Power Side-Channel Analysis with Unsupervised Learning

LSTM Auto-Encoders, Sensitivity Analysis, and ASCAD Implementation

Yahya Mansoub , Supervisors: Dr. Ikram Chairi and Dr. Manal Cherkaoui

Research Question

Main question

Can we recover the AES key from power traces *without* a profiling device or explicit leakage model, by learning features and a leakage model in an unsupervised way?

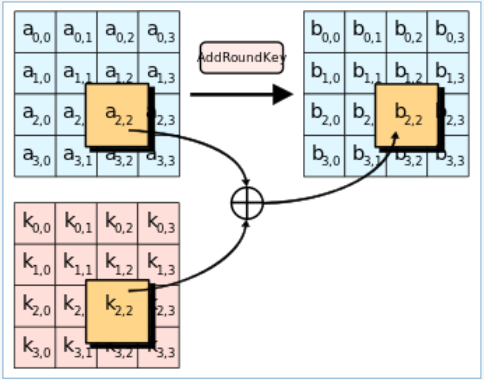
Outline

What is Power Side-Channel Analysis?

- Digital circuits leak information through power consumption.
- Measuring current/voltage $\Rightarrow$ power trace:

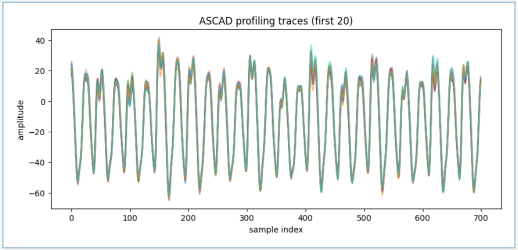
$$T = (t_1, \dots, t_N) \in \mathbb{R}^N.$$

- Goal: recover secret key from many traces and known plaintexts.
- Typical target: intermediate $X = S(P \oplus K)$ in AES round 1.



Example Power Trace

- Each encryption $\Rightarrow$ one waveform.
- Different plaintexts, same key.
- Small parts of the trace depend on S-box operations; rest is noise / unrelated activity.
- Classical side-channel analysis uses statistics at Points of Interest (POIs).



Attack Model and Leakage

Cipher operation under attack:

$$X = F_K(Z) = S(Z \oplus K), \quad Z, P, K \in \mathbb{F}_2^8.$$

We assume mutual information:

$$I(T; X) > 0.$$

Generic leakage model as algebraic normal form:

$$\widetilde{T} = \alpha_0 + \sum_{U \neq 0} \alpha_U X^U + \varepsilon,$$

with monomials

$$X^U = \prod_{i=0}^{m-1} x_i^{u_i}, \quad d = \mathrm{HW}(U).$$

Classical models

- Hamming Weight (HW)
- Hamming Distance (HD)
- Single-bit leakage (MSB, LSB, ...)

Key idea

Correct key $\Rightarrow$ power statistically depends on X;
wrong key $\Rightarrow$ independence.

UM6P College of Computing

Model-Based Attacks: DPA / CPA

For each key guess k^*:

1. Compute $X_{j,k^*} = S(P_j \oplus k^*)$.
2. Build leakage hypothesis (e.g. HW).
3. Cluster traces or correlate with samples.

DPA difference-of-means:

$$\Delta_{k^*}(n) = \mu_1(n; k^*) - \mu_0(n; k^*).$$

CPA correlation:

$$\rho_{k^*}(n) = \frac{\mathrm{Cov}(L_{j,k^*}, t_{j,n})}{\sigma(L_{j,k^*})\,\sigma(t_{j,n})}.$$

Limitations

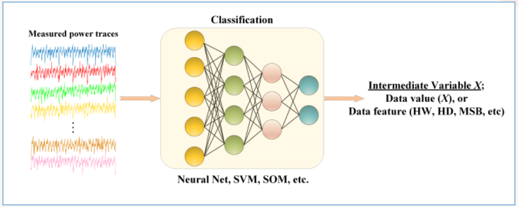
- Need a good leakage model.
- POI selection is manual.
- Misaligned traces break the attack.

Profiling / Supervised ML Attacks

- Profiling phase on clone device:

$$g_\theta : T \to \text{class}(X).$$

- Use CNN / MLP / RNN to learn features + classifier.
- Attack phase: apply g_θ to new traces to rank key hypotheses.



Pros

- Handles misalignment.
- No manual POI selection.

Why Unsupervised?

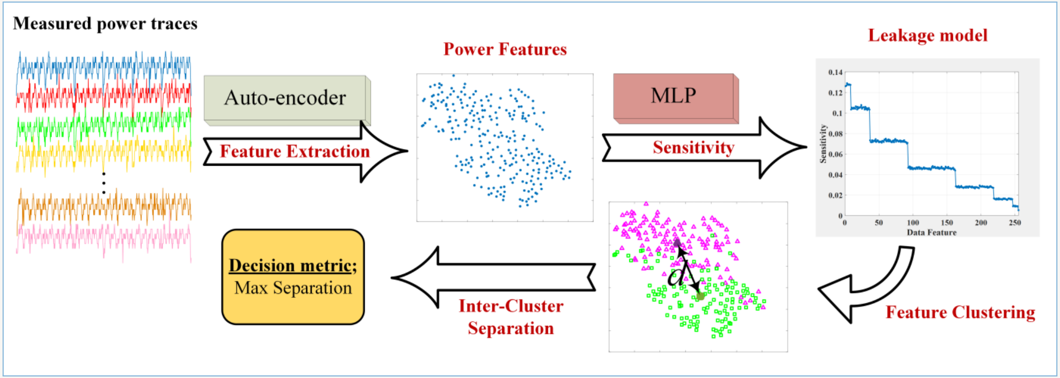
Limitations of profiling

- Requires labeled traces from a clone device.
- Performance drops when training and target devices differ.
- Leakage model is fixed by training labels.

Goal of SCAUL

- Learn power features *without labels*.
- Discover leakage model from the same traces.
- Use them to rank key candidates.

Why Unsupervised? (Overview)



Max-Information Auto-Encoder

Encoder produces features

$$f = e_{\mathbf{W}_e}(\widehat{T}), \quad f \in \mathbb{R}^D$$

from corrupted traces $\widehat{T}$. Mutual information:

$$I(T; f) = H(T) - H(T \mid f).$$

Since $H(T)$ is fixed,

$$\max I(T; f) \iff \min H(T \mid f).$$

Variational objective:

$$\max_{\mathbf{W}_e, \tilde{p}} \mathbb{E}_{T,f}[\log \tilde{p}(T \mid f)].$$

Intuition

- Features should preserve all information about the signal.
- Noise and irrelevant parts are compressed away.
- Later stages operate in this compact feature space.

From Cross-Entropy to MSE

Decoder $d_{\mathbf{W}_d}$ induces $\hat{p}(T \mid \widehat{T}; \mathbf{W}_e, \mathbf{W}_d)$. Objective becomes:

$$\min_{\mathbf{W}_e, \mathbf{W}_d} H\left(p(\widehat{T}) \,\|\, \hat{p}(T \mid \widehat{T})\right).$$

Assume additive Gaussian corruption $\widehat{T} = T + N$:

$$N \sim \mathcal{N}(0, \Sigma).$$

Then minimizing cross-entropy $\Rightarrow$ (up to constants)

$$\min \mathbb{E}[(T - \tilde{T})^\top \Sigma^{-1} (T - \tilde{T})] + H(\tilde{T}).$$

In practice:

$$\mathcal{L}_{\text{AE}} \approx \mathbb{E}\left[\|T - \tilde{T}\|_2^2\right]$$

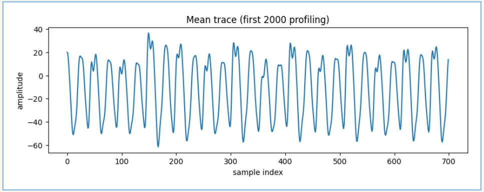
with bottleneck dimension D acting as entropy regularizer.

Takeaway

- MSE-trained AE $\approx$ max-information AE.
- Features f keep what is needed to reconstruct traces.
- Data-dependent leakage survives in f.

ASCAD Dataset

- Public database of power traces for AES-128 on AVR.
- Fixed-key aligned traces (`ASCAD.h5`).
- Each trace: N samples, known plaintext, secret key.
- I use windows around S-box activity in round 1.

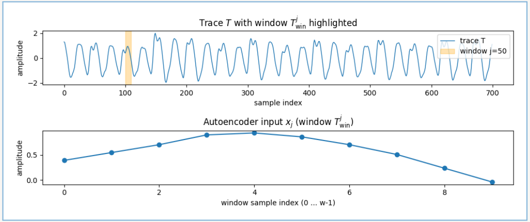


Selecting Windows around S-box

- Average trace shows region where first-round S-box runs.
- For each trace:

$$T_j^{\text{win}} = (t_{j,1}, \ldots, t_{j,N_{\text{win}}}).$$

- Misalignment experiments: enlarge window to include jitter.



Sliding-Window Sequence Construction

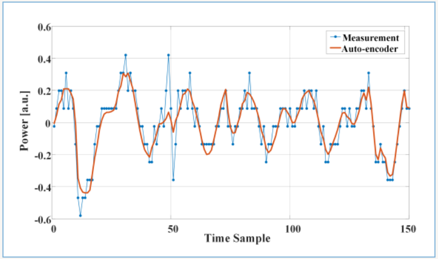
Window length w, stride s:

$$x_t = (t_{j,ts}, \ldots, t_{j,ts+w-1}) \in \mathbb{R}^w.$$

Number of LSTM time steps:

$$T_{\text{steps}} = 1 + \frac{N_{\text{win}} - w}{s}.$$

Each trace $\Rightarrow$ sequence $(x_1, \ldots, x_{T_{\text{steps}}})$ fed to encoder.



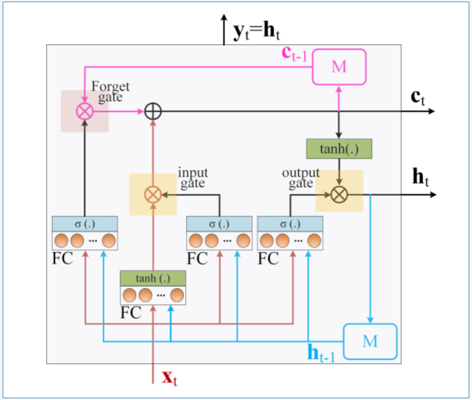
Raw vs. auto-encoder filtered trace.

LSTM Cell Mechanics

For each time step:

$$
\begin{aligned}
f_t &= \sigma(W_f x_t + U_f h_{t-1} + b_f), \\
i_t &= \sigma(W_i x_t + U_i h_{t-1} + b_i), \\
\tilde{c}_t &= \tanh(W_c x_t + U_c h_{t-1} + b_c), \\
c_t &= f_t \odot c_{t-1} + i_t \odot \tilde{c}_t, \\
o_t &= \sigma(W_o x_t + U_o h_{t-1} + b_o), \\
h_t &= o_t \odot \tanh(c_t).
\end{aligned}
$$

Long-term memory: c_t.
Exposed state: h_t.



LSTM Auto-Encoder Architecture

- Input: sequence of sliding windows

$$x_t \in \mathbb{R}^w, \quad t = 1, \dots, T_{\text{steps}}.$$

- Encoder: 2-layer LSTM reads $(x_1, \dots, x_{T_{\text{steps}}})$.
- Decoder: 2-layer LSTM reconstructs the trace (time-reversed).
- Training loss (MSE):

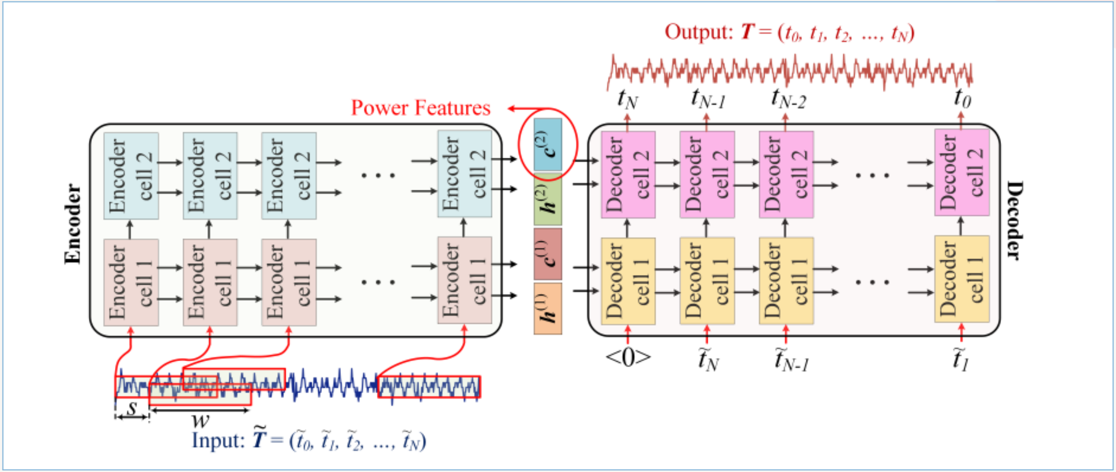
$$\mathcal{L}_{\text{AE}} = \frac{1}{M} \sum_{j=1}^{M} \| T_j - \tilde{T}_j \|_2^2.$$

- Power feature for trace j:

$$f_j = c^{(2)}_{T_{\text{steps}}, j} \in \mathbb{R}^D,$$

i.e. final cell state of top LSTM layer.

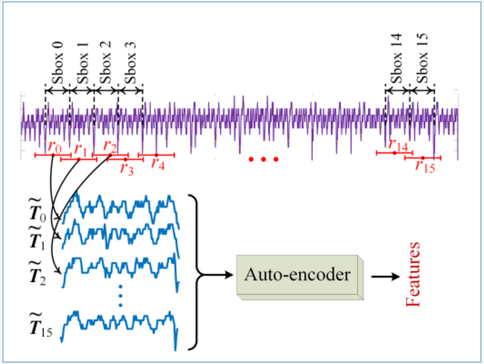
LSTM Auto-Encoder Architecture (Diagram)



Encoder compresses sliding windows into feature vector f_j;
decoder reconstructs the trace from f_j, enforcing information-rich features.

Horizontal Processing of AES Round 1

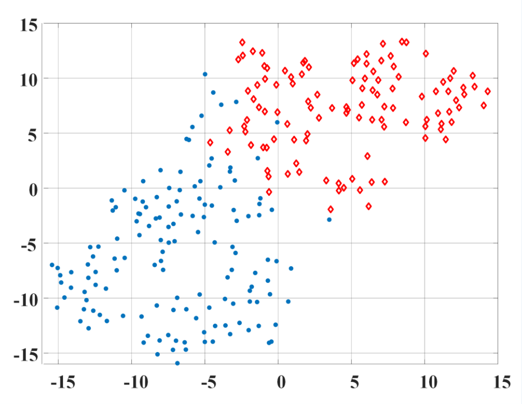
- In SCAUL: 16 S-box windows across round 1.
- Each S-box segment $r_i \Rightarrow$ input trace for auto-encoder.
- Same encoder used for all bytes $\Rightarrow$ horizontal attack.
- Greatly increases effective number of training samples.



- After training AE, each trace $\to f_j$.
- Apply t-SNE / PCA:

$$z_j = \phi(f_j) \in \mathbb{R}^2.$$

- Clusters appear even without using labels.
- Empirical evidence that f_j preserves data-dependent structure.



MLP Mapping Features to Intermediate Bits

Normalize features:

$$\tilde{f}_j = \frac{f_j - \min_\ell f_\ell}{\max_\ell f_\ell - \min_\ell f_\ell}.$$

For key guess k^*:

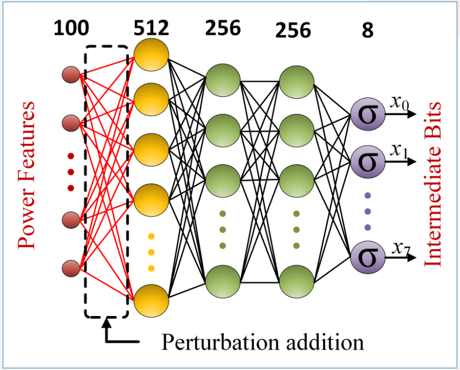
$$X_{j,k^*} = S(P_j \oplus k^*),$$

with bit vector $\mathbf{x}_{j,k^*} \in \{0,1\}^8$. MLP:

$$g_{\theta_{k^*}} : \tilde{f}_j \to \hat{\mathbf{x}}_{j,k^*} \in [0,1]^8.$$

Loss:

$$\mathcal{L}(\theta_{k^*}) = \sum_{j,b} \mathrm{BCE}(x^{(b)}_{j,k^*}, \hat{x}^{(b)}_{j,k^*}).$$



From Fisher Information to Sensitivity

Leakage model:

$$\widetilde{T} = \alpha_0 + \sum_{U \neq 0} \alpha_U X^U + \varepsilon.$$

Fisher information for parameter $\theta = X^U$:

$$I(\theta) = \mathbb{E}_f \left[\left(\frac{\partial}{\partial \theta} \log p(f \mid \theta) \right)^2 \right].$$

Cramér–Rao:

$$\mathrm{Var}(\hat{\theta}) \geq I(\theta)^{-1}.$$

High info $\Rightarrow$ small variance $\Rightarrow$ estimator robust to small perturbations.

Idea

- Perturb MLP weights.
- Observe change in estimated monomials X^U.
- Small change $\Rightarrow$ strong leakage feature.

UM6P College of Computing

Perturbation-Based Sensitivity Measure

Perturb first weight matrix:

$$\widetilde{W}_{0,1} = W_{0,1} + \delta, \quad \|\delta\| \ll \|W_{0,1}\|.$$

For each monomial U:

$$X^{U}_{j,k^*} = \prod_{b} (x^{(b)}_{j,k^*})^{u_b},$$

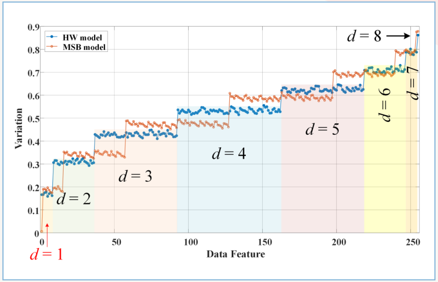
$$\widetilde{X}^{U}_{j,k^*} = \prod_{b} (\tilde{x}^{(b)}_{j,k^*})^{u_b},$$

Variation:

$$\Delta_U = \mathbb{E}_j\left[\left|\widetilde{X}^{U}_{j,k^*} - X^{U}_{j,k^*}\right|\right].$$

Coefficients:

$$\hat{\alpha}_U = 1 - \frac{\Delta_U}{\max_V \Delta_V}.$$



Low-variation features ⇒ strong leakage.

Key Ranking from Learned Leakage

Using selected monomials $\mathcal{U}_{\mathrm{sel}}$:

$$\widehat{L}(X) = \sum_{U \in \mathcal{U}_{\mathrm{sel}}} \hat{\alpha}_U X^U.$$

For each trace and key guess:

$$\ell_{j,k^*} = \widehat{L}(X_{j,k^*}).$$

Cluster features:

$$\mathcal{C}_0(k^*) : \ell_{j,k^*} \leq \tau, \quad \mathcal{C}_1(k^*) : \ell_{j,k^*} > \tau.$$

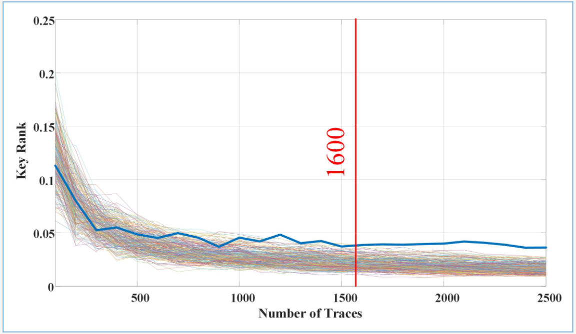
Cluster means:

$$\mu_b(k^*) = \frac{1}{|\mathcal{C}_b|} \sum_{f_j \in \mathcal{C}_b} f_j.$$

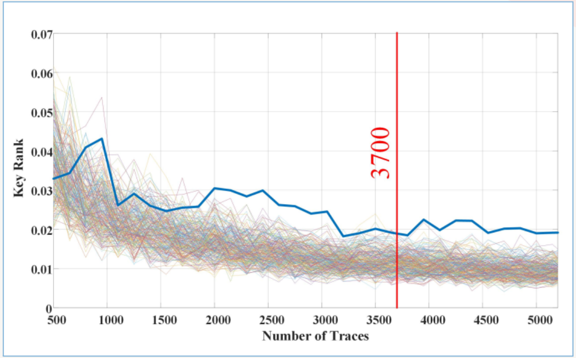
Decision

- Sort candidates by score.
- Correct key should converge to rank 1 as number of traces grows.

Aligned Traces: DPA vs SCAUL



Classical DPA with HW model.



SCAUL with learned leakage model.

- SCAUL recovers correct key with $\sim$ 3,700 traces.
- Classical DPA in original paper needs $\sim$ 1,600 traces.

Misaligned Traces (Summary)

- Random clock jitter creates misalignment $\approx 20\%$ of clock period.
- Classical DPA/CPA on raw samples fails to reveal key.
- LSTM AE still extracts stable features across misaligned traces.
- With SCAUL:
  - Key recovered with $\sim 12{,}300$ measurements (per original paper).
  - Learned leakage model remains valid in feature space.

Conclusions

- **Answer to the research question:**
  Yes – unsupervised features + a sensitivity-based leakage model allow key recovery on ASCAD, without profiling labels and with more traces than classical DPA (and still working under misalignment).
- Implemented SCAUL pipeline on ASCAD:
  - LSTM auto-encoder for unsupervised feature learning.
  - MLP + sensitivity analysis to recover leakage model.
  - Classical key-ranking built on learned model.
- Information-theoretic view explains why MSE-trained AE preserves leakage.
- Features are more robust to noise and misalignment than raw samples.

References

- R. Benadjila *et al.*, "ASCAD: A database for profiling side-channel attacks," *IACR ePrint Archive*, 2018.
- P. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," in *CRYPTO*, 1999.
- F.-X. Standaert, B. Gierlichs, and I. Verbauwhede, "Partition vs. comparison side-channel distinguishers: An empirical evaluation of statistical tests for univariate side-channel attacks against two unprotected CMOS devices," in *International Conference on Information Security and Cryptology (ICISC)*, Springer, 2008, pp. 253–267.
- L. Batina, B. Gierlichs, E. Prouff, M. Rivain, F.-X. Standaert, and N. Veyrat-Charvillon, "Mutual information analysis: a comprehensive study," *Journal of Cryptology*, vol. 24, no. 2, pp. 269–291, 2011.

Questions?