Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Efficient Ensemble Deep Learning for Full Key Recovery in Side-Channel Analysis of Unprotected Ascon Cipher

Septafiansyah Dwi Putra
septa@polinela.ac.id
Politeknik Negeri Lampung
Nurul Qomariyah
Politeknik Negeri Lampung
Agus Ambarwari
Politeknik Negeri Lampung
Adhan Efendi
Universitas Negeri Surabaya
Alde Alanda
Politeknik Negeri Padang
Heriansyah Heriansyah
Johannes Kepler University of Linz

Research Article

Keywords: Side-channel analysis, Ascon, Deep learning, Ensemble learning, Multilayer perceptron, Lightweight cryptography, Guessing entropy

Posted Date: November 4th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11194678/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Efficient Ensemble Deep Learning for Full Key Recovery in Side-Channel Analysis of Unprotected Ascon Cipher

Septafiansyah Dwi Putra[1*], Nurul Qomariyah[1], Agus Ambarwari[1], Adhan Efendi[2], Alde Alanda[3], Heriansyah[4]

[1*]Department of Information Technology, Politeknik Negeri Lampung, Bandar Lampung, Lampung, Indonesia.
[2]Department of Mechanical Engineering Education, Universitas Negeri Surabaya, Surabaya, East Java, Indonesia.
[3]Politeknik Negeri Padang, Padang, West Sumatra, Indonesia.
[4]Johannes Kepler University Linz, Linz, Austria.

*Corresponding author(s). E-mail(s): septa@polinela.ac.id;

Abstract

Ascon was selected by NIST as the standardized lightweight cryptography algorithm, which makes the leakage behaviour of its implementations a practical concern for Internet-of-Things deployments. Ensemble-based deep learning has recently been shown to break Ascon with very few traces, but reproductions are rarely reported with a statistically careful treatment of traces-to-disclosure (TTD), the number of attack traces after which the correct key byte is ranked first and stays there. TTD directly governs how fast the key can be found: fewer traces mean shorter acquisition time and a faster key search. It is right-censored whenever a key byte is not disclosed within the attack budget. This paper presents an efficient profiled attack with a reproducible methodology on the power traces of an unprotected software Ascon-128 implementation. The attack targets two algebraic outputs of the first-round S-box, $\boldsymbol{y_4}$ for the key word $\boldsymbol{K_0}$ and $\boldsymbol{y_2}$ for $\boldsymbol{K_1}$, and uses a per-byte ensemble of five multilayer perceptrons whose log-probabilities are averaged before key scoring. We contribute (i) a complete, byte-wise methodology that recovers all 16 key bytes, (ii) a censoring-aware evaluation using capped medians, a paired Wilcoxon test and a bootstrap confidence interval, and (iii) a contextual comparison with prior Ascon side-channel studies, distinguishing comparable from non-comparable trace counts. The role of AI is decisive for attack effectiveness: instead of assuming a Hamming-weight

leakage model and hand-picked points of interest, the neural networks learn the leakage of the S-box output directly from profiling traces, which turns a partly successful classical attack into a complete key recovery. Using 100,000 profiling traces and 30,000 attack traces, both a single MLP and the ensemble place the correct value of every byte at rank 0, whereas non-profiled CPA with 100,000 traces succeeds for only 2 of 16 bytes. Within a 2,000-trace budget the ensemble discloses 13 of 16 bytes versus 10 for a single MLP, and its capped median TTD is 423.5 versus 995.5 traces (Wilcoxon $\boldsymbol{p = 0.020}$; bootstrap 95% CI of the median difference $\boldsymbol{[10, 802]}$). We discuss why the $\boldsymbol{K_1}$ bytes are markedly easier than the $\boldsymbol{K_0}$ bytes and state the limits of this single-dataset, single-device study.

Keywords: Side-channel analysis, Ascon, Deep learning, Ensemble learning, Multilayer perceptron, Lightweight cryptography, Guessing entropy

1 Introduction

Lightweight authenticated encryption protects data on devices with tight energy, memory and latency budgets, such as sensor nodes and other Internet-of-Things equipment. Ascon, designed by Dobraunig, Eichlseder, Mendel and Schläffer [1], was chosen by NIST as its lightweight-cryptography standard, and its permutation-based, bitsliced design has attracted analysis both of the algorithm [2–5] and of its implementations [6–9]. Cryptanalytic strength, however, does not imply that a device leaks nothing. Since differential and correlation power analysis were introduced [10–12], physical measurements have remained one of the most effective ways to recover a secret key, as our own earlier power-analysis studies of DES, AES and BC3 implementations also showed [13–16], so the leakage of a standardised cipher has to be evaluated on real implementations and not only on paper.

Profiled side-channel analysis models the leakage of a clone device and then attacks the target [17, 18]. Machine learning entered this area early [19–21], and deep learning has since become the dominant profiled approach because it needs no hand-picked points of interest and tolerates misalignment [22–27]. Its results depend strongly on architecture, optimizer, loss and hyper-parameters [28–30], which makes single-network results variable and motivates methods that reduce this variance.

Ensembles are a natural answer: averaging several networks reduces variance and improves generalisation [31–34]. For Ascon in particular, Rezaeezade et al. showed that an ensemble of neural networks recovers an unprotected key with fewer than 100 traces and a protected one with fewer than 3,000 [35], and Weissbart and Picek analysed Ascon on a 32-bit microcontroller [36]. Prior Ascon ensemble studies primarily report the number of attack traces required for key recovery, but to our knowledge they do not evaluate trace-to-disclosure (TTD) under right-censoring or use paired statistical comparisons across key bytes.

This article does not introduce a new attack class, but it improves the ensemble-based profiled attack of Rezaeezade et al. in three respects, which we call *efficient*. First, the pipeline is made compact and byte-wise: one 772–200–200–256 MLP per member, five members per byte and log-probability fusion, so that profiling all 80

networks takes only 287 s on a single GPU. Second, the attack covers both key words of Ascon through two algebraic outputs of the first-round S-box, y_4 for K_0 and y_2 for K_1, with the K_1 stage conditioned on the recovered K_0, on an unprotected software Ascon-128 implementation. Third, the evaluation handles censored TTD correctly, so that the improvement is measured rather than asserted. Under an identical budget, the ensemble improves on a single network of the same pipeline, discloses 13 instead of 10 key bytes within 2,000 traces, and lowers the capped median TTD from 995.5 to 423.5 traces; we report this gain with statistics instead of a single headline number and do not claim it over the original ensemble attack, which we did not reproduce on the same data.

The study makes three methodological contributions. First, we give a complete, reproducible methodology for byte-wise full key recovery on Ascon-128 that covers both key words, with the K_1 stage conditioned on the recovered K_0. Second, we provide a censoring-aware evaluation using capped medians, paired Wilcoxon and sign tests and a bootstrap over bytes, and we show that the naive median over disclosed bytes is misleading. Third, we supply figures that make the algorithm and the leakage targets explicit (Figs. 1 and 3). The rest of the paper is organised as follows: Section 2 reviews related work, Section 3 presents the methodology, Section 4 the results, and Sections 5–7 discuss, conclude and state limitations.

2 Background and related work

2.1 The Ascon-128 algorithm

Ascon-128 keeps a 320-bit state of five 64-bit words $x_0, \ldots, x_4$ and absorbs data at rate $r = 64$ bits [1]. Fig. 1 summarizes the mode. The state is initialised as $IV \,\|\, K \,\|\, N$, the permutation p^a with $a = 12$ rounds is applied, and $0^{192}\|K$ is added. Associated-data blocks are absorbed with p^b, $b = 6$, followed by the domain-separation bit $0^{319}\|1$. Plaintext blocks are XORed into the rate and produce ciphertext blocks; finalization adds the key, applies p^a and adds the key again to obtain the tag.

A round $p = p_L \circ p_S \circ p_C$ adds a round constant to x_2, applies a 5-bit S-box to every one of the 64 columns in bitsliced form, and finally diffuses each word by $x_i \leftarrow x_i \oplus (x_i \ggg r_{i,1}) \oplus (x_i \ggg r_{i,2})$ with rotations $(19, 28)$, $(61, 39)$, $(1, 6)$, $(10, 17)$ and $(7, 41)$ for $x_0, \ldots, x_4$. In the bitsliced form the S-box outputs are

$$
\begin{aligned}
y_0 &= x_4x_1 \oplus x_3 \oplus x_2x_1 \oplus x_2 \oplus x_1x_0 \oplus x_1 \oplus x_0,\\
y_1 &= x_4 \oplus x_3x_2 \oplus x_3x_1 \oplus x_3 \oplus x_2x_1 \oplus x_2 \oplus x_1 \oplus x_0,\\
y_2 &= x_4x_3 \oplus x_4 \oplus x_2 \oplus x_1 \oplus 1,\\
y_3 &= x_4x_0 \oplus x_4 \oplus x_3x_0 \oplus x_3 \oplus x_2 \oplus x_1 \oplus x_0,\\
y_4 &= x_4x_1 \oplus x_4 \oplus x_3 \oplus x_1x_0 \oplus x_1. \qquad (1)
\end{aligned}
$$

Because the S-box inputs at the first round are (IV, K_0, K_1, N_0, N_1), its outputs mix known nonce bits with unknown key bits, which is the attack surface described below.

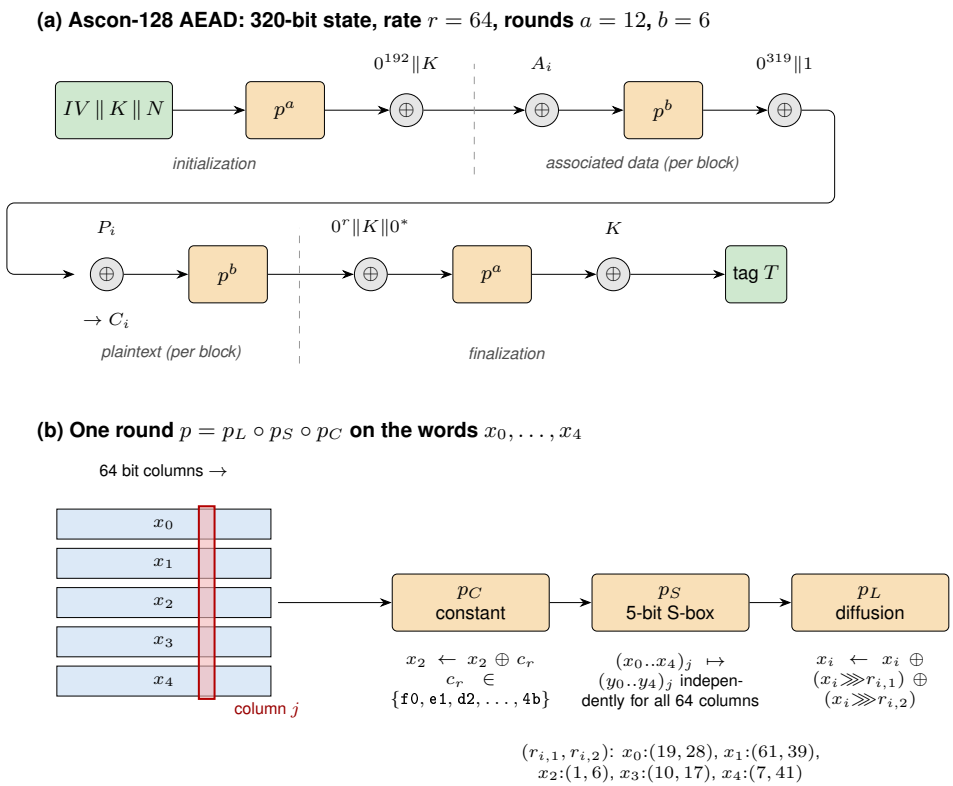


Fig. 1 The Ascon-128 AEAD algorithm. (a) Mode of operation: initialization, associated-data absorption, plaintext/ciphertext processing and finalization with $a = 12$ and $b = 6$ rounds. (b) One permutation round: constant addition p_C, the bitsliced 5-bit S-box layer p_S and the linear diffusion layer p_L. Redrawn by the authors from the specification [1].

2.2 Side-channel analysis and deep learning

Differential and correlation power analysis exploit the dependence of the power consumption on an intermediate value [10, 12], and template attacks profile that dependence with a clone device [17]. Mutual information analysis [37] and the unified framework of Standaert et al. [18] formalise how key-recovery efficiency is measured, which is the origin of guessing entropy (GE). Deep learning removes the need for hand-picked points of interest and tolerates misalignment [22, 23], and public benchmarks such as ASCAD [24] have supported systematic comparisons of architectures, optimizers and losses [26, 28–30, 38, 39]. Recent Springer work has proposed siamese and metric-learning training [40, 41] and interpretability through occlusion [42]. Countermeasures change the picture [43], and transfer or meta-learning address portability across devices [44, 45]. Non-profiled variants avoid a clone device [46–48]. Surveys [21, 25] and studies of lightweight ciphers for IoT [49] complete the context.

Table 1 Related side-channel work on Ascon. "n.c." means that the work does not report a trace count that is comparable with this study.

Work	Publisher	Setting	Reported effort
Rezaeezade et al. [35]	Springer (ACNS)	Unprotected; protected	$<$ 100 traces; $<$ 3,000 traces (ensemble)
Weissbart and Picek [36]	ePrint	32-bit MCU, protected and unprotected	CPA about 8,000; DL about 1,000 (abstract-level)
Groß et al. [6]	IEEE	Hardware implementation	n.c.
Olešák et al. [7]	IEEE	Side-channel comparison	n.c.
Kandi et al. [8]	IEEE	Protected hardware	n.c.
Eliasi et al. [9]	IEEE	Hardware vs. Gaston	n.c.
This work	—	Unprotected software, TRS traces	Rank 0 for 16/16 bytes with 30,000 traces; capped median TTD 423.5

2.3 Earlier power-analysis work by the author

The first author has previously studied power analysis of classical and lightweight-era ciphers: attacks on DES [13], differential power analysis of BC3 [14], a comparison across AES, DES and BC3 [15], first-round and last-round attacks on AES devices [16], correlation power analysis of AES on microcontrollers [50] and of masked AES [51], and Hamming-weight-model attacks on IoT encryption devices [52]. On the defensive side, this line of work proposed DPA countermeasures based on a knowledge-growing system [53] and cognitive artificial intelligence [54]. These studies used the Hamming-weight and correlation-based (non-profiled) tradition that we take as the CPA baseline in Section 3; the present article extends this line to a profiled deep-learning attack on Ascon. The claims here are limited to what those papers' titles state.

2.4 Side-channel work on Ascon

Table 1 contrasts the works most closely related to ours. The claims in the table are limited to what the cited titles and abstracts state; where a work does not report a comparable trace count we say so rather than infer one. Weissbart and Picek analyse Ascon on a 32-bit microcontroller [36], and Rezaeezade et al. introduce the ensemble attack [35]. The hardware-oriented evaluations [6–9] address implementations and countermeasures and therefore do not report trace counts that could be compared with a software profiled attack.

Research gap.

Prior ASCON ensemble studies primarily report the number of attack traces required for key recovery, but to our knowledge they do not evaluate trace-to-disclosure under right-censoring or use paired statistical comparisons across key bytes. Three gaps follow. First, a single trace count hides bytes that are never disclosed within the attack budget, so medians computed over disclosed bytes only are biased towards the easy bytes. Second, it is unclear whether an ensemble beats one of its own members consistently, byte by byte, or only on average. Third, the two key words of ASCON are attacked through different S-box outputs, yet their difficulty is rarely contrasted. We close these gaps with a censoring-aware TTD, paired tests over the 16 bytes, and a separate analysis of K_0 and K_1.

3 Methodology

The methodology is a profiled attack in six steps (Fig. 2). The 200,000-trace dataset is split into a profiling half with random keys and an attack half with one fixed key, and all traces are standardised with profiling statistics only. For every key byte, the known key and nonce of each profiling trace give the identity-model label of the first-round S-box output, y_4 for K_0 and y_2 for K_1. An ensemble of five MLPs is trained on these labels and fuses its outputs by the mean of the log-softmax values. In the attack phase, the fused log-probabilities of the observed traces are summed for each of the 256 key hypotheses, and the hypothesis with the highest score is the recovered byte; for K_1 the hypotheses use the recovered $\hat{K}_0$. Finally, guessing entropy, TTD and final rank quantify the attack, and a non-profiled CPA on the same traces serves as a reference.

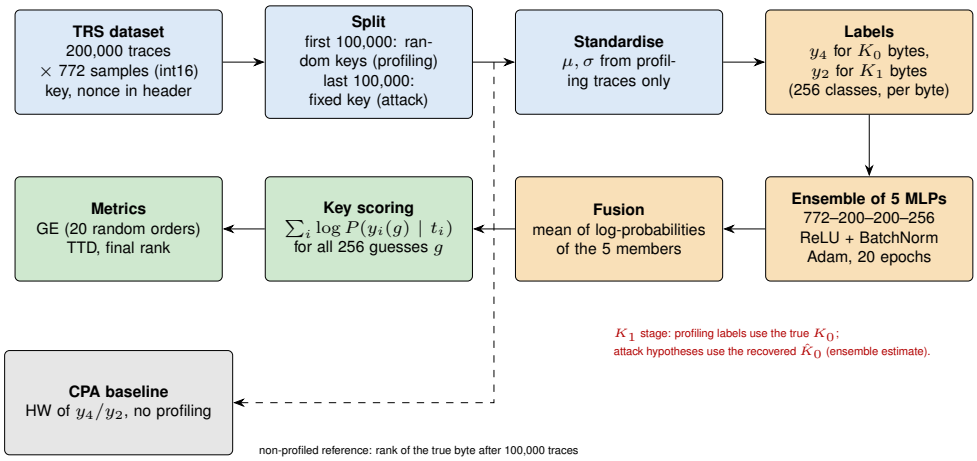


Fig. 2 Methodology: dataset split, standardisation with profiling statistics, byte-wise labels, an ensemble of five MLPs per byte, log-probability fusion, key scoring and metrics. A non-profiled CPA baseline is computed on the same traces.

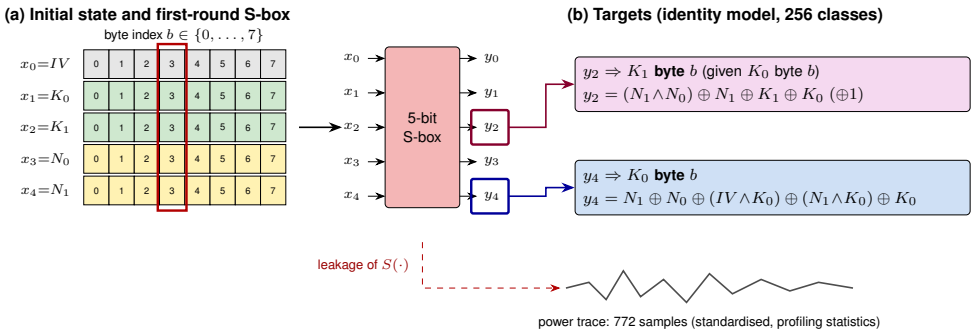


Fig. 3 Leakage targets. (a) The initial state and the bitsliced S-box for byte column b. (b) The outputs y_2 and y_4 are the identity-model targets for K_1 and K_0, respectively.

3.1 Dataset

The dataset is a TRS file of 200,000 power traces with 772 samples each (stored as 16-bit integers), where the key and the nonce of each trace are stored in the header. The first 100,000 traces use random keys and form the profiling set. The last 100,000 traces use one fixed key (`8552f14d974b863615ff78c469cc3622`) and form the attack set, of which the first 30,000 are used for scoring and GE. The traces come from the public Zenodo record of Rezaeezade et al. [55], file `ascon_cw_unprotected.trs` inside `Datasets.zip`, which we use as provided.

3.2 Leakage model

The first-round S-box inputs are $(x_0,\ldots,x_4) = (IV, K_0, K_1, N_0, N_1)$, so for byte index $b \in \{0,\ldots,7\}$ of the 64-bit words, Eq. (1) gives (Fig. 3)

$$y_4[b] = N_1[b] \oplus N_0[b] \oplus (IV[b] \wedge K_0[b]) \oplus (N_1[b] \wedge K_0[b]) \oplus K_0[b], \tag{2}$$

$$y_2[b] = (N_1[b] \wedge N_0[b]) \oplus N_1[b] \oplus K_1[b] \oplus K_0[b] \quad (\oplus\,\text{const}). \tag{3}$$

The value y_4 depends only on K_0 and recovers key bytes 0–7. The value y_2 depends on K_1 given K_0 and recovers bytes 8–15. We use the identity model, so each target is a class in $\{0,\ldots,255\}$. The labels are formed from the byte-wise y_2 expression of Eq. (3) without the constant, which only permutes the classes and does not change the attack.

3.3 Preprocessing and CPA baseline

Traces are standardised with the mean and the standard deviation of the profiling set only, so no information from the attack set enters the model. As a non-profiled reference we compute CPA with the Hamming weight of y_4 (for K_0) and y_2 (for K_1) on all 100,000 attack traces, and score each guess by the maximum absolute correlation over the 772 samples. For K_1 the CPA baseline is granted the true K_0, which favours the baseline.

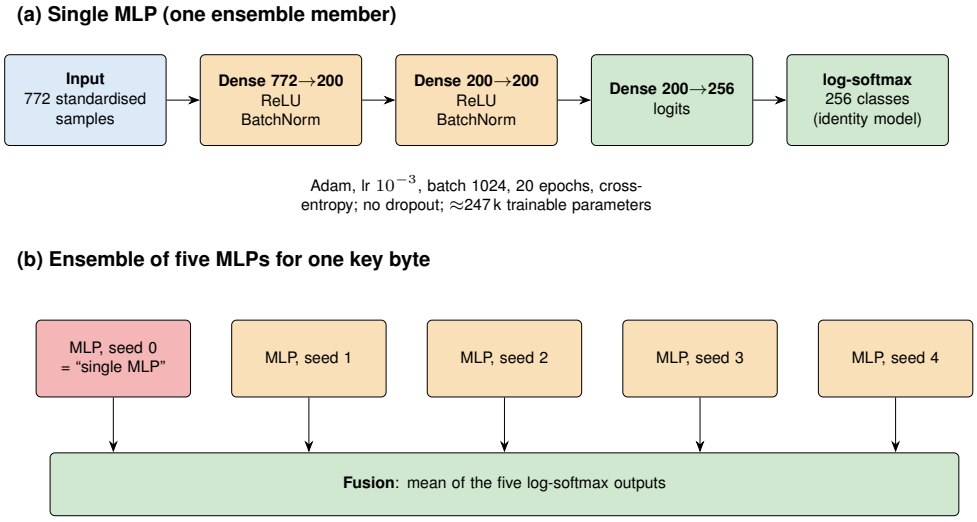


Fig. 4 Model used in this work. (a) The single MLP: 772–200–200–256 with ReLU and batch normalisation. (b) The ensemble of five MLPs (seeds 0–4) fused by the mean of log-softmax outputs. The member with seed 0 is the "single MLP" baseline.

3.4 MLP and ensemble

The *single MLP* is a plain feed-forward network (Fig. 4a). It takes the 772 standardised samples of a trace, passes them through two fully connected hidden layers of 200 units, each followed by a ReLU activation and batch normalisation, and ends in a linear layer with 256 outputs, one per class of the identity model, which is converted to log-probabilities by a log-softmax. It has no convolutional layers and no dropout and about 247,000 trainable parameters. It is trained with Adam (learning rate 10^{-3}), cross-entropy loss, batch size 1,024 and 20 epochs on the 100,000 profiling traces, separately for each key byte.

The *ensemble* (Fig. 4b) trains five such networks per byte with seeds $0, \ldots, 4$ on the same data and fuses them by the mean of their log-softmax outputs, i.e. the logarithm of the geometric mean of the members' probabilities. Table 2 lists the settings. The "single MLP" used for comparison is the first member (seed 0) of the same ensemble, so both methods share the same initial network, training data and budget and the comparison is paired.

3.5 Attack, GE and TTD

For a byte with hypothesis $g \in \{0, \ldots, 255\}$ and attack traces $t_1, \ldots, t_q$, the score is $s(g) = \sum_{i=1}^{q} \log P(y_i(g) \mid t_i)$ where P is the fused network output. For K_1 bytes, $y_i(g)$ uses the recovered $\hat{K}_0$ obtained from the ensemble attack on K_0 (in profiling, the true K_0 is used to form the labels). GE is the mean rank of the true byte over 20 random orderings of the attack traces. TTD is the smallest q for which GE remains 0 up to

Table 2 Model and training settings.

Item	Value
Architecture	772–200–200–256; ReLU + BatchNorm after each hidden layer
Optimizer	Adam, learning rate 10^{-3}
Loss; batch size; epochs	Cross-entropy; 1024; 20
Ensemble size	5 members per byte (seeds 0–4), 80 networks in total
Profiling / attack traces	100,000 / 30,000
GE	20 random trace orders, first 2,000 traces
Profiling time	287 s for all 80 networks
Hardware	NVIDIA A100 (Google Colab)

Algorithm 1 Byte-wise ensemble attack on Ascon-128

Require: Profiling traces with known keys, attack traces $\{t_i\}$ with nonces, ensemble size $M = 5$

Standardise all traces with profiling mean and standard deviation
for $b = 0$ **to** 15 **do**
  Form labels y with Eq. (2) ($b < 8$) or Eq. (3) ($b \geq 8$, true K_0)
  Train M MLPs with seeds $0, \ldots, M-1$ on (profiling traces, y)
  $\ell_i \leftarrow \frac{1}{M} \sum_m \log \mathrm{softmax}_m(t_i)$ ▷ fusion
  $s(g) \leftarrow \sum_i \ell_i[y_i(g)]$ for all g (use $\hat{K}_0$ if $b \geq 8$)
  $\hat{K}[b] \leftarrow \arg\max_g s(g)$; compute GE curve and TTD
end for
return $\hat{K}$, GE curves, TTD per byte

the budget of 2,000 traces. If GE never stays at 0, TTD is right-censored at 2,000. Algorithm 1 summarizes the procedure.

3.6 Statistical protocol

Because TTD is censored at 2,000, the median over disclosed bytes only would exclude the hardest bytes and understate the cost. We therefore report the median with censored bytes set to the cap, the number of bytes disclosed, and paired tests over the 16 bytes: an exact Wilcoxon signed-rank test, a sign test, a bootstrap over bytes for the median difference, and the geometric-mean ratio of capped TTDs (ensemble over single). Because all values equal to the cap are ties, these statistics are conservative for a real improvement.

4 Results

4.1 Profiled attacks versus CPA

Non-profiled CPA with 100,000 traces places the true byte at rank 0 for only two bytes (bytes 8 and 13), with ranks up to 147 for others (Fig. 5). We use the full 30,000

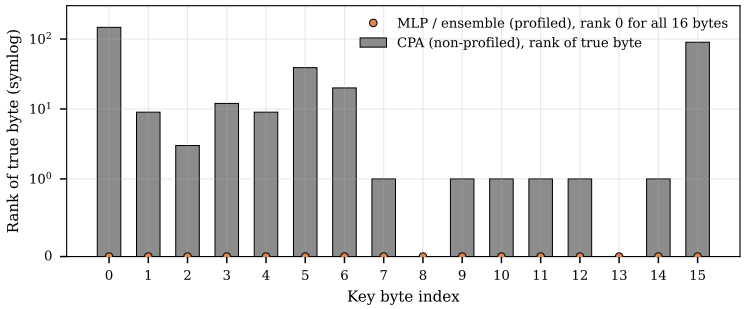


Fig. 5 Rank of the true key byte: CPA (100,000 traces) versus the profiled attack (rank 0 for all bytes). The axis is symmetric-logarithmic.

Table 3 Per-byte results. TTD is in traces; ">2000" is censored. CPA rank uses 100,000 traces (0 = recovered).

Byte	Word	True	CPA rank	Single TTD	Ensemble TTD
0	K_0	85	147	>2000	1864
1	K_0	52	9	>2000	1271
2	K_0	f1	3	>2000	>2000
3	K_0	4d	12	156	180
4	K_0	97	9	1327	611
5	K_0	4b	39	>2000	1600
6	K_0	86	20	>2000	1125
7	K_0	36	1	664	236
8	K_1	15	0	93	90
9	K_1	ff	1	>2000	>2000
10	K_1	78	1	190	68
11	K_1	c4	1	23	28
12	K_1	69	1	221	170
13	K_1	cc	0	1458	>2000
14	K_1	36	1	311	120
15	K_1	22	90	151	69

attack traces to assess final key recovery, while TTD is evaluated only over the first 2,000 traces to quantify early disclosure.

4.2 Traces to disclosure per byte

Table 3 lists the per-byte results and Fig. 6 plots them. Three bytes (2, 9 and 13) are not disclosed by the ensemble within 2,000 traces, and six by the single MLP (0, 1, 2, 5, 6, 9). Byte 13 is the only one where the single MLP is better (1,458 versus censored). Note that the final rank is 0 for every byte, so "not disclosed within 2,000 traces" means that more traces are needed for a stable rank 0, not that the byte is unrecoverable.

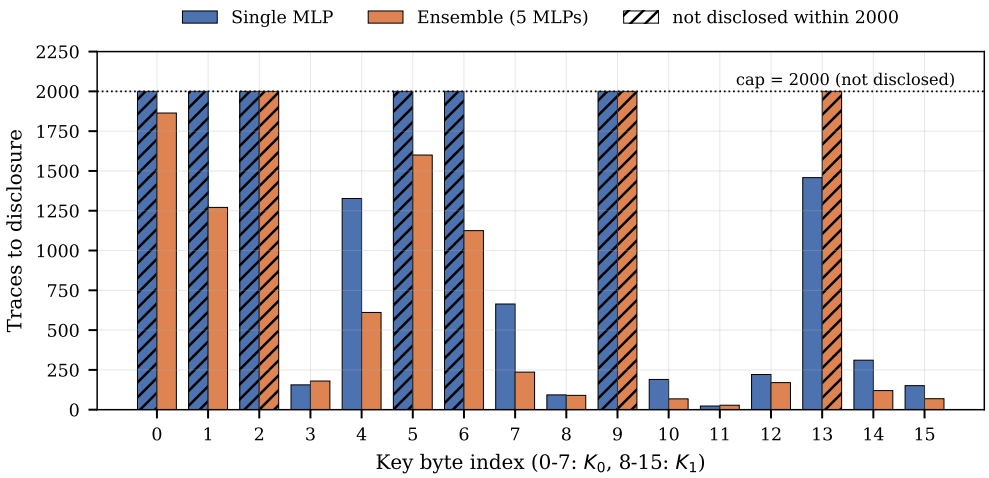


Fig. 6 TTD per key byte for the single MLP and the ensemble. Hatched bars are bytes not disclosed within 2,000 traces (censored).

Table 4 Single MLP versus ensemble (16 bytes, cap = 2,000 traces).

Metric	Single MLP	Ensemble
Bytes at rank 0 (30,000 traces)	16	16
Bytes disclosed within 2,000 traces	10	13
Naive median TTD (disclosed only)	205.5	180.0
Capped median TTD	995.5	423.5
Capped median, K_0 bytes / K_1 bytes	2000 / 205.5	1198 / 105
Networks trained	16	80
Wilcoxon (exact), sign test	$p = 0.020$, $p = 0.057$	
Bootstrap median difference (95% CI)	572 $[10, 802]$	
Geometric-mean ratio (ens./single)	0.71	

4.3 Censoring-aware comparison

Table 4 and Fig. 7 summarise the comparison. The naive median over disclosed bytes is 205.5 (single, 10 bytes) versus 180.0 (ensemble, 13 bytes), but this compares different subsets. With the cap applied the medians are 995.5 and 423.5. The paired Wilcoxon test gives $p = 0.020$, the sign test $p = 0.057$, the bootstrap 95% confidence interval of the median difference is $[10, 802]$ (point estimate 572) and the geometric-mean ratio of capped TTDs is 0.71. These results reinforce the finding of lower TTD under ensemble fusion.

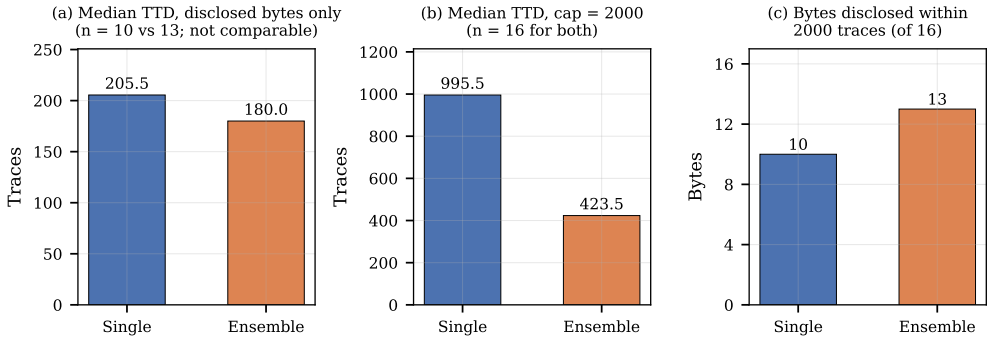


Fig. 7 Summary: naive median (disclosed bytes only), capped median and number of disclosed bytes.

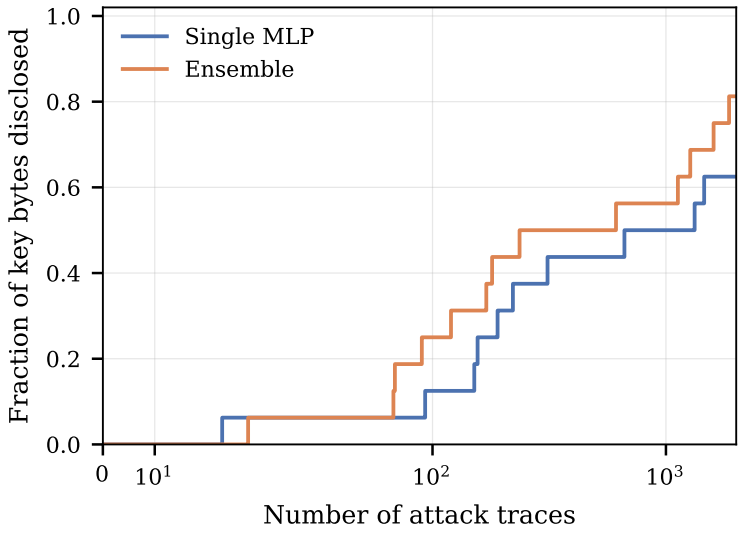


Fig. 8 Fraction of the 16 bytes disclosed as a function of the number of attack traces.

4.4 Guessing entropy

Fig. 9 shows GE for representative bytes with the ensemble and Fig. 10 compares single and ensemble on byte 2. Bytes that are quickly disclosed (for example 8, 11 and 15) reach a stable GE of 0 within roughly 100 traces (TTD 90, 28 and 69), whereas byte 2 stays near GE 0 but does not stay at exactly 0 within the budget.

4.5 Comparison with the literature

Direct comparison with [35] is not fair: their unprotected setting needs fewer than 100 traces, which is about an order of magnitude below our median, but they use a different dataset, multi-task learning and a different network family, and we did not reproduce their setting. Weissbart and Picek report much larger CPA and DL

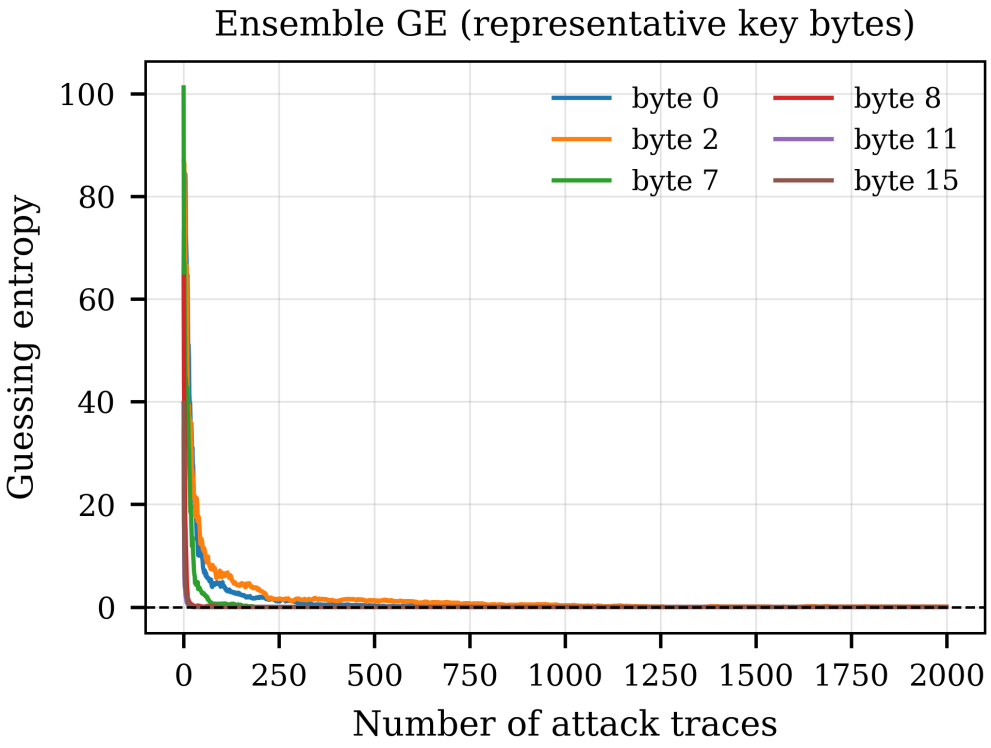


Fig. 9 GE of the ensemble for representative key bytes 0, 2, 7, 8, 11 and 15.

effort on a microcontroller [36], again on a different device. Our contribution to the comparison is therefore the observation that, on this dataset, an ensemble with plain MLPs recovers every byte at rank 0 with 30,000 traces and that the ensemble reduces the capped median TTD relative to its own first member. The IEEE works listed in Table 1 address hardware and do not provide numbers that can be compared. Non-profiled attacks [46, 47] avoid the profiling phase; in our data the non-profiled CPA baseline is clearly weaker than the profiled attack, but we did not evaluate non-profiled deep learning.

5 Discussion

The capped median for the ensemble is 105 traces for the K_1 bytes and 1,198 for K_0. A plausible explanation is algebraic. In Eq. (3) the key bits enter linearly ($K_1 \oplus K_0$), which a Hamming-weight or identity leakage reveals directly, whereas in Eq. (2) the key interacts with nonce bits through AND terms. We treat this as a hypothesis: it is consistent with CPA ranks (0 or 1 for seven of the eight K_1 bytes, the exception being byte 15 with rank 90, versus up to 147 for K_0), but we did not run experiments that isolate the effect, and the K_1 stage also uses the recovered $\hat{K}_0$.

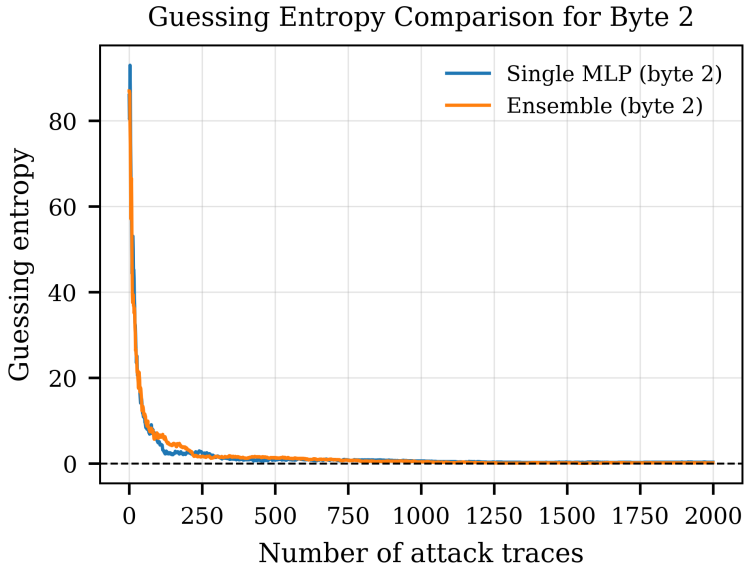


Fig. 10 GE of byte 2: single MLP versus ensemble.

Averaging log-probabilities reduces the variance of individual MLPs [31, 32]. The gain is consistent with this view but the effect is not uniform: byte 13 is worse. With 16 bytes and 20 random orderings, per-byte differences of this size may be noise. The ensemble multiplies the profiling cost by five (287 s in total here), but the attack cost is negligible.

Full key recovery of an unprotected implementation is possible with modest resources, which supports the case for masking or shuffling [43]. The result concerns an unprotected implementation and must not be read as a statement about protected Ascon devices.

6 Conclusion

This article set out to ask whether a compact, byte-wise ensemble of MLPs can recover the full key of an unprotected software Ascon-128 implementation, and whether such an ensemble improves on a single network of the same pipeline once traces-to-disclosure are measured correctly under right-censoring. Both questions are answered by the results. Both a single MLP and the five-member ensemble place every one of the 16 key bytes at rank 0 with 30,000 attack traces, so the full key is recovered end to end from the two algebraic S-box outputs y_4 and y_2, whereas non-profiled CPA with 100,000 traces recovers only 2 of 16 bytes. This confirms that a profiled deep-learning attack, not the classical Hamming-weight baseline, is what makes full key recovery practical on this dataset.

On the second question, the ensemble does improve over its own first member, and the improvement is measurable rather than asserted. Within the 2,000-trace budget used for TTD, the ensemble discloses 13 of 16 bytes against 10 for the single MLP,

and its capped median TTD is 423.5 traces against 995.5 (paired Wilcoxon $p = 0.020$; bootstrap 95% CI of the median difference $[10, 802]$). The gain is therefore real at the level of the whole 16-byte set, even though it is not uniform byte by byte and the underlying evidence remains moderate. A secondary finding, that the K_1 bytes are disclosed far earlier than the K_0 bytes, points to the algebraic structure of the two S-box targets as the likely explanation, although we did not isolate this effect experimentally.

Taken together, these results answer the research problem for the setting studied here: an efficient ensemble pipeline, evaluated with censoring-aware statistics, both recovers the full Ascon-128 key and discloses it faster than a single network under an identical budget. We stress that this answer is scoped to one public dataset, one fixed key and one device, and Section 7 states what would be needed to generalise it.

7 Limitations and future work

The study uses a single dataset, a single fixed key and a single device, and the ensemble comparison is limited to 16 paired bytes, so the statistical evidence is moderate. The single MLP is the first ensemble member rather than an independently tuned baseline, and no hyper-parameter search was done, so the pipeline is better described as efficient than as optimized. The K_1 stage depends on the correctly recovered K_0, so errors would propagate. CPA baselines used the same traces but no pre-processing beyond standardisation. Protected implementations, cross-device portability [44, 45], CNN architectures [23], focal loss [29] and automated tuning [30] are natural next steps. An untested idea, multi-coordinate leakage fusion over several S-box output bits, is left as future work and is not evaluated here.

Acknowledgements. The authors thank the providers of the dataset and the Cyber Physical System Laboratory of Politeknik Negeri Lampung.

Declarations

- **Competing interests:** The authors declare no competing interests.
- **Funding:** The authors declare that no funds, grants, or other support were received during the preparation of this manuscript.
- **Clinical trial number:** Not applicable. This study did not involve a clinical trial.
- **Ethics approval and consent to participate:** Not applicable. The study analyses power traces of a public research benchmark of an unprotected implementation for defensive leakage evaluation.
- **Data availability:** The dataset is public: Zenodo record 15614359, file `ascon_cw_unprotected.trs` [55], DOI 10.5281/zenodo.15614359. No new data were collected.
- **Code availability:** The source code will be shared after the manuscript is accepted at https://github.com/Septafiansyah/ASCON.git.
- **Use of AI tools:** AI-based tools were used to improve the language and writing of the manuscript. The authors reviewed and edited the content and take full responsibility for it.

References

[1] Dobraunig, C., Eichlseder, M., Mendel, F., Schläffer, M.: Ascon v1.2: Lightweight authenticated encryption and hashing. Journal of Cryptology **34**(3), 33 (2021) https://doi.org/10.1007/s00145-021-09398-9

[2] Li, Y., Zhang, G., Wang, W., Wang, M.: Cryptanalysis of round-reduced ASCON. Science China Information Sciences (2016) https://doi.org/10.1007/s11432-016-0283-3

[3] Baek, S., Kim, G., Kim, J.: Preimage attacks on reduced-round Ascon-Xof. Designs, Codes and Cryptography, 2197–2217 (2024) https://doi.org/10.1007/s10623-024-01383-0

[4] Degré, M., Derbez, P., Lahaye, L., Schrottenloher, A.: New models for the cryptanalysis of ASCON. Designs, Codes and Cryptography, 2055–2072 (2025) https://doi.org/10.1007/s10623-025-01572-5

[5] Mandal, K., Saha, D., Sarkar, S., Todo, Y.: Sycon: A new milestone in designing ASCON-like permutations. Journal of Cryptographic Engineering **11**, 305–327 (2021) https://doi.org/10.1007/s13389-021-00272-9

[6] Groß, H., Wenger, E., Dobraunig, C., Ehrenhöfer, C.: Suit up! Made-to-Measure hardware implementations of ASCON. In: 2015 Euromicro Conference on Digital System Design (DSD), pp. 645–652 (2015). https://doi.org/10.1109/DSD.2015.14

[7] Olekšák, M., Miškovský, V.: Is ASCON the best choice regarding the side-channel analysis? In: 2023 12th Mediterranean Conference on Embedded Computing (MECO), pp. 1–5 (2023). https://doi.org/10.1109/MECO58584.2023.10154974

[8] Kandi, A., Baksi, A., Gan, P., Guilley, S., Gerlich, T., Breier, J., Chattopadhyay, A., Shrivastwa, R.R., Martinásek, Z., Bhasin, S.: Side-channel and fault resistant ASCON implementation: A detailed hardware evaluation. In: 2024 IEEE Computer Society Annual Symposium on VLSI (ISVLSI), pp. 307–312 (2024). https://doi.org/10.1109/ISVLSI61997.2024.00063

[9] Eliasi, P.A., Mella, S., Batina, L.: Assessing Gaston: Side-channel security and hardware cost comparison with Ascon-p. In: 2025 IEEE Computer Society Annual Symposium on VLSI (ISVLSI), pp. 1–6 (2025). https://doi.org/10.1109/ISVLSI65124.2025.11130245

[10] Kocher, P., Jaffe, J., Jun, B.: Differential power analysis. In: Advances in Cryptology – CRYPTO'99. Lecture Notes in Computer Science, pp. 388–397 (1999). https://doi.org/10.1007/3-540-48405-1_25

[11] Kocher, P., Jaffe, J., Jun, B., Rohatgi, P.: Introduction to differential power analysis. Journal of Cryptographic Engineering **1**(1), 5–27 (2011) https://doi.org/10.1007/s13389-011-0006-y

[12] Brier, E., Clavier, C., Olivier, F.: Correlation power analysis with a leakage model. In: Cryptographic Hardware and Embedded Systems – CHES 2004. Lecture Notes in Computer Science, pp. 16–29 (2004). https://doi.org/10.1007/978-3-540-28632-5_2

[13] Putra, S.D., Ahmad, A.S., Sutikno, S.: Power analysis attack on implementation of DES. In: 2016 International Conference on Information Technology Systems and Innovation (ICITSI) (2016). https://doi.org/10.1109/icitsi.2016.7858246

[14] Putra, S.D., Yudhiprawira, M., Kurniawan, Y., Sutikno, S., Ahmad, A.S.: Security analysis of BC3 algorithm for differential power analysis attack. In: 2017 International Symposium on Electronics and Smart Devices (ISESD) (2017). https://doi.org/10.1109/isesd.2017.8253361

[15] Putra, S.D., Yudhiprawira, M., Sutikno, S., Kurniawan, Y., Ahmad, A.S.: Power analysis attack against encryption devices: a comprehensive analysis of AES, DES, and BC3. TELKOMNIKA (Telecommunication Computing Electronics and Control) (2019) https://doi.org/10.12928/telkomnika.v17i3.9384

[16] Putra, S.D., Sumari, A.D.W., Asrowardi, I., Subyantoro, E., Zagi, L.M.: First-round and last-round power analysis attack against AES devices. In: 2020 International Conference on Information Technology Systems and Innovation (ICITSI), pp. 410–415 (2020). https://doi.org/10.1109/icitsi50517.2020.9264976

[17] Chari, S., Rao, J.R., Rohatgi, P.: Template attacks. In: Cryptographic Hardware and Embedded Systems – CHES 2002. Lecture Notes in Computer Science, pp. 13–28 (2003). https://doi.org/10.1007/3-540-36400-5_3

[18] Standaert, F.-X., Malkin, T.G., Yung, M.: A unified framework for the analysis of side-channel key recovery attacks. In: Advances in Cryptology – EUROCRYPT 2009. Lecture Notes in Computer Science, pp. 443–461 (2009). https://doi.org/10.1007/978-3-642-01001-9_26

[19] Hospodar, G., Gierlichs, B., De Mulder, E., Verbauwhede, I., Vandewalle, J.: Machine learning in side-channel analysis: A first study. Journal of Cryptographic Engineering **1**, 293–302 (2011) https://doi.org/10.1007/s13389-011-0023-x

[20] Martinasek, Z., Hajny, J., Malina, L.: Optimization of power analysis using neural network. In: Smart Card Research and Advanced Applications (CARDIS 2013). Lecture Notes in Computer Science, pp. 94–107 (2014). https://doi.org/10.1007/978-3-319-08302-5_7

[21] Picek, S., Heuser, A., Jovic, A., Ludwig, S.A., Guilley, S., Jakobovic, D., Mentens,

N.: Side-channel analysis and machine learning: A practical perspective. In: 2017 International Joint Conference on Neural Networks (IJCNN), pp. 4095–4102 (2017). https://doi.org/10.1109/IJCNN.2017.7966373

[22] Maghrebi, H., Portigliatti, T., Prouff, E.: Breaking cryptographic implementations using deep learning techniques. In: Security, Privacy, and Applied Cryptography Engineering (SPACE 2016). Lecture Notes in Computer Science, pp. 3–26 (2016). https://doi.org/10.1007/978-3-319-49445-6_1

[23] Cagli, E., Dumas, C., Prouff, E.: Convolutional neural networks with data augmentation against jitter-based countermeasures. In: Cryptographic Hardware and Embedded Systems – CHES 2017. Lecture Notes in Computer Science, pp. 45–68 (2017). https://doi.org/10.1007/978-3-319-66787-4_3

[24] Benadjila, R., Prouff, E., Strullu, R., Cagli, E., Dumas, C.: Deep learning for side-channel analysis and introduction to ASCAD database. Journal of Cryptographic Engineering **10**, 163–188 (2020) https://doi.org/10.1007/s13389-019-00220-8

[25] Hettwer, B., Gehrer, S., Güneysu, T.: Applications of machine learning techniques in side-channel attacks: A survey. Journal of Cryptographic Engineering **10**, 135–162 (2020) https://doi.org/10.1007/s13389-019-00212-8

[26] Wu, L., Perin, G., Picek, S.: On the evaluation of deep learning-based side-channel analysis. In: Constructive Side-Channel Analysis and Secure Design (COSADE 2022). Lecture Notes in Computer Science, pp. 49–71 (2022). https://doi.org/10.1007/978-3-030-99766-3_3

[27] Ou, Y., Li, L.: Side-channel analysis attacks based on deep learning network. Frontiers of Computer Science **16**(2) (2022) https://doi.org/10.1007/s11704-020-0209-4

[28] Perin, G., Picek, S.: On the influence of optimizers in deep learning-based side-channel analysis. In: Selected Areas in Cryptography (SAC 2020). Lecture Notes in Computer Science, pp. 615–636 (2021). https://doi.org/10.1007/978-3-030-81652-0_24

[29] Kerkhof, M., Wu, L., Perin, G., Picek, S.: Focus is key to success: A focal loss function for deep learning-based side-channel analysis. In: Constructive Side-Channel Analysis and Secure Design (COSADE 2022). Lecture Notes in Computer Science, pp. 29–48 (2022). https://doi.org/10.1007/978-3-030-99766-3_2

[30] Wu, L., Perin, G., Picek, S.: I choose you: Automated hyperparameter tuning for deep learning-based side-channel analysis. IEEE Transactions on Emerging Topics in Computing **12**(2), 546–557 (2024) https://doi.org/10.1109/TETC.2022.3218372

[31] Breiman, L.: Bagging predictors. Machine Learning **24**(2), 123–140 (1996) https:

//doi.org/10.1007/BF00058655

[32] Hansen, L.K., Salamon, P.: Neural network ensembles. IEEE Transactions on Pattern Analysis and Machine Intelligence **12**(10), 993–1001 (1990) https://doi.org/10.1109/34.58871

[33] Igelnik, B., Pao, Y.-H., LeClair, S.R., Shen, C.Y.: The ensemble approach to neural-network learning and generalization. IEEE Transactions on Neural Networks **10**(1), 19–30 (1999) https://doi.org/10.1109/72.737490

[34] Yang, J., Zeng, X., Zhong, S., Wu, S.: Effective neural network ensemble approach for improving generalization performance. IEEE Transactions on Neural Networks and Learning Systems **24**(6), 878–887 (2013) https://doi.org/10.1109/TNNLS.2013.2246578

[35] Rezaeezade, A., Basurto-Becerra, A., Weissbart, L., Perin, G.: One for all, all for Ascon: Ensemble-based deep learning side-channel analysis. In: Applied Cryptography and Network Security Workshops (ACNS 2024). Lecture Notes in Computer Science, pp. 139–157 (2024). https://doi.org/10.1007/978-3-031-61486-6_9

[36] Weissbart, L., Picek, S.: Lightweight but Not Easy: Side-channel Analysis of the Ascon Authenticated Cipher on a 32-bit Microcontroller. Cryptology ePrint Archive, Paper 2023/1598. https://eprint.iacr.org/2023/1598 (2023)

[37] Gierlichs, B., Batina, L., Tuyls, P., Preneel, B.: Mutual information analysis. In: Cryptographic Hardware and Embedded Systems – CHES 2008. Lecture Notes in Computer Science, pp. 426–442 (2008). https://doi.org/10.1007/978-3-540-85053-3_27

[38] Robissout, D., Zaid, G., Colombier, B., Bossuet, L., Habrard, A.: Online performance evaluation of deep learning networks for profiled side-channel analysis. In: Constructive Side-Channel Analysis and Secure Design (COSADE). Lecture Notes in Computer Science, pp. 200–218 (2021). https://doi.org/10.1007/978-3-030-68773-1_10

[39] Liu, J., Zheng, S., Gu, L.: Effective deep learning-based side-channel analyses against ASCAD. In: 2021 IEEE 20th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), pp. 514–523 (2021). https://doi.org/10.1109/TrustCom53373.2021.00081

[40] Li, D., Li, L., Ou, Y.: Side-channel analysis based on Siamese neural network. The Journal of Supercomputing **80**(4), 4423–4450 (2024) https://doi.org/10.1007/s11227-023-05631-3

[41] Li, K., Liang, Y., Meng, H., Zhou, Z.: Deep metric learning-based side-channel analysis with improved robustness and efficiency. Applied Intelligence **55**(10) (2025) https://doi.org/10.1007/s10489-025-06586-z

[42] Yap, T., Picek, S., Bhasin, S.: OccPoIs: Points of interest based on neural network's key recovery in side-channel analysis through occlusion. In: Progress in Cryptology – INDOCRYPT 2024. Lecture Notes in Computer Science, pp. 3–28 (2024). https://doi.org/10.1007/978-3-031-80311-6_1

[43] Maghrebi, H.: Assessment of common side channel countermeasures with respect to deep learning based profiled attacks. In: 2019 31st International Conference on Microelectronics (ICM), pp. 126–129 (2019). https://doi.org/10.1109/ICM48031.2019.9021728

[44] Thapar, D., Alam, M., Mukhopadhyay, D.: Deep learning assisted cross-family profiled side-channel attacks using transfer learning. In: 2021 22nd International Symposium on Quality Electronic Design (ISQED), pp. 178–185 (2021). https://doi.org/10.1109/ISQED51717.2021.9424254

[45] Yu, H., Shan, H., Panoff, M., Jin, Y.: Cross-device profiled side-channel attacks using meta-transfer learning. In: 2021 58th ACM/IEEE Design Automation Conference (DAC), pp. 703–708 (2021). https://doi.org/10.1109/DAC18074.2021.9586100

[46] Do, N.-T., Hoang, V.-P., Doan, V.S.: Performance analysis of non-profiled side channel attacks based on convolutional neural networks. In: 2020 IEEE Asia Pacific Conference on Circuits and Systems (APCCAS), pp. 66–69 (2020). https://doi.org/10.1109/APCCAS50809.2020.9301673

[47] Kwon, D., Hong, S., Kim, H.: Optimizing implementations of non-profiled deep learning-based side-channel attacks. IEEE Access **10**, 5957–5967 (2022) https://doi.org/10.1109/ACCESS.2022.3140446

[48] Ramezanpour, K., Ampadu, P., Diehl, W.: SCAUL: Power side-channel analysis with unsupervised learning. IEEE Transactions on Computers **69**(11), 1626–1638 (2020) https://doi.org/10.1109/TC.2020.3013196

[49] Thakor, V.A., Razzaque, M.A., Khandaker, M.R.A.: Lightweight cryptography algorithms for resource-constrained IoT devices: A review, comparison and research opportunities. IEEE Access **9**, 28177–28193 (2021) https://doi.org/10.1109/ACCESS.2021.3052867

[50] Nuradha, F.R., Putra, S.D., Kurniawan, Y., Rizqulloh, M.A.: Attack on AES encryption microcontroller devices with correlation power analysis. In: 2019 International Symposium on Electronics and Smart Devices (ISESD) (2019). https://doi.org/10.1109/isesd.2019.8909447

[51] Putra, S.D., Ahmad, A.S., Sutikno, S., Kurniawan, Y.: Attacking AES-masking encryption device with correlation power analysis. International Journal of Communication Networks and Information Security (IJCNIS) (2022) https://doi.org/10.17762/ijcnis.v10i2.2884

[52] Putra, S.D., Sumari, A.D.W., Asrowardi, I., Subyantoro, E.: Power analysis in Hamming weight model: Attacking IoT encryption devices. In: 2021 4th International Conference on Signal Processing and Information Security (ICSPIS) (2021). https://doi.org/10.1109/icspis53734.2021.9652185

[53] Putra, S.D., Ahmad, A.S., Sutikno, S.: DPA-countermeasure with knowledge growing system. In: 2016 International Symposium on Electronics and Smart Devices (ISESD) (2016). https://doi.org/10.1109/isesd.2016.7886757

[54] Putra, S.D., Sumari, A.D.W., Ahmad, A.S., Sutikno, S., Kurniawan, Y.: Cognitive artificial intelligence countermeasure for enhancing the security of big data hardware from power analysis attack. In: Combating Security Challenges in the Age of Big Data. Advanced Sciences and Technologies for Security Applications, (2020). https://doi.org/10.1007/978-3-030-35642-2_4

[55] Rezaeezade, A., Yap, T., Jap, D., Bhasin, S., Picek, S.: Breaking the Blindfold: Deep Learning-based Blind Side-channel Analysis. Zenodo dataset. https://zenodo.org/records/15614359 (2025). https://doi.org/10.5281/zenodo.15614359