Hardware Security EL9423
LAB 3

Nimisha Limaye (nsl278)
Ashik H. Poojari (ap4613)

March 25, 2017

Abstract

In this lab, we propose to leak the round key from Data Encryption Standard (DES) by means of differential fault analysis technique. The fault is introduced in the right register, after 15^{th} round. The algorithm is obtained from [3] and the DES encryption code from [1] is modified to contain the fault analysis. We leak the individual 6 bit keys given to each SBOX by running multiple plaintexts, to obtain maximum number of keys satisfying the algorithm. After performing analysis, we found that, on an average 8 plaintexts were needed to leak the round key.

1. Differential Fault Analysis

The algorithm for Differential Fault Analysis (DFA) - attack on 16^{th} round is obtained from **??**. The attacker records correct and faulty ciphertext. In order to generate faulty ciphertext, the attacker introduces faults during the computation of the cryptographic algorithm, typically towards the end of its execution (e.g. before the final round). In the case of hardware solutions, this is typically done by altering the environmental conditions of the device. For example, voltage glitching or laser fault injection could be used to introduce faults.

2. DFA on DES

2.1 Leaking the round key

Step 1: In order to recover the last round key (K16) using a DFA, the attacker injects faults after the execution of round 15, in right register, by flipping one or more bits. These flipped bits go through expansion process and key XORing, and finally affect one or more SBOXes. Using the correct and erroneous ciphertexts, we can write the following equations:

$$\triangle R_{16} = f_{K_{16}}(L_{16}) \oplus f_{K_{16}}(L'_{16}) \tag{2.1}$$

where L_{16} is the left portion of correct ciphertext and L‘$_{16}$ is the left portion of erroneous ciphertext[3]. $\triangle R_{16}$ denotes the error difference between right portion of correct and erroneous ciphertexts.

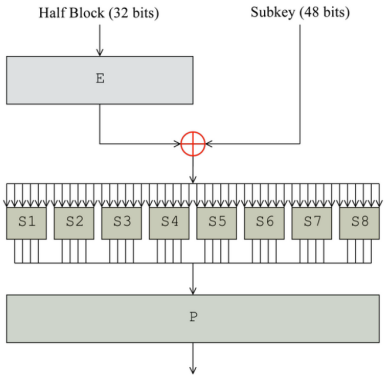


Figure 1: Round Operations of DES encryption

Step 2:
From the Feistel function in fig. 1, we can see that DES uses 6 round key bits to compute 4 output bits in each round, making it possible to solve this equation in chunks of 6 bits of the round key. In particular, for each individual S-Box the following equation needs to be satisfied for correct 6 bit key:

$$P^{-1}(R_{16}^{'} \oplus R_{16})_i = S_i(E(R_{15}) \oplus K_{16i}) \oplus S_i(E(R_{15}^{'}) \oplus K_{16i}) \tag{2.2}$$

Where E and P represent the expansion and permutation steps of the Feistel function, respectively. This equation can be easily solved by exhaustive search of 2^6.
Typically, this results in a number of candidates for each affected sub-key for each fault, since the relationship between SBOX input and output is many to one. Therefore, an attacker needs to iterate this process until only one candidate remains for each key [2].

3. Results

We use a counting method instead of discarding key candidates, since in some cases, when the faults are not injected in the exact way as expected by the attack, it is possible to discard a correct key. For each SBOX, we compute a set of 6 bit key candidates satisfying Eq. 2.2 and increment the counter for the respective candidates. Once all faults are analyzed, the candidate with the highest count for each sub-key is selected as the correct 6 bit key. So in fig. 3a, you can see that value 10 has the highest number of count. Hence it is normalized to one. The key with highest normalized value is selected as the key for that i^{th} SBOX.
The algorithm was implemented in C language and the result is as shown in fig. 2, which shows the reference round key using key schedule function, the individual 6 bit round keys corresponding to each SBOX and the final round key obtained by combining the individual 6 bit round keys expressed in bytes.

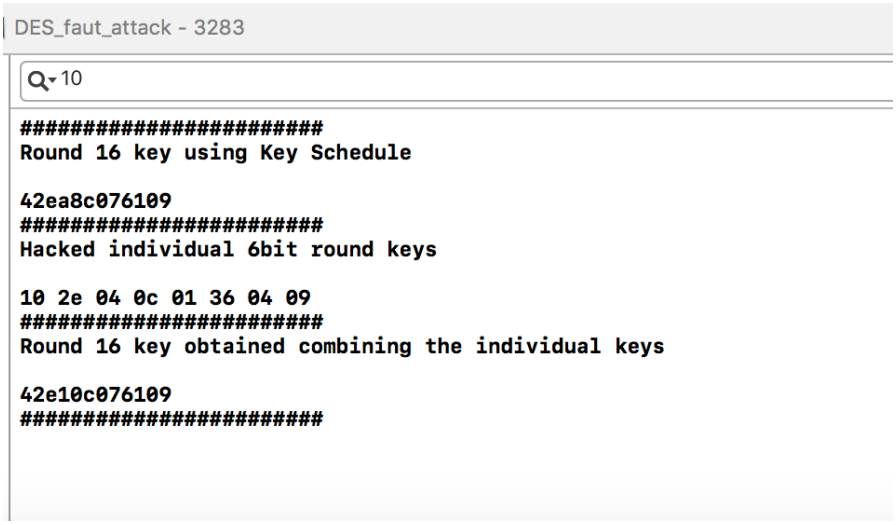
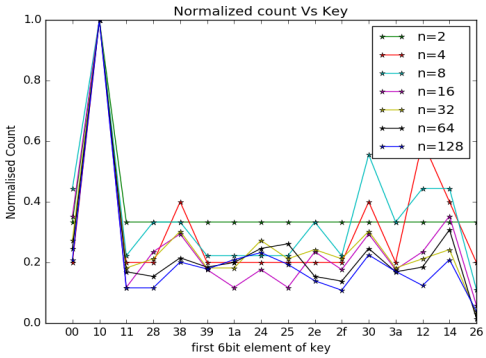
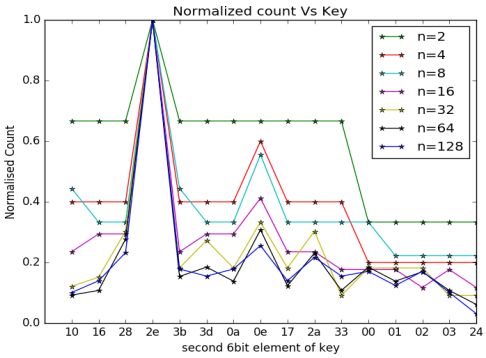


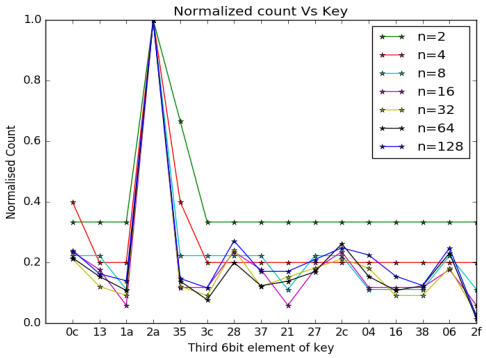
Figure 2: Hacked 48 bit round key generated by DFA



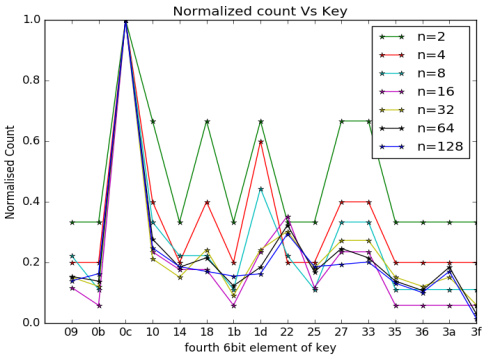
(a) Key value '10' has the highest value



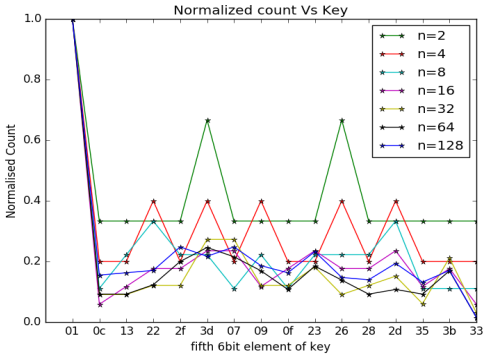
(b) Key value '2e' has the highest value



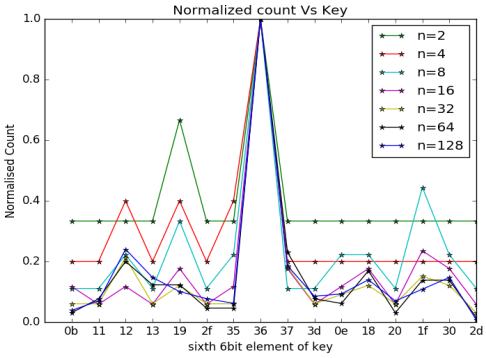
(c) Key value '2a' has the highest value



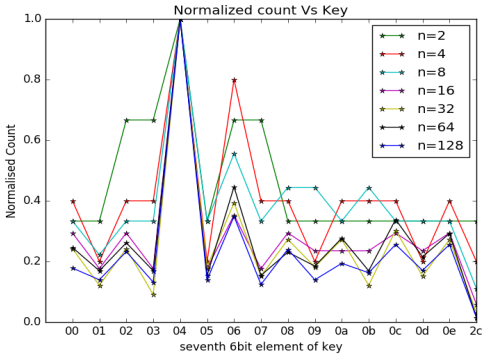
(d) Key value '0c' has the highest value



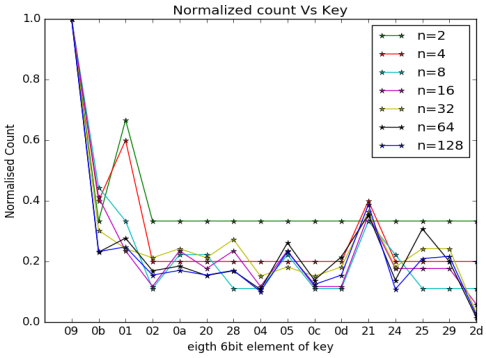
(e) Key value '01' has the highest value



(f) Key value '36' has the highest value



(g) Key value '04' has the highest value



(h) Key value '09' has the highest value

Figure 3: Normalized Count for individual 6 bit keys of sample 48 bit round key. Here the 6 bit key with the highest normalized score will be selected among the other keys. n is the number of plaintexts.

4. Conclusion

The round key was leaked using differential fault analysis by attacking on 16^{th} round. On an average, it took us 8 plaintexts to leak the round key.

References

[1] URL: https://github.com/B-Con/crypto-algorithms.

[2] URL: https://www.blackhat.com/docs/eu-15/materials/eu-15-Sanfelix-Unboxing-The-White-Box-Practical-Attacks-Against-Obfuscated-Ciphers-wp.pdf.

[3] Joye,Marc Tunstall, Michael. *Fault Analysis in Cryptography*. Springer Heidelberg New York Dordrecht London, 2012.