_2 \oplus x_4), M_2)\end{aligned}$$

On Producing a Tag Forgery

On Producing a Tag Forgery

- Let (M_1, AD_1) be a pair of a 128 bit message block and a 128 bit associated data block encrypted by MORUS under key K and IV
- $S^0 = (s_0, s_1, s_2, s_3, s_4)$ be the output of the initialization phase and
- $S^1 = (x_0, x_1, x_2, x_3, x_4) = \mathrm{StateUpdate}(S^0, \mathrm{AD}_1)$ be the output of the next phase (processing of associated data).

We want to find a 128 bit block ΔM and a 128 bit associated data block $\mathrm{AD}_2 \neq \mathrm{AD}_1$ that will relate $\mathrm{StateUpdate}(S^0, \mathrm{AD}_1)$ to $\mathrm{StateUpdate}(S^0, \mathrm{AD}_2)$ as follow:

$$\begin{aligned}\mathrm{StateUpdate}(S^0, \mathrm{AD}_2) &= (x_0, x_1 \oplus \Delta M, x_2 \oplus \Delta M, x_3 \oplus \Delta M, x_4 \oplus \Delta M) \\ &= (x_0, x_1, x_2, x_3, x_4) \oplus (0^{(n)}, \Delta M, \Delta M, \Delta M, \Delta M) \\ &= \mathrm{StateUpdate}(S^0, \mathrm{AD}_1) \oplus (0^{(n)}, \Delta M, \Delta M, \Delta M, \Delta M).\end{aligned}$$

On Producing a Tag Forgery

If such AD_2 and ΔM exist, then using Corollary we can construct the message $M_2 = M_1 \oplus \Delta M$ that will produce a collision in the internal state after the encryption phase:

$$
\begin{aligned}
&\mathrm{StateUpdate}(\mathrm{StateUpdate}(S^0, \mathrm{AD}_1), M_1) \\
&= \mathrm{StateUpdate}((x_0, x_1, x_2, x_3, x_4), M_1) \\
&= \mathrm{StateUpdate}((x_0, M_1 \oplus M_2 \oplus x_1, M_1 \oplus M_2 \oplus x_2, M_1 \oplus M_2 \oplus x_3, M_1 \oplus M_2 \oplus x_4), M_2) \\
&= \mathrm{StateUpdate}((x_0, \Delta M \oplus x_1, \Delta M \oplus x_2, \Delta M \oplus x_3, \Delta M \oplus x_4), M_2) \\
&= \mathrm{StateUpdate}(\mathrm{StateUpdate}(S^0, \mathrm{AD}_2), M_2)
\end{aligned}
$$

Since both pairs (AD_1, M_1) and (AD_2, M_2) have the same `adlen` and `msglen`, the collision ultimately results in the same tag for the messages M_1 and M_2.

On Producing a Tag Forgery

As to finding the required blocks ΔM and AD_2 we show that this is equivalent to solving the following system of equations for $(x_0, x_1, \Delta A, \Delta M)$, where $\Delta A = \mathrm{AD}_1 \oplus \mathrm{AD}_2$:

$$\begin{cases} \mathrm{Rotr\_xxx\_yy}(\Delta M \ggg w_3, b_1) = \Delta A \\ \mathrm{Rotr\_xxx\_yy}(\Delta M \ggg w_4, b_2) = \Delta A \\ \mathrm{Rotr\_xxx\_yy}(\Delta M, b_3) = (\Delta M \ggg w_3) \oplus \Delta A \\ \mathrm{Rotr\_xxx\_yy}(\Delta M, b_4) = (x_0 \wedge x_1) \oplus (x_0 \wedge (x_1 \oplus \Delta M)) \oplus (\Delta M \ggg w_4) \end{cases}$$

But,

No solution was found apart from the trivial one:
$(\Delta M, \Delta A) = (0, 0)$.

Conclusions

We presented the first external analysis of the authenticated cipher MORUS.

We reported the following new observations:

- (1) distinguisher with probability 1 in a nonce-reuse scenario,
- (2) differential biases in the words of the state after initialization and
- (3) collision on the StateUpdate function of MORUS.

Our results do not threaten the security of the scheme and indicate that MORUS is a well-designed cipher.

Thank you for your attention!