Hello Crystals team.

I was wondering if it’s possible to use smaller parameter sets to decrease the signature size, instead of relying on aggressive compression techniques used in the current form of Dilithium.

Specifically, if we were to preserve the $\mathbb{Z}_q[X]/(X^{256}+1)$ module and matrix row and column counts in the current specification, and decrease ‘q’ and acceptance threshold of signature variable ‘z’, and avoid public-key compression, would we be able to achieve comparable bandwidth efficiency and security? Had the team done some calculation on this?

Thanks.
Dear all,

Dustin and the NIST team pointed out to us on Friday that one of our internal hash functions, which converts the message to a digest (line 10 in Figure 4), is hashing to a 384-bit value. This was large enough for NIST level 3 security (which was our maximum security level in round 2), but would need to be 512 for NIST level 5 security. We forgot to make this change and we thank the NIST team for pointing it out. We have since updated our code and spec at

https://github.com/pq-crystals/dilithium
and
https://pq-crystals.org/dilithium/resources.shtml

Since this is an internal variable, it has no impact on the parameter sizes. The runtime differences are also negligible. Since this change does affect the KATs, we took it as an opportunity to make a few additional small changes that we were holding off on. To harmonize the randomness expansion function in the key generation and signing, we are now using SHAKE-256 with 512-bit secret seeds for both. We also reduced the output size of the public key hash (tr on line 7) to 256 bits. And we simplified the pseudo-code throughout the spec by merging the names of H and CRH to just H (since they both use SHAKE-256 with varying output sizes).

Best,

Vadim
(On behalf of the Dilithium team).