Differential Random Fault Attacks on certain CAESAR Stream Ciphers
(Supplementary Material)

Kenneth Koon-Ho Wong, Harry Bartlett, Leonie Simpson, and Ed Dawson

Queensland University of Technology, Brisbane, Australia
{kk.wong, h.bartlett, lr.simpson, e.dawson}@qut.edu.au

Abstract. This document contains supplementary material to the paper with the same title available from the proceedings of the International Conference on Information Security and Cryptology (ICISC) 2019. In this supplementary material, we demonstrate that the random fault attack strategy described in the full paper can be applied to ciphers in the MORUS family, resulting in partial state recovery for these ciphers.

Keywords: AEGIS, CAESAR, differential fault attack, fault attack, MORUS, random faults, side-channel attack, stream ciphers, Tiaoxin

1 Partial state recovery attacks

Our main paper [1] describes a novel random fault attack strategy which can be applied to any cipher in which the output function contains a bitwise AND operation. Based on this attack strategy, several key and state recovery attacks are presented against the AEGIS family, a CAESAR finalist, and Tiaoxin, a CAESAR third-round candidate. The paper also provides a brief discussion regarding the potential application of this attack to another CAESAR finalist, the MORUS cipher family.
This supplementary material expands on that discussion, presenting details of the random fault attack on the MORUS ciphers. The attack provides partial state recovery for each of these ciphers but we have so far been unable to extend this to full state recovery.

1.1 MORUS

There are three variants in the MORUS family of stream ciphers, namely MORUS-640-128, MORUS-1280-128 and MORUS-1280-256 [2]. Each of these variants of MORUS has five registers with a total size of 640 or 1280 bits depending on the variant. The internal state is updated at each timestep through five transformations, shown in Figure 1. See [2] for further algebraic details of the transformations. The rotation operations *Rotl* and $\lll$ have specific parameters for each variant, but the individual parameter values make no difference to our analysis.

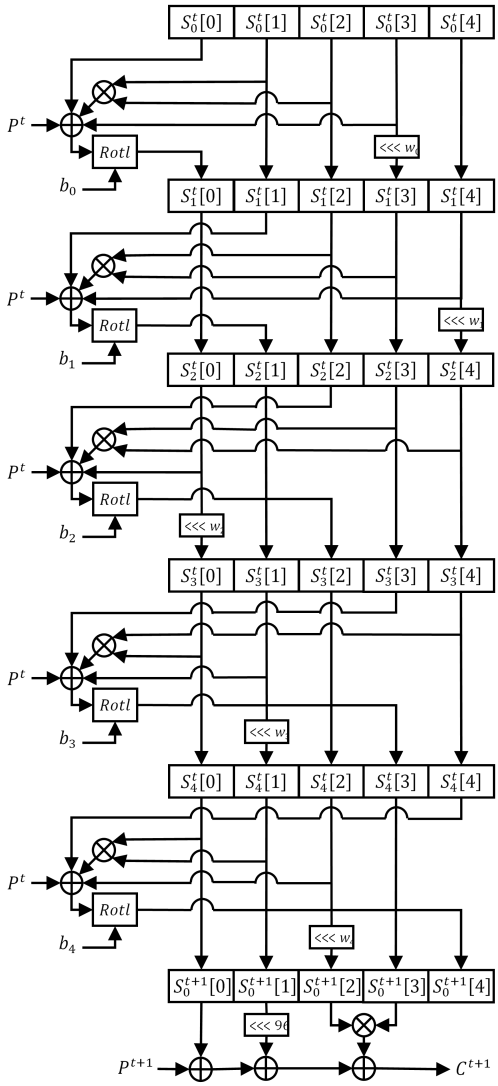

Fig. 1. MORUS State Update

At the encryption stage, after each internal state update, the following output function is used to compute the ciphertext block from the plaintext block and the current internal state.

$$C^t = P^t \oplus S_0^t[0] \oplus (S_0^t[1] \lll 96) \oplus (S_0^t[2] \otimes S_0^t[3]) \tag{1}$$

This function is of the form identified in [1, Equation 2], and so the random fault attack described in [1, Section 2] can be applied to this cipher.

Attack Procedure Comparing with the generic form in [1, Equation 2], we can obtain the following parameter sets:

$$(f(S^t), g(S^t), S^t[u]) = (P^t \oplus S_0^t[0] \oplus (S_0^t[1] \lll 96), S_0^t[2], S_0^t[3]) \quad (2)$$

$$(f(S^t), g(S^t), S^t[u]) = (P^t \oplus S_0^t[0] \oplus (S_0^t[1] \lll 96), S_0^t[3], S_0^t[2]) \quad (3)$$

Therefore, using the parameter set given in Equation 2, the attacker can inject random faults into $S_0^t[3]$ to recover the contents of $S_0^t[2]$ as per the analysis in [1, Section 2]. Alternatively, using Equation 3, random faults can be applied on $S_0^t[2]$ to recover $S_0^t[3]$ in a similar manner. Furthermore, if $S_0^{t+1}[2]$ and $S_0^{t+1}[3]$ are also recovered using the same method, the attacker can follow linear paths backwards and forwards in the state update function to retrieve the following intermediate state contents.

$$\begin{aligned}
S_1^t[2] &= S_0^t[2] & S_1^t[3] &= S_0^t[3] \lll w_0 \\
S_2^t[2] &= S_1^t[2] = S_0^t[2] & S_2^t[3] &= S_1^t[3] = S_0^t[3] \lll w_0 \\
S_3^t[2] &= S_4^t[2] = S_0^{t+1}[2] \ggg w_4 & S_3^t[3] &= S_2^t[3] = S_0^t[3] \lll w_0 \\
S_4^t[2] &= S_0^{t+1}[2] \ggg w_4 & S_4^t[3] &= S_0^{t+1}[3]
\end{aligned}$$

This means that all intermediate stages $S_{i,2}^t, S_{i,3}^t$ where $0 \leq i \leq 4$ can be recovered for any timestep t by attacking timesteps t and $t+1$. However, due to the nonlinear updates path across different stages, recovering $S_{i,0}^t$, $S_{i,1}^t$ and $S_{i,4}^t$ does not seem straightforward. It may be possible to recover some of the other stages deterministically or probabilistically by analysing the state transformations and the output function and known plaintext inputs. This is beyond the scope of our paper [1].

2 Summary

For MORUS, as shown above, it is only possible to recover the values of two of the five stages at any timestep, as well as all the intermediate values between any two timesteps for those two stages. This is because linear paths exist both forwards and backwards from a given timestep. From those intermediate values, it may be possible to recover values from other stages, but it is not straightforward because each update path across stages is nonlinear at some point. It remains a future research question whether our attack on MORUS can be extended to retrieve further information, such as a full state recovery.

References

1. Wong, K., Bartlett, B., Simpson, L., Dawson, E. Differential Random Fault Attacks on certain CAESAR Stream Ciphers. Information Security and Cryptology – ICISC 2018. (LNCS, to appear). Springer.
2. Wu, H., Huang, T. The Authenticated Cipher MORUS (version 2). CAESAR competition. Available from `https://competitions.cr.yp.to/round3/morusv2.pdf`.