Profiles for the Lightweight Cryptography Standardization Process

Public Comments Received by June 16, 2017

From the Keccak Team:

Dear NIST,

We globally agree. Nevertheless, please find below our comments.

Kind regards,
Gilles, Guido, Joan, Michaël and Ronny
The Keccak Team

Page 2, lines 63-66: Why would rekeying come at a high cost? In many smart card protocols there is continuous re-keying, where session keys are derived from a static key.

Page 2, lines 67-76. Why not immediately ask for a XOF instead of a hash function? In that way you don't have to determine what the output length is, just the security strength level. As an example, when generating ephemeral scalars in elliptic curve signatures, one typically needs more bits than the order size for uniformity.

Page 2, line 79: Practical implementations almost always have some hardware and some software. The question is instead: how dedicated is the hardware to the crypto to be implemented? The real constraints are things like energy, power, latency and throughput. It is hard to put numbers on that, but these will be the numbers that define what lightweight actually means.

Profile I, design goals "The message length shall be an integer of bytes": Would proposals that support bit strings as input be allowed?

Profile I, performance characteristics:
- Why would FPGA matter for lightweight crypto?
- We have the impression that with the advent of very light 32-bit CPU cores, 8-bit and 16-bit architectures are being abandoned in industry.

Profile I, security characteristics:
- Why require 128-bit keys and then settle for 112 bits of security?

Profile II, performance characteristics:
- Why would FPGA matter for this ultra-lightweight crypto?
- About the requirement "Flexibility to support various implementation strategies (low energy, low power, low latency)": These requirements may

imply further splitting profile II. Applications that require low latency are typically different from those that require, e.g., low energy. Mandating flexibility excludes having specialized designs for low-latency and other ones for low energy. More and more we understand "lightweight crypto" as "crypto dedicated for specific use cases". A good example of this is the Skinny offering that has a dedicated block cipher for low latency called Mantis.

From Matt Robshaw:

Dear NIST –

I read the document “Profiles for the Lightweight Cryptography Standardization Process” (https://beta.csrc.nist.gov/publications/detail/white-paper/2017/04/26/profiles-for-lightweight-cryptography-standardization-process/draft) with considerable interest. The comments that follow are my personal view.

I would like to congratulate NIST on launching the lightweight cryptography initiative. The draft document “Profiles for the Lightweight Cryptography Standardization Process” provides a good start to the process and describes two initial profiles. The focus on functionality, rather than primitives, will likely be helpful.

However, I have doubts that these two initial profiles will, in themselves, result in the development of particularly lightweight solutions. For instance, while the document observes that an AEAD scheme can be used in a simple fashion in a basic challenge-response protocol (lines 53-58) the requirements for Profiles I and II imply that submissions must support a wide range of uses. Such generality is likely to come at an implementation cost that may, for many applications, be unnecessary.

So, while the initial profiles are interesting and a useful start, it would be helpful if NIST could outline plans for the development of additional profiles. Generally, I would expect profiles with a narrower focus, and with parameters that are more finely-tuned to constrained environments, to be more likely to result in more efficient solutions.

Yours sincerely,

Matt Robshaw
Technical Fellow, Security Solutions

From ARM:

Comments regarding the "Profiles for the Lightweight Cryptography Standardization Process"

Dear NIST team,

Following two workshops on lightweight cryptography you have decided to start the standardization process for a lightweight cryptographic algorithm. As part of this process you asked for feedback regarding the two profiles. We have participated in the 2015 workshop [1, 2], where we shared our experiences with cryptographic algorithms on ARM-based microcontrollers. We want to use this opportunity to share our views and concerns with the criteria outlined in [3].

Our experience is based on the work with ARM Cortex-M processors [4] widely used in microcontrollers, embedded devices, and Internet of Things (IoT), but also with more powerful Cortex-A and Cortex-R processors. Our processors enjoy success in a wide variety of products; as you may know, ARM processors have shipped in more than 100 billion chips to date [5].

In our daily work, we are trying to help developers to build more secure products and get them to market faster. All ARM processors are tailored to provide state of the art energy efficiency. ARM is also developing a development environment, operating system, and an entire protocol stack for the IoT. The protocol stack includes implementations of a number of cryptographic algorithms as well as a TLS/DTLS library. More details can be found at [6].

In this context we have been very interested in analysing the performance of cryptographic algorithms on a range of ARM microcontrollers and SoCs (system-on-chip).

Work on lightweight cryptography is an optimization problem and many optimization problems are about making trade-offs, and thus require a good understanding of the requirements.

Our main concern with the proposed standardization effort is the focus on symmetric key primitives. In our experience, symmetric key cryptography with existing primitives such as AES generally has adequate performance. In the embedded and IoT domain, efficiency of asymmetric crypto represents a more acute problem, but is not covered in your work to date. If we were to optimize the performance and efficiency of an embedded/IoT device then we would be focusing on the part that is most problematic, namely the public key crypto. Hardware accelerated AES is already widely deployed, and offers excellent performance, efficiency, and security.

While there are some applications that impose extraordinary constraints on devices, often other

factors need to be taken into account too. Here is an example: at the end of last year we announced the ARMv8.3-A architecture [7], which supports a system for cryptographic authentication of pointers, to help protect against attacks such as return-oriented programming. As part of this feature, we make use of a new lightweight cipher called QARMA [8]. Due to the application, and the demanding requirements (especially latency) imposed by doing cryptographic operations within the CPU pipeline, existing standardised ciphers such as AES were found to be unsuitable.

We have not seen documents from NIST that describe envisioned use cases, or offer an understanding of the perceived limitations of existing standardized symmetric cryptography. We would like to see rationale provided for the need for new algorithms, and the requirements thereof. Accordingly, we perceive a risk that, despite heroic standardization efforts, the results may not be solving real problems.

Our second concern, with reference to lightweight crypto on constrained devices, is the range of microcontrollers being addressed. For many applications, the market has been moving to 32-bit microcontrollers. Prices have dropped, performance has increased, tool support is better than ever, and there have been major improvements in energy efficiency. Still, you list 8- and 16-bit microcontrollers as targeted platform – in 2017. Standardization itself takes some time, as does incorporation into standardized security protocols such as TLS/DTLS. Then there is a delay before chip manufacturers ship products with the new algorithms in hardware, and finally customers incorporate the new chips into their products. For these reasons we believe that 32-bit and 64-bit processors should be the focus of the investigation. Even today, many manufacturers offer 32-bit microcontroller products with lower cost and better energy efficiency than 8 and 16-bit microcontrollers.

Standardizing more algorithms does not necessary lead to more security. While crypto-agility is an accepted design principle (see [9]), it is less well understood how it should be applied to devices that need to support a very long lifetime, especially where hardware acceleration is used. Additionally, the existence of more algorithms in protocols lowers the level of interoperability and increases implementation and verification costs.

We hope that our input helps NIST to calibrate the requirements and the scope of the work so that the outcome is more useful for industry players, including ARM and our customers. We are happy to provide further input and to share our experience.

Best regards, ARM

Reference

[1] H. Tschofenig, M. Pegourie-Gonnard, "Position Paper about Performance of State-of-the-Art Cryptography on ARM-based Microprocessors", NIST Lightweight Cryptography Workshop, July 2015, URL: http://csrc.nist.gov/groups/ST/lwc-workshop2015/papers/session7-tschofenig-paper.pdf

[2] H. Tschofenig, M. Pegourie-Gonnard, H. Vincent, "Slides about Performance of State-of-the-Art Cryptography on ARM-based Microprocessors", NIST Lightweight Cryptography Workshop, July 2015, URL: http://csrc.nist.gov/groups/ST/lwc-workshop2015/presentations/session7-vincent.pdf

[3] L. Bassham, at al., "Profiles for the Lightweight Cryptography Standardization Process", April 2017, URL: http://csrc.nist.gov/publications/drafts/whitepapers/2017/profiles-lwc-std-proc-draft.pdf

[4] ARM, "Cortex-M Series Family", June 2017, URL:

http://www.arm.com/products/processors/cortex-m

[5] ARM, “Inside the numbers: 100 billion ARM-based chips”, Feb 2017, URL: https://community.arm.com/processors/b/blog/posts/inside-the-numbers-100-billion-arm-based-chips-1345571105

[6] ARM, "mbed OS", June 2017, URL: https://www.mbed.com/en/platform/mbed-os/ [7] ARM, "ARMv8-A architecture – 2016 additions", June 2017, URL:

https://community.arm.com/processors/b/blog/posts/armv8-a-architecture-2016-additions

[8] R. Avanzi, “The QARMA Block Cipher Family”, May 2016, IACR Transactions on Symmetric Cryptology, URL: http://eprint.iacr.org/2016/444.pdf

[9] R. Housley, "Guidelines for Cryptographic Algorithm Agility and Selecting Mandatory-to-Implement Algorithms", Nov. 2015, URL: https://tools.ietf.org/html/rfc7696

From NSA:

NSA's comments on NIST Draft of *Profiles for the Lightweight Cryptography Standardization Process*, dated April 26, 2017

POC: Deb Cooley, decoole@nsa.gov, 410-854-3961

General Comment: Consider these options for follow on profiles:

1. While these profiles suggest key sizes of 128 bits or longer, consider a profile that has a minimum of 256 bit key size.
2. For hardware implementations, consider the on-chip footprint required by the algorithms, energy required to operate the hardware, and the cost to manufacture in large quantities.
3. For software implementations, consider optimizations that can be leverage to reduce power consumption and consider algorithms that can utilize other processing opportunities (i.e. GPUs).
4. For microcontroller implementations, consider reducing the need for complex math operations, large operating register bit-widths, and clock cycles needed to execute.
5. Support re-sync methods.

Line 9-10: Insert "a" between "for" and "submissions".
Line 21, p. 1: "Physical characteristics of the environment the implementation is to reside~~s~~ in…"
Line 31, p. 1: "…satisfy performance ~~requirements~~ characteristics on specific platforms…"
(Replace "requirements" because that word is already used in line 30.)
Line 42-44, p. 2: This is a long sentence, recommend dividing it into two sentences, or at least make this change, "predictable, and various implementation…"
Line 51, p. 2: "…authentication, confidentiality, and data authentication." (clarity)
Line 53: Replace "transforms it" with "transforms them".
Line 61 and 62: Is the assumption that nonce won't be repeated a reasonable assumption for lightweight devices, or are they the sort of environment that may have trouble counting?
Line 61 and 62: "…will not be repeated for multiple ~~encryptions~~ messages processed under the same…"
Line 63-65: Consider using a key update/KDF to change keys, even in the pre-shared case. This would allow the enforcement of data limits.

Line 72: Replace "against" with "to".
Line 79-80: Given that Profile I is for AEAD and hashing, does this mean submissions that consist of just hash functions are expected for Profile I? Or, when a hash function is submitted to Profile I, does it need to accompany an AEAD?
Line 101, p. 3: "…is intended for ~~the~~ those applications that require ~~a~~ good performance characteristics in both…"
P. 4: Table: third bullet of the Design Goals: Replace "integer" with "integral".
P. 4: Table: Performance characteristics: "The performance on ASICs and FPGAs should consider various standard cell libraries, and have the flexibility…"
P. 4: Table: sixth bullet of the Security characteristics: replace "can be processed" with "shall be able to be processed". This will make this bullet consistent with the other bullets, since they all use "shall", rather than "can".

P. 4: Table: Security characteristics, 6th bullet AEAD, 1st bullet hashing: If a minimum key length of 128 bits is required then the attacks should require 2^{128} computations. If attacks requiring as few as 2^{112} computations are allowed, then the key length should be 112 bits.
Line 104: There is a somewhat awkward mix of structure in the bullets, in that some bullets are complete sentences while others are just phrases.
P. 5: Table: third bullet of the Design Goals: replace "integer" with "integral".
P. 5: Table: first bullet of performance characteristics: "The performance on ASICs and FPGAs should…"
P. 5: Table: sixth bullet of the Security characteristics: replace "can be processed" with "shall be able to be processed". This will make this bullet consistent with the other bullets, since they all use "shall", rather than "can".
P. 5: Table: bullet 7 of the Security characteristics: If a minimum key length of 128 bits is required then the attacks should require 2^{128} computations. If attacks requiring as few as 2^{112} computations are allowed, then the key length should be 112 bits.
Line 111: There is a somewhat awkward mix of structure in the bullets, in that some bullets are complete sentences while others are just phrases.

From Hirotaka Yoshida:

Dear NIST,

Here are my comments to "Profiles for the Lightweight Cryptography Standardization Process", Draft Whitepaper of April 2017.

1) Performance characteristic
According to professor Ingrid Verbauwhede's presentation slides at page 9 in the following, different metrics (low power, low energy, high throughput) need different SW/HW platform for optimization:
https://summerschool-croatia.cs.ru.nl/2017/slides/hardware%20design%20for%20cryptographers:%20dreams%20and%20reality.pdf

There could be a submission A that has an outstanding performance in terms of one metric, which could be on only one platform. On the other hand, there could be another submission B that has good performance in terms of plural metrics, which could be on plural platforms.
It is not clear to me that NIST has any preference between A and B.

2) Hash function over other primitives in terms of RAM size
A hash function requiring a decent security level typically has a larger state size than other primitives such as MACs, block ciphers, and permutations. There might be some system or a device that requires integrity only but small RAM size for crypto. In this case, dedicated

lightweight MACs might be preferable over NIST profile algorithms that always require a hash function. But it looks to me that NIST does not want lightweight MACs.

3) Bottleneck clarification
One of the questions which I sometimes hear from industry is where symmetric-key (SK) based crypto is bottleneck.
They could argue that other things such as communication costs, or computational costs of public-ky crypto are bottleneck in some cases.
In these cases, even if symmetric-key based crypto achieves Lightweightness, it seems to me that this cannot help a lot.
If NIST can tell some information on some devices, some systems, or use cases where NIST cares security and NIST really needs lightweight SK-based crypto there, I think that this will probably help attract people from industry to this NIST project.
Examples could be smart grid systems and smart meters or automotive systems and ECUs (Electronic Control Units)? Though, I am not sure if the security of any of these examples is taken speciously by NIST.

Best regards,
Hirotaka Yoshida

From Rene Henriquez Garcia:

Dear NIST Lightweight crypto team,

Please find below inputs from Intel related to your recent draft on lightweight crypto profiles.

- More detailed guidelines on side-channel attacks might be needed. Currently, profiles only mention at a very high and general way resistance against side channel attacks. We believe profiles should be a bit more detailed here and specify exactly what type of resistance is really required specially since we're talking about extremely resource constrained devices and there's no room to implement everything
- Fault attacks? It's not mentioned in the current profiles draft
- NIST provides a blank number for security (i.e. 128 bits for block cipher, 112 for hash, etc.). What about cases where implementers are Ok with less security? Are you providing guidelines? I'm sure this was pointed out during the last workshop, but seems the comments might be ignored for now
- What's the minimum length they're considering for hash/block ciphers? It mention maximum length and it also says that algorithms should be optimized to be efficient for

short messages (e.g. as short as 8 bytes). We're aware of cases where minimum length would be 16 bits! What's the situation with these extreme cases?

- Very few lightweight crypto algorithms (such as Chaskey) lend themselves to inherent security by design. Is NIST planning to consider only those (as mentioned in 'Security Characteristics" of Profile I & II?
- What's the situation with those scenarios where we won't need collision resistant security? (i.e. RFID authentication)
- What is the area, power, timing budget for the lightweight crypto implementations that NIST plans to recommend?

Thanks,
Rene.

From Honeywell:

Background

Honeywell is one of the world's largest multinational corporations with deep concern for cyber security. Honeywell has cyber-physical control products in every critical-infrastructure sector. We have participated in standardization efforts involving cyber security and encryption for various industry sectors and continue to do R&D in these areas. Honeywell has the largest industrial cyber security research capabilities in that market and is continuing to increase its industrial cyber security offerings.

A large and important application area for lightweight cryptography (LWC) includes the safety- and security-critical cyber-physical systems that make up a large part of our critical infrastructure. The cyber elements of these systems are almost always embedded isochronous real-time control systems. While these systems have the resource constraints often discussed for lightweight cryptography, there are also a number of other characteristics of the systems that impact cryptography that are not often discussed. Some of these characteristics have been described in the 2002 Fast Software Encryption conference paper "BeepBeep: Embedded Real-Time Encryptions", a version of which is attached and the original can be downloaded from https://link.springer.com/content/pdf/10.1007%2F3-540-45661-9_13.pdf. The characteristics and constraints described in this paper are as true today as they were 15 years ago, with some additions. These additions include power and cost. Today, there are many more applications that

run on power from batteries or harvested/scavenge power. The large number the nodes in anticipated IoT systems requires a very small cost for each node. While the power and cost constraints were not discussed in the 2002 paper, they probably should have been. The Supervisory Control and Data Acquisition (SCADA) systems, which control much of our cyber-physical critical infrastructure, were in existence well before this paper. And, experience over the past couple of decades has shown that, while the need for encrypting SCADA communications has been widely acknowledged, encryption has made very little inroads into SCADA systems. We believe this is due to the relatively high cost to install encryption and that wide-spread acceptance of SCADA encryption won't happen until total installed cost is reduced by 10x to 100x vs current products. A significant part of this installed cost is providing power for added encryption devices, thus tying these two new constraints together. This cost could be eliminated if newly installed encryption devices could use scavenged power from the EIA/RS- 232 links that connect remote SCADA node electronics to their communication modems.

While SCADA is used as one example above, many other safety- and security-critical cyber-physical systems have the same or very similar requirements to those of SCADA. For many of these systems, particularly for retrofits, any added encryption must have nearly zero impact on bandwidth usage and system timing. This means negligible increases in message sizes.

Comments on current profiles

General observation on options: Given the wide diversity of lightweight encryption applications, there should be a number of options. And, these options should be compile- or synthesis-time options, not just run-time options. Run-time options still incur hardware or software memory costs plus the cost to evaluate the run-time option. These costs can be a significant part of the whole, given that the whole already must be lightweight. Furthermore, requirements of the form "shall be supported" should include such options. Additionally, systems employing lightweight cryptography will tend to be closed networks. That is, some organization is in full control of what elements can be legitimately connected to the network; and thus, the particular cryptography implementation can be controlled. Given the cost sensitivity of lightweight encryption applications, the controlling organization should be free to select the lowest cost implementation(s).

Questions posed in the "call for comments" email (in italics):

- elements of the Draft Profiles that you agree with, or that you would like to argue against?

In Physical characteristics, there are many more characteristics and constraints more than just "compact hardware" and low RAM and ROM (e.g., section 1 of attached paper). In Performance characteristics, trying to get a "one-size-fits-all" algorithm that runs on 8-bit, 16-bit and 32-bit microcontroller architectures may preclude algorithms that perform much better over the vast majority of platforms that are really used in security-critical applications. With 32- bit SoC processors falling in cost (some less than a $1), current consumption (μA range, static), and board area (25 sq mm), applications would have to use a very large number of processors to make smaller processors a significantly cheaper alternative. At such large numbers, dedicated hardware becomes attractive.

What does "efficient" mean for preprocessing of key? In Profile II: Why is hardware assumed to be the solution for the most constrained applications? We have found that for some constraints (e.g., power) software in a processor is better than FPGA hardware.

- Are there particular elements that you think are missing from the Draft Profiles?

Why are numeric values given for security characteristics but not for performance or physical characteristics? No latency or jitter requirements given (needed for embedded real-time control systems). No key agility requirements given (needed for cyber physical systems where remote nodes have no personnel guarding them). No power constraints stated (probably will have to be specified indirectly via proxies like gate-hertz, state-size, or example software platforms). Rationale is scarce to nonexistent. For example, why attacks with 2^{112} computations, a vestige of 3DES effective key size? Why hash outputs of 256 bits?

-how suitable are the Draft Profiles for your target application?

Honeywell has a large number of applications that could benefit from LWC. We are unlikely to create our own LWC hardware, but would likely use hardware developed by others, as long as it is:

a) Supported by a major silicon vendor b) Takes significantly less time for common operations
c) Produces smaller signatures

When will the software profile be addressed? We have candidates that meet the stringent performance requirements described in these comments and attached paper.

What is your opinion on replacing "128 bits" by "64 bits?" [for integrity tags]

The profiles should make clear that integrity tags need not be strictly added bits. Embedded real-time control systems with any kind of safety implications already have CRC or checksum protections against naturally occurring failures. For example, data messages using the DNP3 protocol (the most common SCADA protocol in North America and among the top SCADA protocols in the rest of the world) contain from 32 to 272 bits of CRC. An algorithm should be

allowed to exploit the existence of these bits for integrity checking, in lieu of adding any more tag bits. This can be done by replacing the existing bits with integrity tags or to use a non-malleable algorithm that carries message changes forward so that any change in the ciphertext would change the CRC/checksum fields in a way that can't be predicted by an attacker. This would allow the required nearly zero impact on bandwidth and latency, while providing good integrity coverage for most integrity-critical situations. The number of tag bits should be a compile- or synthesis-time option; and also the option to exploit existing integrity checking in the plaintext protocol.

What is your opinion on replacing "128 bits" by "96 bits? [for nonces]

Similar to the discussion on integrity tags, some characteristics of embedded real-time control systems can be used to supply the functionality of nonces, at least when used for initialization vectors (IVs), without explicitly adding a full field of bits. For example, most embedded real-time control systems are isochronous. This isochrony can be used instead of transmitting IVs. That is, existing plaintext protocol sequence numbers or time can be used for nonces, with these being possibly being a mix implicitly and explicitly transmitted bits. This should be another compile- or synthesis-time option.

... should it be mandatory or optional for [associated data]

This should be a compile- or synthesis-time option.