CS4150: CPA - Correlation Power Analysis

Bas van't Spijker
1497944

Yuhang Tian
5219728

May 20, 2021

1 Model Description

The schematic diagram has been shown in fig. 1 below which represents the leakage model of power consumption of the S-box. The XOR result from the sub plaintext and subkey after being mapped by an S-box, when it is stored into the memory, will fillip the previous bits in the memory, which will cause the dynamic power change. The power change can be captured by the chip whisperer and the filliping can be roughly simulated by Hamming weight. Thus, the model is a Hamming weight power consumption leakage model.

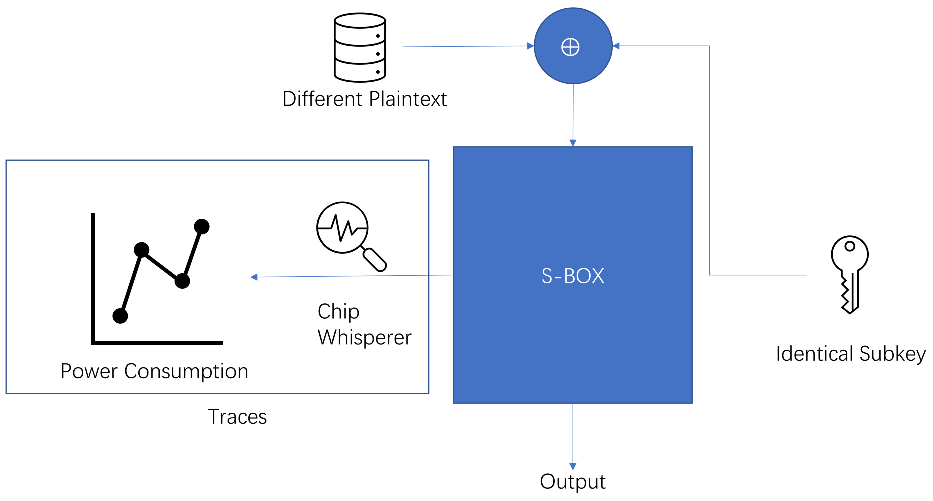


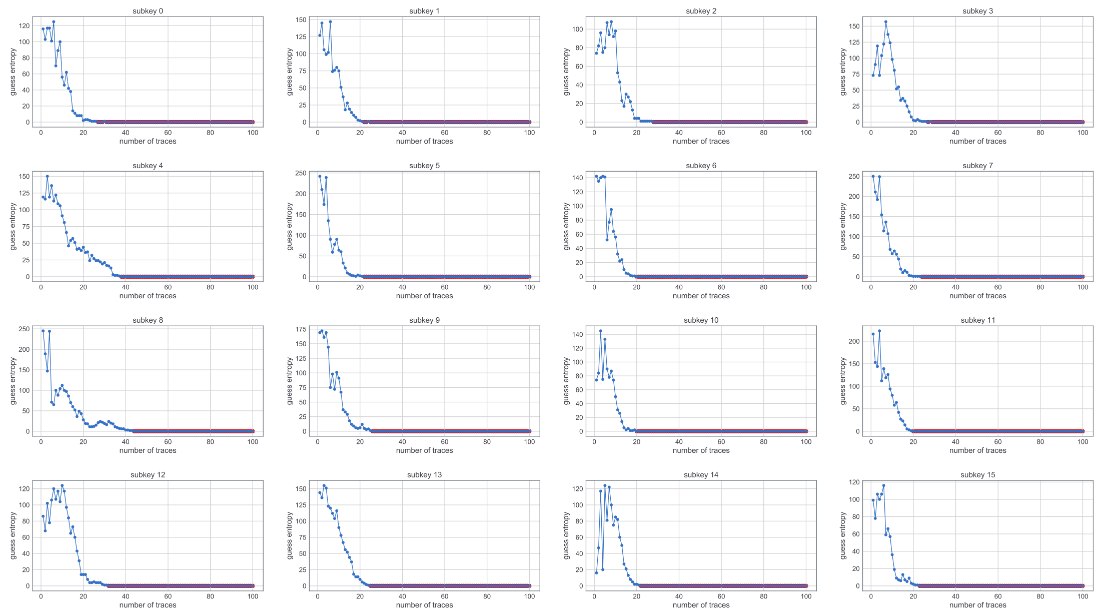
Figure 1: CPA Leakage Model Schematic Diagram

As the subkey is identical, if different and adequate plaintext is sequentially fed in, from the measurement, we could obtain a bunch of features/points of interest for this subkey. These features describe how this subkey together with various plaintext affect the bits in the memory and this kind of effect is nonidentical for different subkey. From another perspective, if we got enough features, we could brute force the subkey (256 possibles) to find a matching one with our measurements, then the subkey was cracked. If we could measure this information from a 128-AES process, the key of the AES could be found in a lower complexity, since each subkey could be broken individually.

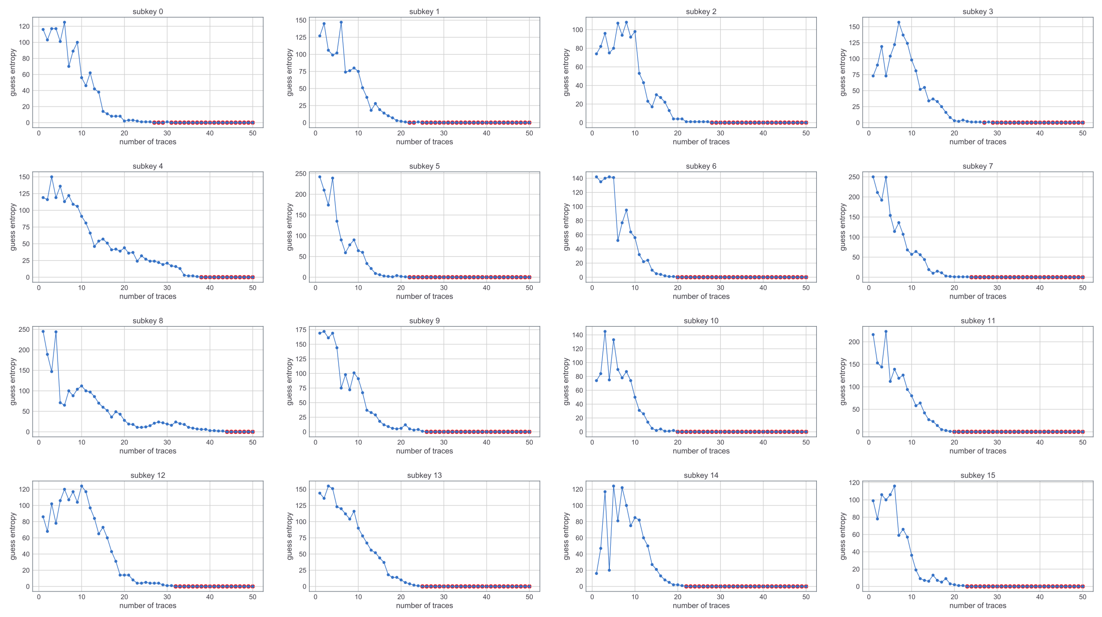
The key is to define the relationship between features (power consumption) and hamming weight (filliping bits). We use the Pearson Correlation coefficient in eq. 1 which measures the linear dependency between two variables, x, and y.

$$Pearson(x,y) = \frac{\Sigma_{i=1}^{N}((x_i-\overline{x})(y_i-\overline{y}))}{\sqrt{\Sigma_{i=1}^{N}(h_{d,i}-\overline{x})^2}\sqrt{\Sigma_{i=1}^{N}(y_i-\overline{y})^2}} \tag{1}$$

2 CPA



(a) Number of Traces Needed: 1 to 100



(b) Number of Traces Needed: 1 to 50 (Zoom In Version)

Figure 2: CPA: Traces vs. Guess Entropy

In order to find how many traces actually we needed to find each subkey, we fed in traces starting from the number of 1 to 100. The repetition of the experiment is 10 and for each time we use a different part from the traces' set. The results of all subkeys are shown in fig. 2. Both fig. 2(a) and fig. 2(b) reveal the trend of guess entropy of each subkey when the number of traces increases and the only difference is the latter is the zoomed-in version of the former.

In the beginning, there are some fluctuations because the input number of traces is so limited that the prediction almost like a random guess leading to unstable guessing results. After traces become sufficient, the guessing entropy keeps decreasing with the increasing number of traces and finally approaches zero. The subkeys which are firstly guessed correctly are subkey[6], subkey[10], and subkey[11], whereas the slowest one is subkey[8]. Therefore, in general, it needs a total of 44 traces so as to find all correct subkeys. It can be more clearly observed from fig. 3 where the overall performance is shown.

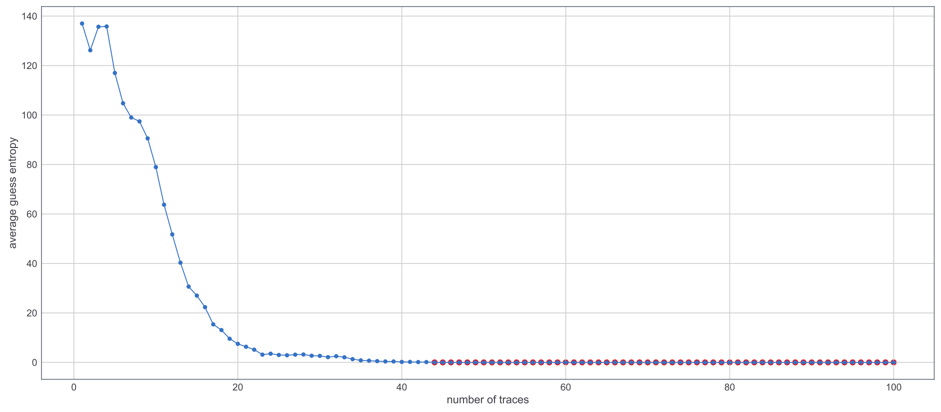


Figure 3: General Performance of Guessing

3 Online CPA

The method shown in the previous section has a great disadvantage, for instance, when the number of traces increases from 50 to 100, the model needs to restart from 0 to 100. This section will introduce online CPA which allows the newly added traces to be directly appended in and continue from the previous result.

In eq. 1, the main barriers for the new sample put in are $\overline{x}$ and $\overline{y}$ since they need to be pre-calculated for certain x_is and y_is. However, after opening the brackets and simplification, they can be removed and the equation becomes eq. 3.

$$Pearson(h,t) = \frac{\Sigma_{d=1}^{D}((h_{d,i}-\overline{h_i})(t_{d,j}-\overline{t_j}))}{\sqrt{\Sigma_{d=1}^{D}(h_{d,i}-\overline{h_i})^2}\sqrt{\Sigma_{d=1}^{D}(t_{d,j}-\overline{t_j})^2}} \tag{2}$$

$$Pearson_{online}(h,t) = \frac{D\Sigma_{d=1}^{D}h_{d,i}t_{d,j} - \Sigma_{d=1}^{D}h_{d,i}\Sigma_{d=1}^{D}t_{d,j}}{\sqrt{((\Sigma_{d=1}^{D}h_{d,i})^2 - D\Sigma_{d=1}^{D}h_{d,i}^2)((\Sigma_{d=1}^{D}t_{d,j})^2 - D\Sigma_{d=1}^{D}t_{d,j}^2)}} \tag{3}$$

In the submission code, online_cpa takes three arguments:

- nsubkey: the number of guessing subkeys, ranging from 1 to 16.
- startpos: the starting position of the input traces.
- numtraces: the number of input traces.

We gave two examples, the first one is finding the first subkey by inputting traces one by one while the second one is finding all subkeys by inputting traces separately. Online CPA can tackle each subkey more efficiently and find the number of traces needed easier.

Project Website: CS4150-SystemSecurity