# Comment on Post-Quantum Cryptography Requirements and Evaluation Criteria

David Jao <djao@math.uwaterloo.ca>

Thu 9/15/2016 10:57 PM

To: pqc-comments <pqc-comments@nist.gov>;

To whom it may concern:

My name is David Jao. I am the designer of post-quantum cryptosystems based on the isogeny problem over supersingular elliptic curves.

I would like to draw your attention to the following areas in your post-quantum cryptography draft requirements and evaluation criteria which I believe would benefit from further clarification:

(Section 2.B.1) In prior NIST standardization processes, there was only one functionality being evaluated (e.g. block ciphers for AES, and hash functions for SHA3). In this draft, we have potentially up to three distinct pieces of functionality (encryption, signatures, and key exchange) being evaluated at the same time. Will NIST be evaluating all the algorithms in a single submission package together, or will the three types of schemes be evaluated separately? If the latter, why not just accept separate submissions in each category rather than combining schemes of each type into one submission? It would be helpful if NIST could clarify the rationale behind accepting multiple items in one submission. For example: "If a submission includes more than one type of scheme, NIST will evaluate the schemes of each type separately. However, submitters may choose to combine different types of schemes into a single submission in order to share software code among multiple schemes within the submission."

(Section 2.C.1) Does the requirement for ANSI C source code preclude the use of assembly language optimizations? Your draft proposal does not specifically address this question. I would like to see assembly optimizations (at least inline ASM) allowed for the optimized implementation, because otherwise the implementation would not be representative of real-world conditions, especially for number-theoretic cryptography which relatively speaking benefits more from assembly optimization than other families of cryptosystems. It seems to me to be a little inconsistent to specify a target platform (Intel x64) and not allow platform-specific optimizations.

(Section 4.A.2) IND-CCA2 makes perfect sense for public-key encryption, as well as key transport, but does not apply to key establishment in isolation. It is not clear what security model NIST is proposing for key establishment. All existing security models for key establishment that I'm aware of are rather heavyweight, and the vast majority are tailored to authenticated key exchange, which you mention only in Section 4.C.1. As I am not an expert in security models for key establishment, I defer to others on the question of what model to use. If NIST requires

external assistance in this regard then a public request for input would be appropriate.

(Section 4.A.4) Typically, it is not possible to tune the classical and quantum security levels of a scheme separately; a given choice of parameters will imply a fixed classical security level and a fixed (possibly different, but not independently tunable) quantum security level. For example, any isogeny-based scheme with 128-bit classical security automatically has 80 bits quantum security; therefore security level number 1 in this section is superfluous for isogenies, as any such parameter choice automatically satisfies security level number 2. It would be helpful to have explicit guidance on what to do in such situations. I suggest adding an explicit guideline to ignore such inapplicable security levels.

(Section 4.A.4) In your FAQ (http://csrc.nist.gov/groups/ST/post-quantum-crypto/faq.html) you state that "quantum security should be defined as the minimum possible value of log(depth*(squareroot (space))) PLUS A CONSTANT" (emphasis added). The phrase "plus a constant" in my interpretation allows for some fudge factor (note the sign of the constant is not restricted to being positive!), so that something which (for example) strictly speaking might provide only 125 bits of security could be considered to provide 128 bits. Unfortunately, this phrasing does not appear in the PDF of your actual draft proposal. Instead, the draft proposal uses the phrasing "meet or exceed" which is less flexible. For number-theoretic cryptography, crossing a machine-level word size boundary incurs a huge performance penalty, and for this reason it is extremely common to use parameters which meet a given security level only up to the addition of a small constant (e.g. Curve25519 provides only 125-bit security). Therefore I would like to ask that the draft proposal be amended to include the "plus a constant" phrasing.

(Section 4.B) This section lists cost considerations which apply specifically to public-key cryptosystems and signature size, but does not list any cost considerations which apply specifically to key exchange. I would suggest that some attempt be made to specify some cost considerations for key exchange protocols, or else explicitly request comments on this topic from the public. Examples of cost considerations specific to key exchange include the number of rounds of communication, the number of static keys and ephemeral keys required, and whether or not the protocol supports (or alternatively requires) synchronous and/or asynchronous communication.