CAA - Lab 03

Fault Attack

15 juin 2021

Student
Doran KAYOUMI

Teacher
Alexandre DUC
Assistant
Nathan SÉVILLE

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

Table des matières

1 Fault Attack

1.1 Attack

For the attack, I used `AES` with **1** and **2** rounds and a plaintext with **32 zeros**.

When executing AES with 1 round and the plaintext, we obtain the **first half** of the key. This is due to the fact that with 1 round will just do an AddRoundKey (i.e. a xor) with the plaintext and the first half of the key. And in this case, we're xoring the first half of the key with zeros, which outputs the first half of the key.

Now with 2 rounds things get slightly trickier. First thing to do, is to cipher the plaintext (with 2 rounds). Next, we cipher the plaintext once again, but we "stop" it right before the last AddRoundKey, which gives us an intermediate state. And now, we can simply xor the ciphertext and the intermediate state to get the second half of the key.

Finally, we can concatenate both halves, and we have the full key.

Here is my secret key :
a166d9f1868c2f597a665bbf76688226f97e37decff685931ae254d6e995da8a

1.2 Code

Here is the code I used to perform the attack :

```python
def break_aes():
    key_first_half = process_key('a166d9f1868c2f597a665bbf76688226', Nk=4)
    two_round_res = process_block('ff381120a0402c268426e75ed1fce3f4')
    block = process_block('00000000000000000000000000000000')

    w = KeyExpansion(key_first_half, Nk=4, Nr=1)
    Nr = 2
    Nb = 4

    state = AddRoundKey(block, w[:Nb])
    state = SubBytes(state)
    state = ShiftRows(state)
    state = MixColumns(state)

    key_second_half = AddRoundKey(state, two_round_res)

    print("Secret Key: {}{}".format(str_block_line(
        key_first_half), str_block_line(key_second_half)))
```

1.3 Smartcard Attack

To perform this attack on a smartcard, we can use a laser to heat up the **counter value**, so we can reduce the number of rounds to **1**, then **2** and finally stop it before the last instruction with 2 rounds.

1.4 Possible fixes

A fix would be to use a combination of different **masking** methods for all the functions used in the AES rounds. **Boolean masking** for the linear functions (ShiftRows, MixColumns and AddRoundKey) and **Multiplicative masking** for the non-linear functions (SubBytes).

By looking at the code, is that the final round is only performed when doing 14 rounds. A fix to the attack, would be to always do the final round. (i.e. remove the if Nr != 14)