Comments on NIST's Requirements for an Accordion Cipher

John Preuß Mattsson, Ben Smeets, Erik Thormarker
Ericsson

We welcome NIST's plans [1] to develop a new tweakable, variable-input-length-strong pseudorandom permutation (VIL-SPRP) and derived functions. We think that a tweakable VIL-SPRP is the correct target, and we really like the name accordion cipher. Building derived functions such as AEAD, tweakable encryption, key wrap, etc., as functions derived from the accordion cipher seems like the correct approach. A well-designed accordion cipher with derived functions could provide significantly improved properties compared to many of the cipher modes that NIST currently approves. In addition to very strong cryptographic properties, we think the derived functions should provide good usability and usable security. Interfaces and guidelines should be chosen to minimize the demands on users and implementers, as well as the adverse consequences of human mistakes [2]. The practical use of the new accordion cipher will heavily depend on its performance and other properties.

We have the following comments on the accordion cipher and its derived functions:

- **Restriction to only block ciphers:** We are not convinced by the suggestion to restrict the underlying primitives to only block ciphers. NIST has not provided any motivation for this restriction. During the Third NIST Workshop on Block Cipher Modes of Operation 2023 [3], a suggested outcome was for NIST to approve use of TurboSHAKE [4] and incorporate it as part of the accordion cipher. There might also be significant benefits in building the accordion cipher on the AES round function, as done in AEGIS [5] and Rocca-S [6]. We think the requirement should be a reduction proof to a small set of primitives and that primitives other than block ciphers should be allowed. The end result might still be that the only used primitive is a block cipher. Performance has historically been very important for large-scale adoption. It is unclear to us how a restriction to only allow block ciphers will affect the properties of the accordion cipher.

- **Replacing other modes:** We think that the accordion cipher, in the long term, can (and should) replace AES-ECB, AES-CBC, AES-CFB, AES-OFB, AES-CRT, XTS-AES, AES-CCM, AES-KW, and AES-KWP. Due to its slower performance, we do not think it will be a general replacement for AES-GCM and other high-performance algorithms. The future accordion cipher is expected to have only about half the performance of AES-GCM, will not have the online property of AES-GCM, AES-GCM-SST [7], and AEGIS [5], and it will take many years until the final standard is published. As traffic volumes continue to grow and NIST's zero trust requirements [8] include mandatory encryption of all data, high performance AEAD schemes are very important. We, therefore, still think [9] there is a need for NIST to quickly approve use of AES-GCM-SIV [10], AES-GCM-SST, and a very high-performance cipher like AEGIS.

- **Message lengths.** We think it should be a strong requirement that $g \leq 8$ to support all byte strings within a certain range. A cipher with $g = 128$ would limit the number of use cases and require padding. Padding causes additional message expansion and cannot be used when length preservation is required. For software implementations, it might actually be preferred if g is not smaller than 8. When specifying hedged ECDSA and EdDSA signatures [11], we noted that relying on the NIST bit-oriented KMAC interface caused interoperability and complexity problems as many implementations of KMAC only support byte strings. That AES-GCM only supports plaintexts of length 2^{36} bytes is a limitation. It would be beneficial if the accordion cipher supported plaintexts of length significantly greater than 2^{36}, e.g., $bg = 2^{48}$ or 2^{64} bytes. We assume that ag will likely be equal to the block length of the block cipher, which is acceptable.

- **Block size of the underlying cipher**. We strongly agree that the accordion cipher should also support 256-bit block ciphers. We think NIST should approve the use of Rijndael with 256-bit keys and blocks [12]. At the third encryption workshop [3], there was a discussion about whether NIST should approve Simpira [13] instead. Rijndael with 256-bit blocks is a more straightforward choice for 256-bit blocks. Simpira has received less analysis and does not seem to have any significant performance benefits. The additional functionality of Simpira (even larger block sizes) would be provided by the accordion cipher. A block cipher with 256-bit blocks is an important building block for many other cryptographic constructions striving for 256-bit security, such as MILENAGE-256 [14].

- **Key size**. We agree with NIST that the accordion cipher must support 256-bit keys. We think it would be acceptable if the accordion cipher only supports 256-bit keys. A reason to support 128-bit keys would be if the accordion cipher can be instantiated with AES-128. Some hardware only have support for AES-128 or AES-256. We do not think the accordion cipher needs to support 192-bit key sizes or AES-192. When approving Rijndael with 256-bit keys and blocks, NIST could even remove AES-192. We are not aware of any use of AES-192 anywhere. We think it would be good if the accordion cipher can be instantiated with AES-128, AES-256, and Rijndael with 256-bit keys and blocks.

- **An alternative to AES to enable cryptographic agility.** Cryptographic agility is the ability to switch between cryptographic primitives without the need to modify or replace the surrounding infrastructure. The importance of cryptographic agility has been emphasized by several US agencies [15–17]. A necessity for cryptographic agility is to have a cryptographic primitive to switch to. With the deprecation of Triple DES, NIST does not have a standardized alternative to AES to be used in the event that AES is broken. Ascon is not recommended as a general replacement for AES and standardizing new algorithms takes many years. At the third encryption workshop [3], we requested a NIST-approved alternative to AES to enable cryptographic agility [9]. NIST suggested that an accordion cipher could address all the issues we pointed out in [9]. An accordion mode based on block ciphers does not provide an alternative to AES to enable cryptographic agility unless NIST standardized an alternative block cipher to AES/Rijndael. NIST has previously discussed the standardization of an AEAD mode based on Keccak. We think NIST should standardize an AEAD mode of Keccak to enable cryptographic agility.

- **Nonce hiding**. Nonce hiding [18] is an interesting feature that has recently seen broad adoption in modern security protocols such as TLS 1.3 [19], DTLS 1.3 [20], and QUIC [21]. The are several potential security problems with sending nonces in the clear along with the ciphertext. If the nonce is a sequence number, this enables tracking and potentially identification of the sender and the receiver. If the nonce is a random number, it might reveal information to an attacker who has backdoored the PRNG. In TLS 1.3, the nonce is implicit and not sent in the packet. In DTLS 1.3 and QUIC, the nonce is a combination of an implicit IV not sent in the packet and an explicit sequence number sent in the package. The sequence number in DTLS 1.3 and QUIC is encrypted using a second pass with the first pass ciphertext as input. The nonce-hiding API would need to be able to only hide the explicit part of the nonce (the sequence number). The API would also need to output the explicit part of the nonce (the sequence number) as that is not only used for decryption but also replay protection and message ordering. A standardized nonce hiding AEAD would be welcome as it simplifies sequence number encryption in both security protocols and when the AEAD is used on its own. Fully Encrypted Protocols (FEP) [22] are important to avoid network censorship and for military communication systems. A nonce hiding AEAD enables implementation of a FEP with less overhead.

- **Replay protection.** An alternative design to nonce hiding that improves usable security are AEAD constructions such as Authenticated Encryption with Replay prOtection (AERO) [23] which use a tweakable VIL-SPRP to provide both nonce hiding and replay protection. We think NIST should standardize an AERO derived function. An AERO API offers several advantages over other methods: it is simpler to use, encourages the use of replay protection, and has more compact messages. AERO can be considered a stateful and self-synchronizing authenticated encryption method that provides protection from replay attacks. The four interfaces are:

  E = EncryptionInit(K)
  D = DecryptionInit(K)
  C = E(P, A)
  P = D(C, A)

  The parameters T, W, V [23] can be part of the algorithm or part of the initialization. We have in the past seen a lot of practical and serious vulnerabilities due to weak or missing replay protection. Higher-layer protocols are very often designed based on the assumption of strong replay protection in lower layers. Missing or weak replay protection in the lower layer typically compromises confidentiality, integrity, and availability at higher layers, i.e., the whole CIA triad. Replay protection should be a mandatory requirement in all modern security protocols. Understanding of the importance of replay protection is, however, often severely lacking even among people working with security. It would therefore be very welcome if NIST increased its recommendations regarding replay-protection and standardized easy-to-use replay protection. Users should ideally not have to deal with nonces or replay protection.

- **Performance targets.** We think smartphones, wearables, and IoT devices would also greatly benefit from the accordion cipher. Today, most of these devices have relatively powerful processors, and the processing energy costs are typically almost negligible compared to the

energy costs for transmitting and receiving radio [24]. An AERO derived function [23] based on the accordion cipher could therefore lower total energy use in wearables and IoT devices even if the energy costs for encryption is higher. We agree that NIST should initially not target constrained devices. Such devices would benefit from an accordion mode based on Ascon, but that can be done later. We strongly agree with NIST that the accordion cipher should allow for substantial parallelism for large input sizes. Performance which is half that of AES-GCM seems like a possible and likely target. Much faster performance than that is likely not possible. An accordion cipher with half the performance as AES-GCM and superior security properties would have many use cases; however, it would not replace AES-GCM. The is also industry need for algorithms like AEGIS with much higher performance than AES-GCM.

- **Concealment of plaintext length**. A required feature in most modern security protocols such as IPSec [25], SSH [26], TLS [19], DTLS [20], QUIC [21], EDHOC [27], and MLS [28], is the ability to conceal the plaintext length and to send packets with no message and only padding. The IND-CCA2 game assumes that all messages have the same length. In practice, this is not the case, and without padding, a passive attacker can trivially determine the length of the message, which can be extremely revealing. Sending packets with no message and only padding complicates traffic flow analysis. Military systems sometimes take this to the extreme by always sending at full bandwidth so that an adversary cannot determine when actual communication is happening. We think plaintext length concealment is an important cryptographic property, and we suggest that the derived functions support padding. NIST's current proposal for an AEAD derived function in Section 3.1 of [1] already includes padding "to ensure that the input is an allowed size for the accordion cipher". We think this should be changed so that the amount of padding is determined by an input parameter.

- **Security properties.** NIST writes that the security definition for a VIL-SPRP in the single user setting automatically promises some important security properties [1]. It would be very beneficial if NIST could describe the promised security properties. In addition to *concealment of plaintext length* and *replay protection* discussed in earlier bullets, some notable security properties that should be discussed are:

    - *Security against release of unverified plaintext (RUP).* The importance of this property was mentioned at the 2023 workshop [3] and listed as a high priority be UK government in [29]. We agree that this is an important property. It seems likely that the AEAD construction in Section 3.1 [1] provides good RUP security as any bit flips results in a completely random unverified plaintext.

    - *Reforgeability Resilience / Robust Authenticated Encryption (RAE).* We think it should be a requirement that after a successful forgery, it is still hard to find subsequent forgeries [30]. Optimally, finding one or more forgeries should not increase the adversary's ability to make further forgeries. It seems likely that the construction in Section 3.1 of [1] has good reforgeability resistance properties.

    - *Resistance to side-channel and fault attacks.* We think something similar to what NIST wrote in [31] should be a goal, i.e. "the ability to provide it easily and at low cost is highly desired."

- **Key wrap, DAE, and MRAE.** The key-wrap problem, as described in [32–33], is to provide confidentiality and integrity protection without the use of nonces and without relying on strong random number generators. A key wrap encryption function KW(K, P) takes as input a key and a plaintext and outputs a ciphertext. One solution to the key wrapping problem is deterministic authenticated-encryption (DAE) [33], but DAE leaks information to an attacker in the sense that an attacker can trivially see that two plaintexts are equal. An alternative solution to the key wrapping problem, using the same interface, is to build a hedged key wrap encryption function as a derived function of a Misuse-Resistant AE (MRAE) [33] encryption function:

  Hedged-KW(K, P):
  1. Let N be a random nonce. Let A be the empty string.
  2. C' = MRAE(K, N, P, A)
  3. Return C = N || C'

  This provides strictly better security properties than DAE at the expense of slightly more message expansion. Hedged key wrap might also be implemented with a nonce-hiding MRAE. Even with a bad random number generator (RNG), this hedged construction gives IND-CCA2 security with DAE security as a worst case. The addition of randomness likely increases security against side-channel attacks and fault attacks. While not basing security on a strong RNG is a requirement to key wrap, we don't think determinism is a requirement; it just happened to be a feature of the solutions. We think NIST should specify an accordion-based MRAE instead of a DAE. An MRAE with empty strings as nonce and associated data is a DAE.

- **Tweak size.** We agree with NIST that the accordion cipher should support long variable-length tweaks. To securely handle AEAD encryption with random nonces in the derived construction suggested by NIST, the tweak size needs to be at least 256 bits. We think 256-bit nonces (or close to 256-bit) should be standard in new 256-bit AEADs. A 128-bit tweak size is too small, as an attacker can find a (key, nonce) collision with an attack complexity of $2^{129} / r$, where r is the number of random nonces [9]. With a 256-bit nonce, the security with random nonces is $\approx 257 - \log_2 r$. We note that the attack complexity is likely higher if the AEAD mode is nonce hiding [18], and that the damage caused by the attack is lower if the AEAD is misuse resistant [33]. The associated data can be very large but could be hashed (with or without a key) before being used as a tweak. Long variable-length tweaks would be beneficial, as that eliminates hashing in the derived functions. Additionally, 128-bit nonces are too small to provide enough randomness against pre-computation attacks such as Hellman's TMTO attack when the key size if 256-bits. We think it is important that AEADs based on the accordion mode support nonces longer than 128 bits.

- **Beyond birthday-bound security**. We agree that the accordion cipher should support AES with a 128-bit block. However, we think that the accordion cipher should support both AES with a 128-bit block as well as Rijndael with a 256-bit block from the start. Approving use of Rijndael with a 256-bit block seems like a much easier task than standardizing the accordion cipher. We would have liked to see NIST already begin the task of approving Rijndael with a 256-bit block, which was one of the suggested outcomes of the third encryption workshop [3]. There is a need for AEAD schemes with confidentiality advantages better than $\sigma_a^2/2^{129}$, where

σ_a is the number of encrypted 128-bit blocks. This would reduce the need for rekeying and enable the encryption of very large plaintexts. We don't think the term "beyond birthday-bound security" should be used. The goal should be to have a high concrete security level, and the easiest way to achieve this is to use a block cipher with a 256-bit block length. A confidentiality advantage of $\sigma_r^2/2^{257}$, where σ_r is the number of encrypted 256-bit blocks, is a good solution but is not "beyond birthday-bound security".

- **Key and context commitment**. While we welcome the recent discussion and academic work on commitment, we think key and context commitment is and will remain a niche use case. We think there should be derived functions without tags (for file system encryption in existing file systems) as well as short tags. Many of the commitment attacks are on systems using passwords for encryption. Passwords are inherently insecure and should only be used as part of multi-factor authentication. In many use cases, such as file encryption [34], a committing MAC in the header is a better solution than a committing AEAD with large tags. We hope the derived functions provide commitment with security strength of half the tag length. Systems assuming that AEADs provides commitment seems common. One example not mentioned in the literature is COSE [35], which performs trial encryption to find the "correct" key. People will not start using 256-bit tags unless they understand that they need commitment, but having 64-bit commitment security would practically be a lot better than 0 bits of commitment security.

- **Collaborative approach**. A collaborative process for developing the new accordion cipher seems like the right approach. At the end of the collaborative effort, we expect NIST to have a longer review period than for FIPS 203–205. NIST publishing a draft for an accordion cipher and inviting cryptanalysis would likely attract many researchers. NIST could even have a competition with prizes (money or honor) for the best attacks.

References

[1] “Proposal of Requirements for an Accordion Mode”
https://csrc.nist.gov/files/pubs/other/2024/04/10/proposal-of-requirements-for-an-accordion-mode-dis/iprd/docs/proposal-of-requirements-for-an-accordion-mode-discussion-draft.pdf

[2] “NIST Cryptographic Standards and Guidelines Development Process”
https://nvlpubs.nist.gov/nistpubs/ir/2016/nist.ir.7977.pdf

[3] “The Third NIST Workshop on Block Cipher Modes of Operation 2023”
https://csrc.nist.gov/events/2023/third-workshop-on-block-cipher-modes-of-operation

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

[5] “The AEGIS Family of Authenticated Encryption Algorithms”
https://datatracker.ietf.org/doc/html/draft-irtf-cfrg-aegis-aead

[6] “Encryption algorithm Rocca-S”
https://datatracker.ietf.org/doc/html/draft-nakano-rocca-s

[7] “Galois Counter Mode with Secure Short Tags (GCM-SST)”
https://datatracker.ietf.org/doc/html/draft-mattsson-cfrg-aes-gcm-sst

[8] “Zero Trust Architecture”
https://datatracker.ietf.org/doc/html/draft-mattsson-cfrg-aes-gcm-sst

[9] “Proposals for Standardization of Encryption Schemes”
https://csrc.nist.gov/csrc/media/Events/2023/third-workshop-on-block-cipher-modes-of-operation/documents/accepted-papers/Proposals%20for%20Standardization%20of%20Encryption%20Schemes%20Final.pdf

[10] “AES-GCM-SIV: Nonce Misuse-Resistant Authenticated Encryption”
https://www.rfc-editor.org/rfc/rfc8452.html

[11] “Hedged ECDSA and EdDSA Signatures”
https://datatracker.ietf.org/doc/html/draft-irtf-cfrg-det-sigs-with-noise

[12] “AES Proposal: Rijndael”
https://csrc.nist.gov/csrc/media/projects/cryptographic-standards-and-guidelines/documents/aes-development/rijndael-ammended.pdf

[13] “Simpira v2: A Family of Efficient Permutations Using the AES Round Function”
https://eprint.iacr.org/2016/122.pdf

[14] “SAGE-23-01 Specification of Milenage-256 finalized”
https://www.3gpp.org/ftp/TSG_SA/WG3_Security/TSGS3_112_Goteborg/Docs/S3-234134.zip

[15] “Cryptographic Agility”
https://www.dhs.gov/sites/default/files/2022-05/22_0512_plcy_2966-01_cryptographic-agility-infographic.pdf

[16] “National Security Memorandum on Promoting United States Leadership in Quantum Computing While Mitigating Risks to Vulnerable Cryptographic Systems”
https://www.whitehouse.gov/briefing-room/statements-releases/2022/05/04/national-security-memorandum-on-promoting-united-states-leadership-in-quantum-computing-while-mitigating-risks-to-vulnerable-cryptographic-systems/

[17] “Report on Post-Quantum Cryptography”
https://nvlpubs.nist.gov/nistpubs/ir/2016/nist.ir.8105.pdf

[18] “Nonces Are Noticed: AEAD Revisited”
https://eprint.iacr.org/2019/624.pdf

[19] ”The Transport Layer Security (TLS) Protocol Version 1.3”
https://www.rfc-editor.org/rfc/rfc8446.html

[20] ”The Datagram Transport Layer Security (DTLS) Protocol Version 1.3”
https://www.rfc-editor.org/rfc/rfc9147.html

[21] ”Using TLS to Secure QUIC”
https://www.rfc-editor.org/rfc/rfc9001.html

[22] “Security Notions for Fully Encrypted Protocols”
https://www.petsymposium.org/foci/2023/foci-2023-0004.pdf

[23] “Authenticated Encryption with Replay prOtection (AERO)”
https://datatracker.ietf.org/doc/html/draft-mcgrew-aero

[24] “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

[25] "IP Encapsulating Security Payload (ESP)"
https://www.rfc-editor.org/rfc/rfc4303.html

[26] "The Secure Shell (SSH) Transport Layer Protocol"
https://www.rfc-editor.org/rfc/rfc4253.html

[27] "Ephemeral Diffie-Hellman Over COSE (EDHOC)"
https://www.rfc-editor.org/rfc/rfc9528.html

[28] "The Messaging Layer Security (MLS) Protocol"
https://www.rfc-editor.org/rfc/rfc9420.html

[29] "GLEVIAN and VIGORNIAN: Robust beyond-birthday AEAD modes"
https://eprint.iacr.org/2023/1379.pdf

[30] "Robust Authenticated-Encryption: AEZ and the Problem that it Solves"
https://eprint.iacr.org/2014/793.pdf

[31] “Status Report on the Final Round of the NIST Lightweight Cryptography Standardization Process”
https://nvlpubs.nist.gov/nistpubs/ir/2023/NIST.IR.8454.pdf

[32] “Request for Review of Key Wrap Algorithms”
https://eprint.iacr.org/2004/340.pdf

[33] “Deterministic Authenticated-Encryption: A Provable-Security Treatment of the Key-Wrap Problem”
https://eprint.iacr.org/2006/221.pdf

[34] “age file Encryption”
https://github.com/FiloSottile/age

[35] “CBOR Object Signing and Encryption (COSE)”
https://www.rfc-editor.org/rfc/rfc9052.html