

Pre-Draft Public Comments on Proposed Revision of SP 800-38F

Comment period: May 5 – July 10

On May 5, 2026, NIST’s cipher modes team [requested pre-draft public comments](#) for the planned revision of SP 800-38F: *Recommendation for Block Cipher Modes of Operation: Methods for Key Wrapping*, 2012.

The comments that NIST received during the comment period are collected below.

Information about NIST’s block cipher modes of operation is available at the [project site](#). Ongoing discussion of technical questions may be found at the [NIST cipher modes forum](#).

LIST OF COMMENTS

1.	Niranjan Sharma, May 6, 2026.....	2
2.	Anupam Kumar Pandey, May 8, 2026.....	3
3.	Ericsson (John Preuß Mattsson), June 2, 2026.....	4
4.	Amazon Web Services (Alicia Squires, Panos Kampanakis), June 19, 2026	15
5.	Thales (COSTA Graham), July 6, 2026.....	16
6.	Entrust (Richard Kettlewell), July 8, 2026.....	21
7.	IBM (Richard Kisley), July 9, 2026.....	23
8.	Arpan Sharma, July 9, 2026.....	26
9.	National Aeronautics and Space Administration (Craig Biggerstaff) , July 10, 2026.....	31
10.	Trusted Computing Group, July 10, 2026.....	33
11.	Department of Energy OCIO (Tasha Barnes), July 10, 2026	34

1. Niranjn Sharma, May 6, 2026

Three notes on the pre-draft call.

1. TKW removal — yes, but the legacy decrypt path needs explicit treatment

Removing TKW from approved methods is the right call — TDEA is disallowed for encryption per SP 800-131A Rev. 3 since January 2024. The gap is what happens to the installed base of TKW-wrapped keys: HSM key escrow blobs, archival KMIP servers, regulated archives that still need to recover keys wrapped years ago.

Two options, pick one explicitly:

Allow TKW unwrap only for legacy recovery, with a sunset date. Aligns with the way SP 800-131A r3 handles TDEA decryption of legacy data.

Require migration: operators re-wrap under AES-KWP before a stated date.

Either is fine. Saying nothing is the failure mode — operators with TKW archives will not know whether they can still unwrap them and FIPS 140-3 vendors will not know what to put in their security policies during the transition.

2. Section 3.1 — yes, replace the blanket approval with an explicit list (NIST Q2)

The current language in Section 3.1 ("approved combinations of an approved encryption mode with an approved authentication method ... are approved for the protection of cryptographic keys") is a foot-gun. Bellare–Namprempre 2000 and the long line of follow-up work showed that arbitrary composition of encryption + MAC isn't generically secure — only Encrypt-then-MAC over the IV/nonce + ciphertext gives IND-CCA. The current text approves MAC-then-Encrypt and Encrypt-and-MAC compositions just as readily, as long as the two primitives are independently approved.

Recommendation: replace the open-ended approval with a closed list. For each entry, state:

Composition order shall be Encrypt-then-MAC over the IV/nonce + ciphertext.

Encryption key and MAC key shall be cryptographically independent (no shared secret, no derivation from the same input without explicit domain separation).

Per-operation IV/nonce handling and the failure mode if reused.

Also include a mapping table from each approved combination to the corresponding PKCS#11 mechanism IDs (CKM_AES_KEY_WRAP, CKM_AES_KEY_WRAP_KWP, CKM_AES_KEY_WRAP_PAD) and KMIP wrap profile names. Without that mapping, FIPS 140-3 vendors will keep asserting ad-hoc combinations in their security policies and validators will keep accepting them — which defeats the point of the restriction.

3. Ad-hoc combinations actually deployed today (NIST Q1)

What I see in production:

JOSE / JWE — RFC 7518 §5.2. A128CBC-HS256, A192CBC-HS384, A256CBC-HS512. AES-CBC + HMAC-SHA-2 with explicit key splitting and Encrypt-then-MAC. This is the most widely deployed ad-hoc combination — every OAuth / OIDC system that uses JWE relies on it. Correctly constructed.

TLS 1.2 cipher suites. TLS_WITH_AES_CBC_SHA / SHA256 — AES-CBC + HMAC-SHA1 / SHA256, MAC-then-Encrypt. Deprecated in TLS 1.3 (which is AEAD-only) but still in production at scale.

PKCS#11. CKM_AES_CBC + separate CKM_SHA256_HMAC over the wrapped key blob. Common in HSM vendor wrap profiles.

KMIP key wrap. Client-defined "MAC/sign + encrypt" combinations. The most uncontrolled surface — clients can request almost any ordering.

CMS / S/MIME (RFC 5652). AES-CBC content encryption + separate authenticated digest wrapper.

If NIST limits the approved set:

The JOSE ACBC-HS family should make it. It's correctly constructed (EtM, key separation in the spec) and ubiquitous; removing it would invalidate most of the federated identity stack overnight.

The TLS 1.2 CBC+HMAC suites should not. MAC-then-Encrypt, and TLS 1.3 has already moved past them. AES-GCM, AES-CCM, AES-GCM-SIV are AEAD primitives in their own right and don't belong in this discussion — they're already specified in SP 800-38C / 38D.

Best regards,
Niranjn Sharma
IEEE Senior

Dear Team NIST,

Greetings 🙏

My inputs [Feedback] as follows:

🔗 Feedback on Ad Hoc Combinations in Practice:

Current Practice:

In industrial and federal environments, many implementations still rely on "generic composition" to protect cryptographic keys when integrated Authenticated Encryption (AE) modes like AES-KW or GCM are not supported by the underlying architecture

Recommended Feedback:

NIST should acknowledge that while AES-KW is the gold standard for dedicated key wrapping, generic Encrypt-then-MAC (EtM) remains a vital fallback in systems where specialized hardware instructions for KW are unavailable. However, NIST should explicitly document that MAC-then-Encrypt and Encrypt-and-MAC are no longer appropriate for key protection due to known side-channel and padding oracle vulnerabilities.

🔗 Feedback on Limiting Approval to Explicit Sets:

The Proposal: NIST is considering whether to restrict approval to a specific list of combinations found in "prominent protocols" (e.g., those in SP 800-52 for TLS).

Recommended Feedback: "NIST should move toward an explicitly approved set of combinations to reduce the 'cryptographic flexibility' that often leads to implementation errors.

🔗 NIST has explicitly stated in the 2026 call for comments that TKW (Triple DES Key Wrap) should be removed from SP 800-38F because TDEA is no longer approved for encryption. Feedback should support this transition to ensure federal systems remain compliant with current security benchmarks.

Please let me know if need any Clarification.

Best wishes 🙏

Anupam Kumar Pandey [SME]

Senior Cyber Security Professional

WG Member: IEEE, IEC, ITU, NIST 📄

NICE CyberSec Career Ambassador

3. Ericsson (John Preuß Mattsson), June 2, 2026

Dear NIST,

Thanks for your continuous efforts to produce well-written open-access security documents. Please find attached our comments on SP 800-38F.

Best Regards,
John Preuß Mattsson

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 $\text{AES-CTR}(K, N, L)$ or $\text{KMAC}(K, N, L)$ 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 PDPC 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^{64} 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^{i_6} 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:

$$\begin{aligned} 0 &< i_1 < m \\ i_1 + m &\leq i_2 \\ i_2 + m &\leq i_3 \\ i_3 + m &\leq i_4 \\ i_4 + m &\leq i_5 \\ i_5 + m &\leq i_6 < 6m \end{aligned}$$

Similarly, we believe the text should read: “ $A^j = B^j$ for each index j such that $5m \leq j < i_6$, or $4m \leq j < i_5$ ”, “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 $m < 2^{64}$, 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., $\text{AES}(A||R) = A||R$, 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., $\text{AES}(A||R) = B||S$ and $\text{AES}(B||S) = A||R$, 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 $C_1 \oplus [6m]_{64} || S_2 || \dots || S_m || C_2$ now have a guaranteed collision $A^{6m-1} = B^{6m-1} = ICV2 || [8]_{32}$. The probability that $C_1 \oplus [6m]_{64} || S_2 || \dots || S_m || C_2$ 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>

Dear NIST,

We agree and applaud the removal of TWK.

Section 3.1 approves theoretical combinations of an approved encryption mode with an approved authentication method, but these are not specified. Key wrapping use-cases often use AES-256-GCM authenticated encryption. Section 3.1 should explicitly approve the most prevalent authenticated encryption method for key wrapping too. Other methods like AES-CBC with HMAC authentication or AES-CCM which are used in other protocols should not be included because of padding and other issues of the past and because they are not used widely for key wrapping. AES-256-GCM is the status quo and supported almost everywhere. There is no need for added complexity.

Thank you,
Alicia, Panos

5. Thales (COSTA Graham), July 6, 2026

Dear NIST,

Thank you for the opportunity to submit pre-draft comments on SP 800-38F Rev 1, Initial Preliminary Draft. Thales would like to respond to the two primary questions proposed:

1. What adhoc combinations - approved encryption mode and approved authentication method - are currently implemented, in practice?
2. Should NIST limit the approval to an explicit set of such combinations that occur within prominent protocols?

Responses are included below this email. In preparation for this review, Thales has considered the current draft of SP 800-38F, the NIST 'Authentication for Confidentiality Modes' page alongside the papers for category II referenced from it.

Thales has already submitted to NIST related comments on proposals to drop composite schemes as was already proposed by NIST in SP 800-131Ar3.

Should you have further questions or wish to discuss our responses in more depth, please do not hesitate to contact us.

Kind Regards,

Graham COSTA (he/him)

Security and Certifications Manager

Cyber Security Products / Cybersecurity and Digital Identity / UK
Thales

THALES
Building a future we can all trust

Thales response to NIST's, Pre-Draft call for comments on SP 800-38Fr1.

The following responses use terminology (Encrypt-and-MAC, MAC-then-Encrypt and Encrypt-then-MAC) as defined in: *Bellare and Namprempre's, 'Authenticated Encryption: Relations among notions and analysis of the generic composition paradigm'*.

Question 1 Response: What Adhoc Combinations?

Thales implements dedicated security products targeting a wide range of industries and where in particular, we'd like to draw attention to the following protocols using separate encryption from authentication.

Please do not consider this an exhaustive list where we assume NIST is already aware of the position and wide-spread use of composite schemes used by widely deployed IETF protocols such as IPsec, TLS (pre-1.3) and SSH etc:

FIDO Authenticators: FIDO protocols, supports options for Encrypt-then-MAC using either HMAC or CMAC alongside options for encryption using AES for key transport.

It is recorded that in addition, these standards also support options for Key Wrapping using KW / KWP, GCM and CCM.

To support continued integration of modern FIDO enabled systems with existing token, we strongly encourage the allowance of the suites above.

Devices supporting FIDO, commonly target FIPS 140 certification and are used by US Federal agencies in situations where they are unable to use PIV solutions.

Reference: <https://fidoalliance.org/specifications-overview/> - general standards.

<https://www.cisa.gov/resources-tools/resources/phishing-resistant-multi-factor-authentication-mfa-success-story-usdas-fast-identity-online-fido> - example of federal use case of deployment of FIDO as an alternative to PIV.

Global Platform, SCP03 and SCP04: These standards are used as the basis for secure channels to smartcard, secure elements and similar devices.

This standard currently uses **Encrypt-then-MAC**, with CBC mode used for encryption and then AES-CMAC used for authentication (different keys are used for both CBC encryption and MAC).

Given the resource constrained nature of many SmartCard and Secure Elements, the approach above offers a balance of performance with provable security for these devices.

SmartCard and similar devices widely target FIPS 140 for certification and where SCP03 and SCP04 protocols are implemented by the platform layers of these device that then support dedicated applets for Federal use cases such as PIV. The platform layers are widely used for card personalization (including key and application loading) including for PIV cards.

Reference: <https://globalplatform.org/specs-library/secure-channel-protocol-03-amendment-d-v1-2/>

[GlobalPlatform Secure Channel Protocol '04' – Amendment K v1.0.2 | GPC_SPE_182 - GlobalPlatform](#)

ICAO Standards for Machine Readable Travel Documents (e.g. Passports)

Secure messaging protocols used during inspection of travel documents based on the ICAO suite of protocols use **Encrypt-then-MAC** with AES-CBC used to encrypt data objects (including keys) and AES-CMAC used then used for authentication.

Relevant protocols are defined in ICAO Doc 9303 Part 11, Machine Readable Travel Documents, Part 11, Security Mechanisms for MRTDs.

Whilst passports don't typically target certification to standards such as FIPS 140 - devices such as HSM that support generation and interoperability with passports do.

Reference: https://www.icao.int/sites/default/files/publications/DocSeries/9303_p11_cons_en.pdf.

Use of Cryptographic Messaging Syntax (CMS) (RFC 5692) with PKCS#12 (RFC 7292) for private key archival

In addition to the examples above using MAC, we would also like to draw the attention to the use of CMS for key archival with PKCS#12. With these standards, objects are bundled and encrypted ahead of an outside signature being applied to the package. This is often used to bundle a private key alongside its complete certificate chain.

This is a common format used to store and exchange keys used by Public Key Infrastructure (PKI) and where the significance of signature goes to help authenticate the origin of the key bundle where asymmetric methods are the dominant method used based on the use of the certificate chain for authentication.

For this style of packaged key, we don't see any constraints on the signature algorithm being used which could be any currently approved by NIST.

Reference:

<https://www.rfc-editor.org/info/rfc5652>

<https://www.rfc-editor.org/info/rfc7292/>.

Question 2 Response: Should NIST limit the approval to an explicit set of such combinations?

Thales is in support of removing some options for unspecified combinations of an approved encryption mode with an approved authentication method in future draft of SP 800-38F.

In particular:

- **Thales is in support of removing options for 'Encrypt-and-MAC'** - acknowledging that some niche use cases exist for this construct may exist, we do not see the effort required by NIST to confirm its use is acceptable on a per-application basis as justified;
- **Thales would like to see continued unrestricted options for 'encrypt-then-MAC' - this option is provably secure** and leaves open options for future protocol designers to trade-off the merits of existing integrated authenticated encryption schemes against use of a composite scheme. This option in many cases is easier to accelerate for resource constrained hardware; and
- **Thales would like to see 'MAC-then-Encrypt' permitted for existing standardized protocols using this mechanism and that are widely deployed** (e.g. TLS 1.2 and earlier, DTLS 1.2). Whilst there are established risks linked to this paradigm, devices being developed today still need to support these established protocols for interoperability with older systems (and in particular, older applications) even though continued use would be by exception and likely disabled by default.

We strongly encourage that NIST limit the order of operations (i.e. MAC-then-Encrypt or Encrypt-then-MAC (for specific protocols)) but do not attempt to list specific combinations of algorithms (e.g. AES-CBC with HMAC).

We think listing specific algorithm combinations is counter-productive where the main security concern being address relates to the order of use of each algorithm.

Given the different levels of threats linked to object authentication vs decryption - we do **not** recommend that NIST constrain that authentication methods must be of an equal or equivalent strength to encryption methods use. This would in some cases force the use of stronger forms of authentication that may not be warranted based on the lifetime (and consequences of substitution) of a given key object.

Additional Comment - Future validation against SP 800-38F:

In order to retain the flexibility, present today, we do not suggest that NIST define a fixed format for testing combined schemes under the NIST CAVP program. This would lead to an unnecessary need to constrain the structures used and would add significant levels of assurance over-and-above having explicit checks performed during FIPS 140 certification.

We do suggest adding a specific section to SP 800-38F to outline a series of rules embodied in 'shall' statements that can be confirmed during FIPS 140 testing. An example statement could consist of the following:

"During FIPS 140 validation activities, the tester shall:

- *confirm that components of any combined authenticated encryption scheme are CAVP certified;*
- *For schemes claiming 'Encrypt-then-MAC':*
 - *confirm that all sensitive data is encrypted prior to calculation of the MAC over the resulting ciphertext; and*
 - *Confirm that no MAC (or other integrity data) linked to the plaintext data is included outside the ciphertext data object.*
- *For schemes claiming 'MAC-then-Encrypt':*
 - *confirm use is exclusively in support of one of the protocols identified in section X of SP 800-38Fr1."*

6. Entrust (Richard Kettlewell), July 8, 2026

Hi,

These are my comments on <https://csrc.nist.gov/pubs/sp/800/38/f/r1/iprd> “PRE-DRAFT Call for Comments: Recommendation for Block Cipher Modes of Operation: Methods for Key Wrapping”.

What ad hoc combinations—approved encryption mode and approved authentication method—are currently implemented, in practice?

In the nShield HSM’s FIPS compliance mode we use the following combinations as part of the functioning of the product:

- AES-256-CBC with HMAC-SHA256 or HMAC-SHA512
- AES-256-CBC with AES-256-CMAC
- AES-256-CTR with HMAC-SHA256 or HMAC-SHA512

In all of these cases, the encryption and MAC mechanisms are separately keyed. The choices here are motivated variously by performance concerns and the functionality available on fixed hardware components.

We offer the following additional combinations to our customers:

- AES-(any size)-CTR with HMAC-(any approved hash). The encryption and MAC mechanisms are separately keyed in this case.
- AES-(any size)-CBC with AES-(any size)-CMAC. The encryption and MAC keys are provided as separate parameters.

Algorithm choices here are generally motivated by specific customer requirements.

Should NIST limit the approval to an explicit set of such combinations that occur within prominent protocols?

NIST should mandate use of secure constructions for key wrapping, meaning:

- Permit fully integrated constructions that inherently enforce authentication (category I)
- Permit all separately keyed encrypt-then-MAC constructions with approved components of adequate strength (i.e. the secure subset of category II)
- Forbid ad-hoc constructions which leave the ciphertext malleable e.g. MAC-then-encrypt or hash-then-encrypt (i.e. the remainder of category II).
- Forbid unambiguously insecure constructions e.g. those with no authentication component at all (category III)

NIST should *not* limit approval of encrypt-then-MAC schemes to an explicit list of combinations. While a review of prominent protocols will, no doubt, capture a large proportion of deployments, it may nevertheless omit niche combinations, forcing a disruptive and expensive transition process on impacted users. We have managed wrapping-related transitions before, both impacting our internal cryptography (which creates compatibility breaks and expensive migration requirements for our customers) and our customers' use cases (which also creates costs for both us and our customers, and can take multiple years to complete).

Unless there are genuine security issues with any particular combination, there is no benefit to removing it from FIPS-validated products, and potentially substantial costs.

ttn/rjk



Richard Kettlewell he/him

Principal Software Security Architect

7. IBM (Richard Kisley), July 9, 2026

Hi,

Please accept these comments:

Updating 38F is a good idea. The areas of concern:

(1) NIST goal: removal of TKW

(2) NIST goal: tightening of Section 3.1 approval of 'ad hoc' combinations of approved encryption and authentication mechanisms, in this sentence: "*Previously approved authenticated-encryption modes—as well as combinations of an approved encryption mode with an approved authentication method—are approved for the protection of cryptographic keys, in addition to general data.*"

(3-IBM) Alignment with other industry keywrap methods.

Taking each of these in turn:

(1) NIST goal: removal of TKW

Industry alignment: There are still many use cases for TDES key wrap that will be need to be maintained for the next decade. Removal of TKW will be one more push in the direction of AES-based key protection, which is a positive for the industry. However, TKW-based unwrapping should still be allowed for legacy use in FIPS 140-3 validated modules.

(2) NIST goal: tightening of Section 3.1 approval of 'ad hoc' combinations of approved encryption and authentication mechanisms.

NIST added 2 questions.

(NIST 2.1) What ad hoc combinations—approved encryption mode and approved authentication method—are currently implemented, in practice?

ISO 20038 presents 2 keywrap methods that are both AES-based and should continue to be allowed by NIST SP800-38F.

Reasons why:

1. The financial services industry has been moving to X9.143 and ISO 20038 over the last 20 years - first to TDES-variant based methods and now AES-based methods - with considerable investment. A NIST denouncement of these methods will throw the industry into chaos and cause hundreds of millions of \$ to be spent on different solutions that improve security only in an academic sense.
2. ISO 20038 Method D (CMAC, CBC encryption) is secure against the attacks noted. Multiple AI analyses (which I agree with) summarizes:
 1. Against the (Bellare, Krawczyk) papers: "**Bottom line:** classified under either paper's taxonomy, this is MAC-then-Encrypt with a strong, independently-keyed MAC and CBC encryption — exactly the case both papers (Krawczyk explicitly, Bellare–Namprempre implicitly) show can be made secure, and the MAC-as-IV twist plus randomized padding closes the specific gap Krawczyk warns is fragile in naive MtE-CBC implementations. It is not susceptible to the named attacks in either paper, conditional on CMAC being sound, the keys being properly independent (which the KDF ensures), and the padding actually being random in practice."

2. Further consideration of the other 2 papers: "So, cryptographically: the X9.143 CBC binding method is designed to avoid these attacks. Operationally: it remains vulnerable to the spirit of Vaudenay/Black-Urtubia if the unwrap implementation leaks anything about unauthenticated decrypted plaintext before the CMAC check succeeds."
3. ISO 20038 Method E (CMAC, CTR encryption) is secure against the attacks noted. Multiple AI analyses summarize:
 1. **"Bottom line:** Not susceptible to the specific counterexample constructions in either paper (weak-encoding, artificially-malleable-cipher, or weak-MAC attacks). But the modification is materially riskier than the CBC original: it converts a birthday-bound collision event with a *contained* consequence into one with a *catastrophic* consequence, and it does so by using a MAC output as a stand-in for the non-repeating counter that Krawczyk's own security proof for stream-cipher AtE says you actually need. If you make this change, the practical mitigation is the same one real SIV designs impose: hard usage limits on KBPK/KBK lifetime well below the 2^{64} birthday bound, enforced by policy, since the algorithm alone (unlike a true incrementing counter) can't guarantee it."
4. Yes - there are issues to mitigate and implementation pitfalls to avoid, which is true for many cryptosystems. With proper implementations, the 20038 constructions are secure.

(NIST 2.2) Should NIST limit the approval to an explicit set of such combinations that occur within prominent protocols?

Qualified 'no'. NIST should limit the approval to crypto systems that mitigate the issues noted in the papers. NIST should survey the techniques reported by industry and note features that strengthen them against the attacks. These techniques should be synthesized into guidance.

(3-IBM) Alignment/collaboration of 38F with financial services industry-standard keywrap methods.

Mis-alignment with certain use cases of financial services has been an issue for a long time. Review of SP800-38F offers a fresh opportunity to come back together and agree on fundamental security of keywrap approaches, healing one of the major issues. SMEs at X9 and ISO would jump at the opportunity to return the favor of NIST and NSA participation over the years to collaborate on an SP800-38F update that takes these into account:

(3.1) recognition of X9.102:2020 AESKW as compliant along the lines of SP800-38F KW, with addition to ACVTS. NIST 140-3 IG (Apr 2025) approves SP800-38F:2012. In the last paragraph of SP800-38F section B.1 it explicitly says that the KW method is equivalent to X9.102 AESKW but the payload format is not, so X9.102-2008 "cannot be compliant". There is no material cryptographic difference between the methods noted. There is a binary compatibility difference in bytes 6 & 7 of the payload:

1. NIST SP800-38F: $ICV = (0xA6 * 8)$
2. X9.102: $ICV = (0xA6 \times 6) || \text{padlen} || \text{hlen}$; where padlen is the bits padding the key to a block size, and hlen is byte length of the associated data hash

While the SP800-38F KW construction is slightly simpler, the binary incompatibility created in 2012, against the comments submitted by industry, means that a CMVP reviewed module cannot be compliant with X9.102, and an X9.102-wrapping module used in financial services cannot be compliant with CMVP.

(3.2) recognition of X9.102:2020 method AKW3, which is also the X9.143 and ISO 20038 method 'D', along with addition to ACVTS. As noted above this is the current transition target of the financial

services industry and recognition of compliance along with helpful implementation guidance would be very valuable.

(3.3) recognition of ISO 20038 method 'E', along with addition to ACVTS. As noted above this is the current transition target (although less so than method 'D') of the financial services industry and recognition of compliance along with helpful implementation guidance would be very valuable.

Richard Kisley

Firmware & Security Architect,
IBM Senior Technical Staff Member, Master Inventor
Payment Card Industry Professional (PCIP)
IBM Cryptographic Technology Development

Dear NIST Cipher Modes Team,

Please find attached my public comment on the pre-draft call for comments for NIST SP 800-38F Revision 1, *Recommendation for Block Cipher Modes of Operation: Methods for Key Wrapping*.

My comments address both of the questions raised in the announcement regarding Section 3.1's approval of ad hoc encryption-and-authentication combinations, and offer six recommendations for the revision, informed by the role AES Key Wrap now plays in standardized post-quantum cryptography (PQC) key-establishment protocols (e.g., RFC 9629, RFC 9936).

Thank you for the opportunity to contribute to this process.

Best regards,

Arpan Sharma
Independent Post-Quantum Cryptography Researcher

Public Comment — PRE-DRAFT Call for Comments: NIST SP 800-38F Revision 1

Recommendation for Block Cipher Modes of Operation: Methods for Key Wrapping

Submitted by: Arpan Sharma
Independent Post-Quantum Cryptography Researcher
[EDITED EMAIL]
Date: 9 July 2026

1. Introduction and Summary of Recommendations

Thank you for the opportunity to provide preliminary feedback on the planned revision of SP 800-38F (2012). My research focuses on the practical engineering of NIST post-quantum cryptography (PQC) standards: performance benchmarking of FIPS 203/204/205 implementations in enterprise Java environments, and analysis of integration constraints in protocols and credential systems. I previously submitted a public comment on the initial public draft of NIST SP 800-133 Rev. 3.

My comments make one overarching point: **Revision 1 will be the version of this Recommendation in force throughout the federal PQC migration, and it should be scoped with that role in mind.** When SP 800-38F was published in 2012, key wrapping was a comparatively specialized key-management technique. In 2026, AES Key Wrap (KW) is the mandatory final step of ML-KEM key establishment in IETF standards, and the keys being wrapped are increasingly multi-kilobyte PQC private keys rather than 128–256-bit symmetric keys. I offer six recommendations:

- **R1.** Limit the Section 3.1 approval to explicitly specified encryption-plus-authentication compositions (responses to both of NIST's questions are given in Section 2).
- **R2.** Recognize and preserve the role of KW/KWP as the key-establishment bridge in standardized PQC protocols, and add guidance on KEM-derived key-encryption keys (KEKs).
- **R3.** Update the robustness-versus-throughput discussion for multi-kilobyte PQC plaintexts, including seed-form wrapping.
- **R4.** Update Appendix A.2 with an explicit mapping from PQC security categories to minimum KEK strength.
- **R5.** Address the associated-data gap: provide guidance on binding key metadata to prevent algorithm/parameter confusion in dual-stack deployments.
- **R6.** Remove TKW with a short migration note, and clarify the revision's relationship to the cryptographic accordion program.

2. Responses to NIST's Questions on Section 3.1 (R1)

Q1 — What ad hoc combinations are currently implemented in practice? Combinations of an approved encryption mode with an approved authentication method that are deployed at scale include:

- **TLS 1.2 and earlier:** AES-CBC with HMAC in MAC-then-encrypt order — the canonical cautionary example. The padding-oracle and Lucky 13 attack lineage arose from composition and implementation details (padding validation, timing of MAC checks), not from weakness in AES or HMAC themselves.
- **JOSE/JWE:** the AES_CBC_HMAC_SHA2 family (e.g., A128CBC-HS256, RFC 7518) — an encrypt-then-MAC composite that is *fully specified*, including key splitting, IV handling, MAC scope, and tag truncation. A useful model of what “specified composition” should mean.
- **PKCS#11 / HSM key export and backup:** application-defined pairings of CKM_AES_CBC (or CBC_PAD) with separate HMAC mechanisms, where padding, IV, and MAC scope are implementation choices.
- **Kerberos:** the aes128/aes256-cts-hmac-sha1-96 encryption types (RFC 3962) and their SHA-2 successors (RFC 8009).
- **IPsec ESP:** AES-CBC confidentiality paired with a separately negotiated HMAC-SHA2 integrity algorithm.
- **SSH:** CBC ciphers paired with HMAC MACs, including encrypt-then-MAC variants whose MAC scope differs from the original construction.

Q2 — Should NIST limit the approval to an explicit set of combinations? **Yes.** The blanket approval treats generic composition as safe by default, while two decades of attacks show that failures concentrate precisely in the composition details left to implementers: MAC order and scope, padding validation, IV handling, and error/timing behavior. If any generic pattern is retained, I recommend restricting it to encrypt-then-MAC with independently derived keys and the MAC computed over the entire ciphertext including the IV; beyond that, approval should

attach to a named, versioned list of fully specified compositions occurring in prominent protocols, with a mechanism for adding entries.

The PQC transition strengthens the case for tightening now. Hybrid and composite schemes and dual-stack credentials are multiplying the algorithm combinations in deployment. Cryptographic agility should mean the ability to swap among fully specified, validated constructions — not the freedom to assemble constructions ad hoc. Revising Section 3.1 prevents the composition-failure class from being carried forward into the post-quantum stack.

3. KW Is Now the “Last Mile” of PQC Key Establishment (R2)

The revision should be informed by how KW is actually used in 2026. RFC 9629 defines the KEMRecipientInfo structure for key-encapsulation mechanisms in the Cryptographic Message Syntax (CMS), and RFC 9936 specifies ML-KEM (FIPS 203) in CMS with AES Key Wrap as the required key-encryption algorithm: implementations supporting ML-KEM-512 must support AES-Wrap-128, and implementations supporting ML-KEM-768 or ML-KEM-1024 must support AES-Wrap-256. KEM-based certificate enrollment and messaging designs follow the same encapsulate → KDF → KW pattern. In short, KW has become the standard bridge between a KEM-derived secret and the protected content-encryption key. I recommend that Revision 1:

- **Retain KW and KWP functionally unchanged** (algorithms, test vectors, CAVP validation, interoperability with the RFC 3394/RFC 5649 profiles). Destabilizing these algorithms mid-migration would ripple through the PQC protocol standards that now hard-depend on them.
- **Add informative guidance on the KEM-derived-KEK usage pattern**, in which the KEK is derived from an ML-KEM shared secret via an approved KDF (SP 800-56C) and used to wrap a single key.
- **Note the security benefit explicitly**: a fresh, single-use KEM-derived KEK also neutralizes the plaintext-equality leakage of deterministic encryption discussed in Appendix A.1, since no two ciphertexts are ever produced under the same KEK.

4. Performance Guidance for Wrapping Multi-Kilobyte PQC Keys (R3)

Section 3.1 of the 2012 publication motivates KW's robustness for cases where “the length of the keys to be protected is greater than the block size” — in 2012, a factor of two to four. For the FIPS 203/204 private keys now being wrapped in practice, the factor is 150–300:

- **Expanded private-key sizes**: ML-KEM-768 decapsulation key = 2,400 bytes (300 semiblocks); ML-KEM-1024 = 3,168 bytes (396 semiblocks); ML-DSA-65 = 4,032 bytes (504 semiblocks); ML-DSA-87 = 4,896 bytes (612 semiblocks). SLH-DSA private keys, at 64–128 bytes, are the exception.
- **Cost model**: the wrapping function W performs $s = 6(n-1)$ block cipher invocations — roughly **12 AES calls per 128-bit block of plaintext, versus one** for counter-mode AEAD. Wrapping a single ML-DSA-87 private key costs approximately 3,672 AES-256 invocations, compared with roughly 306 blocks of AES processing under AES-GCM for the same payload.
- **Operational relevance**: this overhead is immaterial for an individual HSM operation but becomes material at migration scale — KEK rotation across large wrapped-key vaults, key-migration pipelines, and high-volume signing services.

I recommend that the revision (a) update the performance discussion with representative figures for FIPS 203/204/205 key sizes, so implementers can make an informed robustness-versus-throughput decision rather than an uninformed one; and (b) note that FIPS 203 and FIPS 204 both permit private keys to be retained in compact seed form (64 bytes and 32 bytes, respectively), and that wrapping seeds rather than expanded keys restores KW's original performance envelope. Guidance connecting the seed-storage options of those standards to key wrapping does not currently exist anywhere in the NIST portfolio and would be genuinely useful.

5. Map PQC Security Categories to Minimum KEK Strength in Appendix A.2 (R4)

Appendix A.2 correctly states that the KEK should be at least as strong cryptographically as any key it protects, but it predates the NIST PQC security categories. I recommend the revision add an explicit mapping:

- **Category 1** keys (e.g., ML-KEM-512): KEK of at least 128 bits (AES-128).
- **Category 3** keys (e.g., ML-KEM-768, ML-DSA-65): KEK of at least 192 bits; in deployed practice AES-256 (cf. RFC 9936's wrap-algorithm requirements).
- **Category 5** keys (e.g., ML-KEM-1024, ML-DSA-87, SLH-DSA-256 parameter sets): AES-256.

Such a table would codify what deployed practice already assumes, align the Recommendation with CNSA 2.0's requirement of AES-256 for National Security Systems, and close the gap in which a Category 5 post-quantum private key could be wrapped — compliantly but inconsistently — under a 128-bit KEK.

6. The Associated-Data Gap and Algorithm-Confusion Risk (R5)

Appendix B.2 of the 2012 publication already concedes that, unlike GCM and CCM, the key-wrap variants provide no means of authenticating associated data such as “the description of the usage of the protected key.” During the dual-stack transition this gap acquires new significance: the plaintext of a wrapped blob may be an ML-KEM-768 key, an ML-KEM-1024 key, a classical EC key, or a component of a composite scheme, and nothing in KW binds the algorithm identifier, parameter set, or key usage to the ciphertext. Algorithm and parameter-set confusion at unwrap time is a realistic cross-protocol failure mode in mixed classical/PQC estates. I recommend that the revision:

- Provide **informative guidance on cryptographically binding key metadata** to the wrap operation — for example, by including algorithm identifiers and usage context in the KDF context information from which the KEK is derived (as the RFC 9629 structure enables), or by wrapping a self-describing key encoding.
- **Evaluate specifying a key-wrap-with-associated-data variant.** The AESKW mechanism of ANS X9.102 — already reference [2] of the current publication — is an established precedent for wrapping keys together with associated data.

7. TKW Removal, Migration Note, and Coordination with the Accordion Program (R6)

- **I support removing TKW.** TDEA encryption has been disallowed since the end of 2023 under SP 800-131A Rev. 2. I recommend the revision include a short migration note directing operators of legacy TKW/TDEA-wrapped material to fold rewrapping into the cryptographic inventories they already maintain for the PQC migration (e.g., under OMB M-23-02) — one coordinated migration instead of two.
- **Clarify the relationship to cryptographic accordions.** NIST IR 8552 identifies deterministic authenticated encryption for key wrapping as a target derived function of the accordions under development (SP 800-197A). I recommend Revision 1 state explicitly that KW and KWP remain the approved, stable deterministic authenticated-encryption methods for key wrapping for the foreseeable future, and that any future accordion-based key-wrapping function will be introduced with transition guidance. Protocol standards with hard dependencies on KW (Section 3 above) should not be destabilized in the middle of the PQC migration.

Thank you for considering these comments. I would be glad to discuss any of them or to provide supporting measurement data on the wrapping of FIPS 203/204/205 key material in enterprise environments.

Respectfully submitted,

Arpan Sharma

Independent Post-Quantum Cryptography Researcher

[EDITED EMAIL]

References

- [1] NIST SP 800-38F, *Recommendation for Block Cipher Modes of Operation: Methods for Key Wrapping*, December 2012.
- [2] FIPS 203, *Module-Lattice-Based Key-Encapsulation Mechanism Standard*; FIPS 204, *Module-Lattice-Based Digital Signature Standard*; FIPS 205, *Stateless Hash-Based Digital Signature Standard*; NIST, August 2024.
- [3] RFC 9629, *Using Key Encapsulation Mechanism (KEM) Algorithms in the Cryptographic Message Syntax (CMS)*, IETF, August 2024.
- [4] RFC 9936, *Use of ML-KEM in the Cryptographic Message Syntax (CMS)*, IETF.
- [5] RFC 7518, *JSON Web Algorithms (JWA)*, IETF, May 2015 (AES_CBC_HMAC_SHA2).
- [6] RFC 3962 and RFC 8009, *AES encryption types for Kerberos 5*, IETF.
- [7] NIST SP 800-131A Rev. 2, *Transitioning the Use of Cryptographic Algorithms and Key Lengths*, March 2019.
- [8] NIST SP 800-56C Rev. 2, *Recommendation for Key-Derivation Methods in Key-Establishment Schemes*, August 2020.

- [9] NIST IR 8552, *Requirements for Cryptographic Accordions*, 2025; and NIST SP 800-197A (pre-draft call for comments), *NIST Launches Development of Cryptographic Accordions*, June 2025.
- [10] ANS X9.102-2008, *Symmetric Key Cryptography for the Financial Services Industry — Wrapping of Keys and Associated Data*.
- [11] NSA, *Commercial National Security Algorithm Suite 2.0 (CNSA 2.0)*.
- [12] OMB Memorandum M-23-02, *Migrating to Post-Quantum Cryptography*, November 2022.

Dear NIST,

Thank you for the opportunity to provide feedback on revising SP 800-38F, [*Recommendation for Block Cipher Modes of Operation: Methods for Key Wrapping*](#) (2012). This comment is in regard to the following statement from your public announcement: *“The approval in Section 3.1 of unspecified combinations of an approved encryption mode with an approved authentication method should be revisited because such combinations may introduce unexpected security vulnerabilities compared to fully specified methods for authenticated encryption.”*

The current SP 800-38F states in section 3.1: *“Nevertheless, there is no requirement to protect cryptographic keys with a distinct cryptographic method. **Previously approved authenticated-encryption modes**—as well as combinations of an approved encryption mode with an approved authentication method—are approved for the protection of cryptographic keys, in addition to general data.”*

(emphasis mine)

While it may be sensible to limit the combinations of separate encryption and authentication allowed for wrapping, it would be undesirable to further limit the allowed methods within already approved authenticated-encryption (AEAD) modes.

The [*Consultative Committee for Space Data Systems \(CCSDS\)*](#), an international standards group (ISO TC20/SC13) created by civilian space agencies, has published a control-plane standard for managing space data link security in [*CCSDS 355.1-B “Space Data Link Security Protocol—Extended Procedures”*](#). This specification’s procedure for Over-The-Air Rekeying (OTAR) explicitly requires authenticated encryption to protect keys for upload. While 355.1-B does not mandate any specific key wrapping method, its (non-normative) baseline implementation specifies AES-GCM as the method. The rationale for selecting AES-GCM is that it is also a CCSDS-recommended method for protecting space data link RF communications traffic, so implementation of a single algorithm/mode in flight systems can satisfy both needs.

Spacecraft historically have had severe constraints in onboard CPU and memory resources. Requiring the implementation of a separate method to be used exclusively for key wrapping, in addition to implementation of the method used for protecting other communications, would impede the timely development, verification, and flight qualification of spacecraft software and hardware.

Best regards,



Craig Biggerstaff

Mission Control Center Systems Architect

Mission Systems Operations Contract

10. Trusted Computing Group, July 10, 2026

Dear NIST Comment Reviewer,

The Trusted Computing Group (TCG) would like to provide feedback in response to the following questions.

Q1: What ad hoc combinations—approved encryption mode and approved authentication method—are currently implemented, in practice?

The Trusted Platform Module (TPM), standardized by the Trusted Computing Group (TCG), uses AES-CFB for confidentiality and HMAC for integrity and authentication in an encrypt-then-MAC construction. This combination is deployed across TPM 2.0 implementations and is used for key wrapping, for example, to protect TPM objects, TPM contexts, and credentials.

The TPM standard does not support authenticated-encryption schemes, such as AES-KWP, AES-CCM, or AES-GCM, for key wrapping. Adding support for such schemes would require considerable effort and would impact backward compatibility.

Q2: Should NIST limit the approval to an explicit set of such combinations that occur within prominent protocols?

TCG does not take a position on whether NIST should limit approval to an explicit set of approved encryption and authentication combinations.

However, if NIST chooses to approve only an explicit set of such combinations, TCG requests that the AES-CFB + HMAC (encrypt-then-MAC) combination used by TPM 2.0 be included. Excluding this combination would cause the majority of TPM services to become non-approved, despite their standardization and widespread deployment.

Please let us know if you need any further information regarding our comments. Thank you.

Best Regards,

The Trusted Computing Group Board of Directors



11. Department of Energy OCIO (Tasha Barnes), July 10, 2026

Good evening,

Please find attached DOE OCIO feedback on NIST SP 800-38F - Techniques for Key Wrapping.

Respectfully,

Tasha Barnes

IM-1 Executive Secretariat, Communications Specialist
Contractor Supporting the Office of the Chief Information Officer
U.S. Department of Energy

Comment #	Type (General / Editorial / Technical)	Section	Starting Line # *	Comment (include rationale)*	Suggested Change*
1		Should NIST limit the approval to an explicit set of such combinations that occur within prominent protocols?		NIST should consider limiting the approval to an explicit set of combinations that has been verified and tested with prominent protocols. Thus, lowering the potential security risk of unspecified ad hoc combinations.	