ERICSSON

Date: June 2, 2026

Ericsson AB
Group Function Technology
SE-164 80 Stockholm
SWEDEN

Comments on SP 800-38F (Methods for Key Wrapping)

Dear NIST,

Thanks for your continued efforts to produce well-written, user-friendly, and open-access security documents. We welcome NIST's plans to revise SP 800-38F and to remove TKW from the specification. Below we provide our feedback on NIST's questions [1], as well as additional comments on other aspects of SP 800-38F [2].

General comments and feedback on NIST's questions [1]:

- We agree with [1] that ad hoc combinations of approved encryption and authentication methods can introduce significant security vulnerabilities. Securely composing encryption and authentication is non-trivial, and most users cannot be expected to implement such combinations correctly. Historical experience illustrates this clearly: several compositions used in TLS and SSH exhibited serious weaknesses, whereas the compositions adopted in 3GPP NAS, 3GPP PDCP, and IPsec ESP are secure. This underlines the importance of involving cryptographic expertise, such as ETSI SAGE and IETF CFRG in the standardization process.

  The problem of insecure compositions is not limited to key wrapping, but applies more broadly to encrypted data in general. We therefore suggest that NIST provide general guidance, in a specification separate from NIST SP 800-38F, on how to securely combine a keystream generator, such as AES-CTR(K, N, L) or KMAC(K, N, L) with a MAC algorithm in an Encrypt-then-MAC (EtM) composition, including guidance on nonce handling and associated data. Such guidance could follow an approach similar to that used for composite KEM construction in Section 4.6.1 of NIST SP 800-227 [3]. We see little justification for the continued use of ECB, CBC, CFB, and OFB modes, and believe these modes should be retired. ECB is not even IND-CPA secure; CBC is vulnerable to padding oracle attacks; and CFB and OFB are seldom used today. We suggest that NIST makes it very clear that encryption without integrity is insufficient, as it does not provide security against active attackers. Unauthenticated encryption is a significant practical problem. Encryption shall be authenticated. [1]

[1] Except in cases such as message 2 of the SIGMA-I protocol, where unauthenticated encryption can be proven sufficient.

It is very problematic that IETF uses the term ECB for header protection in QUIC [4] and DTLS 1.3 [5]. We have encountered cases where this terminology has led practitioners to believe that ECB is fine for general data protection. It also increases the likelihood that cryptographic libraries will continue to expose ECB interfaces, when they should remove ECB support entirely. We therefore suggest that NIST withdraw the ECB specification as soon as possible and replace it with a narrowly scoped mode, Single Block Encryption (SBE), intended for use cases such as header protection and PIN encryption. Alternatively, the requirement in FIPS 197 [6] stating that AES *"shall be used in conjunction with a FIPS-approved or NIST-recommended mode of operation"* could be revised.

– Examples of currently implemented and widely used secure compositions include: SRTP, which uses AES-128-CTR + HMAC-SHA1 in an EtM composition; Secure Frame (SFrame) [7], which uses AES-128-CTR + HMAC-SHA-256 in an EtM composition; and 4G and 5G 3GPP protocols, where NAS uses AES-128-CTR + AES-128-CMAC in an EtM composition, while PDCP uses AES-128-CTR + AES-128-CMAC in a MAC-then-Encrypt (MtE) composition. Future 5G systems are expected to optionally support AES-256-CTR + AES-256-GMAC-SST [8] in the same compositions.

  We agree with NIST's guidance in 800-38F [2] to segregate keying material from other data. A prime example of the security risks inherent in failing to segregate keys from other types of data is SDP Security Descriptions [9] also known as Security Descriptions Enabling Surveillance (SDES). In SDES, keys are transmitted within the signaling channel itself. This means the keys are exposed in plaintext to any party with access to the signaling channel, including passive eavesdroppers and active intermediaries such as Session Border Controllers, which terminate and inspect signaling by design. Furthermore, because signaling traffic is routinely captured for troubleshooting and auditing, these plaintext keys are frequently written to persistent system logs and diagnostic traces, where they remain vulnerable long after the session has ended. See [10] for a detailed analysis. We strongly suggest that NIST explicitly discourage all use of SDP Security Descriptions for key transport.

– To answer the question in [1], we do not believe that NIST should limit approval exclusively to NIST's currently approved AEAD modes or to compositions already standardized in existing protocols. Assuming sufficiently large key sizes such that brute-force key recovery attacks can be neglected, the expected number of successful forgeries against a strong integrity mechanism should, for all practical values of v, be $\approx v/2^t$, where v is the number of forgery attempts and t is the tag length. This property holds for AES-CTR combined with a 96- or 128-bit KMAC authentication tag, assuming the composition is instantiated in an Encrypt-then-MAC construction with cryptographically separated keys. In contrast, the forgery bounds of GCM and CCM are substantially weaker and deviate significantly from the expected $v/2^t$ behavior. Both IETF [7] and 3GPP [8] have recently standardized new compositions, reflecting the fact that NIST's existing AEADs are insufficient for some use cases. We therefore believe that new secure compositions following NIST guidance should be approved.

– Given the expected advantages of Accordion-based constructions [11–12], in particular Rijndael-256-based Acc256, we no longer believe that NIST should approve AES-SIV. We

instead suggest that derived functions based on Accordions should supersede both AES-KW and AES-SIV.

Detailed comments on SP 800-38F:

– *"KW, KWP, and TKW are each approved for the protection of general data, as well as cryptographic keys."*

"Each provides an option for protecting keys in a manner that is distinct from the methods that protect general data. Segregating keys from general data can provide an extra layer of protection."

As KW and KWP are approved for protecting general data, we do not consider it accurate to describe them as "distinct from the methods that protect general data." However, we believe that only IND-CCA-secure encryption schemes should be approved for the protection of general data. Moreover, simply using a distinct algorithm does not, by itself, provide meaningful security benefits. The additional protection instead arises from segregating key protection from general data protection, using independent and potentially longer keys for key protection, and selecting a more robust algorithm for protecting keys.

We wish AES-KW(P) had seen broader use in applications where performance is not critical. The KW construction was a good design of its era and pioneered the standardization of robust authenticated encryption, utilizing a variable-length block cipher design and an encode-then-encipher approach. Authenticated-encryption modes derived from Accordions [11–12] should retain the advantages of KWP, while addressing its limitations, including improved performance, nonce-misuse resistance, IND-CCA2 security, support for associated data, variable-length padding, larger ICVs, formal security proofs, and AERO-style replay protection.

– *"In 2001, NIST posted a document entitled "AES Key Wrap Specification" on NIST's Computer Security Resource Center web site as an unofficial suggestion for the protection of cryptographic keys. That algorithm is essentially equivalent to KW as specified in this Recommendation. In 2002, two industry groups published specifications of key-wrap algorithms that were based on the specification that NIST posted."*

We suggest that NIST reuploads this important historical document [13] as well as the "NIST Key Wrap requirement" it refers to. Archives are important for long-term traceability, enabling stakeholders to review how decisions were made and to clearly identify who proposed, supported, or challenged specific choices. Transparency and openness are essential for building trust in cybersecurity and cryptography.

– The NIST Key Wrap requirements stated in [13] are well-designed and sound. However, the requirements and related statements in other documents such as [14] and [15] lack clear rationale and contain technical inaccuracies. The ANSI document [14] correctly identifies IND-CCA2 as the intended security goal, but it does not explain why nonces are excluded, nor why algorithms that do not even meet IND-CPA security are considered. Similarly, [15] incorrectly equates the absence of nonces with the absence of randomness and suggests that a

deterministic scheme without nonces is "just as good" as IND-CCA2 for transporting cryptographic keys.

We do not believe that determinism and the absence of nonces should be design criteria for key wrapping, as these choices lead to the vulnerability “Equality of Plaintexts” described in Appendix A. Instead, any future design should focus on mitigating this vulnerability while still ensuring DAE security [15] in the presence of a poor random number generator or repeated nonces.

- *"ICV1 The 64-bit default ICV for KW: 0xA6A6A6A6A6A6A6A6"*
  "ICV2 The 32-bit default ICV for KWP: 0xA65959A6"

  The term “default” suggests that an implementation could choose alternative ICV values. However, Section 8 states that W and W^{-1} are not approved for use independently of AES-KW and AES-KWP. We therefore suggest removing “default”.

  The revision could benefit from an explanation of how the ICV constants were chosen. We note that 0xA6 has Hamming weight 4 and that 0x59 is its bitwise complement. The use of special constants raises questions about whether the algorithm remains secure under alternative choices. We welcome the fact that NIST is planning to use all-zero ICVs in Accordion-derived functions [12], which suggests that the security does not depend on selecting non-trivial or carefully crafted constant values.

- *“If the designated cipher function for a key-wrap algorithm is chosen to be the AES decryption function, then $CIPH^{-1}_K$ will be the AES encryption function.”*

  NIST should consider limiting the KW-AE and KWP-AE constructions to use only the forward cipher operation (AES encryption). Allowing a choice between the forward and inverse cipher operations does not improve security and can hinder interoperability.

- We recommend that Section 5.1 explicitly require KW and KWP to utilize cryptographically independent KEKs.

- *“KW can accept longer inputs; nevertheless, the plaintext for KW-AE shall be limited to fewer than 2^{54} semiblocks.”*

  We suggest removing the statement “KW can accept longer inputs,” as Appendix A.4 demonstrates that KW cannot securely support longer inputs.

- The plaintext and ciphertext limits for KWP are not aligned. A $2^{32} - 1$ octet plaintext results in a $2^{32} + 8$ octet ciphertext, corresponding to $2^{29} + 1$ semiblocks. We therefore suggest adjusting the ciphertext limit for KWP to “2 to $2^{29} + 1$ semiblocks”.

- We suggest that Section 6 reference RFC 3394 [16] as a complementary algorithm specification. The index-based alternative algorithm description on page 5 of RFC 3394 is more easily implemented in software and can be implemented in-place, a property we recommend

NIST explicitly highlight. This in-place capability is particularly relevant in memory-constrained environments, such as low-power IoT devices, where avoiding additional memory allocations can be important.

– We suggest that NIST provide guidance on implementation security. The intermediate variables A and R shall be destroyed (zeroized) before returning from both the wrapping function W and the unwrapping function W^{-1}, noting that compilers may optimize away naive memset calls. In addition, on failure, implementations shall not release any portion of the output of W^{-1} to the caller before all ICV and padding checks have completed successfully. The output of W^{-1} must be destroyed (zeroized) prior to returning FAIL. The current pseudocode for KWP-AD "leaks" whether a FAIL is due to the ICV check or padding validation. Since W^{-1} is assumed to behave as a wide-block cipher, this information might not provide a practical advantage to an attacker, but the revision should clarify whether this is acceptable in production code or whether constant-time error handling is required.

– Depending on the timing of NIST's publication of the Acc256 Accordion and its derived functions [11–12], we suggest that NIST consider adding support of 256-bit block sizes, such as Rijndael-256 with 128-bit semiblocks. This modification appears conceptually straightforward, and we believe the current document would have been cleaner if W and TW had been specified as a single function with the block size treated as a variable, rather than as separate fixed-size designs. In most block cipher modes, the narrow 128-bit block size of AES is a limiting factor. As shown in Appendix A.4, this also applies to AES Key Wrap.

– We recommend that the revision explains that, without additional padding, AES-KW reveals the length of the wrapped key. This information may allow an attacker to focus efforts on the shortest wrapped keys or passwords. We suggest describing how applications can mitigate this weakness by adding application-layer variable-length padding to the AES-KW or AES-KWP plaintext. Similarly, context binding can be achieved by appending application-specific context information to the plaintext before wrapping. These measures also reduce the probability of successful forgery attacks.

– We suggest that NIST adds a paragraph on replay protection. Replay protection is an essential security property. While GCM and CCM with sequential nonces support protocol-level replay protection, AES-KW does not use nonces and therefore requires augmentation with a separate replay protection mechanism. We also recommend that NIST explain that the wrapping function W and encode-then-encipher can be used to provide not only integrity protection, but also AERO-style replay protection [17], with lower overhead than conventional approaches.

– *"In particular, if the adversary chooses a string at random with a valid ciphertext length, the probability that the string will be a genuine ciphertext is exactly 1 in 2^{64} for KW, and approximately 1 in 2^{64} for KWP. The probability that a randomly chosen ciphertext will appear to be genuine for TKW is greater, 1 in 2^{32}"*

This is not a realistic attack model, since an adversary would mount adaptive chosen-ciphertext attacks designed to maximize the success probability. As explained in Appendix A.4, such

attacks slightly increases the success probability. We recommend clarifying this point and stating that the forgery probability for KW is approximately 1 in 2^{64}. However, the bound $P < 2^{-64}$ is insufficient to demonstrate this, as the total forgery probability is $P_f = 2^{-64} + P$. For KW with $m < 2^{54}$ the forgery probability is $P_f \lesssim 2^{-64} + 2^{-60}/6! \approx 2^{-63.97}$.

For KWP, the forgery probability is $P_f \approx 2^{-64} + 2^{-72} \approx 2^{-63.99}$, but it can be significantly lower if the receiver enforces a specific plaintext length. For example, when protecting a 66-byte P-521 private key, the 6-byte zero padding required by KWP adds 48 bits of constraint. If the receiver validates the exact expected length, the effective forgery probability drops to approximately 1 in 2^{112}.

For TKW with $m < 2^{28}$ the forgery probability is $P_f \lesssim 2^{-32} + 2^{-24}/6! \approx 2^{-31.56}$.

– We suggest that NIST note that, for AES-KW, the expected number of successful forgeries after v forgery attempts is $\approx v/2^{64}$, even after a successful forgery. In practical deployments, this can compare favorably to CCM and GCM despite their 128-bit tags. In fact, after only $\approx 2^{33}$ queries, AES-CCM with 128-bit tags is expected to experience more forgeries than AES-KW.

– *"if S is extremely long, about 2^{67} semiblocks"*

The value 2^{67} appears to be slightly inaccurate. A collision probability of $P = 0.5$ seems to occur already at $m \approx 2^{65.41}$, while for $m = 2^{66}$, $P \approx 1$.

– The use of the variable S as input to W^{-1} in Appendix A.4 may be confusing since, according to Section 6.1, S is the input to W, whereas C is the input to W^{-1}. If possible, the notation A^{i6} should be changed to A^{i_6} for clarity. We also note that for $i_6 = 6m$, there is always a collision $A^{6m} = B^{6m} = S_1$, and this collision does not imply collisions for every index j such that $5m < j < i_6$. We believe the "collision conditions" should be stated as follows:

$$\begin{gathered} 0 < i_1 < m \\ i_1 + m \le i_2 \\ i_2 + m \le i_3 \\ i_3 + m \le i_4 \\ i_4 + m \le i_5 \\ i_5 + m \le i_6 < 6m \end{gathered}$$

Similarly, we believe the text should read: "$A^j = B^j$ for each index j such that $5m \le j < i_6$, or $4m \le j < i_5$", "as described in Equation 1 with $d = i_1 + 1$", and "the first i_1 semiblocks of the resulting plaintext".

We further note that, in addition to modifying the last ciphertext semiblocks of a valid ciphertext to produce a forgery, an adversary may also modify the first ciphertext semiblock. We suggest that NIST explicitly mention this possibility.

- *"Moreover, the first i_1-1 semiblocks of the resulting plaintext will be identical to the plaintext from which S||T was generated."*

  It should be clarified that this implies an attacker with a decryption oracle will learn a prefix of the unknown plaintext S. That a forgery leads to attacks on confidentiality is common; however, in this case it shows that AES-KW does not fulfil the NIST Key Wrap requirement [13, 16] that "each ciphertext bit should be a highly non-linear function of each plaintext bit, and (when unwrapping) each plaintext bit should be a highly non-linear function of each ciphertext bit." This should be mentioned in the revision.

- The observation in Appendix A.4 also yields a theoretical encryption-only distinguishing attack. Suppose an attacker encrypts x plaintexts of length $\approx 2^{54}$ semiblocks, where all semiblocks are identical except P_1. Then the probability that the last d semiblocks of two ciphertext collide is $P \approx x^2 \cdot 2^{-59}/6!$. Setting $P = 0.5$ gives $x \approx 2^{34.75}$ and a total data complexity of $\approx 2^{88.75}$ semiblocks.

- The attack in Appendix A.4 appears to rely on all six passes using the same direction. Reversing the direction of one or more passes, e.g., performing the final pass in the opposite direction, seems to mitigate the attack.

- *"Consequently, for KW, the probability, P, that the forgery attack succeeds in this model is"*

  "Consequently, if m < 2^{54}, as required in Sec. 5.3.1, then P < 2^{-64}"

  Note that P is the additional forgery probability; the total forgery probability is $P_f = 2^{-64} + P$. The requirement is therefore that $P \ll 2^{-64}$, which holds for $m < 2^{54}$.

- *"For TKW, the analogous conclusion is that if m < 2^{28}, as required in Sec. 5.3.1, then P < 2^{-32}"*

  For $m < 2^{28}$ the forgery probability is $P_f \approx 2^{-32} + 2^{-24}/6! \approx 2^{-31.56}$. We believe the limit should have been $m < 2^{27}$.

- *"possibly even in a particularly strong manner, i.e., with "beyond-birthday-phenomenon security."*

  The observation in Appendix A.4 demonstrates that the wrapping function W does not achieve beyond-birthday-bound (BBB) security. Furthermore, our distinguishing attack on AES-KWP below demonstrates an additional way in which it fails to achieve BBB security. We recommend removing the quoted phrase and adding a clarification in Appendix A.4 that the observation shows that AES-KW falls short of BBB security, which is not an attack but an important qualification of its security properties.

- Rogaway et al. provide a critique [15] of AES-KW based on the specification in [14]. While we agree with many of their observations, we do not agree with the following points:

  "mysterious aspects of the construction (eg, the byte-reversals or xoring-in counters)"

"The xoring of the counter 6t + i into the blockcipher output is not explained; why is this done?"

We view the injection of a round-dependent constant or counter as not only sensible but standard cryptographic hygiene and best practice. XORing the counter t into the state is an efficient way to break symmetry between iterations and prevent short cycles, fixed points, and slide-like behaviors. Without the round counter, AES-KW can be viewed as a fixed permutation Q composed with itself six times, i.e., $W = Q^6$. We suggest that NIST include text along the lines above to motivate the design of the wrapping function W.

As a concrete example, when the round counter t is omitted, W does not behave like a random permutation, since cycles in AES induces a large family of fixed points in W. If $A\|R$ is a fixed point of AES, i.e., $\mathrm{AES}(A\|R) = A\|R$, then $A\|R\|R$, $A\|R\|R\|R$, ... are all fixed points of W. Similarly, if $A\|R$ lies on a 2-cycle, i.e., $\mathrm{AES}(A\|R) = B\|S$ and $\mathrm{AES}(B\|S) = A\|R$, then $A\|R\|S\|R$, $B\|S\|R\|S$, $A\|R\|S\|R\|S\|R$, $B\|S\|R\|S\|R\|S$, ... are all fixed point of W. 3- and 6-cycles also induce fixed points in W. For a random permutation, the expected number of fixed points is 1. In contrast, for W without the round counter t, the expected number of fixed points is 2−5, depending on the input length.

– We dislike the design choice in KWP [18, 2] to treat plaintexts that fit within a single semiblock differently, including the decision to use the same key K as for longer plaintexts. Conceptually, this is similar to reusing the same key across two distinct modes of operation. Different cryptographic functions should use independent keys or derived subkeys. Similar patterns of key reuse across distinct constructions have led to serious vulnerabilities in the past, for example, when the same key is used for both CBC and CBC-MAC, or for CMAC and a Key Check Value (KCV) [19].

  The inclusion of short 128-bit ciphertexts in KWP enables a trivial encryption-oracle distinguisher: an attacker can query the encryption of all 2^{64} 8-octet strings and observe the absence of ciphertext collisions. We suggest that NIST include this observation in the revision.

  Additionally, for KWP the observation in Appendix A.4 can be extended as follows. Given a valid ciphertext $S \| T$, the attacker queries the decryption of $S_1 \oplus [6m]_{64} \| T$. If the result is accepted as valid, it will with high probability have the format $ICV2 \| [8]_{32} \| X$. The attacker then queries the decryption of $ICV2 \| [8]_{32} \| Y$, obtaining the ciphertext $C_1 \| C_2$. The ciphertexts $S_1 \| T$ and $C_1 \oplus [6m]_{64} \| S_2 \| \ldots \| S_m \| C_2$ now have a guaranteed collision $A^{6m-1} = B^{6m-1} = ICV2 \| [8]_{32}$. The probability that $C_1 \oplus [6m]_{64} \| S_2 \| \ldots \| S_m \| C_2$ is a successful forgery is therefore

$$P_f \approx \frac{1}{2^{64}} + \frac{1}{5!} \cdot \left(\frac{m}{2^{64}}\right)^5 .$$

  Consequently, if $m < 2^{51}$, then $P_f \lesssim 2^{-64} + 2^{-71.91} \approx 2^{-63.99}$. This shows that the use of a 64-bit length field in KWP without $ICV2$ would have been a questionable design choice.

  We believe that KW made the right design choice in disallowing 64-bit plaintexts. We suggest that NIST raise the minimum allowed plaintext length in KWP to 9 octets. In practice, the use of 1−8 octet plaintexts is likely negligible or non-existent since Single-DES was disallowed.

- As noted by Rogaway [15], "*The number of blockcipher calls seems large: roughly 12 per block of data (the same price paid for X or H). This is six times more than that used for AKW1".* There is no justification for the choice of six passes beyond the observation in Appendix A.4. It may have been preferable to use fewer passes together with stricter plaintext-length limits. For example, using three passes and $m < 2^{40}$ gives $P_f \lesssim 2^{-64} + 2^{-3(64-40)}/3! \approx 2^{-64.00}$, while substantially reducing the computational cost.

- We note that in addition to varying the number of rounds, one could also consider AES-KW with an ICV of length β, an accumulator length α, a plaintext limit of 2^{μ} blocks of size $128 - \alpha$, and r rounds. Assuming $\alpha > \mu$ and $r \gg 1$, against the attack in Appendix A.4, the total forgery probability P_f appears to satisfy:

$$P_f \lesssim 2^{-\beta} + 2^{-r(\alpha-\mu)}/r!$$

John Preuß Mattsson,
Expert Cryptographic Algorithms and Security Protocols
On behalf of the Ericsson Cryptography Team

References

[1] NIST Requests Public Pre-Draft Comments on SP 800-38F Rev. 1 IPD
https://csrc.nist.gov/pubs/sp/800/38/f/r1/iprd

[2] Recommendation for Block Cipher Modes of Operation: Methods for Key Wrapping
https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-38F.pdf

[3] Recommendations for Key-Encapsulation Mechanisms
https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-227.pdf

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

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

[6] Advanced Encryption Standard (AES)
https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197-upd1.pdf

[7] Secure Frame (SFrame): Lightweight Authenticated Encryption for Real-Time Media
https://www.rfc-editor.org/rfc/rfc9605.html

[8] Galois Counter Mode with Strong Secure Tags (GCM-SST)
https://www.ietf.org/archive/id/draft-mattsson-cfrg-aes-gcm-sst-18.html

[9] Session Description Protocol (SDP) Security Descriptions for Media Streams
https://www.rfc-editor.org/rfc/rfc4568.html

[10] SDP Security Descriptions is NOT RECOMMENDED and Historic
https://datatracker.ietf.org/doc/html/draft-mattsson-dispatch-sdes-dont-dont-dont

[11] NIST Launches Development of Cryptographic Accordions
https://csrc.nist.gov/pubs/sp/800/197/a/iprd

[12] Requirements for Cryptographic Accordions
https://nvlpubs.nist.gov/nistpubs/ir/2025/NIST.IR.8552.pdf

[13] AES Key Wrap Specification
https://web.archive.org/web/20041031122517/http://www.csrc.nist.gov/CryptoToolkit/kms/key-wrap.pdf

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

[15] Deterministic Authenticated-Encryption
https://eprint.iacr.org/2006/221.pdf

[16] Advanced Encryption Standard (AES) Key Wrap Algorithm
https://www.rfc-editor.org/rfc/rfc3394.html

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

[18] Advanced Encryption Standard (AES) Key Wrap with Padding Algorithm
https://www.rfc-editor.org/rfc/rfc5649.html

[19] Ericsson comments on SP 800-38B The CMAC Mode for Authentication
https://csrc.nist.gov/csrc/media/Projects/crypto-publication-review-project/documents/initial-comments/sp800-38b-initial-public-comments-2024.pdf