From: 'Patrick Longa' via pqc-forum <pqc-forum@list.nist.gov>
Sent: Friday, December 11, 2020 12:24 PM
To: Markku-Juhani O. Saarinen; pqc-forum
Subject: RE: [pqc-forum] ROUND 3 OFFICIAL COMMENT: FrodoKEM -- CCA Bug

Hi Markku,
Thanks for reporting this bug.

We have replaced the offending line of code in our GitHub repository (https://github.com/microsoft/PQCrypto-LWEKE):

```
r = (-(int16_t)r) >> (8*sizeof(uint16_t)-1);
```

By this one:

```
r = (-(int16_t)(r >> 1) | -(int16_t)(r & 1)) >> (8*sizeof(uint16_t)-1);
```

Cheers,
Patrick (on behalf of the FrodoKEM team)

From: pqc-forum@list.nist.gov <pqc-forum@list.nist.gov> **On Behalf Of** Markku-Juhani O. Saarinen
Sent: Thursday, December 10, 2020 7:11 AM
To: pqc-forum <pqc-forum@list.nist.gov>
Subject: [pqc-forum] ROUND 3 OFFICIAL COMMENT: FrodoKEM -- CCA Bug

Hello,

It looks like the FrodoKEM team also fixed the timing oracle [GJN20] badly and caused a more serious security problem while trying to do that. This seems to affect FrodoKEM-976 and FrodoKEM-1344, but not FrodoKEM-640. I'd be happy to be wrong about this, but otoh I think it's better to let people know directly rather than sit on it while these things are in production somewhere.

The code rarely rejects bad decryptions so this reduces these KEMs to CPA security, allowing direct adaptive attacks. Note that you don't need a timing oracle to exploit this vulnerability.

In Round3 submitted code, "util.c" (there are 9 equivalent instances of this file in the submission -- for Reference, Optimized, and Additional):

```
111:int8_t ct_verify(const uint16_t *a, const uint16_t *b, size_t len)
112:{ // Compare two arrays in constant time.
113:  // Returns 0 if the byte arrays are equal, -1 otherwise.
114:    uint16_t r = 0;
115:
116:    for (size_t i = 0; i < len; i++) {
117:        r |= a[i] ^ b[i];
118:    }
119:
120:    r = (-(int16_t)r) >> (8*sizeof(uint16_t)-1);
```