Breaking AES through DPA

Venkatesh Rajagopalan

September 27, 2024

1 Introduction

The purpose of this assignment is to provide a hands-on introduction to power- based side-channel analysis. You are given a set of power traces taken during AES-ECB executions with the associated input/output values of AES, and you are asked to extract the 128-bit secret key via Differential Power Analysis (DPA).

2 Problem 1: Analyze power traces

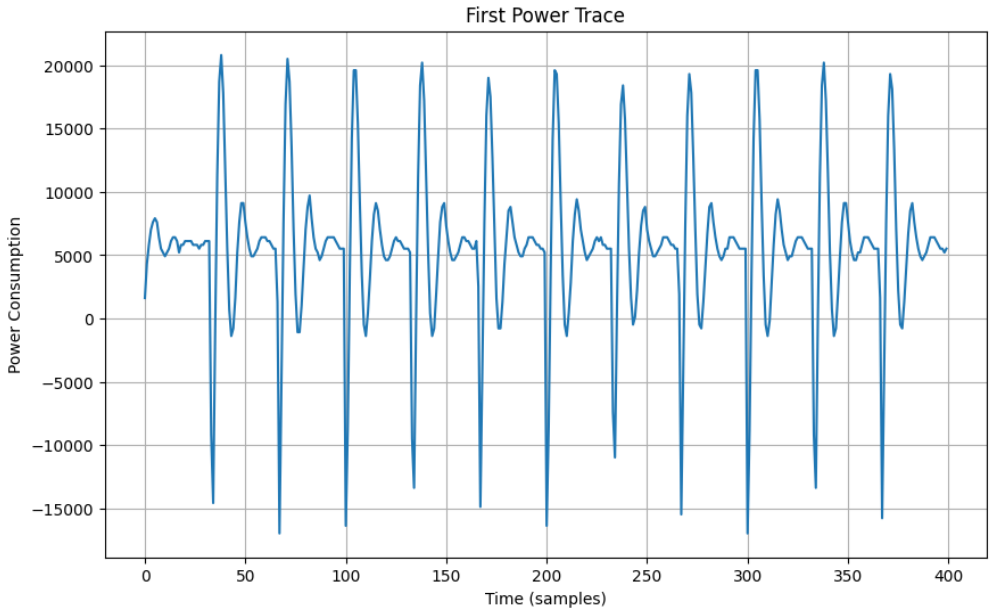

Figure 1: first power trace, corresponding to the first encryption operation.

Q1: Analyze the power trace. What are the peaks in the trace? How is AES implemented?

1. Analyzing the Power Trace

The power trace typically reflects the power consumption of a cryptographic algorithm, such as AES, during its execution. When examining the power trace, the peaks often correspond to key operations in the algorithm, such as:

- **S-Box Substitution**: Peaks at certain clock cycles may indicate S-Box lookups, as these are computationally intensive operations.
- **Key Schedule**: The AES key expansion algorithm might produce a distinct pattern of peaks.
- **Round Transitions**: Each AES round involves multiple operations (SubBytes, ShiftRows, MixColumns, AddRoundKey), and transitions between rounds can produce visible peaks.

These peaks are useful for identifying where sensitive operations take place, which can guide the DPA attack.

2. How AES is Implemented

AES is a block cipher that follows a structured process of encryption. Its implementation can be summarized in the following key steps:

- **Initial AddRoundKey**: Before starting the rounds, the input plaintext is XORed with the initial key.
- **Rounds (10 for AES-128)**: Each round consists of four main operations:
  1. **SubBytes**: A non-linear byte substitution using a pre-defined S-Box.
  2. **ShiftRows**: A row-wise shift operation that rearranges the bytes of the state matrix.
  3. **MixColumns**: A column-wise mixing transformation that provides diffusion. This step is skipped in the final round.
  4. **AddRoundKey**: The result is XORed with the round key.
- **Final Round**: The final round is similar to the previous rounds but without the MixColumns operation.

Each of these steps contributes to the overall power consumption, which is reflected in the power trace. Understanding where these operations occur in the trace helps in aligning the attack with the cryptographic process.

3 Problem 2: Perform the DPA Attack

Q2: What is a good power model for the target operation? Justify it. Feel free to try out different power models before you commit to one.

Power Model Justification: Hamming Distance

The `AddRoundKey` operation in AES performs a bitwise XOR between the plaintext (or intermediate state) and the round key. This XOR operation leads to bit transitions (from 0 to 1 or 1 to 0), which directly affect power consumption in most hardware implementations.

The Hamming Distance power model estimates the power consumption by counting the number of bit transitions between the previous state and the result of the XOR operation. The best two key guesses for our first byte (LSB) is shown below

Max Correlation: 0.11458946420228228, Key Guess: 0xf0, Time Sample: 43
Min Correlation: -0.11458946420228222, Key Guess: 0x0f, Time Sample: 38

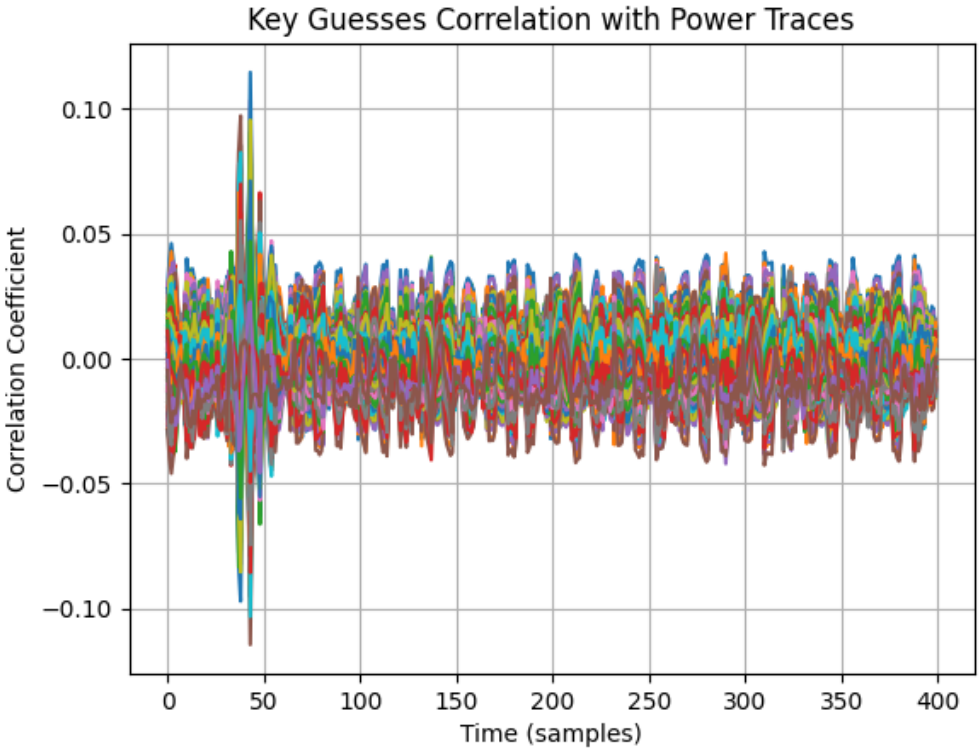

Figure 2: DPA on the first byte of the key.

Q3: Which one is more likely to be the correct key guess? Why do you specifically see these two results?

Based on the correlation results from the DPA attack, we observe the following:

Max Correlation: 0.11458946420228228, **Key Guess:** 0xf0, **Time Sample:** 43
Min Correlation: -0.11458946420228222, **Key Guess:** 0x0f, **Time Sample:** 38

Among these two key guesses, 0xf0 appears more likely to be the correct key guess, and the rationale for this choice is outlined below:

Correlation Significance: The correlation value associated with 0xf0 is positive (0.114), indicating a specific relationship between the hypothesized key and the power consumption. The negative correlation for 0x0f suggests a potential mismatch with the expected power trace behavior. This contrast implies that 0xf0 may indicate a more aligned relationship with the observed power traces, particularly in the context of the AddRoundKey operation.

Correlation Analysis Context: In DPA, the goal is to identify key guesses that closely match the power consumption patterns. The positive correlation for 0xf0 indicates that this key guess aligns more closely with the known behavior of the algorithm during encryption. Conversely, the observed negative correlation with 0x0f could indicate that this guess does not adequately represent the actual transitions expected from the cryptographic operation.

Bit Transition Analysis: The AddRoundKey operation in AES is fundamentally a bitwise XOR operation, which exhibits specific power consumption patterns. The guess 0xf0 may correlate with a distinct set of bit transitions occurring in the power trace, reflecting the actual operation performed during encryption. A key guess that aligns with expected transitions and shows a significant positive correlation is more plausible.

Statistical Confidence: If multiple trials yield a consistent correlation pattern favoring 0xf0, this would strengthen the confidence in it being the correct guess. Additionally, while 0x0f shows a negative correlation, it is essential to highlight that we compared it with the actual power trace for that time sample. This comparison may reveal discrepancies between the predicted and observed power consumption patterns, thereby reinforcing the rationale for preferring 0xf0 over 0x0f.

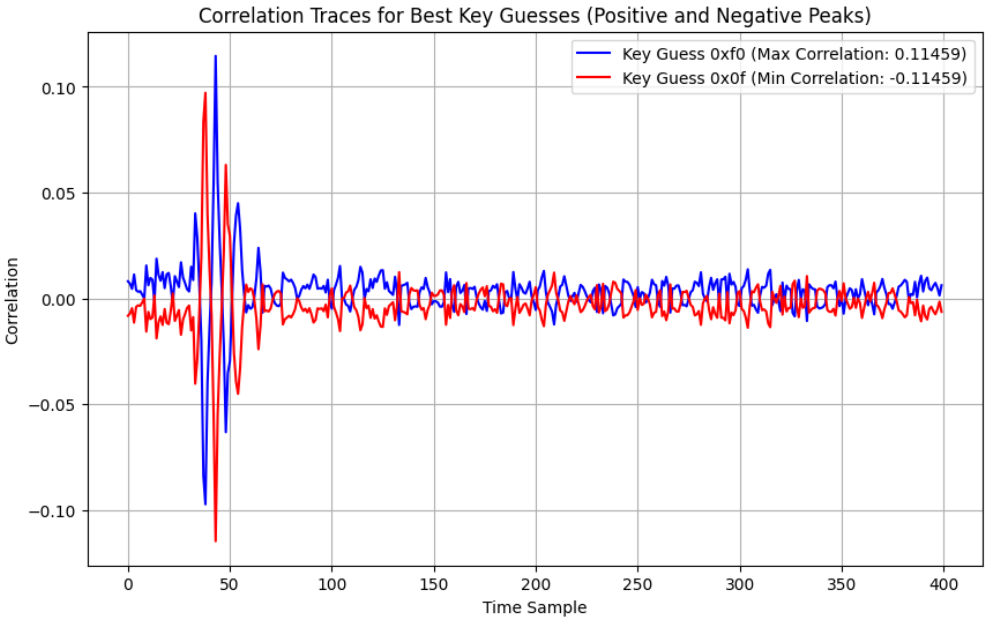

Figure 3: Maximum likelihood of two best key guesses.

Q4 What is the maximum leak point (in time domain)? Plot the "evolution" of all key hypothesis for 10000 measurements at this particular time instance

The maximum leak point i.e, the time at which the operations are "leaking" from the device. The significant correlations are happening at time window [30-50] for all the byte operations.

Plotting the key evolution graph for 10,000 is practically time consuming, so I plotted a key evolution graph for 2000 traces. Luckily, these were more than enough to observe the minimum number of measurements needed to leak the AES operations

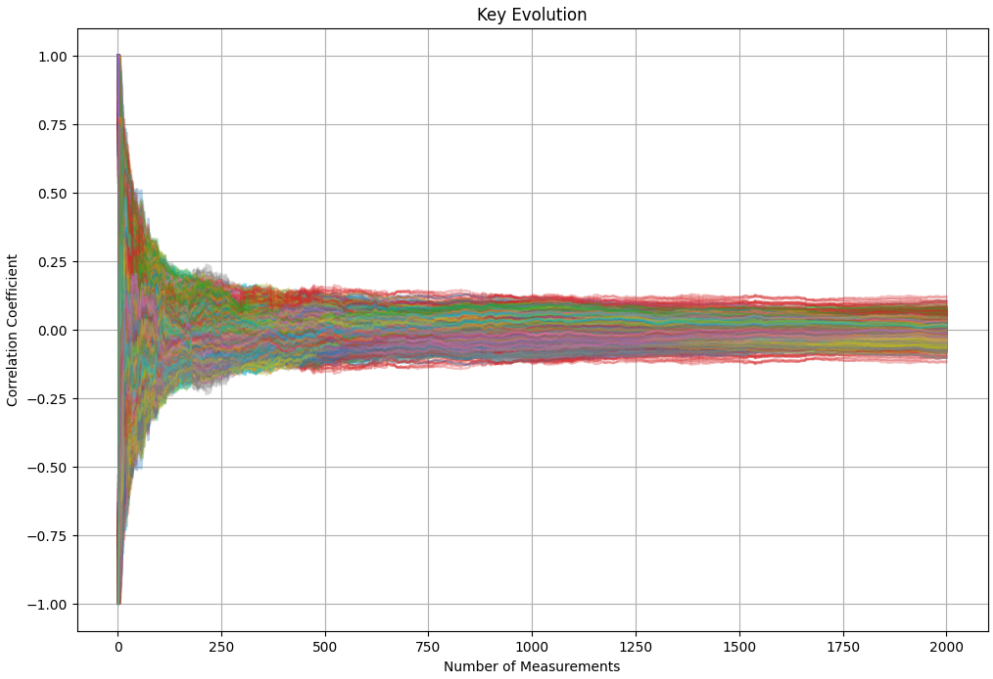

Figure 4: Key evolution plot.

Q5: Starting at how many traces, does the key guess with maximum correlation show highest correlation?

The key guess with the maximum correlation shows the highest correlation at around 250 measurements. Since the populated graph is crowded for 10,000 traces, I had to limit the traces to save some time.

Q6: Apply the DPA on the entire key. What is the 128-bit AES secret key? Note that not all key values could be 100your targeted computation, i.e. there can be false positives. If this is the case, you may need to take the first n possible results and apply a brute-force search within these reduced search space.

The DPA attack generated likely keys, one for a +ve correlation and one for a -ve correlation. One of them is a false positive. To be sure we are populating the correct keys and not "ghost keys" we take the best four guesses out of the key set and then the keys chosen based on the power trace behaviour on the specific time stamps.

The 16 byte secret key after applying DPA is: 0x000102030405060708090a0b0c0d0e0f

To make sure we are not getting any false positives, we will brute force the DPA key space. We will select top four key guesses for each byte and generate a key space for the entire secret key. This key space will be fed into a brute force algorithm to find out the correct key.

The "correct" secret key after brute forcing the key space: 0x000102030405060708090a0b0c0d0e0f

This verifies that our DPA attack was successful!

4 4 Problem 3: Evaluating DPA

Q7: What is the mean time to disclosure, i.e. average number of traces required to extract the key for your DPA attack?

After applying DPA on all the bytes of the plaintext and plotting the key evolution plot shows how the correlation of the hypothesized keys varies with the number of total measurements. We calculate the mean time to disclosure (MTTD) by averaging the minimum measurements required to estimate each key guess for all the bytes.

$$MTTD = \frac{\sum_{i=1}^{n} T_i}{n}$$

The total mean time disclosure (MTTD) taken for our DPA attack: 250 traces

Q8: Compare this with the theoretical cryptanalysis (i.e. brute- force attack) of AES, what is the reduction ratio in the number of traces required to break AES? Why is DPA so powerful?

Comparison of DPA and Brute-Force Attack on AES

AES uses key sizes of 128, 192, or 256 bits, requiring an average of 2^{127}, 2^{191}, or 2^{255} attempts for brute-force attacks, respectively. In contrast, Differential Power Analysis (DPA) can often disclose the key with significantly fewer traces (e.g., hundreds), leading to a reduction ratio R given by:

$$R = \frac{2^{(n-1)}}{M}$$

where n is the key size and M is the number of traces required by DPA.

In our case, the DPA analysis shows that the key leaks at an average of 250 traces. The reduction ratio (R) in our case becomes:

$$R = \frac{2^{(127)}}{250}$$

$$R \approx 6.81 \times 10^{35}$$

Why is DPA Powerful?

- **Exploits Side-Channel Information:** Uses power consumption patterns to reveal key information.
- **Lower Complexity:** Requires far fewer traces than brute-force attacks.
- **Statistical Analysis:** Employs statistical methods to find correlations, even amidst noise.

In summary, DPA attacks provide a critical advantage over traditional brute-force methods, underscoring the need for robust countermeasures against side-channel vulnerabilities.

Q9: How much time it took you to perform the experiment and complete questions 1–8?

The whole assignmnent, including the report writeup took me about 30 hours to complete. The plots took a long time to compile.

5 Extending the DPA Attack

Now use the traces2.csv, which is a set of measurements obtained from a slightly different implementation. Apply the same DPA attack using these traces.

Q10: Why is it failing now? What is the difference in hardware implementation causing this and how would you change the attack to break the implementation?

The traces2.csv file uses a different kind of implementation. It introduces some kind of external noise and disrupts the DPA attack. We need to preprocess traces2.csv to remove the random noise entering the hardware. Additionally, traces2.csv implements a different kind of power model and could not be broken with the help of hamming distance (HD). We need to come up with another power model in order to break it. A lot of randomness in the correlation graph shows us that

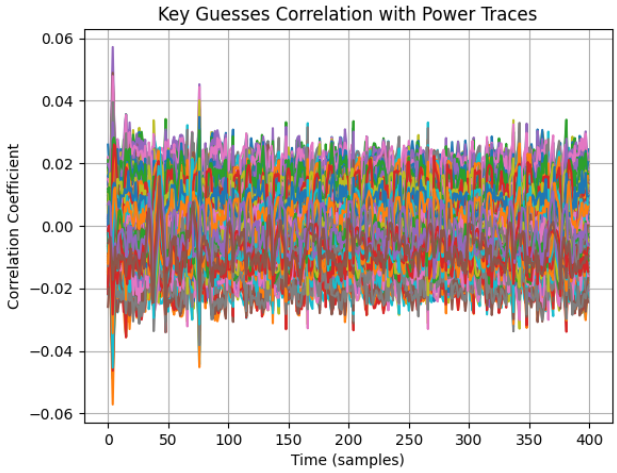

Figure 5: Key guesses for traces2.csv.

the data is not pre-processed properly.

Q11: Assume that you do not have access to input plaintext but only to output ciphertext (i.e. ciphertext-only cryptanalysis), how would you modify the attack?

Ciphertext-Only Differential Power Analysis (DPA) on AES

In a ciphertext-only cryptanalysis scenario, the attack can be modified as follows:

1. **Guess Intermediate Values**: Guess the values of the last round's S-box outputs based on the ciphertext.
2. **Hypothesis Testing**: Formulate hypotheses about the key bytes and compute the hypothetical power consumption.
3. **Correlation Analysis**: Compute the correlation between the hypothetical power consumption and the actual power traces.
4. **Statistical Techniques**: Use statistical methods to refine key hypotheses.
5. **Exploiting Redundancies**: Identify patterns in the ciphertexts to aid in guessing intermediate values.
6. **Advanced Methods**: Consider machine learning techniques to model power consumption patterns.