HW2 writeup

109550127 宋哲頤

AES

首先我們可以從講義的解題步驟得知 DPA Workflow 的作法

1. Choosing an intermediate value
2. Measuring the power consumptions
3. Calculating hypothetical intermediate values
4. Mapping intermediate values to hypothetical power consumption (Power model)
5. Comparing the hypothetical consumption with real power consumption (Statistics test)
6. The 16 bytes are independent before **MixColumns** in the first round

因為我們要解開 key，因此從明文加密開始，從題目提供的 power traces 可以知道有 D = 50 個 record(等於有 50 筆 plaintext)每個 record 有 T = 1086 個 point。

題目為 AES128 加密，所以要分成 16bytes 來一個一個 bytes 分析，每次取出 D 個 record 的第 i　(i = 0 ~ 15) 個 byte 組合並與假設的 0~255 的 key xor 後丟入 sbox，得到　D * K (K = 256)大小的矩陣，並以 power model(HW)的方式以二進位後幾個 1 當作 weight 並和 D * T 大小的 trace 計算 correlation，得出 K * T 大小的所有相關係數的矩陣，我用 np.argmax()找出 correlation 最大值的 index 並除以 1806 得到他在 0~255 中的哪一個 bytes，找到後就知道在這個 bytes 位置加密的 key 是多少了，這方式重複 16 次便能得到 flag=FLAG{18MbH9oEnbXHyHTR}。

```
with open('stm32f0_aes.json') as f:
    data = json.load(f)
    keys=[]
    for i in range (16):
        corr_2d=np.ndarray((256,1806),dtype=float)
        plain_first_byte=[]
        for dic in data:
            plain_first_byte.append(dic["pt"][i])
        for k in range(256):
            print("round %d  %d" % (i, k))
            for t in range(1806): # trace
                x_list=[]
                y_list=[]
                for d in range(len(plain_first_byte)):
                    x_list.append(hamming_weight(sbox[(plain_first_byte[d]^k)%256]))
                    y_list.append(data[d]["pm"][t])
                corr_2d[k][t]=correlCo(x_list, y_list)
        idx=np.argmax(corr_2d)
        idx= idx //1806
        keys.append(idx)

    print(bytes(keys))
```