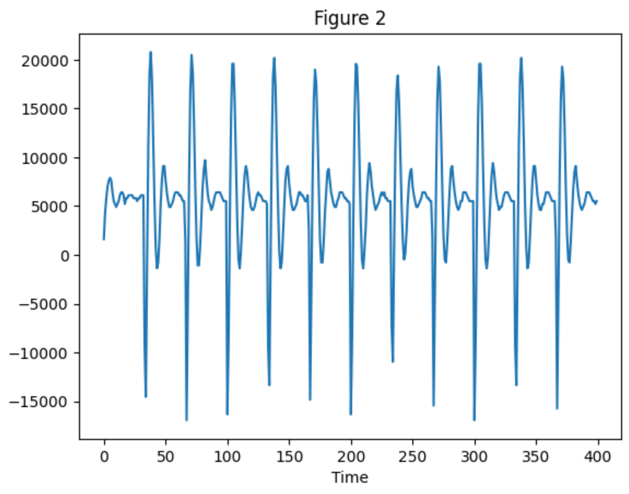
2 Problem 1: Analyze the power trace

Figure 2 Plot the first power trace in the power_traces.csv file and analyze it.
Question 1: What do the peaks in the power trace correspond to? Why are there 11 power peaks and what is happening at each power peak?



The peaks correspond to a heavy, high-power operation. Each power peak probably corresponds to a round key addition operation, since 11 round key addition operations happen each time AES-128 encrypts/decrypts.

3 Problem 2: Perform the DPA attack

Let's execute the DPA attack.
Question 2:What is an easy target operation for DPA? Describe how many bits of the key your DPA attack estimate at a time. Hint: You are given the input, think of the very first operation of AES.
Apply DPA on the first byte of the key, plot correlation results **Figure 3** for all 256 key guesses.
Find the maximum correlation among key guesses (consider both positive and negative peaks) and plot the two best key guesses, on the same figure,

An easy target operation for DPA is the first round key since it is simply an XOR of the plaintext against the key.

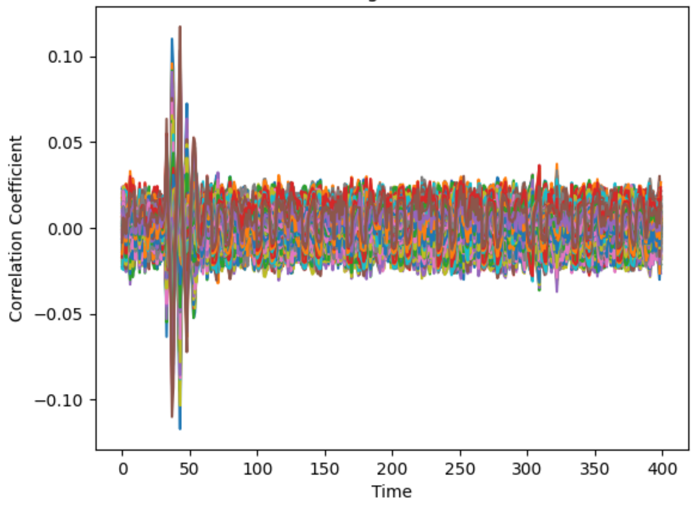


Figure 3

Key guesses 0 and 255 have the highest correlation.

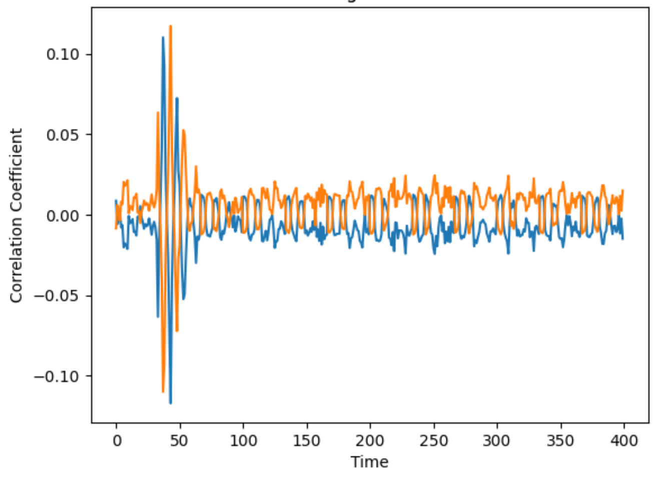


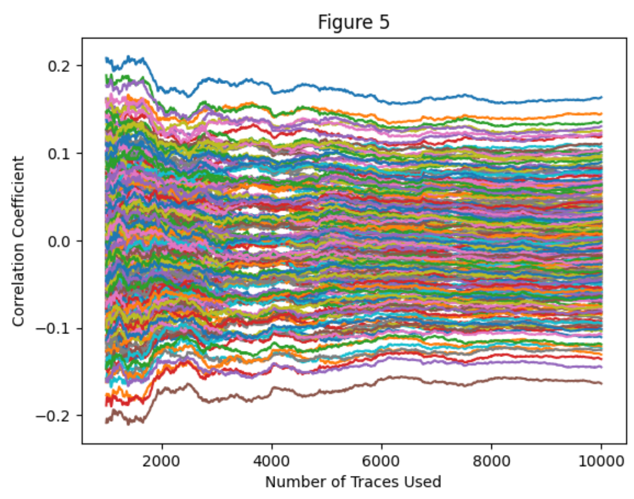
Figure 4

Question 3: Which one is the correct key guess? Why do you see these two results?

0x00 is the correct key guess. 0xFF has a high negative correlation because it is the exact opposite of the correct key, so its hamming distance guesses are negatively correlated with the peak power.

Question 4: Which point in the time domain has the maximum power leak (ie. correlation)? Plot the "evolution" of correlation coefficient for all 256 key guesses for 10000 measurements at this particular time instance **Figure 5**.

The point that has the maximum power leak is the first peak, around index ~48.



Question 5: Starting at how many traces, does the correct key guess show highest correlation?

The correct key guess always showed the highest correlation from the very beginning, but it stabilizes around 3000.

Question 6: Apply DPA on the entire key, write the value of the 128-bit AES key.

0x000102030405060708090a0b0c0d0e0f

4 Problem 3: Analyze the DPA attack

Question 7: What is the mean time to disclosure, ie. the number of traces required to extract the key?

16 bytes, 256 traces per byte -> 16 * 256 = 4096 traces total

Question 8: Compare this with the theoretical crypto-analysis of AES, what is the reduction ratio in the number of traces required to break AES? What is the reason for this reduction?

AES bruteforce: 2^128
DPA: 4096 = 2^12
Reduction ratio: 2^16

The reason for this reduction is we can bruteforce just one byte at a time, avoiding the exponential increase.

Bonus1: Describe a simple pre-processing on power traces that will improve their alignment, and hence the DPA attack efficiency.

The first peak typically varies between index 51 to index 52 for the 10,000 traces. Instead of just always sampling at index 51, you could instead perform cross correlation first to align the traces, and then sample that point. Or you could use a sliding window to find the high peaks and find the value of the highest point there. I implemented the former method (see function track_peaks() in my code).

Bonus2: Assume that you do not have access to input plaintext but only to output ciphertext, how would you modify the attack?

First, you can use DPA to calculate the last round key, since the last step of AES is find the ciphertext by XORing the output of ShiftRows and AddRoundKey. We can then invert ShiftRows and SubBytes trivially.

At this point, we have the output of the second-to-last AddRoundKey. We can then use DPA again with the same concept as before to find the second-to-last round key. With this key, we can find the output of MixColumns. Using InvMixColumns, we can inverse MixColumns and find the output of ShiftRows.

At this point, we have inverted every type of operation in AES128 (other than round key generation). We can then repeat these operations for every single step in AES128 to completely reverse the encryption and find the original key.