From:	pqc-forum@list.nist.gov on behalf of Markku-Juhani O. Saarinen <mjos.crypto@gmail.com>
Sent:	Thursday, December 10, 2020 10:11 AM
To:	pqc-forum
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);
121:    return (int8_t)r;
122:}
```

Diagnosis:

Note that the inputs are 16-bit vectors; these have a full range of values in FrodoKEM-976 and FrodoKEM-1344 which have Q = 2**16. FrodoKEM-640 wtih Q = 2**15 has the high bit set to zero in input vectors, and this function behaves as claimed.

A 16-bit type is also used for comparisons. If even a single word pair has a difference in the high bit, the high bit of r is 1, the negation operation makes it into a positive integer, and the function returns 0 -- as if the arrays were equivalent. Example failures: a=0x0000,b=0x8000 -- returns 1 (illegal value), a=0x0000, b=0x8001, returns 0 even though a != b.

See usage in FO implementation In file "kem.c"

```
197:    int8_t selector = ct_verify(Bp, BBp, PARAMS_N*PARAMS_NBAR) | ct_verify(C, CC,
PARAMS_NBAR*PARAMS_NBAR);
198:    // If (selector == 0) then load k' to do ss = F(ct || k'), else if (selector == -1) load s to do ss = F(ct || s)
199:    ct_select((uint8_t*)Fin_k, (uint8_t*)kprime, (uint8_t*)sk_s, CRYPTO_BYTES, selector);
```

A decryption failure is very likely to be ignored by this rejection mechanism because the selector will be 0 with a high probability even in case of a mismatch, the failed decryption will be used and returned to the caller, just like in SIKE.

This flaw seems to be also present in some derived, current codebases:

https://github.com/microsoft/PQCrypto-LWEKE/blob/6a82d4d2b6bc32eeac3ad4f9c178c48072b506b3/src/util.c#L111
https://github.com/PQClean/PQClean/blob/c380c628bc41df0a617992c24618f027b36a61b7/crypto_kem/frodokem1344shake/clean/util.c#L232

I'd assume that the key recovery attacks of [GJN20] are even easier to mount with this new, more powerful oracle.

[GJN20] Qian Guo, Thomas Johansson, Alexander Nilsson. "A key-recovery timing attack on post-quantum primitives using the Fujisaki-Okamoto transformation and its application on FrodoKEM." https://eprint.iacr.org/2020/743 In IACR-CRYPTO 2020.

Sincerely,
- markku

Dr. Markku-Juhani O. Saarinen <mjos@pqshield.com> PQShield, Oxford UK.

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);
```