THE LEAKAGE MODEL AS THE DOMINANT FACTOR IN CORRELATION POWER ANALYSIS OF AES-128 ON A CORTEX-M0

Gabriela Nedyalkova (1), Magdalena Garvanova (1), Ivan Garvanov (1, 2)*

(1) University of Library Studies and Information Technologies, Sofia; (2) Institute of Mathematics and Informatics, Bulgarian Academy of Sciences, Sofia
Bulgaria

* Corresponding Author, e-mail: m.garvanova@unibit.bg

Abstract: Cryptographic algorithms secure at the algorithmic level may remain vulnerable when executed on physical hardware. This paper characterises correlation power analysis against an unprotected AES-128 implementation on a low-cost ARM Cortex-M0 microcontroller, separating four factors commonly assumed to govern attack success: the leakage model, the trace count, signal preprocessing, and the choice between profiled and non-profiled attacks. Two datasets of 5 000 and 10 000 traces were captured on a ChipWhisperer-Nano and analysed with a custom implementation cross-validated against the ChipWhisperer Analyzer. The complete key was recovered from 55 traces. The Hamming weight model gave a mean peak correlation of 0.6985 against 0.0996 for Hamming distance, which resolved only 2 of 16 bytes; filtering altered correlation by 0.45%. Leakage model choice dominates every other factor examined.

Key words: side-channel analysis, correlation power analysis, leakage model, AES-128, embedded security.

1. INTRODUCTION

Modern cryptographic algorithms are designed to resist mathematical attack. AES-128, standardised by NIST [1], is among the most widely deployed symmetric block ciphers in embedded systems and remains secure at the algorithmic level, with no known practical attack on the full cipher substantially more efficient than exhaustive key search [2]. Executing AES on a physical processor, however, produces measurable side effects: small variations in power consumption that depend on the data being processed. These are not software defects but a consequence of how switching CMOS logic draws current according to the values it handles [3]. Where those values relate to a secret key, a device may be cryptographically secure in theory and physically vulnerable in practice: a gap that NIST's own review of the standard identifies as considerably more damaging than cryptanalytic attacks on the algorithm itself [2].

The gap between algorithmic security and implementation security is not hypothetical. Connected IoT devices numbered 18.5 billion in 2024 and are projected to reach 39 billion by 2030 [4], and many run compact software implementations of AES on low-cost microcontrollers without dedicated side-channel countermeasures [5]. Where hardware is inexpensive and firmware is optimised for speed and code size rather than physical security, secret keys may be extractable through comparatively simple statistical analysis of power consumption [3]. As the number of connected devices grows, so does the practical significance of two questions: how readily such implementations can be attacked, and how many observations an attacker actually requires.

This paper reports a systematic empirical characterisation of correlation power analysis against an unprotected AES-128 implementation on a low-cost ARM Cortex-M0 microcontroller. Four factors plausibly govern attack success: the assumed leakage model, the number of traces, signal preprocessing, and the choice between non-profiled and profiled attack paradigms. All four are evaluated against a single common dataset, so that their contributions can be separated rather than confounded. The results identify one of them as dominant. The measurement campaign and the analysis implementation are the authors' own: two trace sets were captured on hardware configured for this study, and a correlation analysis exposing the full 256-hypothesis space was implemented in Python and cross-validated against the reference tool.

The methodology is built around the ChipWhisperer-Nano, a low-cost board in the open-source ChipWhisperer platform [6], with an integrated target microcontroller. Two datasets of 5 000 and 10 000 traces were captured under a fixed-key, random-plaintext scheme, and the attack was implemented as a vectorised NumPy computation cross-validated against the ChipWhisperer Analyzer. Convergence, minimum traces to disclosure, a comparison of the Hamming weight and Hamming distance leakage models, digital preprocessing and a profiled support vector machine attack were each evaluated on the same data.

2. RELATED WORK

Power consumption in CMOS logic depends on the data processed, since current is drawn predominantly when transistors switch state. Attacks exploit this by relating an intermediate value to expected consumption, most commonly through the Hamming weight of the value or the Hamming distance between successive states [3]. Both are simplifications: real leakage is shaped by transistor-level behaviour, parasitic capacitance and bus routing [7], and simulator work on ARM Cortex-M cores shows behaviour more complex than either captures [8].

Three techniques form a lineage of increasing statistical power. Simple power analysis inspects traces visually and succeeds only where control flow depends on the key [9]. Differential power analysis [9] showed that statistical averaging could extract keys from leakage invisible in any single trace. Correlation power analysis (CPA) replaced its difference-of-means distinguisher with the Pearson correlation coefficient and an explicit leakage model [10], needing fewer traces and allowing models to be compared on the same measurements [3, 10].

Evaluation was later formalised so results could be compared across devices and methods. A framework separating success rate from guessing entropy, the expected rank of the correct candidate, is proposed in [11]. These measures express attack cost quantitatively, in traces, rather than qualitatively [3].

A separate line assumes the attacker can characterise a device in advance. Template attacks [12] model leakage during profiling and apply it to the attacked device. Machine-learning classifiers were later shown to serve the same role, from least-squares support vector machines [13] to comparative studies of classifiers and labelling schemes [14, 15], and then deep learning with public trace databases [16]. Transfer between instances remains difficult, since manufacturing variation alters the leakage profile [17, 18], so single-device evaluations are a favourable case.

Side-channel security has been addressed in this journal from two directions. At the cipher-design level, a lightweight block cipher for constrained IoT devices is proposed in [19], whose constant-time bit-sliced implementation resists cache-timing attacks, benchmarked on ARM Cortex-M targets including the Cortex-M0. At the evaluation level, the same ChipWhisperer-Nano measurement platform and AES implementation employed here, with an external Cortex-M4 target, is used in [20] to measure how correlation power analysis degrades as artificial noise is injected into the target's power supply. The journal has also published practical security assessments of embedded devices carried out with accessible open-source tooling [21]. The experiments reported here were carried out on a ChipWhisperer-Nano measurement board [22] with an integrated STM32F030F4P6 target [23], running the tiny-AES-c implementation of AES-128 [24]. The present work occupies the space between countermeasure design and countermeasure evaluation: rather than proposing a defence or measuring one, it characterises the leakage behaviour of an unprotected implementation across two leakage models and two attack paradigms, establishing the baseline against which such defences are measured.

3. EXPERIMENTAL SETUP AND METHODOLOGY

3.1. Target platform and trace acquisition

All experiments were performed on a ChipWhisperer-Nano (NAE-CWNANO), a low-cost side-channel analysis board belonging to the open-source ChipWhisperer platform [6]. The board combines the power measurement hardware and the target microcontroller on a single circuit board, so no separate oscilloscope or external target device is required. Power consumption is digitised by an 8-bit analog-to-digital converter (ADC) with a maximum sampling rate of 20 MS/s.

The target device is a STMicroelectronics STM32F030F4P6, a 32-bit microcontroller based on the Arm Cortex-M0 architecture. It was clocked at 7.5 MHz, one of the fixed output clock frequencies provided by the CW-Nano and the default value applied by the ChipWhisperer API for this device. Sampling is synchronous, meaning that the ADC is locked to the target clock rather than driven by an independent asynchronous oscilloscope.

The firmware implements AES-128 using the tiny-AES-c library. The implementation is unprotected and applies no masking, no shuffling of operations and

no insertion of random delays. This was a deliberate choice, since an unprotected target provides the clearest baseline for evaluating the CPA methodology itself.

Acquisition parameters were left at the ChipWhisperer defaults, which record 5 000 samples per trace and trigger on a rising edge from a target GPIO pin. A fixed-key, random-plaintext scheme was used, which is the appropriate configuration for a non-profiled attack, since the attacker observes varying plaintexts while attempting to recover a constant secret key. Two datasets were captured under identical settings and differ only in size: a baseline set of 5 000 traces and an extended set of 10 000 traces.

3.2. CPA implementation and leakage models

The attack targets the output of the SubBytes operation in the first AES round. For each trace, the known plaintext byte is combined by XOR with a key hypothesis and the result is passed through the AES S-box, producing an intermediate value that depends on both the public plaintext and the unknown key byte.

The built-in ChipWhisperer Analyzer attack was first run on both datasets as verification, recovering all 16 key bytes with PGE 0 on each. Subsequent experiments required direct access to the correlation values for all 256 key hypotheses across all samples, which the built-in workflow did not expose, so a custom implementation was developed by the authors in Python 3.11 using NumPy, SciPy and scikit-learn, with the Analyzer API retained for verification.

Equations (1) to (3) state the standard definitions applied in this study; the contribution of this section is their controlled comparison on a single dataset. The distinguisher is the Pearson correlation coefficient [10], computed for every key hypothesis k and every sample index t:

$$\rho_{k,t} = \frac{\sum_i (h_{i,k} - \bar{h}_k)(T_{i,t} - \bar{T}_t)}{\sqrt{\sum_i (h_{i,k} - \bar{h}_k)^2 \sum_i (T_{i,t} - \bar{T}_t)^2}} \quad (1)$$

where $h_{i,k}$ is the hypothetical leakage for trace i under hypothesis k, $T_{i,t}$ is the measured sample at index t, and the bar denotes the mean over all traces. The computation is carried out for all 256 values of a key byte across all 5 000 samples, producing a 256 × 5 000 correlation matrix per byte position, and is vectorised in NumPy. The procedure is repeated independently for each of the 16 key bytes.

Two leakage models were evaluated on the same traces. The Hamming weight model assumes that power consumption relates to the number of bits set to 1 in the processed intermediate value [3]:

$$h_{i,k} = \mathrm{HW}\big(\mathrm{SBox}(p_i \oplus k)\big) \quad (2)$$

The Hamming distance model instead assumes that consumption relates to the number of bit transitions between two consecutive register states, here taken as the transition from the plaintext byte to the SBox output [10]:

$$h_{i,k} = \mathrm{HW}\big(p_i \oplus \mathrm{SBox}(p_i \oplus k)\big) \quad (3)$$

Attack performance is quantified using three metrics. Partial guessing entropy (PGE), a per-subkey form of the guessing entropy of [11], is the rank of the correct key byte among the 256 candidates, so a PGE of 0 indicates that the correct value is ranked first. Success rate is here defined as the proportion of the 16 key bytes reaching PGE 0. Minimum traces to disclosure (MTD) are the smallest number of traces at which a byte

first reaches PGE 0, the global MTD being the maximum across the 16 bytes. Random ordering of the candidates would give a mean PGE of approximately 127.5.

3.3 Profiled attack setup

A profiled attack based on a support vector machine was implemented on the same 10 000-trace dataset to compare a learned leakage model against the assumed models above.

Feature selection was applied first, since using all 5 000 samples as inputs would be computationally wasteful and would increase the risk of overfitting. Points of interest were identified for key byte 0 by computing the absolute Pearson correlation between the Hamming weight of the S-box output under the correct key byte and the value at each sample position, evaluated over all 10 000 traces. The ten highest-correlating sample indices were 576, 577, 841, 842, 844, 846, 859, 860, 881 and 931, and each trace was represented by the values at these ten positions.

The classifier was trained to predict the Hamming weight of the S-box output for byte 0, a value between 0 and 8, giving a nine-class problem. The traces were split into 7 000 profiling traces and 3 000 attack traces. Features were standardised with parameters estimated on the profiling set and applied unchanged to the attack set. An SVM with a radial basis function kernel was used with C = 1.0, gamma set to 'scale', probability estimates enabled and a fixed random seed (random_state = 0).

Key recovery does not rely on individual classifications. For each of the 256 candidate values of byte 0 and each attack trace, the plaintext byte is combined with the candidate to give the expected Hamming weight class, and the probability assigned to that class is added in logarithmic form to a running score. After all attack traces have been processed, the candidate with the highest accumulated score is returned as the recovered key byte.

4. RESULTS

4.1. Key recovery and trace requirements

The CPA attack under the Hamming weight model recovered the full 128-bit AES key from both captures. Partial guessing entropy reached 0 for all 16 key bytes in each case, giving a success rate of 100%, matching the key used during acquisition.

Figure 1 shows the correlation traces for key byte 0 over the 10 000-trace capture. The correct hypothesis produces a sharp correlation peak of $\rho = 0.7251$ at sample 576, followed by several weaker peaks between samples 780 and 940 that correspond to later operations on data derived from the same byte. Beyond approximately sample 1 200 the correlation for all hypotheses falls to uniformly low values over the remainder of the 5 000-sample trace, which is not shown. The 255 incorrect hypotheses remain within a narrow band around zero throughout, as expected for estimators of a zero correlation.

Averaged across all 16 bytes, the peak absolute correlation was 0.6970 for the 5 000-trace capture and 0.6985 for the 10 000-trace capture. These are two independent acquisition sessions, and the difference between them is 0.21%, indicating stable measurement conditions rather than a dependence on capture size.

Both captures are therefore far larger than the attack requires. Evaluated on the 10 000-trace capture at eight intermediate trace counts between 500 and 10 000, all 16 bytes were recovered at every point, so convergence occurs somewhere below 500 traces. Evaluation at finer intervals resolves the transition. No bytes were recovered at 10 traces, 2 at 20 traces, 6 at 30, 13 at 40 and 15 at 50, with full recovery of all 16 bytes first observed at 75 traces and sustained at every larger trace count tested.

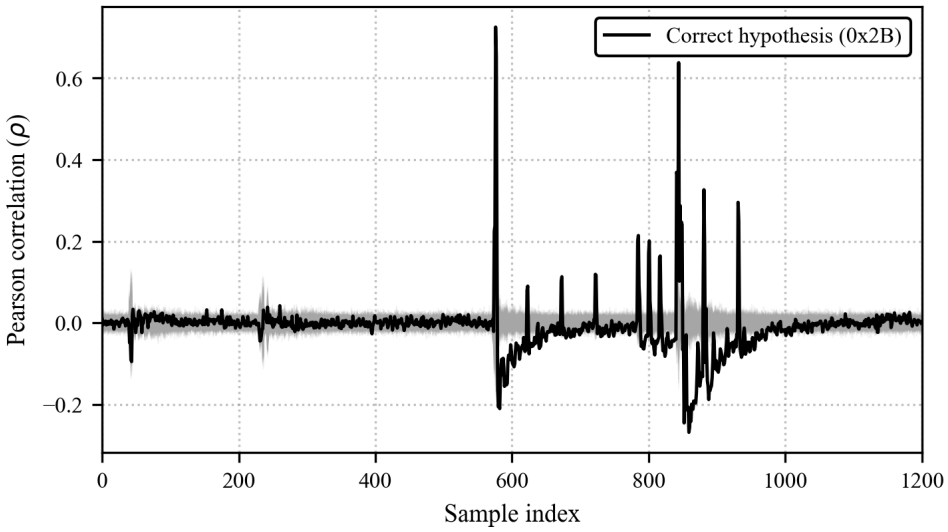


Figure 1. CPA correlation traces for key byte 0 over 10 000 traces, shown for the first 1 200 samples. The correct hypothesis (0x2B) is shown in black and the 255 incorrect hypotheses in grey.

Because that analysis samples only preselected trace counts, the exact threshold was located by a trace-by-trace search. Figure 2 gives the resulting minimum traces to disclosure for each byte. The lowest is 12 traces for byte 2 and the highest is 55 traces for byte 1, a spread of approximately 4.6 to 1. The global MTD is therefore 55 traces, since every byte reaches rank 1 within that count. The coarser sweep above, which re-evaluates all 16 bytes at each point, first showed simultaneous recovery of all 16 at 75 traces, because a byte that first attains rank 1 can fall below it again as further traces are added, a known imprecision when trace counts are read from a single experiment [3]. The two figures therefore reflect different definitions of the trace requirement rather than a discrepancy in the data.

Digital preprocessing was also evaluated. A third-order Butterworth low-pass filter with normalised cutoff Wn = 0.3 was applied with zero phase shift to the first 5 000 traces of the extended capture, and the attack was repeated on both the raw and filtered versions of that subset. The mean peak absolute correlation rose from 0.7062 to 0.7094, an improvement of 0.45%. The improvement is negligible and reflects the acquisition configuration rather than filtering in general: the ADC samples synchronously with the target clock, which already suppresses the asynchronous noise a low-pass filter targets. This baseline is that subset, not the separate 5 000-trace acquisition reported above.

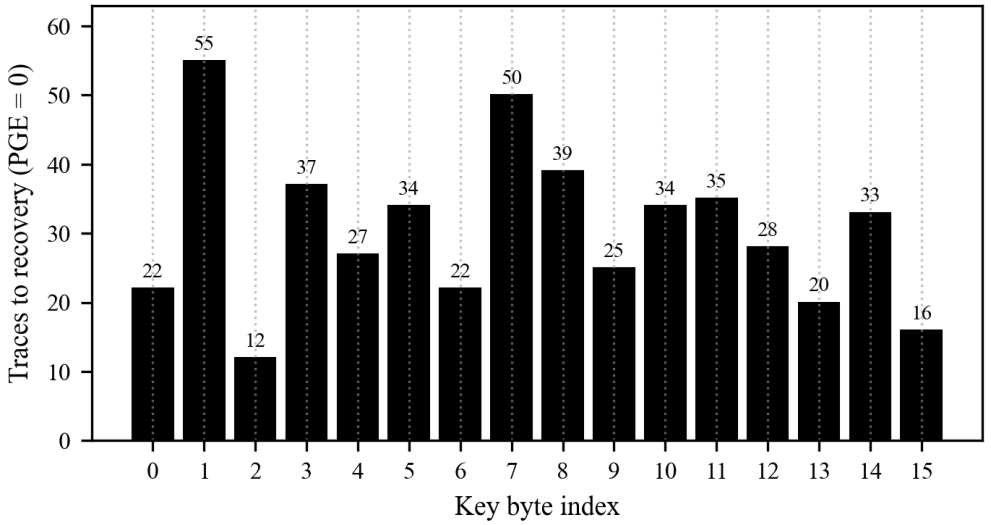


Figure 2. Minimum traces to disclosure for each of the 16 AES key bytes, measured on the 10 000-trace capture under the Hamming weight model.

4.2. Leakage model comparison

The same 10 000 traces were then evaluated under both leakage models. For this comparison the correct key byte was supplied directly rather than searched for, since the objective was to measure how well each model matches the observed leakage rather than to perform key recovery.

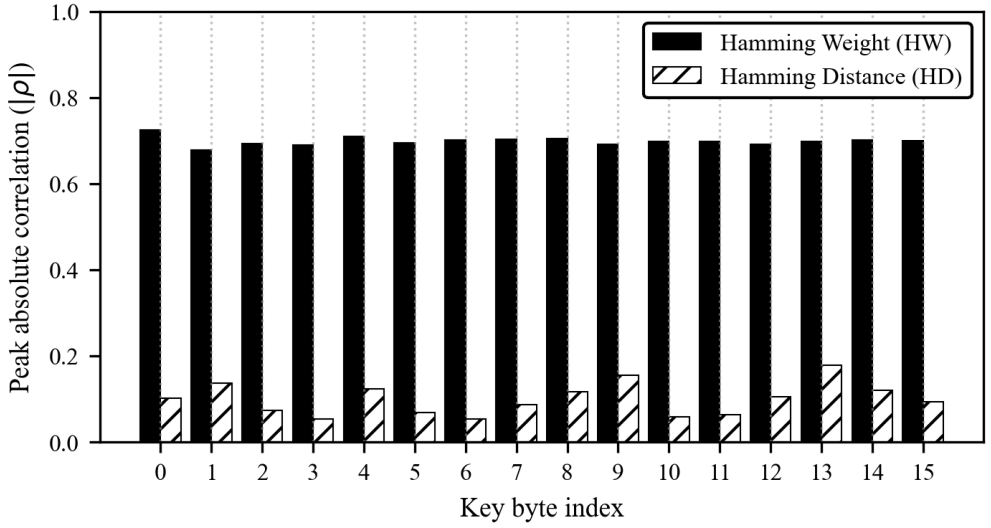


Figure 3. Peak absolute correlation per key byte under the Hamming weight and Hamming distance models, measured on the 10 000-trace capture.

Figure 3 shows the peak absolute correlation obtained for every key byte under each model. Under the Hamming weight model, the values remain high and tightly grouped

across all 16 bytes, falling approximately between 0.68 and 0.73, with a mean of 0.6985. Under the Hamming distance model, they fall approximately between 0.05 and 0.18, with a mean of 0.0996. The Hamming weight model therefore yields a correlation on average about seven times stronger on this dataset and does so at every byte position rather than on average alone.

The effect on attack outcome is correspondingly large. Under the Hamming weight model, the full attack achieves a 100% success rate, a mean final PGE of 0.0 and a global MTD of 55 traces. Under the Hamming distance model only 2 of the 16 key bytes are recovered, a success rate of 12.5%, with a mean final PGE of 66.9 and a global MTD exceeding 10 000 traces. The mean PGE of 66.9 is nevertheless well below the value of approximately 127.5 expected from random ordering, so the Hamming distance model retains a weak association with the measured leakage while remaining unusable for full key recovery in this configuration.

4.3. Profiled attack

The profiled SVM attack was evaluated on the same 10 000-trace capture. On the 3 000 attack traces the classifier reached an accuracy of 52.23% in assigning single traces to one of the nine Hamming weight classes, against approximately 11.1% for random assignment.

Single-trace accuracy at this level is not sufficient to recover a key byte from one trace. Accumulating the class probabilities in logarithmic form across the attack set, however, recovered key byte 0 as 0x2B, matching the true value, with a PGE of 0. The correct value was separated from the remaining candidates by a clear margin, confirming that recovery derives from the accumulation of weak but consistent evidence across many traces rather than from any individual classification.

Both approaches identify the same instant as most informative: the strongest correlation peak in Figure 1 falls at sample 576, which is also the highest-ranked point of interest selected independently during feature selection.

The two approaches differ sharply in what they demand. Correlation power analysis required no prior access to the device and recovered all 16 bytes from 55 traces under a fixed Hamming weight assumption. The profiled attack learned its leakage model from the data instead but needed 7 000 labelled traces for profiling and 3 000 for the attack, recovered only byte 0, and was neither minimised for trace count nor evaluated across separate physical devices.

5. DISCUSSION

A global MTD of 55 traces is the security-relevant result of this study. Each trace corresponds to a single AES-128 encryption, so an attacker with physical access to the measurement point needs to observe only 55 encryptions to recover the complete key. This does not imply that all deployed STM32-based products are equally exposed, since the trace requirement depends on firmware, clock configuration, measurement setup and any countermeasures applied. Independent work using the same measurement platform and AES implementation on a Cortex-M4 target reports a 100% subkey recovery rate

for correlation power analysis in the absence of injected noise [20], which is consistent with the baseline established here.

5.1. Why the leakage model dominates

The dominance of the Hamming weight model is consistent with pre-charge bus behaviour. If the internal data path is driven to a fixed reference state, typically 0x00, between successive operations, then bits switch relative to that reference rather than relative to the previous data value. Since the Hamming distance from 0x00 to any value x equals the Hamming weight of x, the two models coincide, as established in [3, 7, 10]. The Hamming weight model therefore does not describe a property of the data value itself; it captures the same physical mechanism as a correctly formulated Hamming distance model in which the pre-charged zero state, rather than the plaintext byte, serves as the reference. The model tested here assumed the plaintext byte persisted in the target register, which under pre-charge behaviour it would not, and that mismatch accounts for the weak correlation observed.

The Hamming distance model does not fail completely, however. Its mean PGE of 66.9 sits well below the 127.5 expected from random ordering, indicating a residual association with the measured leakage despite the incorrect reference state. Secondary effects such as parasitic coupling between adjacent bus lines, incomplete pre-charge under timing constraints, or data-dependent activity in peripheral logic may contribute such a component; mechanisms of this kind are described in [3, 8], though distinguishing between them would require more targeted analysis. The residual is in any case far too weak for practical key recovery.

This interpretation is an empirical inference from the statistical fit of two competing models, not a direct observation of bus state. Leakage-simulator work on Cortex-M0 cores shows behaviour more complex than any single simplified model captures [8], so the result should be read as a characteristic of this device under this measurement configuration rather than a universal property of the core. Comparing leakage models therefore functions as a form of non-invasive architectural probing: two bus behaviours were distinguished without the decapsulation and localised electromagnetic measurement used in earlier work [7].

5.2 Limitations

Several limitations bound these results. All measurements come from a single physical device, so manufacturing variation across nominally identical parts remains unquantified here, although cross-device studies [17, 18] have measured it directly. In the profiled attack, profiling and attack were performed on the same board, which excludes the cross-device variation a realistic attacker would face and makes 52.23% an optimistic estimate. Points of interest were selected using correlation computed across all 10 000 traces before the profiling and attack sets were separated, which favours the classifier; a stricter evaluation would select features on profiling data alone. The number of attack traces was not minimised, so the trace counts reported for the profiled attack are those used rather than requirements. Finally, the target implementation is unprotected by design, so these results characterise a baseline and do not predict how the same attacks would perform against a protected implementation. The target was clocked at 7.5 MHz

throughout, and the effect of higher clock rates was not evaluated; a faster clock reduces the number of samples acquired per operation and places greater demands on the measurement front-end, so the trace requirement reported here should not be assumed to transfer unchanged. Capture was started by an explicit trigger from a target GPIO pin, so the encryption began at a fixed offset in every trace. Where the timing of the cryptographic operation is not known in advance, trace alignment would be required before analysis, and the trace requirement would rise accordingly.

The four factors were not exercised symmetrically: the leakage model was tested as a correct specification against a deliberately mismatched one, whereas the other three were varied within a single configuration. The comparison is better read as diagnostic of this device than as a balanced contest.

6. CONCLUSION

This work implemented a correlation power analysis attack against an embedded AES-128 implementation, quantified its efficiency using formal evaluation metrics, and examined both the underlying leakage-model assumption and an alternative profiled approach on the same platform.

The full 128-bit key was recovered with PGE 0 on two independent captures at a global MTD of 55 traces, with mean peak correlation near 0.70. Under Hamming distance on identical traces, it fell to 0.0996, resolving only 2 of 16 bytes.

Taken together, the results identify the leakage model as the dominant factor in attack efficiency on this target. Trace volume was not: full recovery occurs at 55 traces and is already complete at every trace count tested from 500 upward. Preprocessing was not: low-pass filtering changed correlation only marginally. Attack sophistication was not free: the profiled approach used roughly 180 times as many traces to recover a single byte. The choice between Hamming weight and Hamming distance, by contrast, separated complete key recovery from failure.

The contribution is fourfold: the four factors were compared on identical traces and a single target, isolating model choice from every other variable; the trace requirement was resolved to a single figure; both attack paradigms were evaluated on one dataset with their asymmetries stated; and a negative result was reported in full, preprocessing having yielded no benefit here.

Several directions follow. Applying the same methodology to a masked implementation would establish whether countermeasures displace the leakage model as the dominant factor or merely raise the trace requirement. Extending the profiled attack to all 16 bytes across separate physical devices would place the two paradigms on stricter terms.

ACKNOWLEDGEMENT

This work is supported by the Bulgarian National Science Fund, Project title “Innovative Methods and Algorithms for Detection and Recognition of Moving Objects by Integration of Heterogeneous Data”, KP-06-N 72/4/05.12.2023.

REFERENCES

[1] National Institute of Standards and Technology. *Advanced Encryption Standard (AES)*, FIPS 197-Upd1, NIST, Gaithersburg, USA, 2023. DOI: 10.6028/NIST.FIPS.197-upd1.

[2] Mouha, N. *Review of the Advanced Encryption Standard.* NIST IR 8319, NIST, Gaithersburg, USA, 2021. DOI: 10.6028/NIST.IR.8319.

[3] Mangard, S., Oswald, E., Popp, T. *Power Analysis Attacks: Revealing the Secrets of Smart Cards*. Springer, USA, 2007, (338 p.). DOI: 10.1007/978-0-387-38162-6.

[4] Sinha, S. *State of IoT 2025: Number of connected IoT devices growing 14% to 21.1 billion globally*, IoT Analytics, October 2025. URL: https://iot-analytics.com/number-connected-iot-devices/ (Visited on 08.08.2026).

[5] Liptak, C., Mal-Sarkar, S., Kumar, S.A.P. Power analysis side channel attacks and countermeasures for the Internet of Things. *2022 IEEE Physical Assurance and Inspection of Electronics (PAINE),* Huntsville, USA, October 2022, pp. 1-7. DOI: 10.1109/PAINE56030.2022.10014854.

[6] O'Flynn, C., Chen, Z. ChipWhisperer: an open-source platform for hardware embedded security research. *Constructive Side-Channel Analysis and Secure Design – COSADE 2014*, Paris, France, April 2014, pp. 243-260. DOI: 10.1007/978-3-319-10175-0_17.

[7] Peeters, E., Standaert, F.-X., Quisquater, J.-J. Power and electromagnetic analysis: improved model, consequences and comparisons. *Integration*, vol. 40, no. 1, 2007, pp. 52-60. DOI: 10.1016/j.vlsi.2005.12.013.

[8] McCann, D., Oswald, E., Whitnall, C. Towards practical tools for side channel aware software engineering: 'grey box' modelling for instruction leakages. *26th USENIX Security Symposium (USENIX Security 17)*, Vancouver, Canada, August 2017, pp. 199-216.

[9] Kocher, P., Jaffe, J., Jun, B. Differential power analysis. *Advances in Cryptology – CRYPTO '99*, Santa Barbara, USA, August 1999, pp. 388-397. DOI: 10.1007/3-540-48405-1_25.

[10] Brier, E., Clavier, C., Olivier, F. Correlation power analysis with a leakage model. *Cryptographic Hardware and Embedded Systems – CHES 2004*, Cambridge, USA, August 2004, pp. 16-29. DOI: 10.1007/978-3-540-28632-5_2.

[11] Standaert, F.-X., Malkin, T.G., Yung, M. A unified framework for the analysis of side-channel key recovery attacks. *Advances in Cryptology – EUROCRYPT 2009*, Cologne, Germany, April 2009, pp. 443-461. DOI: 10.1007/978-3-642-01001-9_26.

[12] Chari, S., Rao, J.R., Rohatgi, P. Template attacks. *Cryptographic Hardware and Embedded Systems – CHES 2002*, Redwood Shores, USA, August 2002, pp. 13-28. DOI: 10.1007/3-540-36400-5_3.

[13] Hospodar, G., Gierlichs, B., De Mulder, E., et al. Machine learning in side-channel analysis: a first study. *Journal of Cryptographic Engineering*, vol. 1, no. 4, 2011, pp. 293-302. DOI: 10.1007/s13389-011-0023-x.

[14] Picek, S., Heuser, A., Jovic, A., et al. Side-channel analysis and machine learning: A practical perspective. *2017 International Joint Conference on Neural Networks (IJCNN)*, Anchorage, USA, May 2017, pp. 4095-4102. DOI: 10.1109/IJCNN.2017.7966373.

[15] Hettwer, B., Gehrer, S., Güneysu, T. Applications of machine learning techniques in side-channel attacks: A survey. *Journal of Cryptographic Engineering*, vol. 10, no. 2, 2020, pp. 135-162. DOI: 10.1007/s13389-019-00212-8.

[16] Benadjila, R., Prouff, E., Strullu, R., et al. Deep learning for side-channel analysis and introduction to ASCAD database. *Journal of Cryptographic Engineering*, vol. 10, no. 2, 2020, pp. 163-188. DOI: 10.1007/s13389-019-00220-8.

[17] Golder, A., Das, D., Danial, J., et al. Practical approaches toward deep-learning-based cross-device power side-channel attack. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 27, no. 12, 2019, pp. 2720-2733. DOI: 10.1109/TVLSI.2019.2926324.

[18] Cao, P., Zhang, C., Lu, X., et al. Cross-device profiled side-channel attack with unsupervised domain adaptation. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, vol. 2021, no. 4, 2021, pp. 27-56. DOI: 10.46586/tches.v2021.i4.27-56.

[19] Thorat, C., Inamdar, V., Jadhav, B. TED: A lightweight block cipher for IoT devices with side-channel attack resistance. *International Journal on Information Technologies and Security*, vol. 12, no. 2, 2020, pp. 83-96.

[20] Garipay, F., Uzgören, K., Kaya, İ. Side channel attack performance of correlation power analysis method in noise. *International Journal on Information Technologies and Security*, vol. 15, no. 1, 2023, pp. 101-111. DOI: 10.59035/UWCB7557.

[21] Nedyalkov, I., Georgiev, G. Kali Linux – a simple and effective way to study the level of cyber security and penetration testing of power electronic devices. *International Journal on Information Technologies and Security*, vol. 16, no. 2, 2024, pp. 103-114. DOI: 10.59035/JMFY4876.

[22] NewAE Technology Inc. *CW1101: ChipWhisperer-Nano datasheet*, NAE-CW1101, 2019. URL: https://media.newae.com/datasheets/NAE-CW1101_datasheet.pdf (Visited on 08.08.2026).

[23] STMicroelectronics. *STM32F030x4/x6/x8/xC datasheet*, DS9773 Rev 5, 2021. URL: https://www.st.com/resource/en/datasheet/stm32f030f4.pdf (Visited on 08.08.2026).

[24] *tiny-AES-c*, kokke, GitHub. URL: https://github.com/kokke/tiny-AES-c (Visited on 08.08.2026).

Information about the authors:

Gabriela Nedyalkova holds a bachelor's degree in computer science from the University of Library Studies and Information Technologies, Sofia, Bulgaria. Her areas of interest are hardware security, side-channel analysis, embedded systems, and applied cryptography.

Magdalena Garvanova is a full-time professor, PhD, Head of Information Systems and Technologies Department at the University of Library Studies and Information Technologies, Sofia, Bulgaria. Her areas of interest are data management and image processing.

Ivan Garvanov is a full-time professor, DSc, Director of Educational Policies at the University of Library Studies and Information Technologies, Sofia, Bulgaria. Professor at the Institute of Mathematics and Informatics, Bulgarian Academy of Sciences, Sofia, Bulgaria. His research interests are signal processing, data processing, and information systems and technologies.

Manuscript received on 21 August 2026