Test Vector Leakage Assessment

Course project for 2020 - CS 873-Cryptographic Engineering

Under the guidance of Prof. Srinivas Vivek

Group Members

- Dasari Sai Venkatesh (IMT2017012)
- Deep Inder Mohan (IMT2017013)
- Kaushal Mittal (IMT2017024)

Github Link

https://github.com/deep-guy/tvla

Table of Contents

1. Introduction

Cryptography has played a major role in data security and integrity. On the other hand, cryptanalysis is used to understand the effectiveness and correctness of cryptographic algorithms. Although secured by cryptography at the software level, devices are vulnerable to devastating side-channel attacks like hardware attacks. There have been multiple countermeasures such as masking and obfuscating power consumption signals to prevent such attacks. Testing these countermeasures involves different methodologies, one is extensive testing with the existing state of the art side-channel attacks but this has its own drawbacks. The other methodology is to detect the presence of any leakage rather than focusing on specifics like what and where. TVLA comes under the 2nd type of validation methodology.

2. Problem Statement

Demonstrate how the TVLA is used for power trace analysis.

3. Background

Test Vector Leakage Analysis (TVLA) was first proposed at NIST sponsored NIAT workshop 2011. The methodology came into existence as a side-channel resistance validation program. The fundamental principles of reproducibility of results and cost-efficiency of testing were foundational requirements for this methodology.

The approach is based on measuring power consumption from devices while performing cryptographic operations with a pre-specified set of input test vectors and then performing a set of pre-specified statistical tests on the collected power measurements. These statistical tests yield confidence scores, using which a clear fail/pass criterion can be established. The methodology uses Welch's t-test to identify statistical differences between power trace recordings. This organized approach offers multiple advantages over other side-channel key recovery attacks: because of the standardization of the test vectors and statistical tests that this method proposes, it becomes procedurally very simple and independent of the evolution of side-channel attacks.

3.1 Related Work

For the purposes of this project, explored the following three papers in detail:-

1. "***A testing methodology for side-channel resistance validation***"[1]: This is the original paper that proposed this methodology. It describes the specific guidelines of data collection and testing for TVLA. Additionally, the authors demonstrate the effectiveness of the methodology by also proposing a specific set of test vectors for three different AES implementations and presenting the corresponding experimental results using them.
2. "***Test Vector Leakage Assessment (TVLA) methodology in practice***"[2]: This paper presents enhancements made to the methodology through more than two years of testing with hundreds of implementations, and names this enhanced version as *TVLA*. Specifically, it demonstrates how TVLA can be sped up using bulk AES encryption modes and describes the design of test vectors to be used against AES implementation with timing and amplitude noise countermeasures.

3. "***How (not) to Use Welch's T-test in Side-Channel Security Evaluations***"[3]: TVLA outputs whether the proposed implementation is safe or not. But sometimes this straightforward assessment is not enough. To back this statement, this paper puts forward a case that looks safe from the TVLA point-of-view but it is very easy to break. This paper will give us a better understanding of the limitations of TVLA.

4 Our Implementation

We will be using power traces from DPA Contest v2 for our analysis. We are using python based tools for the implementation of TVLA validation method with Welch's T-Testing.

4.1 Data and Tools

DPA contest V2 provides traces, which were collected from AES-128 implementation on a SASEBO GII board (smartcard) during the encryption phase. They have collected traces for 32 different keys, each run on 20000 message inputs giving different cipher texts.
The complete dataset has 640000 traces each corresponding to a unique key, message, ciphertext triplet.

We have used

- **Numpy:** used for processing and computing aggregation like mean and variance.
- **Matplot:** used to plot graphs for our results.
- **AES-128 python implementation (https://github.com/boppreh/aes):** Used to determine intermediate values of the rounds given key and message text.

4.2 Our Implementation

Our implementation can be broadly split into 3 steps.

1. Understanding and obtaining the traces from DPA contest v2 and filtering out required traces.
2. Splitting the obtained traces based on intermediate values specified by the DTR .
3. Performing Welch's T-test on the set of traces.

4.2.1 **Obtaining and filtering the traces**

Download traces under public base from DPA-contest v2 website. The complete data has been split into 4 parts, for successful extraction download all 4 parts along with the index file and follow these steps

1. To build all the parts into a single file

```
cat DPA_contest2_public_base_diff_vcc_a128_2009_12_23.tar.bz2.part{0..3}
> DPA_contest2_public_base_diff_vcc_a128_2009_12_23.tar.bz2
```

2. To uncompress bz2 and extract the tar files:

```
unzip2 DPA_contest2_public_base_diff_vcc_a128_2009_12_23.tar.bz2

tar xvf DPA_contest2_public_base_diff_vcc_a128_2009_12_23.tar
```

3. To analyse the number of traces corresponding to each key, download the index file and run

```
awk '{print $1}' DPA_contest2_public_base_index_file | awk '{a[$1]++;}
END{for(i in a) print a[i]"  "i}'

output :
20000  d1310ba698dfb5ac2ffd72dbd01adfb7
.
.
20000  00000000000000003243f6a8885a308d3
```

As the sets of keys does not include any of the specific keys suggested by DVR for TVLA for an AES-128 implementation, we planned to use traces corresponding to the key **00000000 00000003 243f6a88 85a308d3.**

This key seemed to be a good choice over the other available keys since the hamming distance of this key from a key of all zeros was the least. This reasoning behind the choice of the key can be understood by referring [1]. So we have selected only the traces related to our key in one separate folder to ease access to these traces.

4. Based on the selected key (“00000000000000003243f6a8885a308d3” in our case), copy all the trace corresponding to the following key into ‘target’ directory

```
find -maxdepth 1 -name '*k=00000000000000003243f6a8885a308d3*' -exec cp
-t ../target {} +
```

4.2.2 Splitting the traces into two sets

Before splitting the set of traces, for each trace we run an AES-128 python implementation with key and message of the corresponding trace. Then according to the following criterias we will split the set of traces into 2 sets i.e, set0 and set1. Note each of the following criteria corresponds to a specific test as per the original TVLA paper, and M is chosen to be the middle round, which in the case of AES-128 is the 5th round. These criteria cover 768 of the 896 specific tests as listed in the TVLA DTR document.

Criteria 1 (RIRO_M_bit_0 through RIRO_M_bit_127):
For each trace *t*, we compute XOR *x* of i^{th} bit of input of round M and i^{th} bit of output of round M. If *x* is equal to 0 then we place *t* in set0 otherwise in set1. Like this we will split the traces for each i (0 to 127).

Criteria 2 (Rout_M_bit_0 through Rout_M_bit_127):
For each trace t, if the bit of round M output is 0 then we place t in set0 otherwise in set1. Like this we will split the traces for each i (0 to 127).

Criteria 3 (Rout_M_byte_0_is_0 through Rout_M_byte_0_is_255):
For each trace t, for a byte b. We will place trace t in set0 if the first byte of the output for round M is equal b otherwise in set1. Like this we will split for each b (o to 255), each i (0 to 127).

Criteria 4 (Rout_M_byte_1_is_0 through Rout_M_byte_1_is_255):
For each trace t, for a byte b. We will place trace t in set0 if the second byte of the output for round M is equal b otherwise in set1. Like this we will split for each b (o to 255), each i (0 to 127).

4.2.3 Analysing the traces

Now, we have split our original set of 20,000 traces into two subsets, set0 and set1 based on one of the above criteria, we proceed to check for statistical differences between these sets by using the Welch t-test. Unequal sizes of set0 and set1 do not pose any issue as Welch t-test does not require exactly the same number of traces.

Now we will follow the following steps:

1. Scale the power consumption values as per the given scale (1.41351e-5), although this does not affect the final results for the first order TVLA it is an important step when applying higher attack order TVLA.
2. Obtain the mean and variance for the resulting sets for each time-step of all the traces in a particular test.
3. The mean and the variance are used to calculate the t-scores.

$$\frac{X_A - X_B}{\sqrt{\frac{S_A^2}{N_A} + \frac{S_B^2}{N_B}}}$$

N_A and N_B are the size of the sets A and B.

X_A the average of all the traces in group A,
X_B the average of all traces in group B.

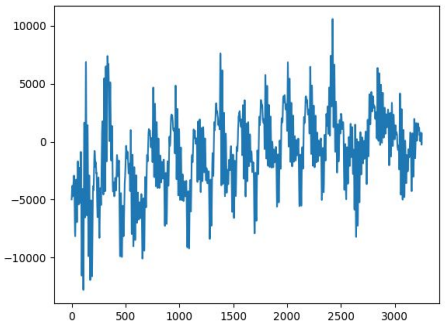
S_A, S_B the sample standard deviation of all the traces in group A and B

Finally after obtaining the t-score for all the specified tests, we plot and check if the t-significance values are crossed. This absolute value of t-score denotes how similar or how different the 2 sets ,i.e, set0 and set1 are.

- Low t-score implies that the set0 and set1 are sampled from same distribution i.e the attacker will not be able to learn anything new from these 2 sets
- High t-score implies that set0 and set1 are samples from different distribution, which indicates their must be some leakage in the traces which is causing this difference. These sets of traces can be used for launching side channel attacks.

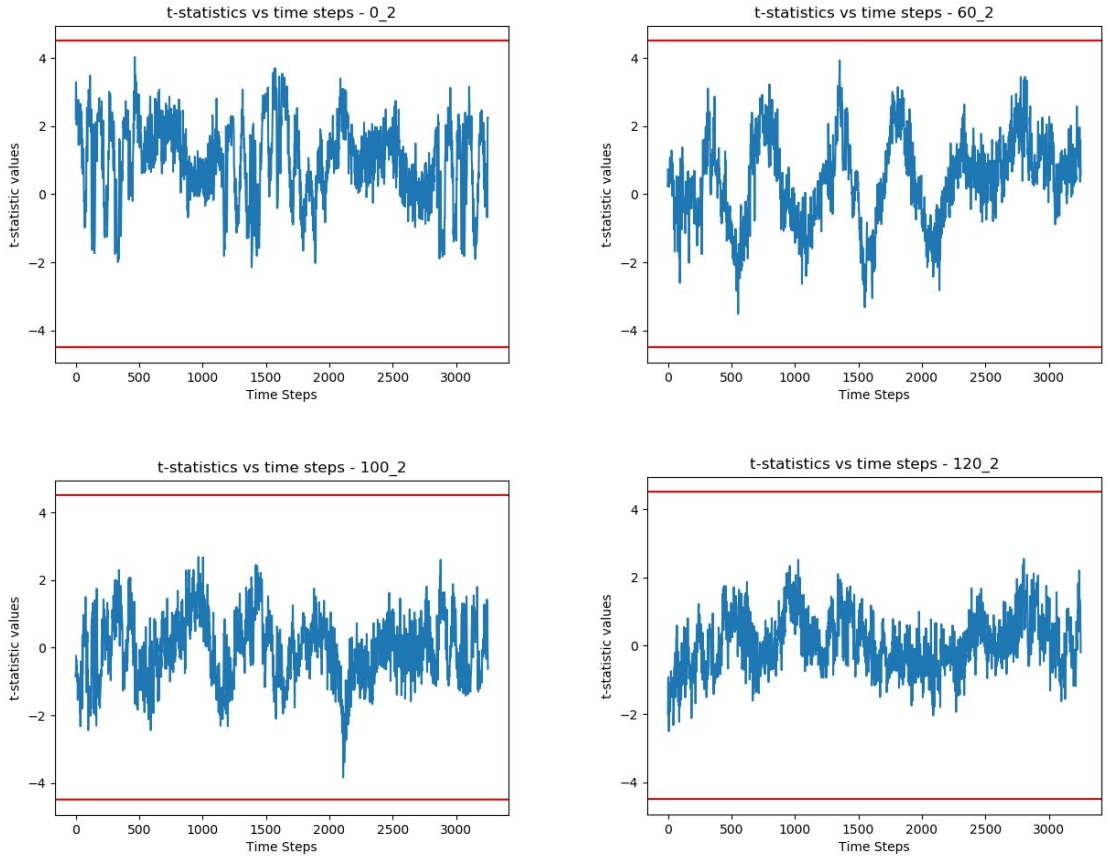
5. Results

We looked at the plot of a trace of specified AES implementation for random key and message pair, we can clearly identify the ten rounds as identified by ten corresponding peaks in between the trace. (figure 1)

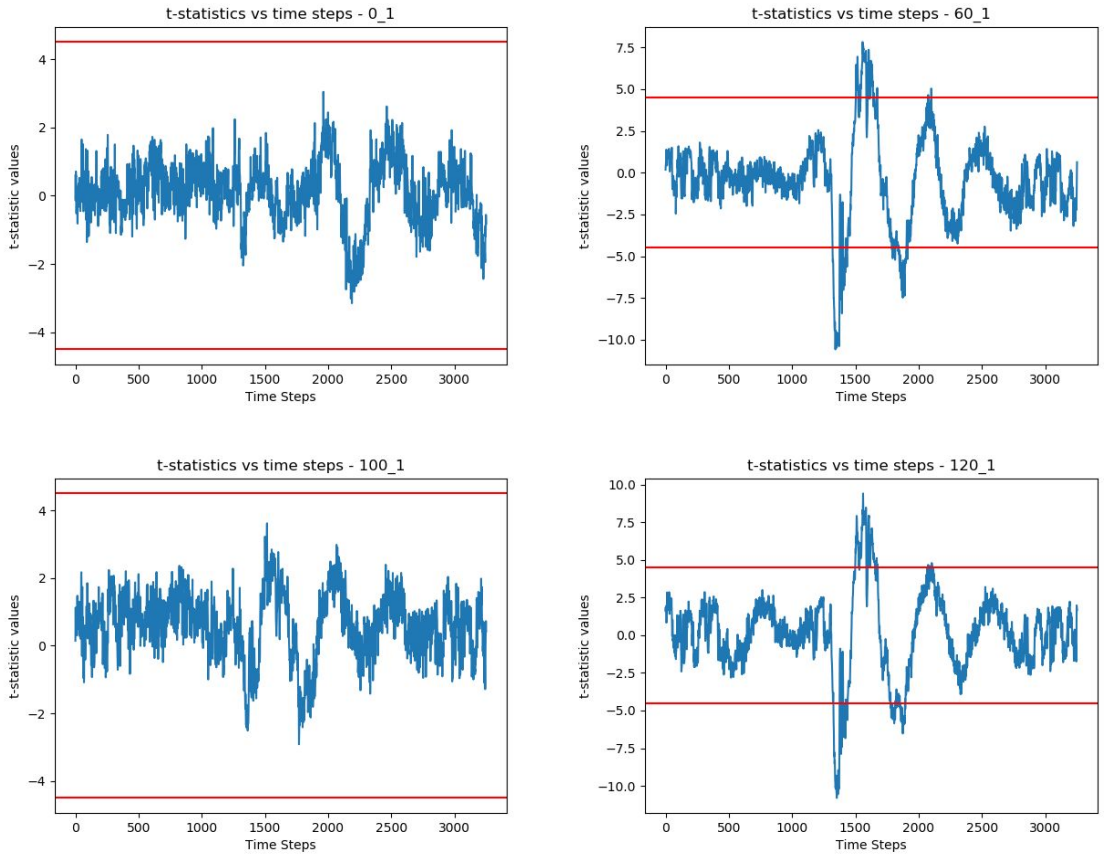


(figure 1: power consumption vs time-step)

When the sets are split based on the input value of the 5th round (criteria 2) on a particular bit, specifically 0^{th},60^{th},100^{th},120^{th} bit. We do not notice any t-statistics values breaking the threshold. Similarly for the test run on other criteria the threshold was not crossed except (criterion 1)

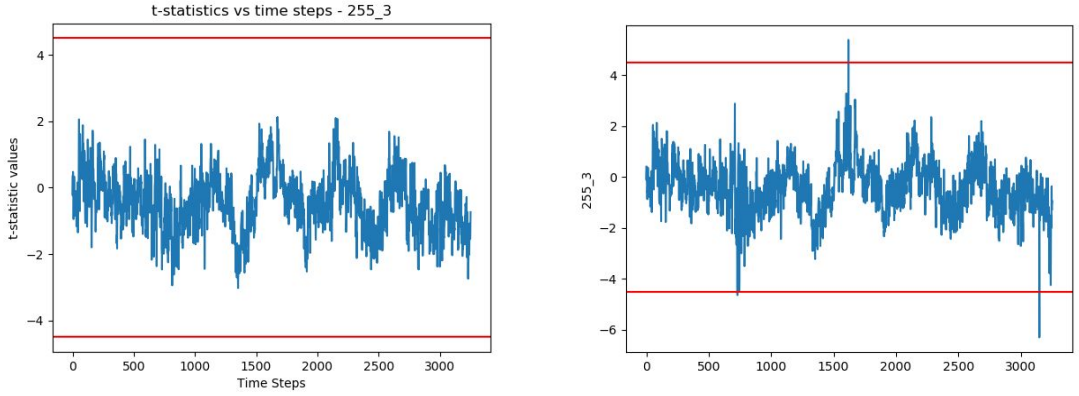


When the sets are split based on the XOR value of the 5th round (criteria 1) on a particular bit, specifically 0^{th},60^{th},100^{th},120^{th} bit. T-statistics value breaks the threshold value when split based on 60^{th} and 120^{th} bit value.



We notice spikes in t-score graphs corresponding to bit 60 and bit 120, this means that the power traces depend on the intermediate values. Specifically XOR of the bits in these positions show that when split on the basis of the XOR values both the sets show significant statistical difference which may reveal the intermediate round values in turn causing a side channel attack.

As a part of our observations, we also notice it was easier to cross the thresholds for higher order traces (3rd attack order). Where 3rd attack order means that all the values in the traces are raised to a 3rd power. This essentially denotes the attacker may launch a successful attack using higher attack order using the power traces. For example, based on the 1st byte value of the 5th round, the first order attack did not cross the threshold whereas we can clearly notice the 3rd attack order breaks the threshold.



First Attack Order

Third Attack Order

6. Conclusion

As expected for the given implementation of AES-128, without any countermeasures, side channel attacks such as DPA, CPA etc are possible and the results obtained by TVLA are representative of the same. Through this we are successfully able to demonstrate the vulnerability in the implementation despite lack of specified key, message pair.

7. References

[1] Goodwill, G., Benjamin Jun, J. Jaffe, and P. Rohatgi. "A testing methodology for side channel resistance." (2011).

[2] Becker, G., J. Cooper, E. DeMulder, G. Goodwill, J. Jaffe, G. Kenworthy, T. Kouzminov, A. Leiserson, M. Marson, P. Rohatgi, and S. Saab. "Test Vector Leakage Assessment (TVLA) methodology in practice" (2013).

[3] Standaert, F.. "How (not) to Use Welch's T-test in Side-Channel Security Evaluations." *IACR Cryptol. ePrint Arch.* 2017 (2017).

[4] DPA Contest V2 (2010)

[5] Open Source AES implementation