Updates on Elephant
(proposal for presentation)

Tim Beyne¹, Yu Long Chen¹, Christoph Dobraunig²³, and Bart Mennink²

¹ KU Leuven and imec-COSIC, Leuven, Belgium
² Radboud University, Nijmegen, The Netherlands
³ Graz University of Technology, Austria

Email: elephant@cs.ru.nl

elephant@cs.ru.nl

Elephant is a nonce-based encrypt-then-MAC style authenticated encryption scheme. It is permutation-based and only evaluates this permutation in the forward direction. It is parallelizable by design, and as such perfectly suitable for small permutations. The Elephant scheme consists of three instances: Dumbo, Jumbo, and Delirium, which are instantiations of Elephant with Spongent-π[160], Spongent-π[176], and Keccak-f[200], respectively.

In this talk, we will discuss a tweak we are planning to apply to the Elephant mode. In a nutshell, the main change consists of moving from a Wegman-Carter-Shoup style authenticator \([3, 6, 7]\) in v1.1 to a protected counter sum style authenticator \([2, 5]\) in v2. Elephant v2 is depicted in Figure 1. We will explain that this planned tweak does not degrade the security and efficiency of Elephant. In addition, we will discuss the main benefit of this tweak: whereas version v1.1 only achieved confidentiality and authenticity against nonce-respecting adversaries, v2 additionally achieves authenticity under nonce-reuse.

Fig. 1. Depiction of Elephant. For the encryption part (top): message is padded as \(M_1 \ldots M_M \leftarrow M\), and ciphertext equals \(C = [C_1 \ldots C_M] \| M\). For the authentication part (bottom): nonce and associated data are padded as \(A_1 \ldots A_{\ell_A} \leftarrow N \| A\| 1\), and ciphertext is padded as \(C_1 \ldots C_{\ell_C} \leftarrow C\| 1\).
In addition, we will discuss novel results on the implementation of Elephant. We will consider our own\(^4\) as well as external [1,4] results on using the inherent parallelism of Elephant.

References


\(^4\) Available at https://github.com/TimBeyne/Elephant.