Location, location, location

Revisiting modeling and exploitation for location-based side channel leakages

Christos Andrikos Lejla Batina Lukasz Chmielewski
Liran Lerman Vasilios Mavroudis Kostas Papagiannopoulos
Guilherme Perin Giorgos Rassias Alberto Sonnino

kostaspap88@gmail.com

https://kpcrypto.net

National Technical University of Athens, NXP Semiconductors Hamburg, Radboud University Nijmegen, Riscure BV, Thales Belgium, University College London

6th December 2019

Introduction to Location-based leakage

Examples for asymmetric and symmetric cryptography

Introduction

- Double-and-add always for scalar multiplication

Secret: $k = k_n k_{n-1} \dots k_0$

$R \leftarrow P$
for $i = n$ downto 1 **do**
    $R_0 \leftarrow 2R$
    $R_1 \leftarrow R_0 + P$
    if $s_i = 0$ **then**
        $R \leftarrow R_0$
    else
        $R \leftarrow R_1$
    end if
end for

- Typically the SCA observes and analyzes data leakage
- A common target can be the value of scalar bit s_i
- Less often SCA focuses on location leakage
- A common target can be the register location, i.e. distinguishing R_0 and R_1
- Both data and location leakage can lead to key recovery

Introduction

- Double-and-add always for scalar multiplication

Secret: $k = k_n k_{n-1} \ldots k_0$

$R \leftarrow P$
for $i = n$ downto 1 **do**
  $R_0 \leftarrow 2R$
  $R_1 \leftarrow R_0 + P$
  if $s_i = 0$ **then**
    $R \leftarrow R_0$
  else
    $R \leftarrow R_1$
  end if
end for

- Typically the SCA observes and analyzes data leakage
- A common target can be the value of scalar bit s_i
- Less often SCA focuses on location leakage
- A common target can be the register location, i.e. distinguishing R_0 and R_1
- Both data and location leakage can lead to key recovery

Introduction

- Double-and-add always for scalar multiplication

Secret: $k = k_n k_{n-1} \ldots k_0$

$R \leftarrow P$
for $i = n$ downto 1 **do**
  $R_0 \leftarrow 2R$
  $R_1 \leftarrow R_0 + P$
  if $s_i = 0$ **then**
    $R \leftarrow R_0$
  else
    $R \leftarrow R_1$
  end if
end for

- Typically the SCA observes and analyzes data leakage
- A common target can be the value of scalar bit s_i
- Less often SCA focuses on location leakage
- A common target can be the register location, i.e. distinguishing R_0 and R_1
- Both data and location leakage can lead to key recovery

Introduction

- Double-and-add always for scalar multiplication

Secret: $k = k_n k_{n-1} \dots k_0$

$R \leftarrow P$
for $i = n$ downto 1 **do**
  $R_0 \leftarrow 2R$
  $R_1 \leftarrow R_0 + P$
  if $s_i = 0$ **then**
    $R \leftarrow R_0$
  else
    $R \leftarrow R_1$
  end if
end for

- Typically the SCA observes and analyzes data leakage
- A common target can be the value of scalar bit s_i
- Less often SCA focuses on location leakage
- A common target can be the register location, i.e. distinguishing R_0 and R_1
- Both data and location leakage can lead to key recovery

Introduction

- Double-and-add always for scalar multiplication

Secret: $k = k_n k_{n-1} \dots k_0$

$R \leftarrow P$
for $i = n$ downto 1 **do**
  $R_0 \leftarrow 2R$
  $R_1 \leftarrow R_0 + P$
  if $s_i = 0$ **then**
    $R \leftarrow R_0$
  else
    $R \leftarrow R_1$
  end if
end for

- Typically the SCA observes and analyzes data leakage
- A common target can be the value of scalar bit s_i
- Less often SCA focuses on location leakage
- A common target can be the register location. i.e. distinguishing R_0 and R_1
- Both data and location leakage can lead to key recovery

Introduction

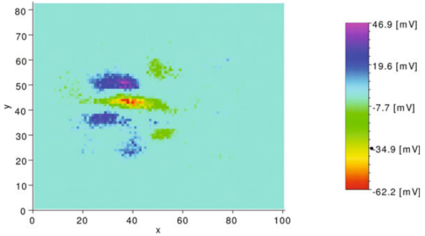
- Double-and-add always for scalar multiplication

Secret: $k = k_n k_{n-1} \dots k_0$

$R \leftarrow P$
for $i = n$ downto 1 **do**
  $R_0 \leftarrow 2R$
  $R_1 \leftarrow R_0 + P$
  if $s_i = 0$ **then**
    $R \leftarrow R_0$
  else
    $R \leftarrow R_1$
  end if
end for

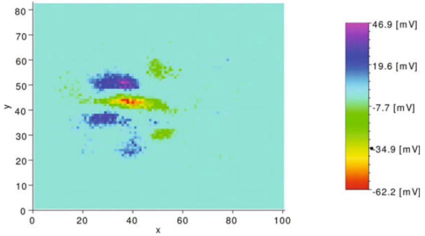
- Typically the SCA observes and analyzes data leakage
- A common target can be the value of scalar bit s_i
- Less often SCA focuses on location leakage
- A common target can be the register location. i.e. distinguishing R_0 and R_1
- Both data and location leakage can lead to key recovery

Introduction



- Depending on the location accessed (R_0 or R_1) the adversary can distinguish the key bit
- Localized electromagnetic analysis of cryptographic implementations, *Heyszl et al.*
- On measurable side-channel leaks inside ASIC design primitives, *Sugawara et al.*

Introduction



- Depending on the location accessed (R_0 or R_1) the adversary can distinguish the key bit
- Localized electromagnetic analysis of cryptographic implementations, *Heyszl et al.*
- On measurable side-channel leaks inside ASIC design primitives, *Sugawara et al.*

Introduction

- Sbox operation for AES cipher, using LUT

Secret: $k = k_7 k_6 \ldots k_0$

$index = input \oplus k$
$y = LUT(index)$

- Data leakage: Observing and analyzing the leakage of k or y
- Location leakage: Observing the table cell accessed
- Learning the accessed position on the LUT is equivalent to learning the value of y
- Thus location information can again lead to key recovery

Introduction

- Sbox operation for AES cipher, using LUT

Secret: $k = k_7 k_6 \ldots k_0$

$index = input \oplus k$
$y = LUT(index)$

- Data leakage: Observing and analyzing the leakage of k or y
- Location leakage: Observing the table cell accessed
- Learning the accessed position on the LUT is equivalent to learning the value of y
- Thus location information can again lead to key recovery

Introduction

- Sbox operation for AES cipher, using LUT

Secret: $k = k_7 k_6 \dots k_0$

$index = input \oplus k$
$y = LUT(index)$

- Data leakage: Observing and analyzing the leakage of k or y
- Location leakage: Observing the table cell accessed
- Learning the accessed position on the LUT is equivalent to learning the value of y
- Thus location information can again lead to key recovery

Introduction

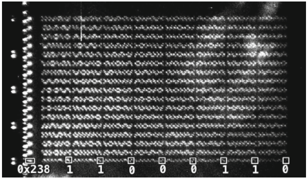
- Sbox operation for AES cipher, using LUT

Secret: $k = k_7 k_6 \ldots k_0$

$index = input \oplus k$
$y = LUT(index)$

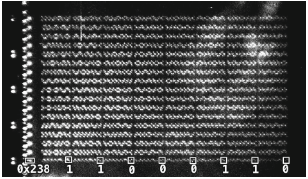
- Data leakage: Observing and analyzing the leakage of k or y
- Location leakage: Observing the table cell accessed
- Learning the accessed position on the LUT is equivalent to learning the value of y
- Thus location information can again lead to key recovery

Introduction



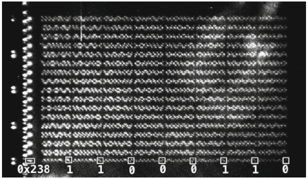
- Depending on the LUT position accessed the adversary can distinguish the AES key byte
- Still this may be hard using the coarse EM emission and requires more precise forms of leakage
- Simple photonic emission analysis of AES, *Schlösser et al.*

Introduction



- Depending on the LUT position accessed the adversary can distinguish the AES key byte
- Still this may be hard using the coarse EM emission and requires more precise forms of leakage
- Simple photonic emission analysis of AES, *Schlösser et al.*

Introduction



- Depending on the LUT position accessed the adversary can distinguish the AES key byte
- Still this may be hard using the coarse EM emission and requires more precise forms of leakage
- Simple photonic emission analysis of AES, *Schlösser et al.*

Introduction

- We have seen location attacks against symmetric & asymmetric cryptography, targeting 2 large registers or 256 small memory cells, with EM leakage or optical emission
- Is it time to revisit it?

1. Have we modeled this line of attack adequately?
2. Can we quantify their impact?
3. Can we exploit their full potential?

Introduction

- We have seen location attacks against symmetric & asymmetric cryptography, targeting 2 large registers or 256 small memory cells, with EM leakage or optical emission
- Is it time to revisit it?

1. Have we modeled this line of attack adequately?
2. Can we quantify their impact?
3. Can we exploit their full potential?

Introduction

- We have seen location attacks against symmetric & asymmetric cryptography, targeting 2 large registers or 256 small memory cells, with EM leakage or optical emission
- Is it time to revisit it?

1. Have we modeled this line of attack adequately?
2. Can we quantify their impact?
3. Can we exploit their full potential?

Introduction

- We have seen location attacks against symmetric & asymmetric cryptography, targeting 2 large registers or 256 small memory cells, with EM leakage or optical emission
- Is it time to revisit it?

1. Have we modeled this line of attack adequately?
2. Can we quantify their impact?
3. Can we exploit their full potential?

Introduction

- We have seen location attacks against symmetric & asymmetric cryptography, targeting 2 large registers or 256 small memory cells, with EM leakage or optical emission
- Is it time to revisit it?

1. Have we modeled this line of attack adequately?
2. Can we quantify their impact?
3. Can we exploit their full potential?

Location Leakage Model

A new model for location-based leakages

Location Leakage Model

- Device: Riscure Pinata, ARM Cortex M4, STM32F417IG
- Decapsulated chip, disabled peripherals
- Probe: ICR HH 100-27 microprobe with 75 um resolution
- XYZ table: 300 $\times$ 300 measurement grid
- Oscilloscope: LeCroy WaveRunner 8404M-MS
- Sampling rate: 1 Gsample/sec

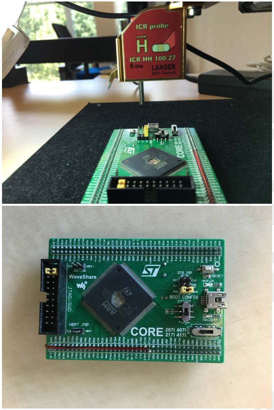


Figure: Microprobe & ARM core

Location Leakage Model

- Device: Riscure Pinata, ARM Cortex M4, STM32F417IG
- Decapsulated chip, disabled peripherals
- Probe: ICR HH 100-27 microprobe with 75 um resolution
- XYZ table: 300 $\times$ 300 measurement grid
- Oscilloscope: LeCroy WaveRunner 8404M-MS
- Sampling rate: 1 Gsample/sec

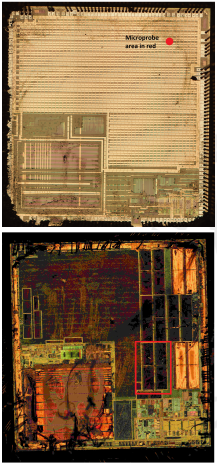
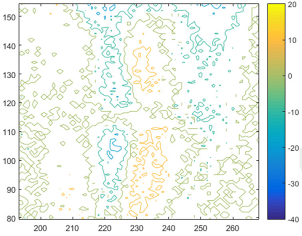


Figure: Chip surface images at layer 1 & layer 2

Location Leakage Model

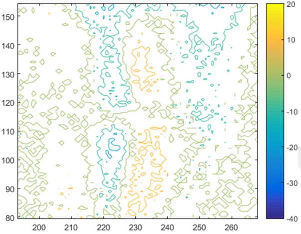
- Activate 2 SRAM regions of 8KBytes each
- Perform difference of means test



- Leaky area is approximately proportional to the memory size
- We integrate this feature in our location leakage model

Location Leakage Model

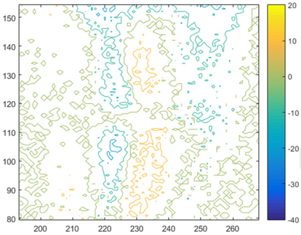
- Activate 2 SRAM regions of 8KBytes each
- Perform difference of means test



- Leaky area is approximately proportional to the memory size
- We integrate this feature in our location leakage model

Location Leakage Model

- Activate 2 SRAM regions of 8KBytes each
- Perform difference of means test



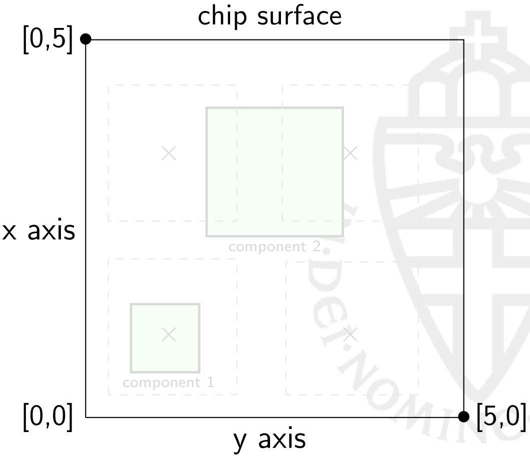
- Leaky area is approximately proportional to the memory size
- We integrate this feature in our location leakage model

Location Leakage Model

- Chip surface area
- Leakage: position & area
- EM probe & grid

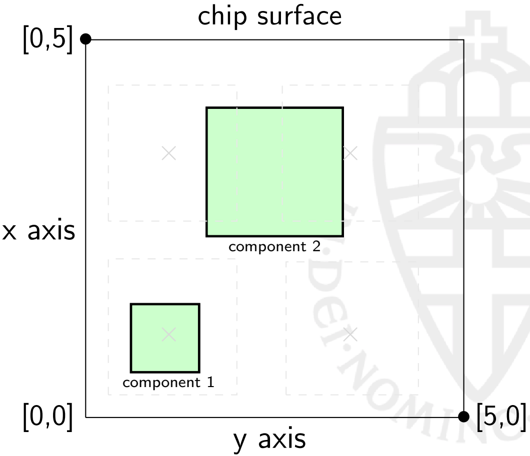
Location Leakage Model

- Chip surface area
- Leakage: position & area
- EM probe & grid



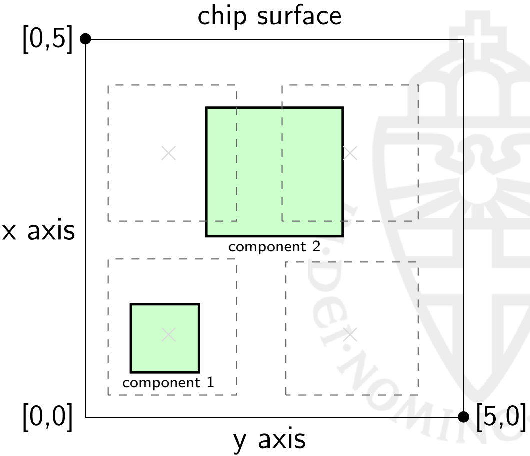
Location Leakage Model

- Chip surface area
- Leakage: position & area
- EM probe & grid



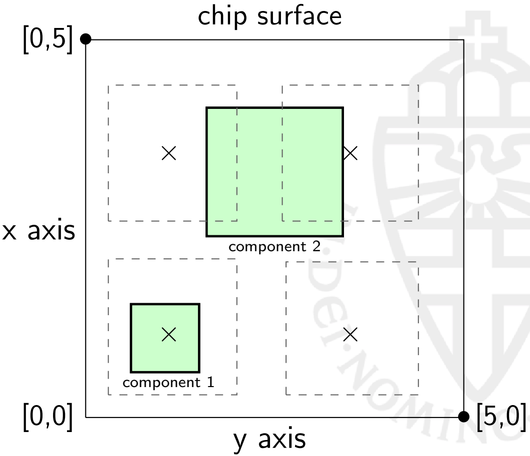
Location Leakage Model

- Chip surface area
- Leakage: position & area
- EM probe & grid



Location Leakage Model

- Chip surface area
- Leakage: position & area
- EM probe & grid

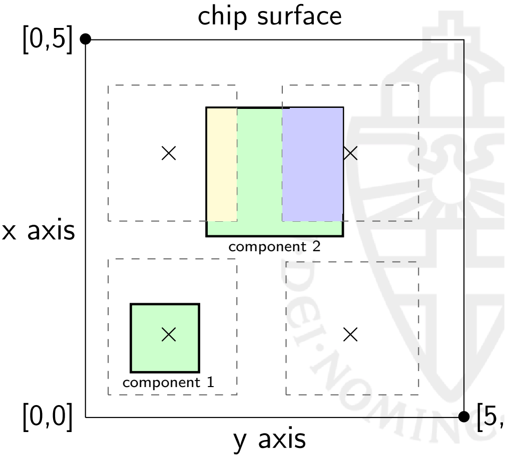


Location Leakage Model

Leakage function at position x,y:
$L_{[x,y]} = l^{det}_{[x,y]} + Noise$

Deterministic part wrt component i:

$$l^{det}_{[x,y]} = \begin{cases} 0 \\ d_i, \ 0 < d_i < area_i \\ area_i, \end{cases}$$



Information-Theoretic Analysis

- A multitude of parameters is defined in an location scan experiment: {*surface, scan grid, leaky components, probe*}
- The model can help the analyst to gauge the impact of the such parameters on the security level
- We simulate a 256-byte LUT using the defined model
- We use the Perceived Information (PI) formula to estimate the security level for different parameters

$$PI(\mathbf{L}; R) = H[R] + \sum_{r \in \mathcal{R}} Pr[r] \cdot \int_{\mathbf{l} \in \mathcal{L}^{g^2}} Pr_{true}[\mathbf{l}|r] \cdot log_2 Pr_{model}[r|\mathbf{l}] \, d\mathbf{l}$$

$\Pr_{model}[r|\mathbf{l}] = \frac{Pr_{model}[\mathbf{l}|r]}{\sum_{r^* \in \mathcal{R}} Pr_{model}[\mathbf{l}|r^*]}$, $Pr_{true}[\mathbf{l}|r] = \frac{1}{n_{test}}$, n_{test} test set size, R: region variable, g: scan grid dimension

Information-Theoretic Analysis

- A multitude of parameters is defined in an location scan experiment: {*surface, scan grid, leaky components, probe*}
- The model can help the analyst to gauge the impact of the such parameters on the security level
- We simulate a 256-byte LUT using the defined model
- We use the Perceived Information (PI) formula to estimate the security level for different parameters

$$PI(\mathbf{L};R) = H[R] + \sum_{r\in\mathcal{R}} Pr[r] \cdot \int_{\mathbf{l}\in\mathcal{L}^{g^2}} Pr_{true}[\mathbf{l}|r] \cdot log_2 Pr_{model}[r|\mathbf{l}]\, d\mathbf{l}$$

$\mathrm{Pr}_{model}[r|\mathbf{l}] = \frac{Pr_{model}[\mathbf{l}|r]}{\sum_{r^*\in\mathcal{R}} Pr_{model}[\mathbf{l}|r^*]}$, $Pr_{true}[\mathbf{l}|r] = \frac{1}{n_{test}}$, n_{test} test set size,
R: region variable, g: scan grid dimension

Information-Theoretic Analysis

- A multitude of parameters is defined in an location scan experiment: {*surface, scan grid, leaky components, probe*}
- The model can help the analyst to gauge the impact of the such parameters on the security level
- We simulate a 256-byte LUT using the defined model
- We use the Perceived Information (PI) formula to estimate the security level for different parameters

$$PI(\mathbf{L}; R) = H[R] + \sum_{r \in \mathcal{R}} Pr[r] \cdot \int_{\mathbf{l} \in \mathcal{L}^{g^2}} Pr_{true}[\mathbf{l}|r] \cdot log_2 Pr_{model}[r|\mathbf{l}] \, d\mathbf{l}$$

$\mathrm{Pr}_{model}[r|\mathbf{l}] = \frac{Pr_{model}[\mathbf{l}|r]}{\sum_{r^* \in \mathcal{R}} Pr_{model}[\mathbf{l}|r^*]}$, $Pr_{true}[\mathbf{l}|r] = \frac{1}{n_{test}}$, n_{test} test set size,
R: region variable, g: scan grid dimension

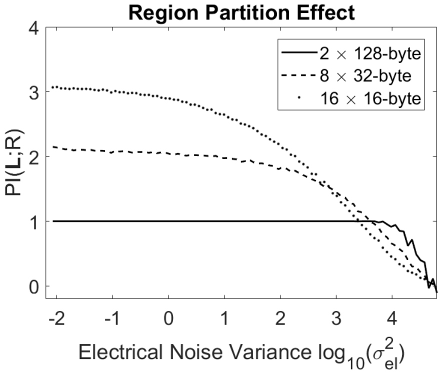
Information-Theoretic Analysis

- A multitude of parameters is defined in an location scan experiment: {*surface, scan grid, leaky components, probe*}
- The model can help the analyst to gauge the impact of the such parameters on the security level
- We simulate a 256-byte LUT using the defined model
- We use the Perceived Information (PI) formula to estimate the security level for different parameters

$$PI(\mathbf{L};R) = H[R] + \sum_{r\in\mathcal{R}} Pr[r] \cdot \int_{\mathbf{l}\in\mathcal{L}^{g^2}} Pr_{true}[\mathbf{l}|r] \cdot log_2 Pr_{model}[r|\mathbf{l}]\ d\mathbf{l}$$

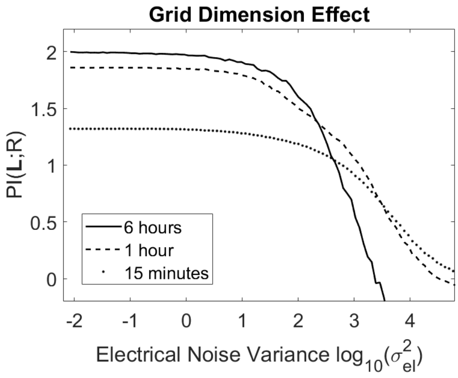
$\Pr_{model}[r|\mathbf{l}] = \frac{Pr_{model}[\mathbf{l}|r]}{\sum_{r^*\in\mathcal{R}} Pr_{model}[\mathbf{l}|r^*]}$, $Pr_{true}[\mathbf{l}|r] = \frac{1}{n_{test}}$, n_{test} test set size, R: region variable, g: scan grid dimension

Information-Theoretic Analysis



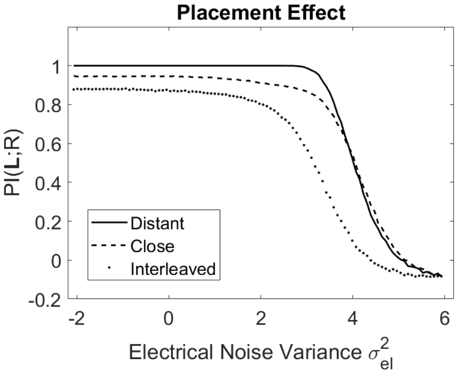
- Higher region granularity can yield more information
- Distinguishing many small regions is hard and doesn't yield max information
- The designer can assess the vulnerability threat in the cipher context

Information-Theoretic Analysis



- Scanning over the surface is time consuming
- Dimensions: $100 \times 100, 40 \times 40, 20 \times 20$
- The model assists us to strike a balance between time and effectiveness

Information-Theoretic Analysis



- Different placement of SRAM cells can hinder the adversary
- Placements: distant, close, interleaved
- High proximity leads to less information

Real-World Attacks

Real-World Attacks

Location-based leakage exploitation

Real-World Attacks

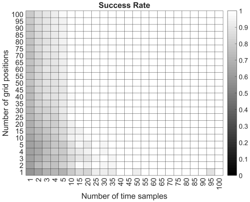
- Using actual measurements, we perform standard multivariate template attacks
- We use correlation-based spatial and temporal POI selection
- As in the IT model, we gauge the effect of experimental parameters
- I.e. we assess the effect of component number, scan grid size and placement on the real-world security level

Real-World Attacks

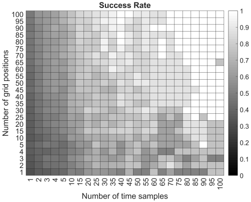
- Using actual measurements, we perform standard multivariate template attacks
- We use correlation-based spatial and temporal POI selection
- As in the IT model, we gauge the effect of experimental parameters
- I.e. we assess the effect of component number, scan grid size and placement on the real-world security level

Real-World Attacks

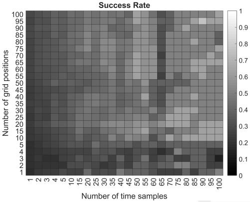
- Success Rate as function of grid positions and attack samples
- Analyzed 3 region/size configurations



(a) 2 regions of 128 bytes each



(b) 4 regions of 64 bytes each

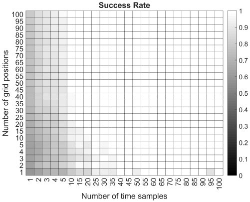


(c) 8 regions of 32 bytes each

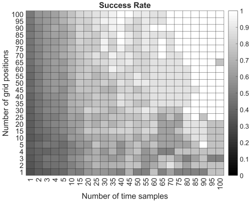
- The real-world experiment follows the model trend
- Distinguishing single bytes is not possible

Real-World Attacks

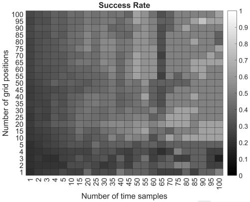
- Success Rate as function of grid positions and attack samples
- Analyzed 3 region/size configurations



(a) 2 regions of 128 bytes each



(b) 4 regions of 64 bytes each

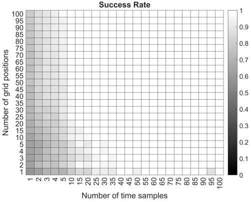


(c) 8 regions of 32 bytes each

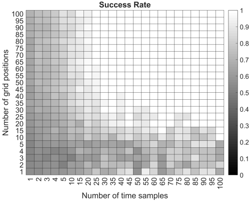
- The real-world experiment follows the model trend
- Distinguishing single bytes is not possible

Real-World Attacks

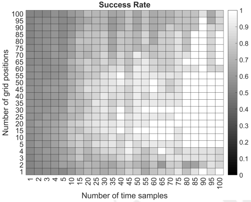
- Analyzed 3 different grid configurations
- The real-world experiment does not follow the model trend



(a) 300 × 300 grid



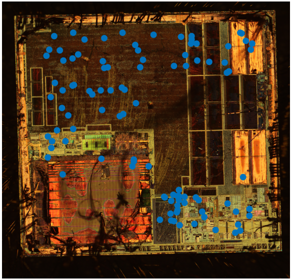
(b) 40 × 40 grid



(c) 10 × 10 grid

Real-World Attacks

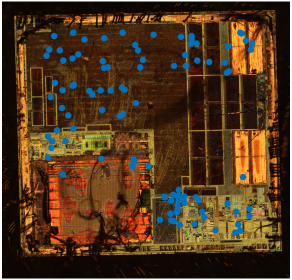
- Can we root-cause this divergence?
- Find which spatial POIs are used by our attack



- Although POI concentration is visible, outlier POIs exist as well!

Real-World Attacks

- Can we root-cause this divergence?
- Find which spatial POIs are used by our attack



- Although POI concentration is visible, outlier POIs exist as well!

Real-World Attacks

- Analyzed 3 different placement configurations
- Interleaved placement is a mild countermeasure

(a) close (b) distant (c) word-interleaved

Real-World Attacks

- Scanning a complex system-on-chip can yield highly irregular data
- Motivation point for the deployment of deep learning techniques
- Can these improve our attack w.r.t. standard templates?

1. We deployed pre-trained Convolutive Neural Networks
2. We deployed custom multi-layer perceptrons

Real-World Attacks

- Scanning a complex system-on-chip can yield highly irregular data
- Motivation point for the deployment of deep learning techniques
- Can these improve our attack w.r.t. standard templates?

1. We deployed pre-trained Convolutive Neural Networks
2. We deployed custom multi-layer perceptrons

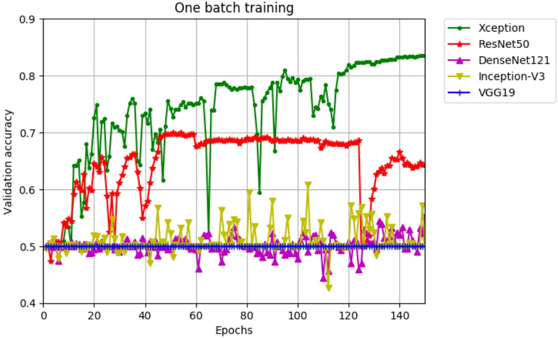
Real-World Attacks

- Scanning a complex system-on-chip can yield highly irregular data
- Motivation point for the deployment of deep learning techniques
- Can these improve our attack w.r.t. standard templates?

1. We deployed pre-trained Convolutive Neural Networks
2. We deployed custom multi-layer perceptrons

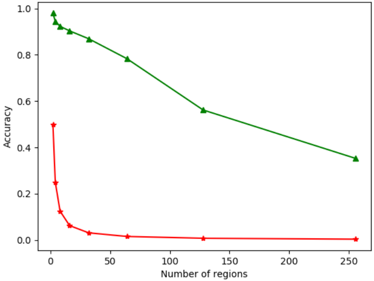
Real-World Attacks

- CNNs Xception and ResNet50 networks yield the best results
- Single-trace attack success rate, distinguishing 2 regions of 128 bytes reaches 88%
- Pretrained CNNs cannot surpass in general template attacks for a higher number of regions



Real-World Attacks

- Deployed MLP with SOFTMAX activation layer
- Single-trace attacks can reach 98% for 2 regions of 128 bytes
- Single-trace attacks reach 32% when attacking 128 bytes separately
- We see potential, since it outperforms Templates and CNNs



Conclusions & Future Work

1. Modern ARM cores are vulnerable to location leakages
2. Not only asymmetric but also symmetric cryptography can be targeted
3. We established a location leakage model, yet we need more realistic simulation
4. Similar efforts in Towards efficient and automated side channel evaluations at design time, *Sijacic et al.* and in Efficient simulation of EM side-channel attack resilience, *Kumar et al.*
5. We demonstrated the efficacy of standard templates, yet underlined the potential of deep learning techniques

Conclusions & Future Work

1. Modern ARM cores are vulnerable to location leakages
2. Not only asymmetric but also symmetric cryptography can be targeted
3. We established a location leakage model, yet we need more realistic simulation
4. Similar efforts in Towards efficient and automated side channel evaluations at design time, *Sijacic et al.* and in Efficient simulation of EM side-channel attack resilience, *Kumar et al.*
5. We demonstrated the efficacy of standard templates, yet underlined the potential of deep learning techniques

Conclusions & Future Work

1. Modern ARM cores are vulnerable to location leakages
2. Not only asymmetric but also symmetric cryptography can be targeted
3. We established a location leakage model, yet we need more realistic simulation
4. Similar efforts in Towards efficient and automated side channel evaluations at design time, *Sijacic et al.* and in Efficient simulation of EM side-channel attack resilience, *Kumar et al.*
5. We demonstrated the efficacy of standard templates, yet underlined the potential of deep learning techniques

Conclusions & Future Work

1. Modern ARM cores are vulnerable to location leakages
2. Not only asymmetric but also symmetric cryptography can be targeted
3. We established a location leakage model, yet we need more realistic simulation
4. Similar efforts in Towards efficient and automated side channel evaluations at design time, *Sijacic et al.* and in Efficient simulation of EM side-channel attack resilience, *Kumar et al.*
5. We demonstrated the efficacy of standard templates, yet underlined the potential of deep learning techniques

Conclusions & Future Work

1. Modern ARM cores are vulnerable to location leakages
2. Not only asymmetric but also symmetric cryptography can be targeted
3. We established a location leakage model, yet we need more realistic simulation
4. Similar efforts in Towards efficient and automated side channel evaluations at design time, *Sijacic et al.* and in Efficient simulation of EM side-channel attack resilience, *Kumar et al.*
5. We demonstrated the efficacy of standard templates, yet underlined the potential of deep learning techniques