# Performance of State-of-the-Art Cryptography on ARM-based Microprocessors

Hannes Tschofenig*, Manuel Pegourie-Gonnard†

*ARM Limited, Email: Hannes.Tschofenig@arm.com

†ARM Limited, Email: manuel.pegourie-gonnard@arm.com

## I. Extended Abstract

ARM has designed many processors, and has extended its product portfolio by diversifying its CPU development. This resulted in the new processor family with the name "Cortex". There are three profiles in this family, namely

1) **Cortex-A**: Application processors that are designed to handle complex applications, such as high-end embedded operating systems. Example products include high-end smart phones, tables, home gateways, televisions, drones, etc. The Raspberry Pi also uses the Cortex-A processor.
2) **Cortex-R**: Real-time, high-performance processors targeted primarily at the higher end of the real-time market. Example products are hard drive controllers, baseband controllers for cellular radio communciation, and automotive systems.
3) **Cortex-M**: Processors targeting applications where low cost, and energy efficiency play an important role. Currently, there are the following Cortex-M processors on the market: Cortex-M0, M0+, M3, M4 and the recently released M7). Each of these processors provides different capabilities for different market segments [1].

Cortex-A and Cortex-R processors are very powerful and do not require special attention from a performance point of view. As mentioned, they are able to perform complex tasks without any problems and are not considered 'constrained' with respect to their computational capabilities. Widely used high-end operating system are available for these processors.

The situation for Cortex-M processors is, however, different since these processors do not offer a memory management unit (MMU) (but a memory protection unit (MPU) instead), contain a less powerful but more energy efficient CPU, and are equipped with less RAM and flash memory. Many modern operating systems assume the presence of a MMU and the limited RAM / flash memory resources often prohibit the use of sophisticated operating systems[1].

Position paper for the 'NIST Lightweight Cryptography Workshop', $20^{th}$ and $21^{st}$ July 2015, Gaithersburg, US.

[1]Note that we assume a platform with 256 KB of flash memory and 32 KB of RAM for use with our new mbed operating system, see [2]. mbed OS also requires a memory protection unit to provide memory access permissions for different memory regions thus improving OS security. In particular, this prevents applications from corrupting/accessing memory used by other applications or by the hypervisor.

Cortex-M processors are very popular with Internet of Things (IoT) products and the ability to offer an online development environment, an operating system, as well as the Internet protocol stack (including standardized security protocols) lowers the barrier of entry for small, innovative companies. Our efforts are therefore focused on ensuring suitable performance for the Cortex-M processor family. Without stating real-world hardware requirements it is difficult to offer meaningful performance numbers and goals for optimization efforts.

For evaluating the performance of an entire IoT product it is, however, also necessary to refer to a reference design since there are many different design patterns in use under the umbrella of IoT. The basic design patterns are described in RFC 7452 [3]. A presentation at the IETF 92 plenary by the Internet Architecture Board summarized the key differences of various design patterns [4]. While it will ultimately be most useful to measure performance in context of the different design patterns no such extended performance analysis has been conducted to our knowledge.

We have focused our performance investigations so far on the most demanding computations required by [5], namely on elliptic curve cryptography. While our work is still ongoing a preliminary presentation has been given to the IETF audience, see [6], with the intention to solicit feedback and to encourage others to offer their performance data. Without proper performance data it is difficult to decide whether cryptographic algorithms available today are (in-)sufficient for a given task. Note that [5] has been written with TLS/DTLS usage with IoT applications and hence the choice of ciphersuites and protocol extensions differs from the use in a typical Web/smart phone app scenario.

While the results are documented in detail in [6] it is useful to summarize a few key aspects. Note that we have used an open source TLS/DTLS library, namely mbedTLS [7], without any hardware optimization or ARM-assembly instructions.

1) ECC requires performance-demanding computations and those take time. What an acceptable delay is depends on the application. Many applications only need to run public key cryptographic operations during the initial (session) setup phase and infrequently afterwards. With session resumption DTLS/TLS uses symmetric key cryptography most of the time.
2) The performance of symmetric key cryptography (keyed hash functions, encryption functions) is negligible.
3) Detailed performance figures depend on the enabled performance optimizations (and indirectly the available RAM size), the key size, the type of curve, and CPU speed. Choosing the right microprocessor based on the expected usage environment is important.
4) Different curves offer quite some differences in performance. The Brainpool curves were slower than NIST curves and Curve25519 shows promise to be even faster than NIST curves.
5) ECDSA signature operation is faster than ECDSA verify operation. ECDH is only slightly faster than ECDHE (when fixed point optimization is enabled). Taking this fact into account can play a role in the overal system design.
6) ECC key sizes above 256 bits are substantially slower than ECC curves with key size 192, 224, and 256. Key sizes around 224 bits are roughly similiar in speed. It is important to note that the chosen key size has to be based on the state of the art recommendations rather than on the pre-selected hardware

platform. Quite often asymmetric cryptography is used on hardware that is not fit for the task and key sizes have to be chosen that are ridiculously small.

7) CPU speed has a significant impact on the crypto performance. Faster CPU speeds often also have a positive impact on energy efficiency because the CPU can finish computations much faster and the sleeping cycles can be longer.
8) Optimizations, such as NIST curve optimization and fixed point optimization, have a significant influence on the performance. There is a performance - RAM usage tradeoff: increased performance comes at the expense of additional RAM usage. We believe that the additional RAM is well spent.
9) An ECC library increases code size (compared to a pure shared secret-based approach).

Since various optimizations have not been utilized so far we believe that asymmetric cryptography using our mbedTLS stack can be used with all processors in the Cortex M family, particularly in context of TLS/DTLS, without noticable delay for most applications. For those applications that require very fast response times (for example due to user interactions) the Cortex-M3 and M4 processors provide good performance at a low cost. The new Cortex M7 will improve performance even further and thereby bridging the gap between M-class and A-class processors.

We are interested to hear what performance data others have gathered using their crypto libraries, maybe using different optimization techniques, and tests executed on different processors. We are also interested in documenting widely used IoT design patterns (as reference designs), which would not only be useful for performance comparisons of IoT systems but also for interoperability testing setups.

## REFERENCES

[1] ARM, "Cortex-M Series," Mar. 2015, http://www.arm.com/products/processors/cortex-m/.

[2] S. Ford, "Announcing our plans for mbed v3.0," Oct. 2014, http://developer.mbed.org/blog/entry/Announcing-our-plans-for-mbed-v30/.

[3] H. Tschofenig, J. Arkko, D. Thaler, and D. McPherson, "Architectural Considerations in Smart Object Networking," Mar. 2015, RFC 7452, Request For Comments.

[4] D. Thaler and H. Tschofenig, "Architectural Considerations in Smart Object Networking," Mar. 2015, http://www.ietf.org/proceedings/92/slides/slides-92-iab-techplenary-2.pdf.

[5] H. Tschofenig and T. Fossati, "A TLS/DTLS Profile for the Internet of Things," Mar. 2015, IETF draft (work in progress), draft-ietf-dice-profile-10.txt.

[6] H. Tschofenig and M. Pegourie-Gonnard, "Presentation at the IETF92 Light-Weight Implementation Guidance (lwig) working group meeting on Crypto Performance," Mar. 2015, http://www.ietf.org/proceedings/92/slides/slides-92-lwig-3.pptx.

[7] ARM, "mbedTLS," Mar. 2015, https://tls.mbed.org.