SCIENCE CHINA
Information Sciences

• LETTER •

March 2019, Vol. 62 039109:1–039109:3
https://doi.org/10.1007/s11432-017-9457-x

Real-time state recovery attack against MORUS in nonce-misuse setting

Tairong SHI* & Jie GUAN

Information Science and Technology Institute, Zhengzhou 450001*, China*

Received 12 May 2017/Revised 6 December 2017/Accepted 4 May 2018/Published online 3 January 2019

Citation Shi T R, Guan J. Real-time state recovery attack against MORUS in nonce-misuse setting. Sci China Inf Sci, 2019, 62(3): 039109, https://doi.org/10.1007/s11432-017-9457-x

Dear editor,
CAESAR [1] is a large-scale cryptographic competition supported by the US National Institute of Standards and Technology, which mainly calls for authenticated ciphers. MORUS [2], an authenticated stream cipher, has been selected as one of the third-round candidates. As a nonce-based scheme, the designer does not make a clear claim on the security margin of MORUS in nonce-misuse setting. In fact, in practical applications, the reuse of nonce is difficult to avoid especially when operating in hostile environments. Herein, a state recovery attack on MORUS in a misuse setting is proposed. It shows that if the initialization vector (IV) is reused seven times, then all 640 bits of the internal state can be revealed with negligible computational complexity and data complexity, and the success probability is one. Finally, we present the forgery attack on the authentication process of MORUS. To our best knowledge, a full-state recovery attack on MORUS does not exist.

As a nonce-based authenticated cipher, the designer does not make a certain claim on the robustness level in nonce-misuse setting. Mileva et al. [3] proposed a distinguishing attack on MORUS in a nonce-reuse setting. However, the designer Wu indicated that the attack in document [3] was non-ideal for a non-full-state recovery. Nozaki et al. [4] studied the power analysis attack on MORUS; their experiments using an FPGA proved the validity of the proposed method and the vulnerability of MORUS against the power analysis attack. A sat-based cryptanalysis of MORUS was performed by Dwivedi et al. [5]. Their analysis revealed that these schemes, as submitted to CAESAR, provided strong resistance against SAT-based state recoveries. In addition, our team has heavily studied MORUS, including the research on the confusion and diffusion properties of the initialization [6], and the theoretical support for MORUS's resistance against collision attacks through the usage of mixed integer programming [7]. However, a valid state recovery attack on MORUS has not been reported hitherto.

Description of MORUS. MORUS is a family of authenticated stream ciphers designed by Wu and Huang, which has been selected as a third-round candidate in the CAESAR competition. Three MORUS versions: MORUS-640-128, MORUS-1280-128, and MORUS-1280-256 are recommended in different internal states and key sizes. In this study, we only analyzed the security of MORUS-640-128, and all the "MORUS" hereafter refers to "MORUS-640-128" without special instructions. MORUS uses five 128-bit registers in its internal state. In each step, it processes a 128-bit plaintext P_i. This function consists of five rounds with similar operations that update the state S^i. The message block is used in rounds 2–5 but not in round 1.

In each step of the encryption, a 128-bit plaintext block P_i is used to update the state. Let msglen represent the bit length of plaintext and

*Corresponding author (email: strwanzi@163.com)

$v = \lceil \text{msglen}/128 \rceil$. We write $S_0^i = S_{0,0}^i, S_1^i = S_{0,1}^i, S_2^i = S_{0,2}^i, S_3^i = S_{0,3}^i, S_4^i = S_{0,4}^i$ hereafter. From $i = 0$ to $v-1$, the encryption and update state operation are as shown below:

$$C_i = P_i \oplus S_0^i \oplus (S_1^i \lll 96) \oplus (S_2^i \ \& \ S_3^i),$$
$$S^{i+1} = \text{StateUpdate}(S^i, P_i).$$

State recovery attack in nonce-reuse setting. We only require pairs of plaintexts with special differences, (P_i, P_{i+1}, P_{i+2}) and $(P_i', P_{i+1}', P_{i+2}')$, as well as the corresponding ciphertexts (C_i, C_{i+1}, C_{i+2}) and $(C_i', C_{i+1}', C_{i+2}')$ at three consecutive steps i, $i+1$ and $i+2$ under the same nonce. Subsequently, we define $\Delta P_i = P_i' \oplus P_i$ and $\Delta C_i = C_i' \oplus C_i$.

By analyzing the encryption of MORUS, we know that the only nonlinear operation in the update function is " & ". To eliminate the influence caused by &, we perform the following observation.

Observation 1. The function $y = f(x) = a \ \& \ x$ is a bijection iff $a = 1^n$, where $x, y, a \in F_2^n$.

To satisfy the condition of bijection in Observation 1, in the process of recovering the internal state, we chose the input differences as the following principle.

Principle 1. Build more linear relations between the known information (such as the input differences, output differences, and known states) and the unknown state bits.

Except for 1^{128}, we also used some special input differences. For the convenience of description, we agree that $\alpha_0 = (1^{32}, 0^{32}, 0^{32}, 0^{32})$, $\alpha_1 = (0^{32}, 1^{32}, 0^{32}, 0^{32})$, $\alpha_2 = (0^{32}, 0^{32}, 1^{32}, 0^{32})$, $\alpha_3 = (0^{32}, 0^{32}, 0^{32}, 1^{32})$.

$e_j = (\gamma_{j_0}, \gamma_{j_1}, \gamma_{j_2}, \gamma_{j_3})$ consists of two 32-bit vectors $\gamma_0 = (1,0,1,0,\ldots,1,0)$, $\gamma_1 = (0,1,0,1,\ldots,0,1)$, where j ranges from 0 to 15 while $j_0, j_1, j_2, j_3 \in \{0,1\}$. The details are as follows: $e_0 = (\gamma_0, \gamma_0, \gamma_0, \gamma_0)$, $e_1 = (\gamma_0, \gamma_0, \gamma_0, \gamma_1)$, $\ldots$, $e_{15} = (\gamma_1, \gamma_1, \gamma_1, \gamma_1)$.

Let $D = \{1^{128}, \alpha_0, \alpha_1, \alpha_2, \alpha_3, e_0, e_1, \ldots, e_{15}\}$ be the set of input differences. We will demonstrate that D is the most efficient set satisfying the principle above in the following.

For convenience, we let $i = 0$, namely we use plaintexts and the corresponding ciphertexts at the first three steps. Our goal is to recover the internal state $S^1 = (S_0^1, S_1^1, S_2^1, S_3^1, S_4^1)$. Using state S_3^1 as an example, the recovery process is illustrated in detail as follows.

Recovering state S_3^1. Let $\Delta P_0 = \alpha_0$ and $\Delta P_1 = \Delta P_2 = 0^{128}$. According to $S^1 = \text{StateUpdate}(S^0, P_0)$, the substrate differences of ΔS^1 are expressed as

$$\begin{cases} \Delta S_0^1 = 0^{128}, \\ \Delta S_1^1 = \alpha_2, \\ \Delta S_2^1 = \alpha_3, \\ \Delta S_3^1 = 0^{128}, \\ \Delta S_4^1 = \text{Rotl}(\alpha_2 \ \& \ S_0^1, \ 13). \end{cases} \tag{1}$$

Subsequently, owing to the encryption function

$$C_i = P_i \oplus S_0^i \oplus (S_1^i \lll 96) \oplus (S_2^i \ \& \ S_3^i), \tag{2}$$

we can obtain

$$\begin{aligned} \Delta C_1 = \Delta P_1 \oplus \Delta S_0^1 \oplus (\Delta S_1^1 \lll 96) \oplus (\Delta S_2^1 \\ \& \ S_3^1) \oplus (\Delta S_3^1 \ \& \ S_2^1) \oplus (\Delta S_2^1 \ \& \ \Delta S_3^1). \end{aligned} \tag{3}$$

Substituting (1) in (3) yields:

$$\begin{aligned} \Delta C_1 &= (\Delta S_1^1 \lll 96) \oplus (\Delta S_2^1 \ \& \ S_3^1) \\ &= (\alpha_2 \lll 96) \oplus (\alpha_3 \ \& \ S_3^1). \end{aligned} \tag{4}$$

We consider each state as four 32-bit blocks; more specifically, a state, S_i^1 is divided into four 32-bit blocks. $S_i^1 = (S_{i,1}^1, S_{i,2}^1, S_{i,3}^1, S_{i,4}^1)$ and $\Delta C_i = (\Delta C_{i,1}, \Delta C_{i,2}, \Delta C_{i,3}, \Delta C_{i,4})$, where $0 \leqslant i \leqslant 4$. From $\alpha_3 = (0^{32}, 0^{32}, 0^{32}, 1^{32})$, we know that the only non-zero block of $\alpha_3 \ \& \ S_3^1$ is $S_{3,4}^1$, the other blocks are 0, and the only non-zero block of $(\alpha_2 \lll 96) = \alpha_3$ is the last block, whereas other blocks are 0. Therefore, $\Delta C_{1,4} = 1^{32} \oplus S_{3,4}^1$, where ΔC_1 is known; hence, $S_{3,4}^1 = \Delta C_{1,4} \oplus 1^{32}$.

The same technique can also be trivially extended to $\Delta P_0 = \alpha_1, \alpha_2, \alpha_3$. Therefore, we can obtain $S_{3,1}^1 = \Delta C_{1,1} \oplus 1^{32}$,$S_{3,2}^1 = \Delta C_{1,2} \oplus 1^{32}$,$S_{3,3}^1 = \Delta C_{1,3} \oplus 1^{32}$.

The uniqueness of differences. In terms of blocks, if the first block of ΔP_0 is not 1^{32}, $\Delta P_{0,0} \neq 1^{32}$, there is at least one bit 0 at the first block. Let β_i refer to a 32-bit vector whose i-th bit is 0 and the other bits are arbitrary. Let $\Delta P_0 = (\beta_i, 0^{32}, 0^{32}, 0^{32})$. The substrate differences of ΔS^1 are expressed as

$$\begin{cases} \Delta S_0^1 = 0^{128}, \\ \Delta S_1^1 = (0^{32}, 0^{32}, \beta_{(i+1) \bmod 32}, 0^{32}), \\ \Delta S_2^1 = (0^{32}, 0^{32}, 0^{32}, \beta_{(i-7) \bmod 32}), \\ \Delta S_3^1 = 0^{128}. \end{cases}$$

Therefore, according to (3), we can obtain $\Delta C_{1,4} = \beta_{(i+1) \bmod 32} \oplus (S_{3,4}^1 \ \& \ \beta_{(i-7) \bmod 32})$. The $(i-7)$ mod 32-th bit of $\beta_{(i-7) \bmod 32}$ is 0. As a result of Observation 1, the value of $(i-7)$ mod 32-th bit in $S_{3,4}^1$ is uncertain.

Thus, α_0, α_1, α_2, α_3 are the most efficient plaintext differences respectively on the condition of different blocks.

The entire recovery process is shown in Table 1.

Table 1 Summary of state recovery

Recovery state	ΔP_0	ΔP_1	ΔP_2	Reuse time
S_3^1	$\alpha_0, \alpha_1, \alpha_2, \alpha_3$	0^{128}	0^{128}	4
S_2^1	$e_j, j \in [0,15]$	0^{128}	0^{128}	1
S_3^2	0^{128}	1^{128}	0^{128}	1
S_2^2	0^{128}	$e_j, j \in [0,15]$	0^{128}	1
S_0^1, S_1^1, S_4^1	0	0	0	0

We can infer from Table 1 that the entire recovery process reuses nonce seven times with negligible computational complexity and data complexity, as well as the success probability of one.

Forgery attack. However, if IV was reused in MORUS, a serious state recovery attack would occur, and the messages can be forged once the state is recovered. According to the authentication, we can tamper ciphertext C' from the arbitrary step and generate the corresponding tag T'. The new pair of (C', T') would pass the verification, and then the forgery process would be realized.

Acknowledgements This work was supported by National Natural Science Foundation of China (Grant Nos. 61572516, 61602514, 61272041, 61272488).

References

1 The CAESAR Committee. Competition for authenticated encryption: security, applicability, and robustness. 2014. http://competitions.cr.yp.to/caesar.html

2 Wu H, Huang T. The authenticated cipher MORUS (v1.1). 2016. http://competitions.cr.yp.to/round2/morusv11.pdf

3 Mileva A, Dimitrova V, Velichkov V. Analysis of the authenticated cipher MORUS (v1). In: Proceedings of International Conference on Cryptography and Information Security in the Balkans, Koper, 2015. 45–59

4 Nozaki Y, Yoshikawa M. Power analysis attack for a fast authenticated encryption MORUS. In: Proceedings of International Conference on Applied System Innovation, Sapporo, 2017. 365–368

5 Dwivedi A D, Klouček M, Morawiecki P, et al. SAT-based cryptanalysis of authenticated ciphers from the CAESAR competition. In: Proceedings of International Conference on Security and Cryptography, Madrid, 2017. 237–246

6 Zhang P, Guan J, Li J Z, et al. Research on the confusion and diffusion properties of the initialization of MORUS. J Cryptol Res, 2015, 2: 536–548

7 Guan J, Shi T R, Li J Z, et al. Analysis of MORUS against collision attack (in Chinese). J Elec Inf Tech, 2017, 39: 1704–1710