From: John Mattsson <john.mattsson@ericsson.com> via pqc-forum <pqc-forum@list.nist.gov>
To: pqc-forum@list.nist.gov
Subject: Re: [pqc-forum] "90s" version parameter sets
Date: Thursday, July 07, 2022 03:47:48 AM ET

Hi,

As it is now clear that NIST is recommending CRYSTALS-Kyber and CRYSTALS–Dilithium as the primary algorithms it might be good to revive this thread.

The current CRYSTALS specifications use the following algorithms:

Kyber: SHAKE128, SHAKE256, SHA2-256, SHA3-512

Kyber 90s: AES-256, SHA-256, SHA-512, SHAKE256

Dilithium: SHAKE256, SHAKE128

Dilithium 90s: SHAKE256, AES-256

Kyber 90s looks very messy with four different primitives and will likely lead to more implementations with side-channel vulnerabilities. I would strongly prefer if NIST only standardized CRYSTALS with Keccak. That would hasten general support for hardware acceleration of Keccak. The best would have been if NIST said clearly 5 years ago that PQC will only use Keccak. That many CPUs today have acceleration of SHA-1 but not SHA-3 is just tragic and should not be rewarded.

The only reason to standardize the 90s versions would be short-term speed improvements before Keccak hardware acceleration is generally available. I am not sure that extra speed is necessary. Kyber and Dilithium already have very good performance and the performance will improve when Keccak hardware acceleration is generally available. Specifying 90s versions would delay general availability of Keccak hardware acceleration and reward vendors that are stuck in the 90s.

I also strongly think NIST should go ahead and specify the AEAD mode of Keccak that we were promised. For vendors wanting crypto-agility and NIST compliance, the lack of an NIST approved Keccak-based AEAD mode is problematic. A lot of constrained hardware and software implementations would like to implement a single NIST approved cryptographic

primitive and use that for CCA-encryption, CPA-encryption, variable-length hash, variable-length MAC, KDF, etc. Currently that is not possible.

Cheers,

John Prueß Mattsson

From: Ruben Niederhagen <ruben@polycephaly.org> via pqc-forum@list.nist.gov
To: pqc-forum@list.nist.gov
Subject: Re: [pqc-forum] "90s" version parameter sets
Date: Thursday, July 07, 2022 04:09:35 AM ET

On 07/07/2022 15:47, 'John Mattsson' via pqc-forum wrote:
> [...] I would strongly prefer if NIST only standardized CRYSTALS
> with Keccak. [...]
>
> The only reason to standardize the 90s versions would be short-term
> speed improvements before Keccak hardware acceleration is generally
> available. [...]

Keccak has a relatively large state which might be an issue for embedded devices with strong resource restrictions. This also applies to the size of hardware accelerators.

Hence, some diversity in the hash-function choice might be desirable for some applications.

Ruben

From: John Mattsson <john.mattsson@ericsson.com> via pqc-forum <pqc-forum@list.nist.gov>
To: Ruben Niederhagen <ruben@polycephaly.org>, pqc-forum@list.nist.gov
Subject: Re: [pqc-forum] "90s" version parameter sets
Date: Thursday, July 07, 2022 04:55:15 AM ET

I agree but the solution is probably not to force implementation of all the four primitives AES-256, SHA-256, SHA-512, and SHAKE256. The future LWC ”winner” might be a good candidate for this.

Cheers,

John

Get Outlook for iOS

From: pqc-forum@list.nist.gov <pqc-forum@list.nist.gov> on behalf of Ruben Niederhagen <ruben@polycephaly.org>
Sent: Thursday, July 7, 2022 10:09:01 AM
To: pqc-forum@list.nist.gov <pqc-forum@list.nist.gov>
Subject: Re: [pqc-forum] "90s" version parameter sets

On 07/07/2022 15:47, 'John Mattsson' via pqc-forum wrote:
> [...] I would strongly prefer if NIST only standardized CRYSTALS
> with Keccak. [...]
>
> The only reason to standardize the 90s versions would be short-term
> speed improvements before Keccak hardware acceleration is generally
> available. [...]
Keccak has a relatively large state which might be an issue for embedded devices with strong resource restrictions. This also applies to the size of hardware accelerators.

Hence, some diversity in the hash-function choice might be desirable for some applications.

Ruben

--