Computing Science

Software implementation of Sonic and SuperSonic

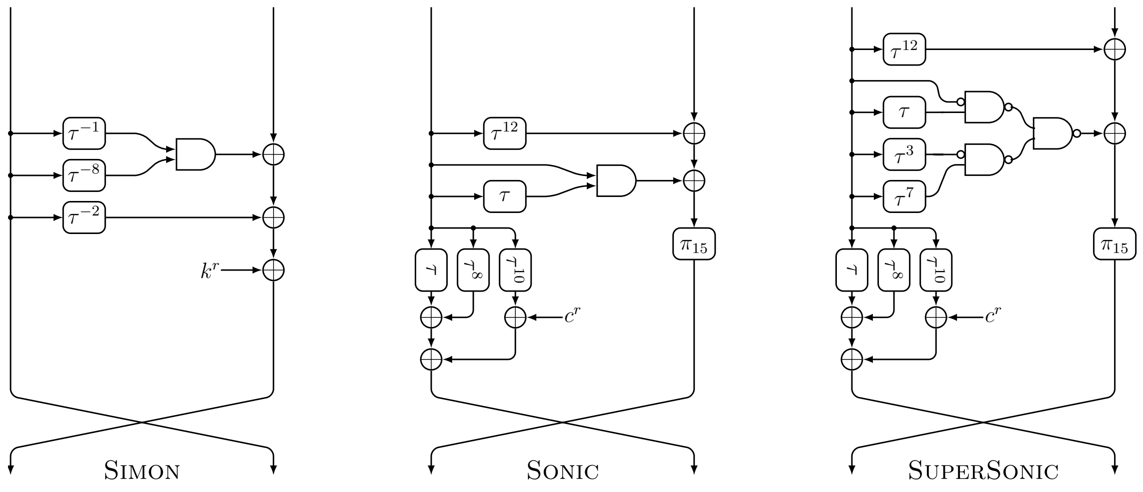
In the long list of cryptographic primitives already existing, there has been a growing category of lightweight design in the past few years: *low-latency*.

Those ciphers' design is focused on the latency of the algorithm, meaning the delay between the moment the input is available and the moment the output is produced.

The performance of a cipher will depend on its implementation. There is a wide range of possibilities, from complete hardware-implementation to full software with no dedicated hardware support.

Hardware implementation means the algorithm will be encoded as a circuit of gates and registers to process the input. Software implementation means that a programming language will be used to implement the algorithm and then be compiled into binary to be executed as a sequence of instructions by the device's processor. Hardware implementations are likely faster than software, but software implementations are more accessible to create, test, and deploy. Also, software implementation can benefit from dedicated instructions, which are processor instructions dedicated to performing part of the algorithm or coprocessor dedicated to cryptographic operations.

The best example is AES, for which many platforms have dedicated instruction. Therefore, designing algorithms to be faster than an AES software implementation using dedicated instruction, is challenging. We tried with Sonic and SuperSonic [**?**] to design hardware dedicated low-latency primitive. They are based on Simon's design, a hardware-oriented block cipher with modifications to reduce the number of rounds, making it low latency when implemented in dedicated hardware. Nevertheless, due to the simple design of their round function and their possible dimensions (64,128,256 and 512-bit), it could be possible to obtain competitive results for software implementation.



The goal of this project is first to produce a software implementation of Sonic and SuperSonic in Python, then a C reference code, and C with assembly instructions for the round function in ARM(V6/7 or 8). Then, evaluate the performances of such implementations and compare the result with other algorithms. If time allows, it will also be possible to program a RUST implementation, compare the RUST and the C implementation side-by-side, and/or import the code on an embedded platform and run performance tests.

This work will lead to a first reference software implementation for Sonic and SuperSonic and evaluate the software performance without a hardware dedicated instruction.

Contact Information

First supervisor: Joan Daemen `joan@cs.ru.nl`

References

[1] Yanis Belkheyar, Joan Daemen, Christoph Dobraunig, Santosh Ghosh, and Shahram Rasoolzadeh. Introducing two low-latency cipher families: Sonic and supersonic. *IACR Cryptol. ePrint Arch.*, page 878, 2023.