ERICSSON

Date: November 20, 2023

Ericsson AB
Group Function Technology
SE-164 80 Stockholm
SWEDEN

Comments on the draft versions of FIPS 203 (ML-KEM), FIPS 204 (ML-DSA), and FIPS 205 (SLH-DSA)

Dear NIST,

Thanks for your continuous efforts to produce well-written open-access security documents. Please find below our comments on FIPS 203 (Draft), FIPS 204 (Draft), and FIPS 205 (Draft).

General comments on all three draft specifications:

– *"secure even against adversaries who possess a quantum computer"*
  "including after the advent of quantum computers"
  "If large-scale quantum computers are realized"
  "resistance to attacks from a large-scale quantum computer"
  "adversaries in possession of a large-scale quantum computer"

  A lot of adversaries already have small, error prone, and currently quite useless quantum computers. "Large-scale" is better but is also not a good term as in addition to being large, the quantum computer must have a sufficiently low error rate to be relevant. We suggest that NIST uses the established and excellent term Cryptographically (or Cryptanalytically) Relevant Quantum Computer (CRQC). This aligns with CNSA 2.0 [1].

– "A number associated with the security strength of a post-quantum cryptographic algorithm"

  We approve NIST sparingly using the term "post-quantum" at all in the draft standards. It is a quite bad term as quantum-resistant algorithms need to be deployed before the advent of CRQCs. We suggest that NIST removes the last few "post-quantum" and replace them with the established and excellent term "quantum-resistant". This aligns with CNSA 2.0 [1].

– "Key search on block cipher with 128-bit key"

  Attacking any sort of 128-bit symmetric cryptography (block cipher or not) requires a drastic number of computational resources comparable to attacking AES-128. As concluded in [2–3], at least cubic (n^3) or quartic (n^4) speedups are required for a practical quantum advantage. The viability of quantum advantage with cubic speedups is still ambiguous [3]. Algorithms with quadratic (n^2) speedup like Grover's algorithm (which is proven to be optimal) will not provide any practical quantum advantage for breaking symmetric cryptography or any other problems.

NIST will soon standardize Ascon-128 [4] which is quantum-resistant but not a block cipher. Stream ciphers like SNOW 3G and sponge-based key derivation and authentication functions like TUAK with 128-bit keys used for protection in 4G and 5G mobile networks are also quantum-resistant. In fact, we would expect that key search on SNOW 3G requires more gates than attacking AES-128. But the exact gate counts are not very important practically. While SNOW 3G and TUAK are not NIST algorithms, we think NIST has an important role in educating the general public that also these types of algorithms are and will remain quantum-resistant.

We suggest that NIST changes the definition of security category 1 to "Key search on cipher with 128-bit key". This aligns with the statement from UK NCSC [5]:

"the security of symmetric cryptography is not significantly impacted by quantum computers, and existing symmetric algorithms with at least 128-bit keys (such as AES) can continue to be used. The security of hash functions such as SHA-256 is also not significantly affected, and secure hash functions can also continue to be used."

We also suggest that NIST provides a statement in some SP or FIPS document estimating for how many decades security category 1 and 128-bit symmetric cryptography will be allowed. The current text in SP 800-57 just states that security strengths 128, 192, and 256 are acceptable beyond 2030. When RSA-1024 was disallowed in 2010 it could almost be broken by the world's fastest supercomputer [6–7]. When RSA-2048 is disallowed in 2030 it is expected to take another 30 years until it can be broken by a supercomputer [6–7]. Using similar security margins (0–30 years), security category 1 should be allowed until 2060–2090. A suggested very conservative statement would be "security category 1 can be used at least until 2060". That would give some needed guidance for industry, but also enable NIST to allow security category 1 to be used longer if Moore's law slows down as many people predict.

- We think it is excellent that ML-KEM and ML-DSA only use SHA-3/Keccak. As stated by Mike Hamburg "SHAKE has a more appropriate interface, comparable or better performance, and is easier to make side-channel resistant" [8]. Hash functions should be designed to provide indifferentiability from a random oracle [9]. Looking at hash performance figures for small output strings it is easy to think that SHA-2 is slightly faster on some platforms, but this is often completely negated by the fact that to use SHA-2 in a secure you need a lot of complex and heavy constructions only designed to overcome the severe shortcomings of SHA-2 such as HMAC, HKDF, MGF1, HASH_DRBG, some function to get short output (NIST defines several ways for SHA-2), and often a mix of SHA-256 and SHA-512. Doing anything secure with SHA-2 is very complex. SHA-2 is not robust.

- We strongly think NIST should produce negative test vectors for all algorithms. Negative test vectors are very important for catching bugs that might have security implications. We think all future algorithm and protocols standards should be accompanied with negative test vectors. It is often claimed that security agencies participate in standardization and production of cryptographic modules with the explicit goal of sabotaging security to enhance their surveillance capabilities. Taking a strong stance on finding security threatening implementation bugs would increase the trust in NIST as a global SDO for cryptography. Two functions that require negative test vectors are ML-KEM.Encaps and ML-KEM.Decaps. FIPS validation shall not be achievable without input validation.

– *"At present, ML-KEM is believed to be secure even against adversaries who possess a quantum computer."*
  "ML-DSA is believed to be secure even against adversaries in possession of a large-scale quantum computer."
  "SLH-DSA is expected to provide resistance to attacks from a large-scale quantum computer."
  "ML-DSA is designed to be strongly existentially unforgeable"

  We suggest removing "At present", which is not suitable for a document that will live for many years. The terms believed to, expected to, and designed to are all used when talking about security properties. We suggest removing "believed". A huge amount cryptanalytic effort targeting lattice-based cryptography has been done before and during the NIST PQC project and US government is planning to protect all national security systems using lattice-based cryptography. Maybe only use the term "designed to", which seems to be the most common in FIPS 203—205 and other NIST specifications.

– "1.3 Differences From the … Submission"

  These sections are probably better suited as appendixes in the final standards.

Comments on FIPS 203 (Draft):

– We strongly disagree with suggestions that NIST should remove ML-KEM-512 based on a heavily contested claim regarding the gate count in one theoretical memory model [10]. We agree with NIST that the cost of breaking ML-KEM-512 is higher than the cost to break AES-128. We think that it is excellent that NIST has specified ML-KEM-512. If ML-KEM-512 is slightly above or below the theoretical security level of AES-128 in one theoretical model is practically completely irrelevant. The important thing practically is that ML-KEM-512 is approximately as hard to break as AES-128. We believe ML-KEM-512 offer a significant security margin for many applications, especially if used in hybrid mode with Curve25519. As stated by UK NCSC [5], ML-KEM-512 provides an acceptable level of security for personal, enterprise, and government information.

  The maximum transmission unit (MTU) on the Internet is typically just around 1300 bytes. The encapsulation key and ciphertext are 800 and 768 bytes in ML-KEM-512 versus 1184 and 1088 bytes in ML-KEM-768. The size difference means that when using ML-KEM-512, a lot more additional information can be sent in the same packet. This significantly reduces latency, which is very important in many applications. We believe that the availability of ML-KEM-512 will increase the adoption rate of quantum-resistant cryptography. We believe most implementations will support all of the security levels so applications should be able to change the security level quickly if needed.

- We are strongly against replacing SHA-3/Keccak in ML-KEM with SHA-2 as suggested in [8]. Such a big change would risk introducing various kinds of security problems, decrease trust in ML-KEM and NIST, lower performance on most/all platforms, and significantly delay deployment of ML-KEM. Using Keccak should not be a problem for organizations that have invested in cryptographic agility.

- "makes the encapsulation key available to Party B. Party B then uses Party A's encapsulation key to generate one copy of a shared secret key along with an associated ciphertext. Party B then sends the ciphertext to Party A over the same channel"

  It does not have to be the same channel. It is quite common that a different channel is used.

- "used by two parties to establish a shared secret key over a public channel"
  "A shared secret key is computed jointly by two parties (e.g., Party A and Party B)"
  "randomness used by the two parties"

  ML-KEM is also very useful for quantum-resistant protection of data at rest using for example Hybrid Public Key Encryption (HPKE) [11]. In such use cases the party encapsulating and decapsulation may be one and the same, i.e., there is only one party. We think this should be mentioned in the specification.

- "As a result, ML-KEM is believed to satisfy so-called IND-CCA security"

  We think it should be described that the encapsulation key can be used several times. That this follows from the IND-CCA security is likely not obvious to most readers. We think this information should be mentioned in FIPS 203 and not just in the future SP 800-227.

- We think it would be good if FIPS 204 also discusses additional security properties. E.g., is ML-KEM believed to have key commitment or not? How does reusing the encapsulation key affect the security bounds. Can anything be said about multi-key security?

- "The scheme K-PKE is not sufficiently secure"

  We suggest that NIST explains is some detail why NIST believes that K-PKE is not sufficiently secure.

Comments on FIPS 204 (Draft):

- Very good that NIST embraced the suggestion to include hedged signatures [12] and made it the default mode. We believe that making this mode the default will increase the practical security in deployed systems.

- Rejection sampling is new to most users of digital signatures. FIPS 203 has an excellent table showing decapsulation failure rate. We think FIPS 204 should have a similar table showing the probability for one or more rejections in the signing algorithm. This is not trivial for most readers to calculate. Users will want to know if the variable signing time is something they need to care about or if the probabilities are so low that the variable signing time can be ignored.

– "ML-DSA is designed to be strongly existentially unforgeable under chosen message attack"

  We suggest also adding the abbreviation SUF-CMA, i.e., "ML-DSA is designed to be strongly existentially unforgeable under chosen message attack (SUF-CMA)". This allows the reader to search for "SUF-CMA" or "CMA".

– "ML-DSA is also designed to satisfy additional security properties beyond unforgeability, which are described in [6]"

  We suggest that FIPS 204 lists the additional security properties that ML-DSA is designed to satisfy. The current text does not say if ML-DSA satisfies all or a subset of the properties, and the paper [13] analyzes a non-standardized version of ML-DSA. It is not clear to most readers if the analysis is still valid for ML-DSA.

Comments on FIPS 205 (Draft):

– *"The 12 parameter sets included in Table 1 were designed to meet certain security strength categories defined by NIST in its original Call for Proposals [21] with respect to existential unforgeability under chosen message attack (EUF-CMA)".*

  We think this is a good selection of parameters. Sections 10.1, 10.2, and 10.3 provide a clear illustration of how much easier it is to work with SHAKE instead of SHA2. The SHA-2 versions are downright inelegant and complexity like this often leads to specification and implementation bugs. We think NIST should also mention if SLH-DSA is (believed to be) SUF-CMA or not. i.e., given a number of different message-signature pairs (m_i, σ) can an attacker create a new signature (m_i, σ') for an already signed message m_i.

– For ML-DSA, hedged signatures are the default, the value rnd should be generated by an approved RBG, and the deterministic mode should not be used on platforms where side-channel attacks are a concern. For SLH-DSA, hedged signatures are not the default, opt_rand does not require use of an approved RBG, and for devices that are vulnerable to side-channel attacks opt_rand may be set to a random value. We suggest that NIST aligns the SLH-DSA specification with use the stronger ML-DSA requirements alternatively explain why it is acceptable for SLH-DSA to have much weaker requirements.

– We think FIPS 205 should describe how the security depends on the number of times the SLH-DSA private key is used to generate signatures. We think it would be helpful for the reader to understand if there are no practical limits for the number of signatures that can be generated or if systems producing a very large number of signatures should change the SLH-DSA private key periodically to keep a high security level. In ECDSA the collision probability can be ignored while AES-256-GCM with r random IVs only provide $\approx 97 - \log_2 r$ bits of security due to the collision probability [14].

- “finding such a collision would be expected to require fewer computational resources than specified for the parameter sets‘ claimed security levels in all cases except SLH-DSA-SHA2-128f and SLH-DSA-SHAKE-128f.”

  We suggest that FIPS 205 describes how much fewer resources would be needed. An application might require different security levels for different properties. It would also be good if NIST stated that it is unknown if SLH-DSA provides the properties exclusive ownership and non re-signability [13].

- “Don‘t support component use.”
  ”cryptographic modules should not make interfaces to these components available to applications“

  We would suggest that this is softened or rewritten. That some hardware implementations do not support important building blocks like the AES round function and the KECCAK-p permutation has turned out to be very limiting for innovation, significantly decreasing performance (or security) of future standards like ML-KEM and meaning that the acceleration cannot be used for algorithms like AEGIS [15], Rocca-S [16], Snow 5G [17], and Simpira [18] that make use the AES round function. We think it is very important that many types of hardware implementations do support component use to enable future innovation and standards. In general, we strongly think that NIST should encourage hardware implementations such as CPUs to have flexible APIs supporting component use.

Best Regards,
John Preuß Mattsson,
Expert Cryptographic Algorithms and Security Protocols

[1] NSA, "Announcing the Commercial National Security Algorithm Suite 2.0"
https://media.defense.gov/2022/Sep/07/2003071834/-1/-1/0/CSA_CNSA_2.0_ALGORITHMS_.PDF

[2] Hoefler, Häner, Troyer, "Disentangling Hype from Practicality: On Realistically Achieving Quantum Advantage"
https://cacm.acm.org/magazines/2023/5/272276-disentangling-hype-from-practicality-on-realistically-achieving-quantum-advantage/fulltext

[3] Babbush, McClean, Newman, Gidney, Boixo , Neven, "Focus beyond Quadratic Speedups for Error-Corrected Quantum Advantage"
https://arxiv.org/pdf/2011.04149.pdf

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

[5] UK NCSC, "Next steps in preparing for post-quantum cryptography"
https://www.ncsc.gov.uk/whitepaper/next-steps-preparing-for-post-quantum-cryptography

[6] CRYPTEC, "Cryptographic Technology Evaluation Committee Activity Report"
https://www.cryptrec.go.jp/symposium/2023_cryptrec-eval.pdf

[7] CRYPTEC, "Japan CRYPTREC Activities on PQC"
https://events.btq.li/Japan_CRYPTREC_Activities_on_PQC_Shiho_Moriai.pdf

[8] NIST PQC Forum, "Comments on FIPS 203/204"
https://groups.google.com/a/list.nist.gov/g/pqc-forum/c/SPTpYEP7vRg

[9] Maurer, Renner, Holenstein, "Indifferentiability, Impossibility Results on Reductions, and Applications to the Random Oracle Methodology"
https://eprint.iacr.org/2003/161

[10] NIST PQC Forum, "Kyber security level?"
https://groups.google.com/a/list.nist.gov/g/pqc-forum/c/W2VOzy0wz_E

[11] IETF RFC 9180, "Hybrid Public Key Encryption"
https://www.rfc-editor.org/rfc/rfc9180.html

[12] Preuß Mattsson, "OFFICIAL COMMENT: CRYSTALS-Dilithium"
https://groups.google.com/a/list.nist.gov/g/pqc-forum/c/1mQjngj_2Po/m/-p4RKXGQAwAJ

[13] Cremers, Düzlü, Fiedler, Fischlin, Janson, "BUFFing signature schemes beyond unforgeability and the case of post-quantum signatures"
https://eprint.iacr.org/2020/1525.pdf

[14] Preuß Mattsson, Smeets, Thormarker, "Proposals for Standardization of Encryption Schemes"
https://csrc.nist.gov/csrc/media/Events/2023/third-workshop-on-block-cipher-modes-of-operation/documents/accepted-papers/Proposals%20for%20Standardization%20of%20Encryption%20Schemes%20Final.pdf

[15] IRTF, "The AEGIS Family of Authenticated Encryption Algorithms"
https://datatracker.ietf.org/doc/draft-irtf-cfrg-aegis-aead/

[16] IRTF, "Encryption algorithm Rocca-S"
https://datatracker.ietf.org/doc/draft-nakano-rocca-s/

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

[18] Gueron, Mouha, "Simpira v2: A Family of Efficient Permutations Using the AES Round Function"
https://eprint.iacr.org/2016/122