From: John Mattsson <john.mattsson@ericsson.com> via pqc-forum <pqc-forum@list.nist.gov>
To: pqc-forum@list.nist.gov
Subject: [pqc-forum] OFFICIAL COMMENT: CRYSTALS-Kyber
Date: Saturday, July 09, 2022 02:41:41 AM ET

NIST should carefully consider which versions of Kyber to standardize as well as the algorithm choices in the standardized versions of Kyber.

- The Kyber specification makes use of no less then 4 different SHA-3 functions (SHAKE128, SHAKE256, SHA2-256, SHA3-512). While the four SHA-3 function can be implemented with a single Keccak API

SHAKE128(M,d) = Keccak[256](M||1111, d)

SHAKE256(M,d) = Keccak[512](M||1111, d)

SHA3-256(M) = Keccac[512](M||01, 256)

SHA3-512(M) = Keccac[1024](M||01, 256)

it looks a bit strange to use four different functions. The fixed-length SHA-3 hash functions (drop-in for SHA-2) have to our knowledge seen little or no practical use. Instead the variable-length SHAKE functions have seen significant practical use in implementations as well as in published and upcoming standards such as EdDSA (RFC 8032), XMSS (RFC 8391), LMS (NIST SP 800-208), CMS (RFC 8702), RSASSA-PSS and ECDSA (FIPS 186-5 (Draft), RFC 8692), COSE (draft-ietf-cose-hash-algs), EDHOC (draft-ietf-lake-edhoc), CPace (draft-irtf-cfrg-cpace), FROST (draft-irtf-cfrg-frost), OPRF (draft-irtf-cfrg-voprf), CRYSTALS-Dilithium, Falcon, and SPHINCS+. Most hash function suggested after SHA-3 such as KangarooTwelve and BLAKE3 are only specified as variable-length. We think NIST should specify CRYSTALS-Kyber using only SHAKE (we assume that the extreme 512-bit preimage resistance of SHA3-512 is not needed for security and that SHA3-512 was chosen just because it had the right output length).

- We think NIST should standardize Kyber512, Kyber768, Kyber1024. This aligns well with the current levels for P-256, P-384, and P-521 where P-256 is for general usage, P-384 is for high security such as CNSA, and P-521 is good to have if there would be any surprising attacks on ECC cryptography.

- We do not think NIST should specify the 90s versions of Kyber. Having 6 different versions are three versions too many and would decrease interoperability. Standardizing the 90s

version would likely lead to more implementations with side-channel vulnerabilities, delay severely needed general availability of SHAKE hardware acceleration in CPUs, and reward vendors that are stuck in the 90s.

Cheers,

John Preuß Mattsson

From: John Mattsson <john.mattsson@ericsson.com> via pqc-forum <pqc-forum@list.nist.gov>
To: Peter Schwabe <peter@cryptojedi.org>
CC: pqc-forum@list.nist.gov
Subject: Re: [pqc-forum] OFFICIAL COMMENT: CRYSTALS-Kyber
Date: Saturday, July 09, 2022 07:24:47 AM ET

Hi Peter,

Thanks for the quick answer. I see now that the design choice of using four different SHA-3 function because of domain separation is clearly explained in the Kyber specification. Sorry for missing that. I don't have a strong opinion. Maybe using four different SHA-3 functions is the best tradeoff. But requiring four different functions like (SHAKE128, SHAKE256, SHA2-256, SHA3-512) or (AES-256, SHA-256, SHA-512, SHAKE256) might not work for future hash functions like the "winner" of the NSIT LWC project. That problem could however be solved with explicit domain separation at a later time.

Cheers,

John

On 2022-07-09, 11:06, "Peter Schwabe" <peter@cryptojedi.org> wrote:

'John Mattsson' via pqc-forum <pqc-forum@list.nist.gov> wrote:

Dear John, dear all,

Just speaking for myself.

> NIST should carefully consider which versions of Kyber to standardize

> as well as the algorithm choices in the standardized versions of

> Kyber.

>

> - The Kyber specification makes use of no less then 4 different SHA-3

> functions (SHAKE128, SHAKE256, SHA2-256, SHA3-512). While the four

> SHA-3 function can be implemented with a single Keccak API

>

> SHAKE128(M,d) = Keccak[256](M||1111, d)

> SHAKE256(M,d) = Keccak[512](M||1111, d)

> SHA3-256(M) = Keccac[512](M||01, 256)

> SHA3-512(M) = Keccac[1024](M||01, 256)

>

> it looks a bit strange to use four different functions. The

> fixed-length SHA-3 hash functions (drop-in for SHA-2) have to our

> knowledge seen little or no practical use. Instead the variable-length

> SHAKE functions have seen significant practical use in implementations

> as well as in published and upcoming standards such as EdDSA (RFC

> 8032), XMSS (RFC 8391), LMS (NIST SP 800-208), CMS (RFC 8702),

> RSASSA-PSS and ECDSA (FIPS 186-5 (Draft), RFC 8692), COSE

> (draft-ietf-cose-hash-algs), EDHOC (draft-ietf-lake-edhoc), CPace

> (draft-irtf-cfrg-cpace), FROST (draft-irtf-cfrg-frost), OPRF

> (draft-irtf-cfrg-voprf), CRYSTALS-Dilithium, Falcon, and SPHINCS+.

> Most hash function suggested after SHA-3 such as KangarooTwelve and

> BLAKE3 are only specified as variable-length. We think NIST should

> specify CRYSTALS-Kyber using only SHAKE (we assume that the extreme

> 512-bit preimage resistance of SHA3-512 is not needed for security and

> that SHA3-512 was chosen just because it had the right output length).

I don't have a very strong opinion about this, but the reason we use

different functions from the Keccak family is that we can rely on their

internal domain separation. Implementing everything with just SHAKE, would require some explicit domain separation. That can of course be done and in a pure software implementation of Kyber only, this is not a problem. However, when you want to implement Kyber using some existing hash API, you need two calls to the update function (one to absorb the domain separation and one to absorb the actual input) or you have to copy domain separation and input to a new buffer. If you have a non-incremental hash API, the former is not an option.

Also, when using explicit domain separation, there is a question if you want to simply concatenate the domain-separation byte and the input, or first pad domain separation to a full block. Padding would increase the number of Keccak permutations, which already now cost most of the CPU cycles of Kyber.

All the best,

Peter

From: Kelsey, John M. (Fed) <john.kelsey@nist.gov> via pqc-forum <pqc-forum@list.nist.gov>
To: John Mattsson <john.mattsson@ericsson.com>, Peter Schwabe <peter@cryptojedi.org>
CC: pqc-forum <pqc-forum@list.nist.gov>
Subject: Re: [pqc-forum] OFFICIAL COMMENT: CRYSTALS-Kyber
Date: Thursday, July 21, 2022 05:34:40 PM ET

Just as an aside, you could use cSHAKE for this, which has a domain separation string (“customization string”) as a built-in component. That’s specified in SP 800-185.

--John

> On 7/9/22, 07:24, "'John Mattsson' via pqc-forum" <pqc-forum@list.nist.gov> wrote:
>
> Hi Peter,
>
> Thanks for the quick answer. I see now that the design choice of using four different SHA-3 function because of domain separation is clearly explained in the Kyber specification. Sorry for missing that. I don't have a strong opinion. Maybe using four different SHA-3 functions is the best tradeoff. But requiring four different functions like (SHAKE128, SHAKE256, SHA2-256, SHA3-512) or (AES-256, SHA-256, SHA-512, SHAKE256) might not work for future hash functions like the "winner" of the NSIT LWC project. That problem could however be solved with explicit domain separation at a later time.
>
> Cheers,
>
> John
>
> On 2022-07-09, 11:06, "Peter Schwabe" <peter@cryptojedi.org> wrote:
>
> 'John Mattsson' via pqc-forum <pqc-forum@list.nist.gov> wrote:
>
> Dear John, dear all,
>
> Just speaking for myself.
>
> > NIST should carefully consider which versions of Kyber to standardize
>
> > as well as the algorithm choices in the standardized versions of
>
> > Kyber.

>

> - The Kyber specification makes use of no less then 4 different SHA-3

> functions (SHAKE128, SHAKE256, SHA2-256, SHA3-512). While the four

> SHA-3 function can be implemented with a single Keccak API

>

> SHAKE128(M,d) = Keccak[256](M||1111, d)

> SHAKE256(M,d) = Keccak[512](M||1111, d)

> SHA3-256(M) = Keccac[512](M||01, 256)

> SHA3-512(M) = Keccac[1024](M||01, 256)

>

> it looks a bit strange to use four different functions. The

> fixed-length SHA-3 hash functions (drop-in for SHA-2) have to our

> knowledge seen little or no practical use. Instead the variable-length

> SHAKE functions have seen significant practical use in implementations

> as well as in published and upcoming standards such as EdDSA (RFC

> 8032), XMSS (RFC 8391), LMS (NIST SP 800-208), CMS (RFC 8702),

> RSASSA-PSS and ECDSA (FIPS 186-5 (Draft), RFC 8692), COSE

> (draft-ietf-cose-hash-algs), EDHOC (draft-ietf-lake-edhoc), CPace

> (draft-irtf-cfrg-cpace), FROST (draft-irtf-cfrg-frost), OPRF

> (draft-irtf-cfrg-voprf), CRYSTALS-Dilithium, Falcon, and SPHINCS+.

> Most hash function suggested after SHA-3 such as KangarooTwelve and

> BLAKE3 are only specified as variable-length. We think NIST should

> specify CRYSTALS-Kyber using only SHAKE (we assume that the extreme

> 512-bit preimage resistance of SHA3-512 is not needed for security and

> that SHA3-512 was chosen just because it had the right output length).

I don't have a very strong opinion about this, but the reason we use different functions from the Keccak family is that we can rely on their internal domain separation. Implementing everything with just SHAKE, would require some explicit domain separation. That can of course be done and in a pure software implementation of Kyber only, this is not a problem. However, when you want to implement Kyber using some existing hash API, you need two calls to the update function (one to absorb the domain separation and one to absorb the actual input) or you have to copy domain separation and input to a new buffer. If you have a non-incremental hash API, the former is not an option.

Also, when using explicit domain separation, there is a question if you want to simply concatenate the domain-separation byte and the input, or first pad domain separation to a full block. Padding would increase the number of Keccak permutations, which already now cost most of the CPU cycles of Kyber.

All the best,

Peter

--

You received this message because you are subscribed to the Google Groups "pqc-forum" group.
To unsubscribe from this group and stop receiving emails from it, send an email to pqc-forum+unsubscribe@list.nist.gov.
To view this discussion on the web visit https://groups.google.com/a/list.nist.gov/d/msgid/pqc-

forum/HE1PR0701MB3050A538123C9BE696521D0489859%40HE1PR0701MB3050.eurprd07.prod.outlook.com.

From: Kelsey, John M. (Fed) <john.kelsey@nist.gov> via pqc-forum <pqc-forum@list.nist.gov>
To: Peter Schwabe <peter@cryptojedi.org>
CC: John Mattsson <john.mattsson@ericsson.com>, Peter Schwabe <peter@cryptojedi.org>, pqc-forum <pqc-forum@list.nist.gov>
Subject: Re: [pqc-forum] OFFICIAL COMMENT: CRYSTALS-Kyber
Date: Wednesday, July 27, 2022 12:42:29 PM ET

That can be precomputed for a given name/customization string. I'm not sure it that would work in your Kyber implementation, but in general, if you know the customization string, you can precompute the resulting starting value for the rest of the SHAKE computation. So if you are in a position where you are using a fixed set of customization strings for each of the five hashes you're calling, you can just precompute the starting states for their customization strings once, and then use them as many times as you need to.

--John

> On 7/22/22, 11:35, "Peter Schwabe" <peter@cryptojedi.org> wrote:
>
> "'Kelsey, John M. (Fed)' via pqc-forum" <pqc-forum@list.nist.gov> wrote:
>
> Dear John,
>
> > Just as an aside, you could use cSHAKE for this, which has a domain
>
> > separation string ("customization string") as a built-in component.
>
> > That's specified in SP 800-185.
>
> Yes, indeed. That would be the standardized solution, but it would be
>
> quite a bit more costly, because domain separation takes an additional
>
> Keccak permutation.
>
> All the best,
>
> Peter

From: Markku-Juhani O. Saarinen <mjos.crypto@gmail.com> via pqc-forum@list.nist.gov
To: Kelsey, John M. (Fed) <john.kelsey@nist.gov>
CC: Peter Schwabe <peter@cryptojedi.org>, John Mattsson <john.mattsson@ericsson.com>, pqc-forum <pqc-forum@list.nist.gov>
Subject: Re: [pqc-forum] OFFICIAL COMMENT: CRYSTALS-Kyber
Date: Wednesday, July 27, 2022 02:05:05 PM ET

On Wed, Jul 27, 2022 at 5:42 PM 'Kelsey, John M. (Fed)' via pqc-forum <pqc-forum@list.nist.gov> wrote:

> That can be precomputed for a given name/customization string. I'm not sure it that would work in your Kyber implementation, but in general, if you know the customization string, you can precompute the resulting starting value for the rest of the SHAKE computation. So if you are in a position where you are using a fixed set of customization strings for each of the five hashes you're calling, you can just precompute the starting states for their customization strings once, and then use them as many times as you need to.

Hi John,

I recall that some earlier-round versions of PQC Lattice algorithms had a running variable (index) as the customization string, which created the two-permutation requirement. This hit their performance hard (as those were *not* fixed strings, as intended for cSHAKE) so it was changed.

For Kyber and Dilithium, the XOF input in the prominent ExpandA phase is a tuple of a "seed" and one or two running indices "i"/"j". The indices allow parallelization of the ExpandA step. Thinking about it, this might look more naturally like a NIST SP 800-185 TupleHashXOF than a cSHAKE use case. But in practice, one just wants to have an efficient, unambiguous domain-separating padding for these inputs -- and only one permutation to generate a block of XOF output.

Current high-end CPU architectures benefit from Keccak parallelization a lot. And as a future consideration, enabling parallelization with appropriate modes shortens the overall "critical path" of the PQC algorithm and thereby also lowers the absolute lower bound on its execution latency. It has been pointed out (thanks, Bas Westerbaan!) that the requirement in Kyber encapsulation op to hash the (ephemeral) public key creates a surprisingly big parallelization bottleneck, so perhaps one wants to enable parallelization there too.

From a hardware design perspective, I need to say that while a cSHAKE "customization string" may work reasonably well in CPUs, it will significantly hit either size or performance of Keccak hardware. The 1600-bit state of Keccak is very large, and having multiple copies of it quickly available is awkward (as kind of a register mux, 1600 bits is enormous). Loading the permutation state over a bus to the internal state can take as long as running the 24-round permutation itself. If any kind of rapid initialization is provided for the state, it is probably zeroization, as that allows short inputs to be rapidly hashed (a very common use case). So an efficient (short!) padding scheme is preferred to resetting the entire 1600-bit state to some random constant.

By the way, I think most cryptographers agree with the suggestion from the Keccak team (Gilles Van Assche, July 8, "Reduced-round Keccak for PQ schemes") that a 12-round Keccak is sufficient for essentially any security level. This saves perhaps 25+% of software power & latency for Kyber and Dilithium (and almost 50% for SP 800-208 and Sphincs+).
Cheers,

- markku

Dr. Markku-Juhani O. Saarinen <mjos@iki.fi>

> --John
>
> > On 7/22/22, 11:35, "Peter Schwabe" <peter@cryptojedi.org> wrote:
> >
> > "'Kelsey, John M. (Fed)' via pqc-forum" <pqc-forum@list.nist.gov> wrote:
> >
> > Dear John,
> >
> > > Just as an aside, you could use cSHAKE for this, which has a domain
> >
> > > separation string (“customization string”) as a built-in component.
> >
> > > That’s specified in SP 800-185.
> >
> > Yes, indeed. That would be the standardized solution, but it would be
> >
> > quite a bit more costly, because domain separation takes an additional
> >
> > Keccak permutation.
> >
> > All the best,

> Peter
>
> --
> You received this message because you are subscribed to the Google Groups "pqc-forum" group.
> To unsubscribe from this group and stop receiving emails from it, send an email to pqc-forum+unsubscribe@list.nist.gov.
> To view this discussion on the web visit https://groups.google.com/a/list.nist.gov/d/msgid/pqc-forum/SA1PR09MB7629DF87E3D4E7931511E69D8A979%40SA1PR09MB7629.namprd09.prod.outlook.com.

--
You received this message because you are subscribed to the Google Groups "pqc-forum" group.
To unsubscribe from this group and stop receiving emails from it, send an email to pqc-forum+unsubscribe@list.nist.gov.
To view this discussion on the web visit https://groups.google.com/a/list.nist.gov/d/msgid/pqc-forum/CA%2BiU_q%3DXz536VHU342mq7NGsdre-1WW-E_SJZLmV2pzfo%2BQN4A%40mail.gmail.com.