Proposals for Standardization of the Ascon Family

John Preuß Mattsson, Göran Selander, Santeri Paavolainen, Ferhat Karakoç
Ericsson

Marco Tiloca
RISE Research Institutes of Sweden

Robert Moskowitz
HTT Consulting

Abstract: NIST has announced the decision to standardize the Ascon family for lightweight cryptography applications. The NIST Lightweight Cryptography project is now entering a new phase. This memo suggests some directions for the upcoming work. The Ascon family consists of AEADs, hash functions, PRFs, and MACs but only the AEADs and hash functions were part of the original NIST submission. An important goal is that most forms of small devices should be able to implement only Ascon without also having to implement any of AES and its various modes of operation (e.g., CCM, CTR, CFB, GCM), SHA-2, SHAKE, HMAC, KMAC, HKDF, etc. Another goal is that the Ascon specification should support a variety of hash function output lengths.

Background

NIST has announced [1][2] the decision to standardize the Ascon family for lightweight cryptography applications. The Ascon family [3][4] consists of AEADs, fixed-length hash functions, variable-length hash functions, PRFs, and MACs but only the AEADs and hash functions were part of the original NIST submission. NIST writes that the newly selected algorithms should be appropriate for most forms of tiny tech, which includes a broad range of devices and applications. Ascon might be implemented in hardware or software on devices that can be powered by energy harvesting, batteries, or a wall socket. Some motivations for new lightweight cryptography mentioned in the initial call for algorithms [5] are reduced latency, area, energy consumption, required power, RAM and ROM, and reduced vulnerability to side-channel attacks. The relevance of each motivation differs significantly depending on the use case.

Lightweight cryptography is expected to be applied in settings where the constrained devices are connected, typically to an infrastructure consuming or producing the device related data. The connectivity comes at a cost on the device performance which in constrained environments can be significant, and the cryptographic constructs used contribute to this cost. The energy consumption for symmetric cryptography computations can be negligible when compared to the energy costs for the physical operation and the radio transmission [6]. However, the message overhead due to the used crypto algorithms contributes to the latter. Message overhead also has an impact on performance of constrained networks [7] where tiny devices are expected to be used. For these

reasons it is important to be able to tailor the added overhead, specifically the length of integrity tags, to the requirements of the application.

Another limitation in constrained devices is the amount of available RAM and ROM, hence the ability to reuse crypto functionality for different purposes can have a significant impact on what applications are possible to implement on a given device. An important requirement for reducing area and ROM usage in constrained devices is to standardize sufficient variants of the cryptographic primitive Ascon, so that constrained devices can implement only Ascon without also having to implement any of AES and its various modes of operation (e.g., CCM, CTR, CFB, GCM), SHA-2, SHAKE, HMAC, KMAC, HKDF, etc. One important metric in this case is the total area or ROM for *all* cryptographic functions considered together, not the area or ROM required for individual cryptographic functions. Having a single cryptographic primitive like Ascon that efficiently provides all the necessary symmetric cryptographic functions is a desire for many IoT developers and manufacturers.

Particularly in this regard, we think that Ascon is an excellent choice for standardization. Ascon exhibits performance advantages over existing standards on various target platforms without introducing security concerns or reducing the achieved security level. Ascon was the winner for Lightweight applications in the Competition for Authenticated Encryption: Security, Applicability, and Robustness (CAESAR) [8] and has received a significant amount of cryptanalysis [9]. Just like the SHA-3 family [10][11], the Ascon family is based on a sponge function. This makes Ascon very flexible and makes it possible to fulfill the goal that most forms of small devices should be able to implement only Ascon. We think that the Ascon family will find its way into a large number of applications.

Proposals for Standardization

The NIST Lightweight Cryptography project is now entering a new phase focused on standardizing the Ascon family. It is important to remember that, in this new phase, the initial call for algorithms [5] is not very relevant anymore. What is relevant is what can be done with the Ascon family [3][4] to best meet the requirements of future lightweight cryptography applications.

Below are our proposals for inclusion in the standardization of the Ascon family. We would like to see them addressed in the initial NIST draft specification, rather than piecemeal added as has been with the organic growth of functions based on AES and SHA. These proposals are based on years of experience working with symmetric encryption and hashing; they are:

- **Variable length hash function only**. The current Ascon specification [3] defines both fixed- and variable-length hash functions. We strongly think NIST should standardize one of the variable-length hash functions Ascon-Xof or Ascon-Xofa. We do not think that NIST should standardize any of the fixed-length hash functions Ascon-Hash or Ascon-Hasha. We do not see any reason whatsoever to standardize one of the fixed-length hash functions, which are just special cases of the variable-length hash-functions. Anybody needing a 256-bit digest can use a variable-length hash function with $l = 256$. Both longer and shorter digests than 256 bits will be needed. Ed25519 [12] does for example require a 512-bit hash. For SHA-2 [13], NIST has afterwards introduced two different variable-length hash functions based on the fixed-length hash functions. These are SHA-256/t (using the terminology from SP 800-208), which relies

on a simple truncation to t bits, and SHA-512/t, which also changes some of the inner constants. The fixed-length SHA-3 hash functions [10] have seen little practical use, while variable-length functions such as SHAKE [10], cSHAKE [11], and KMAC [11] have seen significant practical use in implementations as well as in published and upcoming standards such as EdDSA (RFC 8032), XMSS (RFC 8391), LMS (NIST SP 800-208), CMS (RFC 8702), RSASSA-PSS and ECDSA (FIPS 186-5, RFC 8692), COSE (RFC 9054), DRIP (RFC 9347), EDHOC (draft-ietf-lake-edhoc), CPace (draft-irtf-cfrg-cpace), FROST (draft-irtf-cfrg-frost), OPRF (draft-irtf-cfrg-voprf), Kyber, Dilithium, Falcon, and SPHINCS+.

- **Shorter AEAD tags.** The current Ascon specification [3] only specifies AEADs with 128-bit tags. We would strongly like to see also shorter tags as an available option. This can consist of either variable tag lengths or a set of allowed, smaller tag lengths (e.g., 32, 64, 80, 96, 128). 32-bit tags are standard in most radio link layers including 5G [14]. 64-bit tags are very common in transport and application layers of the Internet of Things. 32- and 64-bit tags are also common for protection of audio frames, also in future proposals like IETF Secure Media Frames (sframe) [15]. The new ETSI SAGE specification of high-performance algorithms for 5G Advance and 6G specifies tag length as a variable between 4 and 16 bytes.

- **Length-preserving IND-CPA encryption**. The current Ascon specification [3] only introduces IND-CCA encryption with 128-bit tags. We would strongly like to see also a standardized lightweight IND-CPA encryption without message expansion as an option, i.e., Ascon encryption without an authentication tag. The IND-CPA encryption that does not need to include operations for tag computation should use the AEAD interface [16] without the input parameter A for the associated data, i.e., $C = E(K, N, P)$. The need of having IND-CPA encryption comes from many use cases in modern protocols such as header encryption in DTLS 1.3 [17] and QUIC [18], field encryption in DRIP [19], and identity protection in SIGMA-based protocols (e.g., message_2 encryption in EDHOC [20]), as well as in systems where message integrity is provided by a signature such as in the group mode of Group OSCORE [21] and in SUIT [22]. Also, the IETF WG COSE has decided to reintroduce IND-CPA encryption such as AES-CTR for these purposes [23].

- **Omit 160-bit keys.** We do not think that NIST should standardize Ascon-80pq [3]. The NIST Post-Quantum Cryptography project specified their quantum security levels based on symmetrical algorithms where security level I is based on AES-128 [24]. Even if a Cryptanalytically Relevant Quantum Computer (CRQC) able to break RSA-2048 in hours is ever built, such a CRQC would not pose any practical threat at all to any symmetrical algorithms including Ascon-128. Using NIST's assumptions about quantum computer performance [24], a huge cluster of one billion CRQCs (according to one estimate costing one billion USD each) would take a million years of uninterrupted calculation to find a single AES-128 key. The time for finding a single Ascon-128 key would not differ from AES-128 in a practically meaningful way. Therefore, we suggest that NIST redefines the quantum security level I to be based on Ascon-128.

- **Specify KDF, PRF, and MAC.** PRFs and MACs based on Ascon are defined in [4]. We strongly think NIST should specify a KDF, a PRF, and a MAC based on Ascon together with an AEAD and a variable-length hash in a single document, instead of in different documents and timepoints as it was done with SHAKE [10] and KMAC [11]. The standard should define

a KDF that can be used with secrets that are not uniformly distributed, such as ECDH shared secrets. If a PRF can be achieved with fewer passes, the standard should also define a PRF that can be used with secret keys that are uniformly distributed. The MAC function should be variable-length. We suggest that NIST standardizes Ascon-Prf, Ascon-PrfShort, and Ascon-Mac [4]. We think that a dedicated MAC function is needed as the Ascon-128 function cannot be used with a fixed nonce.

- **Ed25519 and ECDSA with Ascon**. We would strongly need that NIST specifies the use of an Ascon hash function for Ed25519 and ECDSA [12], preferably already in the initial Ascon draft specification instead of waiting for an update of 186-5 [12]. Ed25519 currently requires SHA-512, while ECDSA with P-256 can, e.g., be used with SHA-256 or SHAKE128.

- **API support for Ascon round function.** We think that NIST should at least strongly recommend or even better mandate that implementations support the Ascon round function. New algorithms like AEGIS [25] and SNOW 5G [26] makes clever use of the AES round function. TurboSHAKE128, TurboSHAKE256, and KangarooTwelve [27] use fewer rounds of the Keccak-p permutation than SHAKE128 and SHAKE256. APIs not supporting the AES round function and Keccak-p cannot support acceleration of these new algorithms, which is thwarting innovation. NIST should avoid the risk of the same thing happening with Ascon.

- **Support for longer nonces.** Ascon-128 and Ascon-128a sets 32 bits of the IV to zero [3]. We think that these 32 bits should be used to support 160-bit nonces. The use of random nonces is a common practice, and increasing the nonce length increases the number of instantiations that can be allowed with a single key and random nonces. The limited number of instantiations and the resulting high collision probabilities with random nonces are big problems with AES-GCM. New AEAD algorithms such as XChaCha [28] and AEGIS-256 [25] support 192-bit or 256-bit nonces and are therefore suitable to use with random nonces.

- **Customizable hash function.** Ascon-Xof and Ascon-Xofa use a 256-bit zero value during initialization. We think that these 256 bits should be used to support function-name bit strings and customization bit strings similar to the parameters N and S in cSHAKE [10]. We think that NIST should only standardize a customizable variable-length hash function based on Ascon. Having a customizable variable-length hash function is essential to build algorithms such as Kyber. Unless there are significant performance differences, we do not think that NIST should standardize both non-customizable and customizable variable-length hash functions as was done with SHAKE [10] and cSHAKE [11].

- **Duplex mode for key derivation.** We think that NIST should consider standardizing the Ascon duplex mode of operation also for key derivation with a suitable interface like "$init(), digest = update(M_i, l)$". The duplex mode can be seen as a generalization of a running hash interface "$init(), update(M_i), digest = finalize(l)$". The duplex interface maps more naturally to how key derivation is done in modern security protocols, without the need to derive intermediate keys whose only use is being input to the key derivation in the next state of the security protocol.

Summary and Conclusions

We think that Ascon is an excellent choice for standardization and that the Ascon family will find its way into a large number of applications. One key aspect of the applicability of the Ascon standard is the ability to tailor the length of the integrity tag to the security requirements and capabilities of the application. Another important goal of the Ascon standardization should be to enable the implementation of only Ascon in constrained devices, without having to also implement any of AES and its various modes of operation (e.g., CCM, CTR, CFB, GCM), SHA-2, SHAKE, HMAC, KMAC, HKDF, etc. The area and ROM for individual cryptographic functions is not a useful metric, and we think that a more useful metric is the total area or ROM of *all* the cryptographic functions available on a device. We suggest that all the important functions based on Ascon should be standardized already in the initial draft specification.

References

[1] NIST, "Lightweight Cryptography Standardization Process: NIST Selects Ascon"
https://csrc.nist.gov/News/2023/lightweight-cryptography-nist-selects-ascon

[2] NIST, "NIST Selects 'Lightweight Cryptography' Algorithms to Protect Small Devices"
https://www.nist.gov/news-events/news/2023/02/nist-selects-lightweight-cryptography-algorithms-protect-small-devices

[3] Dobraunig, Eichlseder, Mendel, Schläffer, "Ascon v1.2 Submission to NIST"
https://csrc.nist.gov/CSRC/media/Projects/lightweight-cryptography/documents/finalist-round/updated-spec-doc/ascon-spec-final.pdf

[4] Dobraunig, Eichlseder, Mendel, Schläffer, "Ascon PRF, MAC, and Short-Input MAC"
https://eprint.iacr.org/2021/1574.pdf

[5] NIST, "Submission Requirements and Evaluation Criteria for the Lightweight Cryptography Standardization Process"
https://csrc.nist.gov/CSRC/media/Projects/Lightweight-Cryptography/documents/final-lwc-submission-requirements-august2018.pdf

[6] Preuß Mattsson, Selander, Smeets, Thormarker, "Constrained Radio Networks, Small Ciphertexts, Signatures, and Non-Interactive Key Exchange"
https://csrc.nist.gov/csrc/media/Events/2022/fourth-pqc-standardization-conference/documents/papers/constrained-radio-networks-pqc2022.pdf

[7] RFC 7228, "Terminology for Constrained-Node Networks"
https://www.rfc-editor.org/rfc/rfc7228

[8] CAESAR, "Competition for Authenticated Encryption: Security, Applicability, and Robustness"
https://competitions.cr.yp.to/caesar-submissions.html

[9] Dobraunig, Eichlseder, Mendel, Schläffer, “Ascon Lightweight Authenticated Encryption & Hashing”
https://ascon.iaik.tugraz.at/

[10] FIPS 202, “SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions”
https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.202.pdf

[11] NIST SP 800-185, “SHA-3 Derived Functions: cSHAKE, KMAC, TupleHash and ParallelHash”
https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-185.pdf

[12] FIPS 186-5, “Digital Signature Standard (DSS)”
https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-5.pdf

[13] FIPS 180-4, “Secure Hash Standard (SHS)”
https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.180-4.pdf

[14] 3GPP TS 33.501, “Security architecture and procedures for 5G System”
https://www.3gpp.org/ftp/Specs/archive/33_series/33.501/33501-i00.zip

[15] IETF, “Secure Frame (SFrame)”
https://datatracker.ietf.org/doc/html/draft-ietf-sframe-enc

[16] RFC 5116, “An Interface and Algorithms for Authenticated Encryption”
https://www.rfc-editor.org/rfc/rfc5116

[17] RFC 9147, “The Datagram Transport Layer Security (DTLS) Protocol Version 1.3”
https://www.rfc-editor.org/rfc/rfc9147

[18] RFC 9002, “Using TLS to Secure QUIC”
https://www.rfc-editor.org/rfc/rfc9001

[19] IETF, “UAS Operator Privacy for RemoteID Messages”
https://datatracker.ietf.org/doc/html/draft-moskowitz-drip-operator-privacy

[20] IETF, “Ephemeral Diffie-Hellman Over COSE (EDHOC)”
https://datatracker.ietf.org/doc/html/draft-ietf-lake-edhoc

[21] IETF, “Group OSCORE - Secure Group Communication for CoAP”
https://datatracker.ietf.org/doc/html/draft-ietf-core-oscore-groupcomm

[22] IETF, “Software Updates for Internet of Things (suit)”
https://datatracker.ietf.org/wg/suit/about/

[23] IETF, “CBOR Object Signing and Encryption (COSE): AES-CTR and AES-CBC”
https://datatracker.ietf.org/doc/html/draft-ietf-cose-aes-ctr-and-cbc

[24] NIST, “Submission Requirements and Evaluation Criteria for the Post-Quantum Cryptography Standardization Process”
https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/call-for-proposals-final-dec-2016.pdf

[25] IRTF, “The AEGIS family of authenticated encryption algorithms”
https://datatracker.ietf.org/doc/html/draft-irtf-cfrg-aegis-aead

[26] Ekdahl, Johansson, Maximov, Yang, “SNOW-Vi: an extreme performance variant of SNOW-V for lower grade CPUs”
https://eprint.iacr.org/2021/236.pdf

[27] IRTF, “KangarooTwelve and TurboSHAKE”
https://datatracker.ietf.org/doc/html/draft-irtf-cfrg-kangarootwelve

[28] CFRG, “XChaCha: eXtended-nonce ChaCha and AEAD_XChaCha20_Poly1305”
https://datatracker.ietf.org/doc/html/draft-irtf-cfrg-xchacha