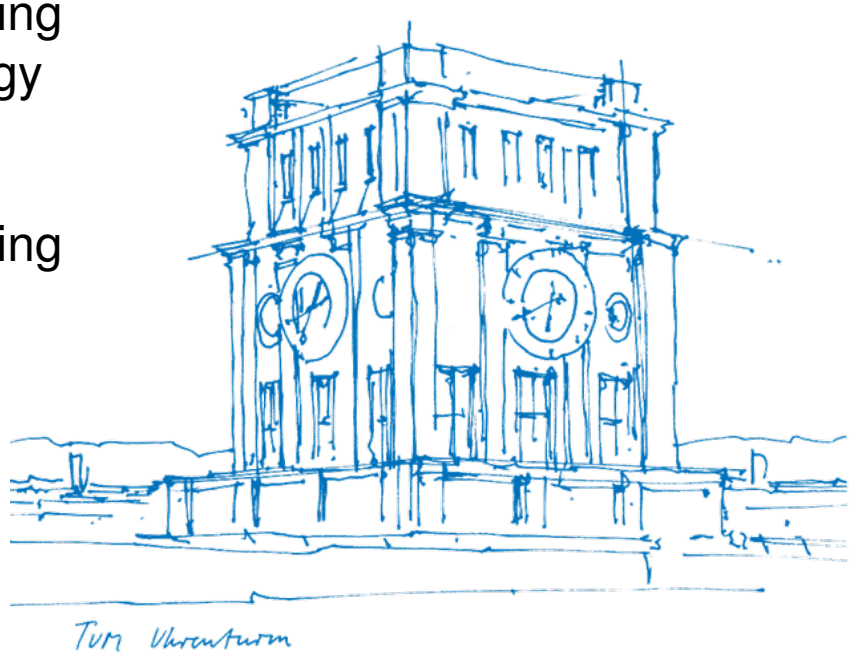
A Second Look at the ASCAD Databases

Maximilian Egger[2] Thomas Schamberger[1] Lars Tebelmann[1] Florian Lippert[1]
Georg Sigl[1]

[1]Technical University of Munich
Faculty of Electrical and Computer Engineering
Institute for Security in Information Technology

[2]Technical University of Munich
Faculty of Electrical and Computer Engineering
Institute for Communications Engineering

12.04.2022



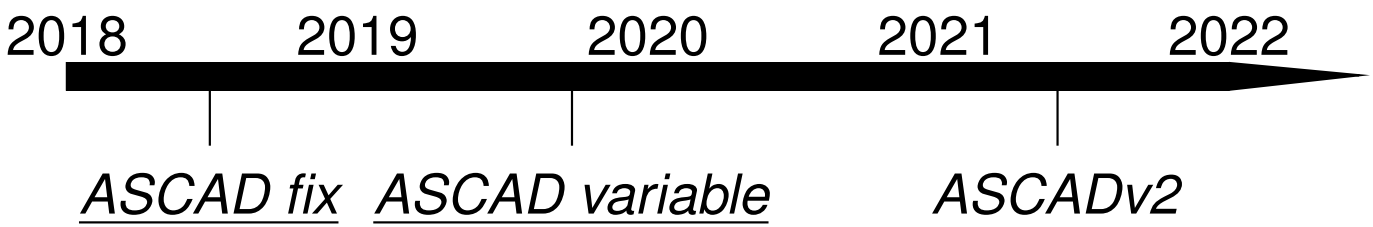
COSADE 2022

Outline

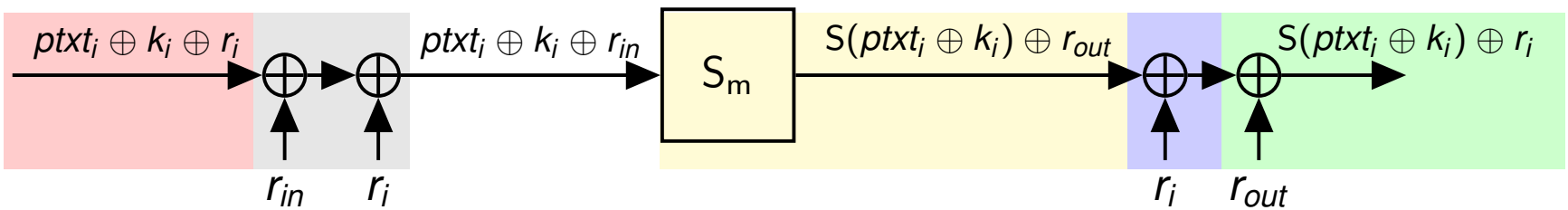
Motivation

- Vast amount of work regarding the ASCAD databases
- Only few papers investigate the dataset
- Additional information (key bytes) is contained in the databases
- This is **not a criticism of the databases**, which are very helpful for the community

ANSSI SCA Databases (ASCAD)

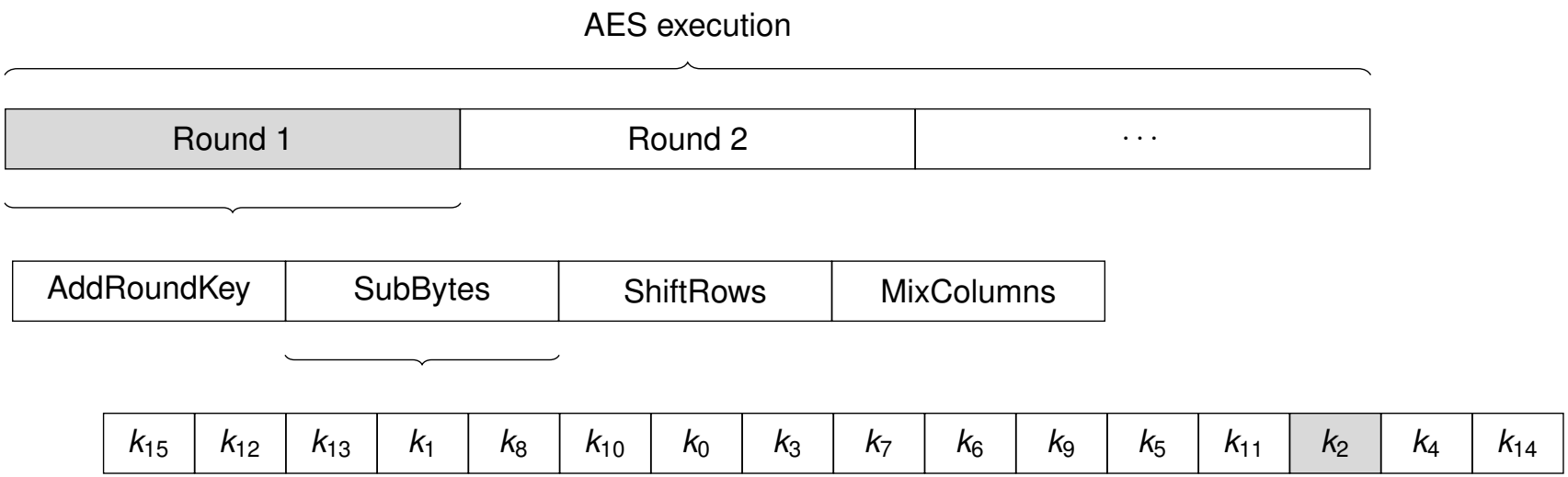


- Power side-channel measurements of software AES
- First-order boolean masked implementation on ATMega8515
- Table re-computation method:

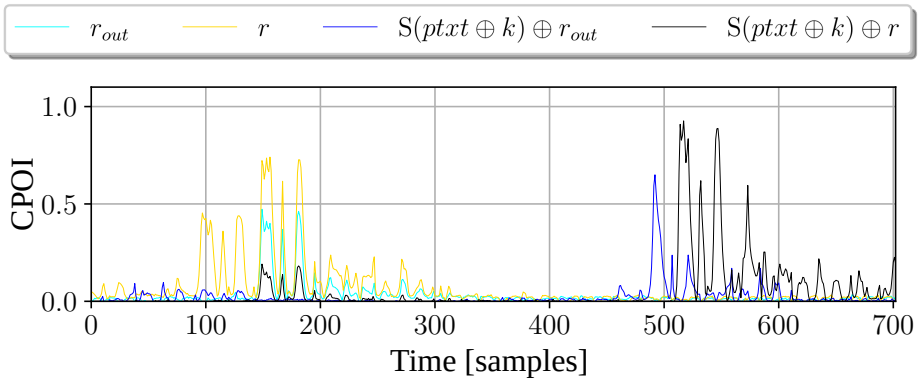


ANSSI SCA Databases (ASCAD)

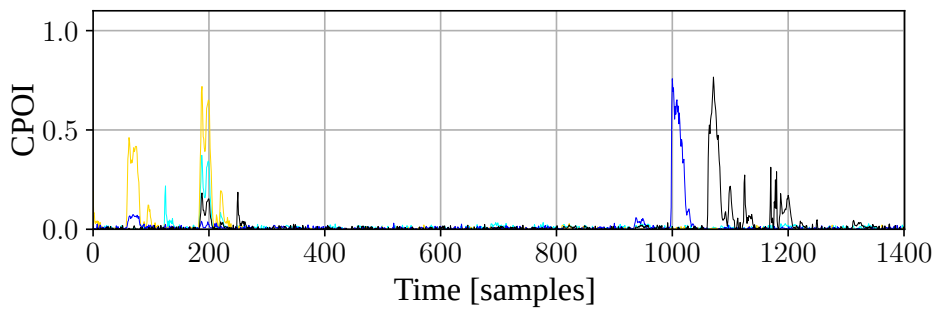
- The databases consist of two components:
  - Raw traces including the whole execution of the first AES round
  - Pre-selected sample range of k_2



Difference between Datasets: Leakage of k_2



ASCAD fix

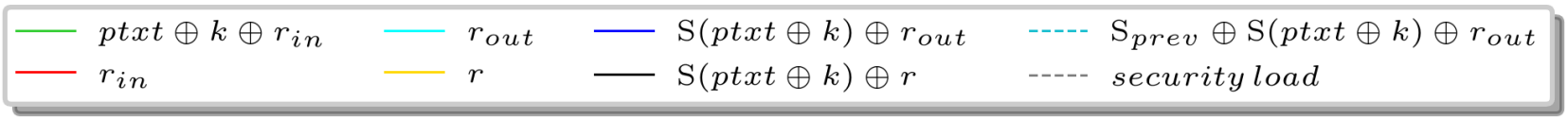


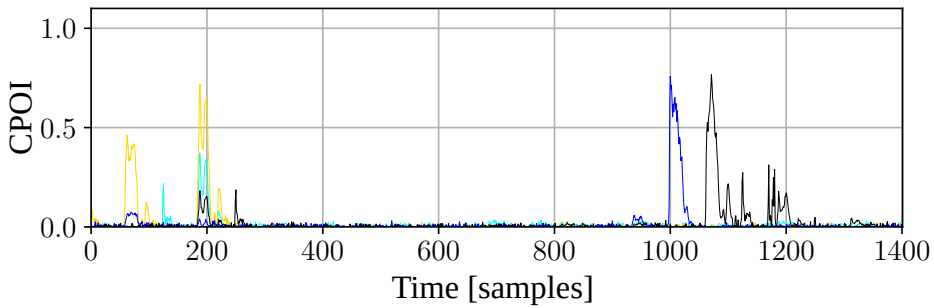
ASCAD variable

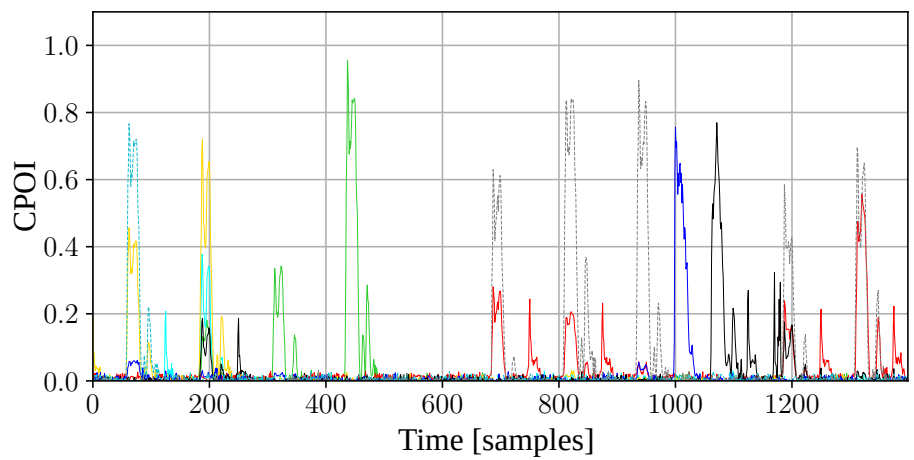
- Difference between datasets:
  - Different time frame captured (f_s)
  - Leakage of *ASCAD fix* spread over multiple clock cycles
- Leakage of intermediates as observed by related work
- Target for alignment for other bytes:

  S-box$(ptxt \oplus k) \oplus r_{out}$

Additional Leakage of Intermediates: k_2

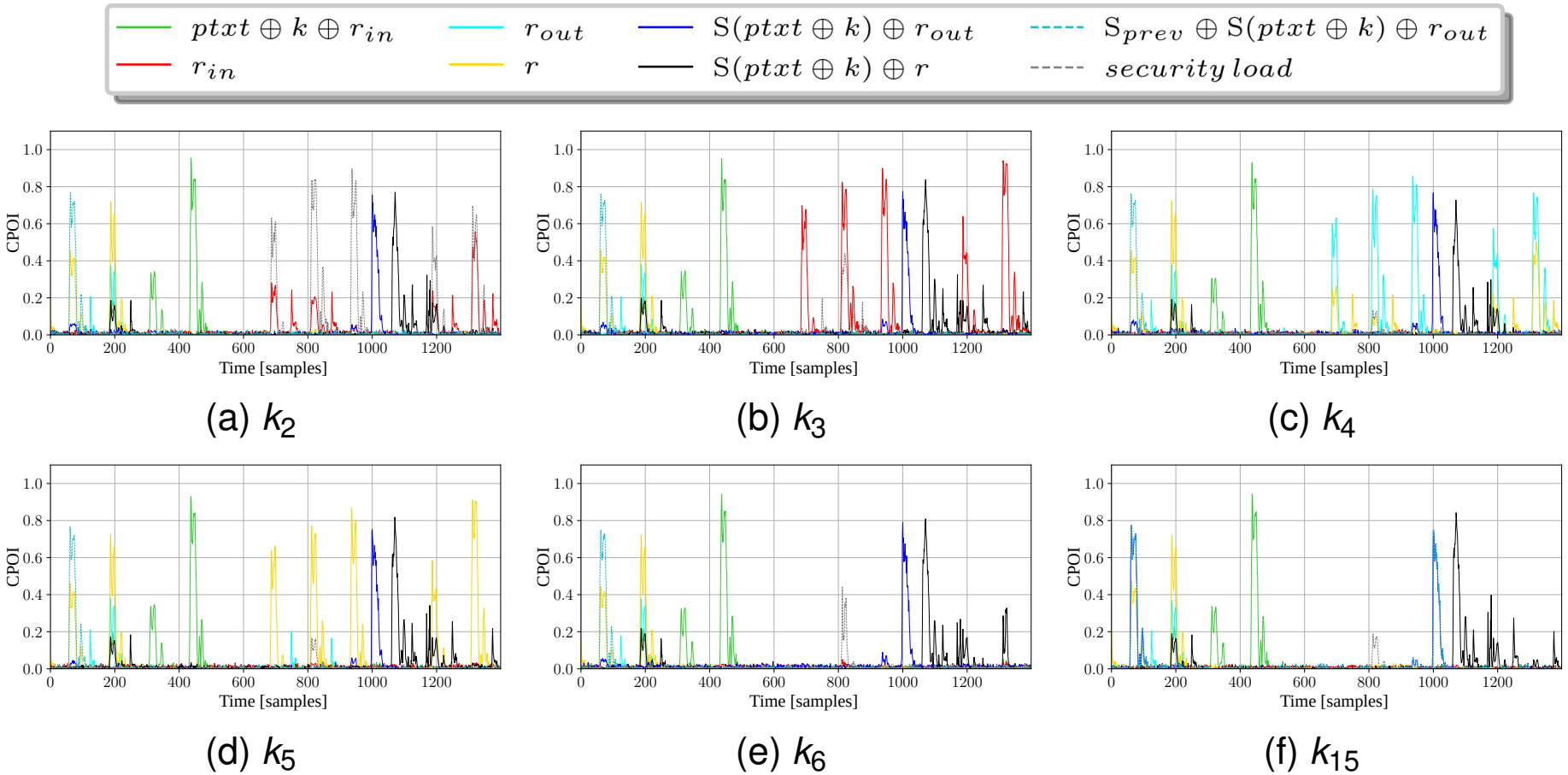






- Additional leakage not observed by related work
- Useful for neural network attribution
- Networks may exploit this leakage

Leakage Difference between Key Bytes



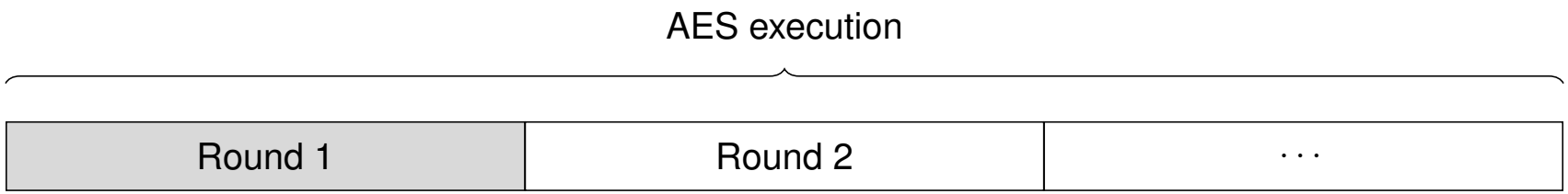
Classical Side-Channel Analysis

- ASCAD includes first-order secure implementation
  $\Rightarrow$ ML-SCA has to perform higher-order attacks
- CPA attack results (trace segments):

	Order	k_0	k_1	k_2	k_3	k_4	k_5	k_6	k_7	k_8	k_9	k_{10}	k_{11}	k_{12}	k_{13}	k_{14}	k_{15}
ASCAD fix	1st	19	12	–	–	–	1960	–	–	–	–	–	–	–	–	–	–
	2nd (uni.)	x	x	5440	2060	4900	3160	4880	9400	5180	2360	2940	5200	8580	7920	1980	2730
	2nd (mult.)	x	x	620	280	540	260	200	480	340	1340	400	460	620	460	240	300
ASCAD variable	1st	10	24	–	–	85700	1580	–	–	–	–	–	–	–	–	–	–
	2nd (uni.)	x	x	–	–	–	–	–	–	–	–	–	–	–	–	–	–
	2nd (mult.)	x	x	560	640	900	540	880	740	680	960	900	1220	1100	1380	520	660

Classical Side-Channel Analysis

- CPA attack results (Raw traces):



	Order	k_0	k_1	k_2	k_3	k_4	k_5	k_6	k_7	k_8	k_9	k_{10}	k_{11}	k_{12}	k_{13}	k_{14}	k_{15}
ASCAD fix	1st	14	14	12960	11220	11640	2280	15240	10220	–	6980	–	27580	–	–	34660	–
	2nd (uni.)	x	x	3960	4460	5160	3120	6540	15560	9820	10380	6780	6400	12000	9160	12840	3100
ASCAD variable	1st	14	16	17160	10900	14060	2260	7760	22220	11720	15480	13800	19360	6120	22200	12740	16520
	2nd (uni.)	x	x	–	–	–	–	–	–	–	–	–	–	–	–	–	–

- Does ML-SCA exploit these first-order or 2nd-order univariate leakages?

Comparison of Databases

	ASCAD fix	*ASCAD variable*
Training traces	50.000 (fixed key)	200.000 (random key)
Attack traces	10.000 (fixed key)	100.000 (fixed key)
sampling frequency (f_s)	~~2 GS/s~~ 200 MS/s	500 MS/s
clock frequency (f_{clk})	4 MHz	4 MHz
sample size (trace segment)	700	1400

ML-SCA on ASCAD

- Use of ID model $\rightarrow$ 256 classes
- ASCAD proposed model (large architecture) [1]:

	$\mathrm{CNN}_{\mathrm{best}}$ [1]
Network architecture	C(64,11,2),P(),C(128,11,1),P(),C(256,11,1),P(),C(512,11,1), P(),C(512,11,1),P(),FLAT,FC(4096),FC(4096),SM(256)
Training parameters	Batch size (200); Epochs (100);

- Reinforcement learning model [2]:

	$\mathrm{CNN}_{\mathrm{small}}$ [2]
Network architecture	C(128,3,1),P(75,75),FLAT,FC(30),FC(2),SM(256)
Training parameters	Batch size (400); Epochs (50); adaptive learning rate

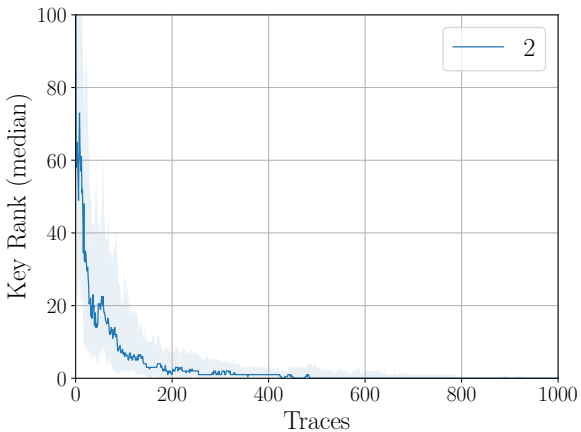
[1] Benadjila et al.: *Deep learning for side-channel analysis and introduction to ASCAD database*, Journal of Cryptographic Engineering, 2019
[2] Rijsdijk et al.: *Reinforcement Learning for Hyperparameter Tuning in Deep Learning-based Side-channel Analysis*, CHES 2021

Fixed Key vs. Variable Key Training

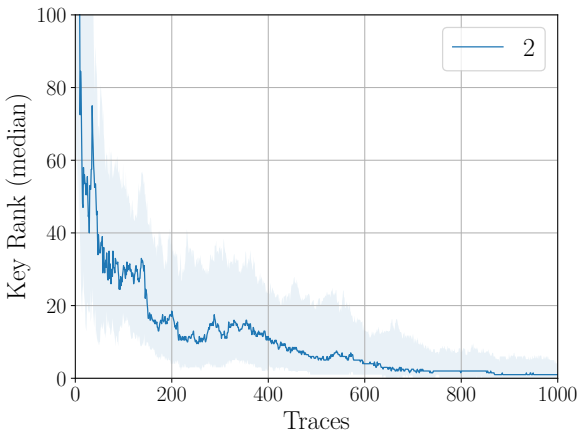
- *ASCAD fix* uses exact **same** key for **training** and **attack** phase
  - Unrealistic scenario
  - Does this fixed key influence the training?

Fixed Key vs. Variable Key Training

- *ASCAD fix* uses exact **same** key for **training** and **attack** phase
  - Unrealistic scenario
  - Does this fixed key influence the training?



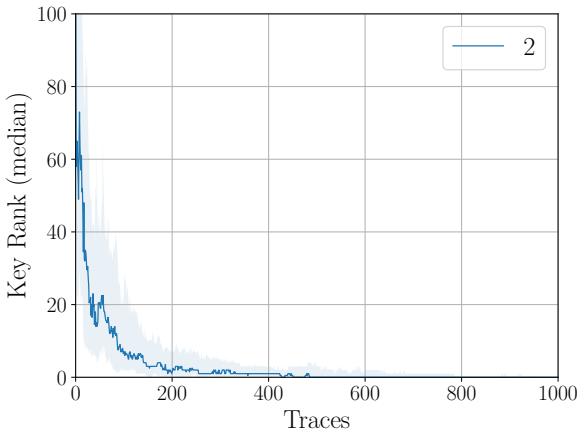
(a) $\mathrm{CNN}_{\mathrm{best}}$: *ASCAD fix*



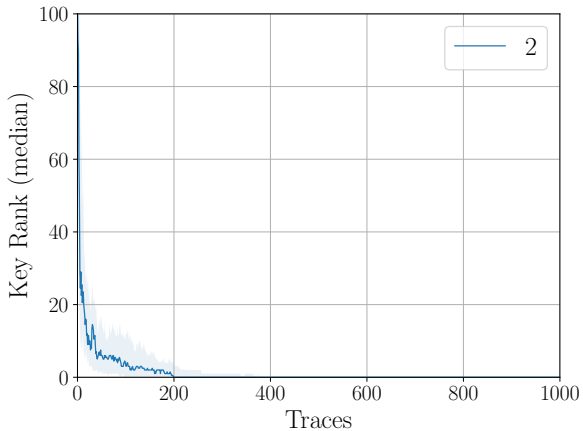
(b) $\mathrm{CNN}_{\mathrm{best}}$: Variable key

Fixed Key vs. Variable Key Training

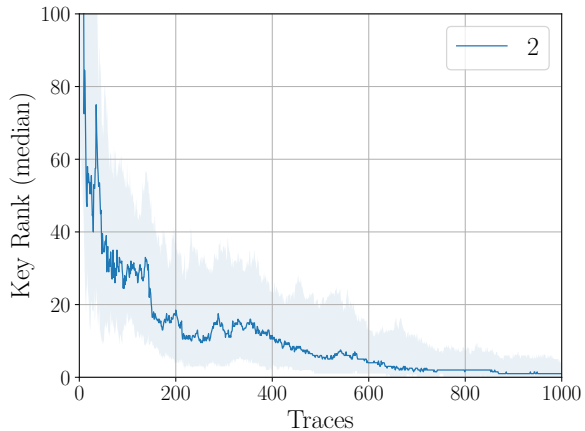
- *ASCAD fix* uses exact **same** key for **training** and **attack** phase
  - Unrealistic scenario
  - Does this fixed key influence the training?



(a) $\mathrm{CNN}_{\mathrm{best}}$: *ASCAD fix*



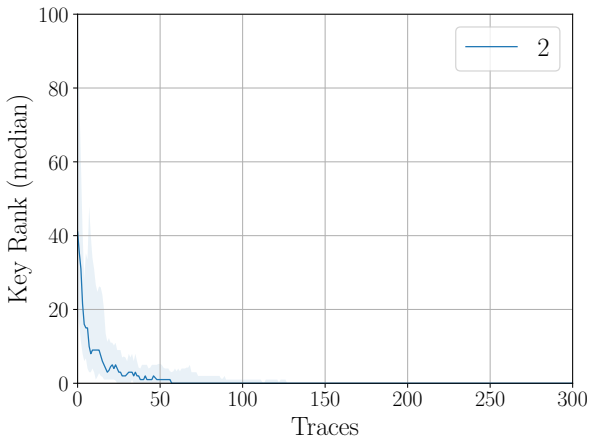
(b) $\mathrm{CNN}_{\mathrm{best}}$: Fixed key



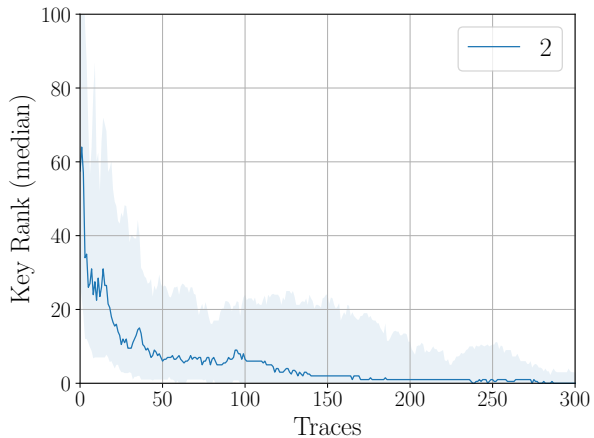
(c) $\mathrm{CNN}_{\mathrm{best}}$: Variable key

Fixed Key vs. Variable Key Training

- *ASCAD fix* uses exact **same** key for **training** and **attack** phase
  - Unrealistic scenario
  - Does this fixed key influence the training?



(a) $\mathrm{CNN}_{\mathrm{small}}$: Fixed key



(b) $\mathrm{CNN}_{\mathrm{small}}$: Variable key

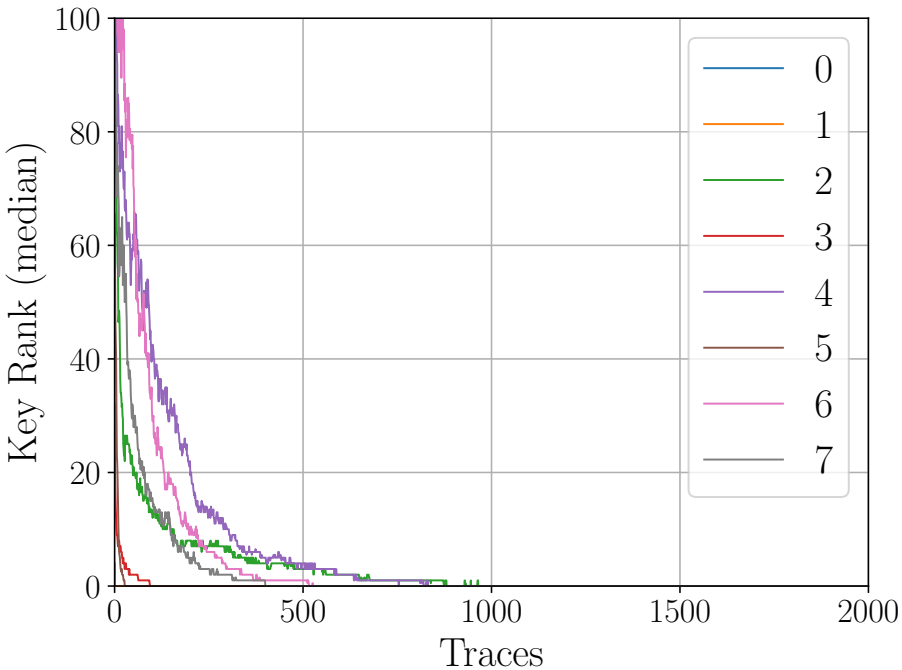
- Fixed key training overestimates attack results

ML-SCA Results for all Key Bytes

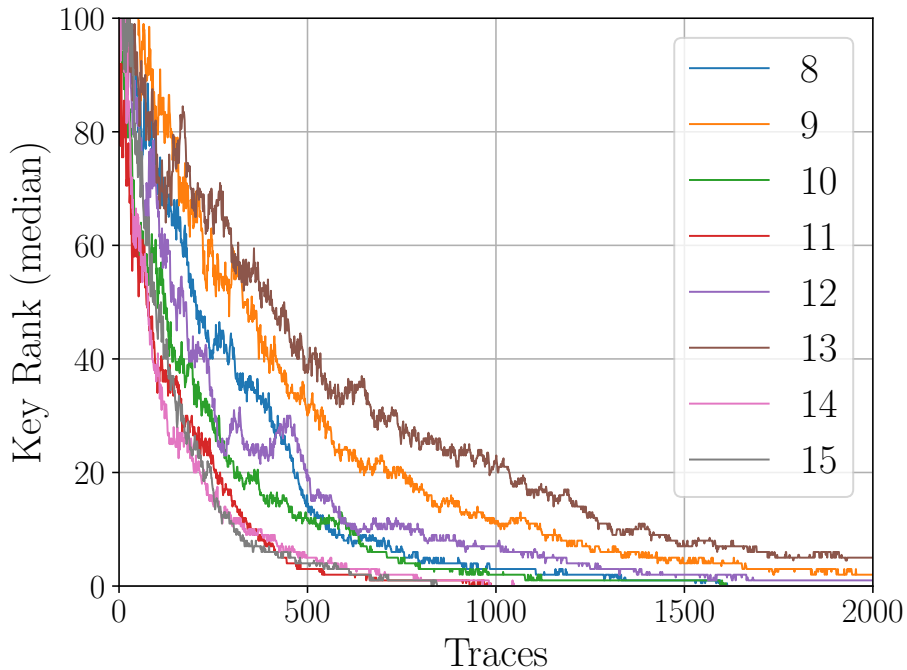
- Different leakage characteristics between the key bytes:
  - Attack result differences between bytes?
  - Are the networks able to generalize between bytes?
- Two experiments:
  - Training and attack on the same key bytes
  - Cross-byte Analysis: Training and attacking on different bytes
- Evaluation on *ASCAD variable*

Training and Attack on same Byte

- $\mathrm{CNN}_{\mathrm{best}}$:



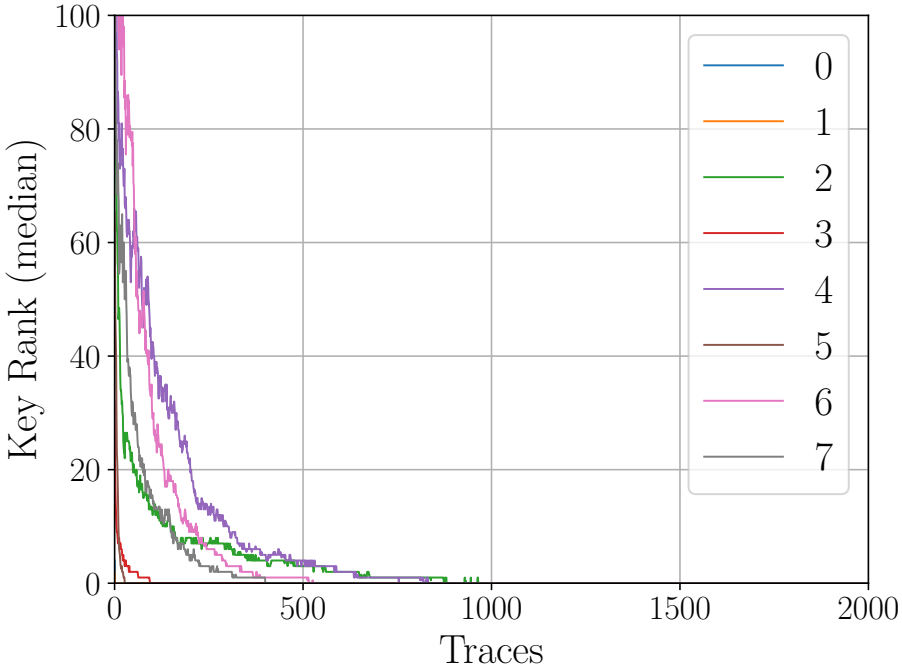
(a) $\mathrm{CNN}_{\mathrm{best}}$: Bytes k_0-k_7



(b) $\mathrm{CNN}_{\mathrm{best}}$: Bytes k_8-k_{15}

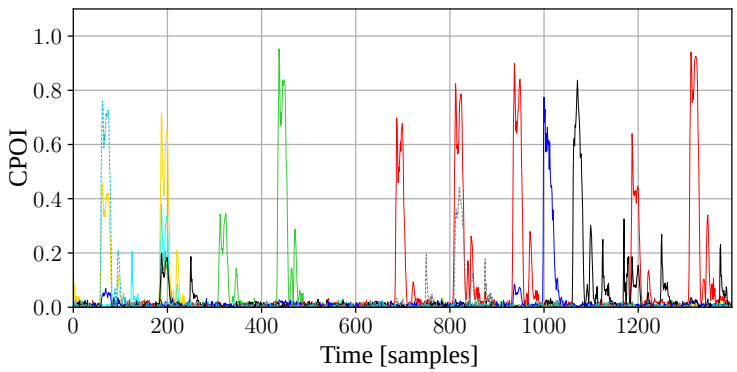
Training and Attack on same Byte

- $\mathrm{CNN}_{\mathrm{best}}$:



(a) $\mathrm{CNN}_{\mathrm{best}}$: Bytes k_0-k_7

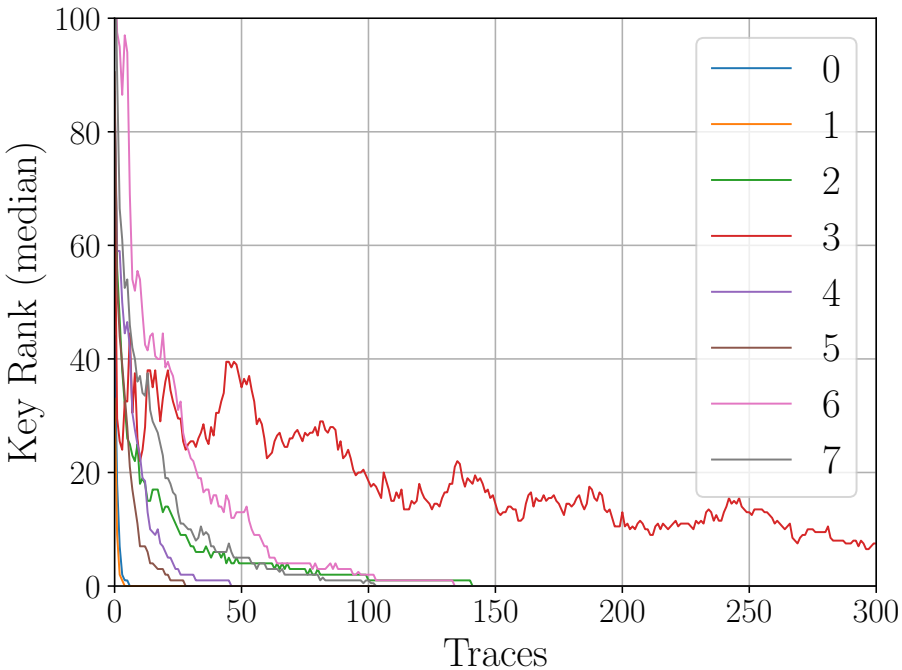
- Difference between bytes clearly visible
- Best bytes k_5 (28 traces) and k_3 (95 traces)
- k_5 byte with first-order leak
- k_3 has additional leakage of r_{in}



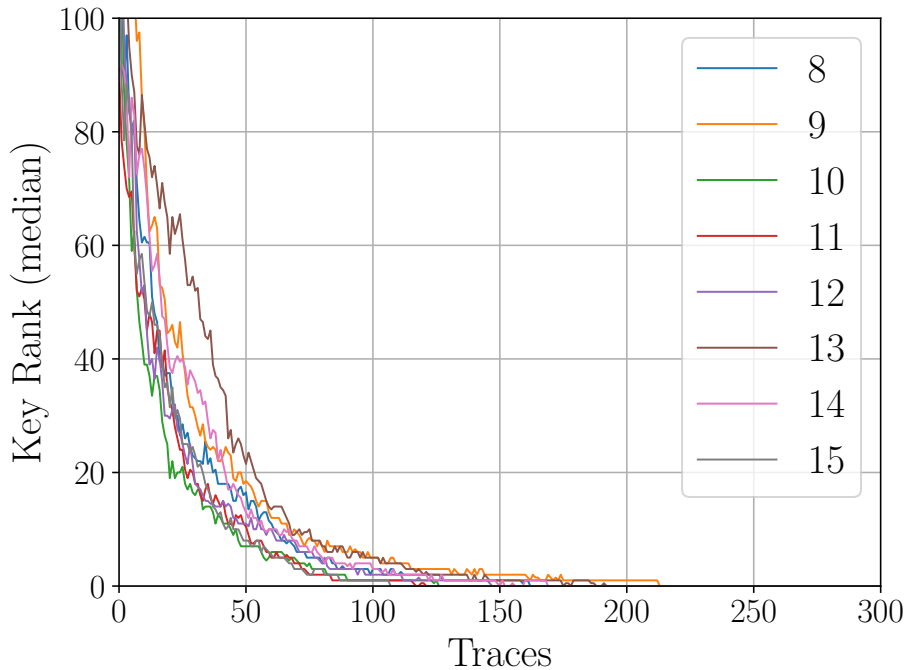
k_3

Training and Attack on same Byte

- $\mathrm{CNN}_{\mathrm{small}}$:



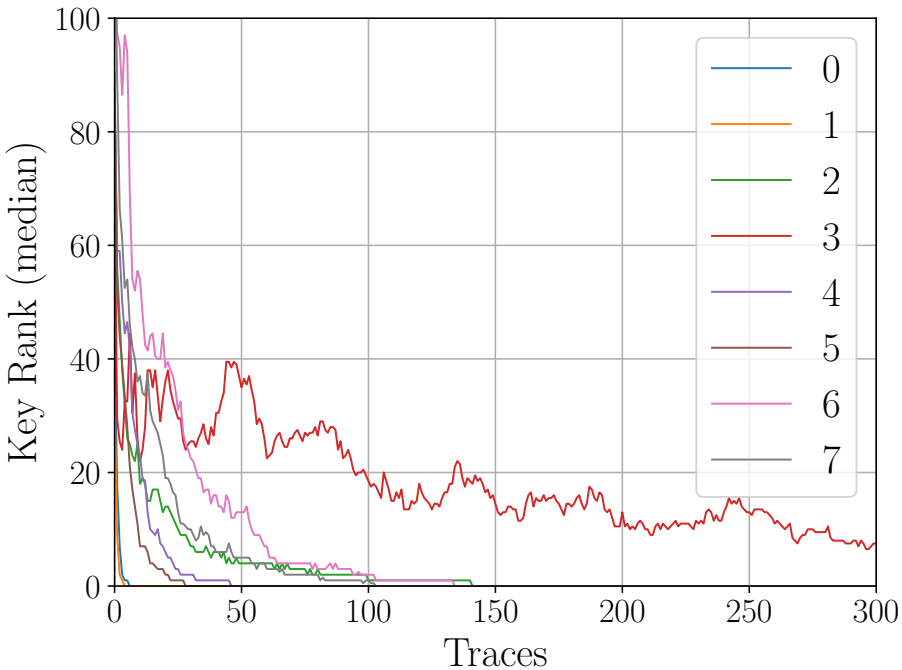
(a) $\mathrm{CNN}_{\mathrm{small}}$: Bytes k_0-k_7



(b) $\mathrm{CNN}_{\mathrm{small}}$: Bytes k_8-k_{15}

Training and Attack on same Byte

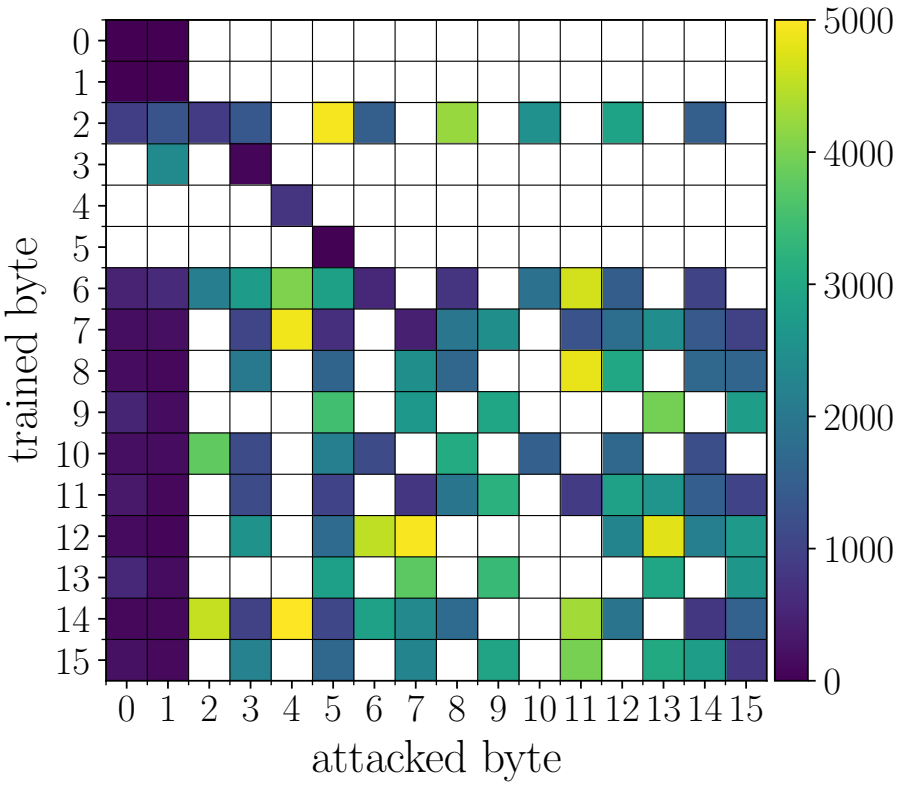
- $\mathrm{CNN}_{\mathrm{small}}$:



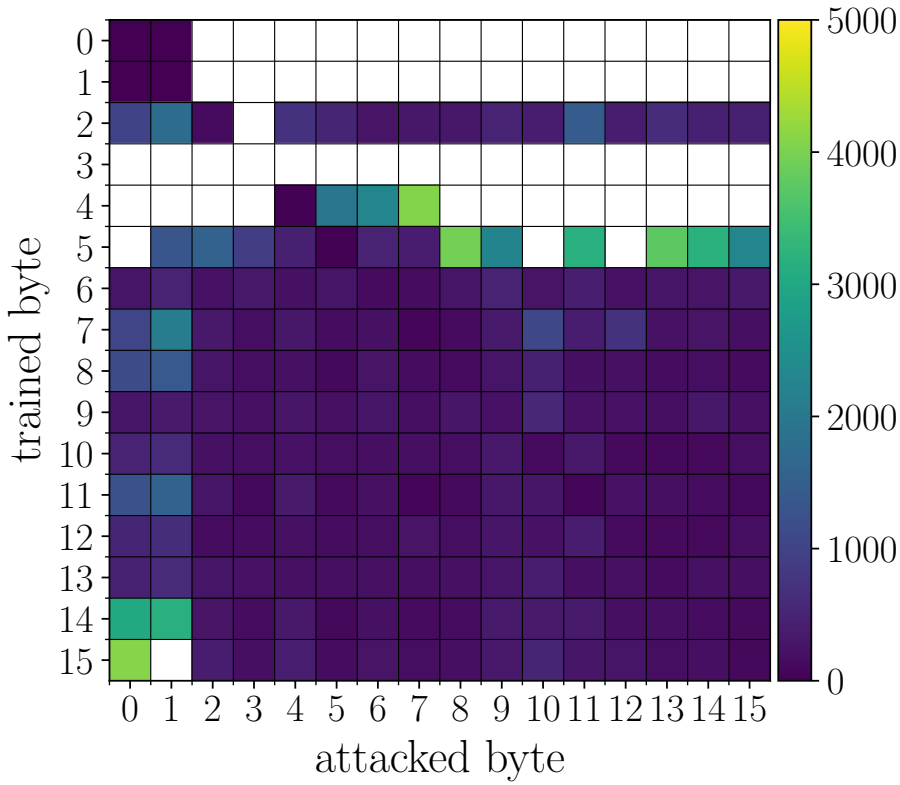
(a) $\mathrm{CNN}_{\mathrm{small}}$: Bytes k_0-k_7

- Best attack results k_4 and k_5 (first-order leak)
- k_3 is an outlier
- Hypothesis:
  - Hyperparameter search for $\mathrm{CNN}_{\mathrm{small}}$ is done on k_2
  - Optimized smaller architecture has problems with different leakage (r_{in})

Cross-byte Analysis



(a) $\mathrm{CNN}_{\mathrm{best}}$



(b) $\mathrm{CNN}_{\mathrm{small}}$

Conclusion

- Interpretation of ML-SCA attack results requires a thorough analysis of the underlying datasets
- Leakage Analysis + Classical SCA:
  - Additional leakage in contrast to ASCAD paper
  - Leakage differs between key bytes
  - First and second-order univariate leakage observable
- Training on a fixed key (*ASCAD fix*) overestimates attack results.
- Training on different bytes:
  - $\mathrm{CNN}_{\mathrm{best}}$ (large architecture): Results differ significantly
  - $\mathrm{CNN}_{\mathrm{small}}$ (k_2): Outlier for different leakage of k_3
  - Cross-byte Analysis: Different leakage bytes (k_3-k_5) difficult

Thank You!

Thomas Schamberger

t.schamberger@tum.de
https://www.sec.ei.tum.de/

Bundesministerium
für Bildung
und Forschung

This work was supported by the German Federal Ministry of Education and Research in the project SIKRIN-KRYPTOV through grant number 16KIS1070.