Solving Concealed ILWE and its Application for Breaking Masked Dilithium

Simon Damm[1], Asja Fischer[1], Soundes Marzougui[2], Alexander May[1],
Leander Schwarz[3], Henning Seidler[3], Jean-Pierre Seifert[3],
Jonas Thietke[1] <jonas.thietke@rub.de>, Vincent Ulitzsch[3,4] <viniul@mit.edu>

[1] Faculty of Computer Science, Ruhr University Bochum
[2] STMicroelectronics, Belgium
[3] Technische Universität Berlin – SecT
[4] Massachusetts Institute Of Technology

Summary - Contribution

1	***Concealed ILWE**:* A variant of the Integer Learning With Errors Problem that naturally occur in side-channel attacks on Fiat-Shamir With Aborts Lattice-Based Signature Schemes.
2	Introducing **robust regression** to cryptanalysis, yielding algorithms to solve concealed ILWE instances not solvable by prior work.
3	A **novel power side-channel attack on first-order masked Dilithium, breaking all security levels.**

Fiat-Shamir With Aborts Signatures Hide a Linear Relationship

Private Key: $s \in \mathbb{Z}^n$ with small coefficients in $\{-\eta, \eta\}$.

Signature: $(z = y + \langle c, s \rangle, c)$

Nonce $y \in \mathbb{Z}$ is sampled so that z does not reveal any information about s . y's coefficients are in $\{-\gamma_1, \dots, \gamma_1\}$; $\gamma_1 \gg \eta$.

$c \in \mathbb{Z}^n$ is a **public** challenge **derived from the message** to be signed.

Fiat-Shamir With Aborts Signatures Hide a Linear Relationship

Private Key: $s \in \mathbb{Z}^n$ with small coefficients in $\{-\eta, \eta\}$.

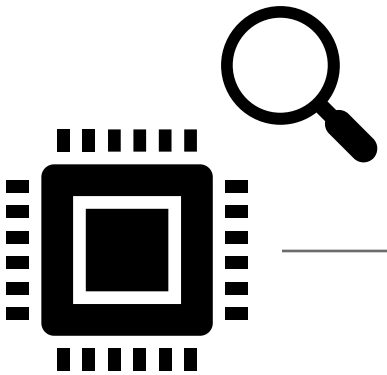
Signature: $(z = y + \langle c, s \rangle, c)$

Nonce $y \in \mathbb{Z}$ is sampled so that z does not reveal any information about s . y's coefficients are in $\{-\gamma_1, \dots, \gamma_1\}$; $\gamma_1 \gg \eta$.

$c \in \mathbb{Z}^n$ is a **public** challenge **derived from the message** to be signed.

- Side-channel and fault-injection leak information about the nonce y.
- This breaks the zero-knowledge property of Fiat-Shamir With Aborts signatures.

Side-Channel Can Reveal Information About y



Attacker uses side-channel to guess whether $\boldsymbol{y = 0}$ and adds equation $z = \langle \boldsymbol{c}, \boldsymbol{s} \rangle$ to equation system.

Aggregate resulting signatures in an equation system and solve for secret key.

$$z = \langle \mathbf{c} = \begin{pmatrix} 0 \\ \vdots \\ -1 \\ 1 \end{pmatrix}, \mathbf{s} = \begin{pmatrix} 1 \\ \vdots \\ 1 \\ 2 \end{pmatrix} \rangle + \mathrm{y}$$

$$\begin{aligned} z_1 &= \langle \boldsymbol{c_1}, \boldsymbol{s} \rangle + y_1 \\ z_2 &= \langle \boldsymbol{c_2}, \boldsymbol{s} \rangle + y_2 \\ z_3 &= \langle \boldsymbol{c_3}, \boldsymbol{s} \rangle + y_3 \end{aligned}$$

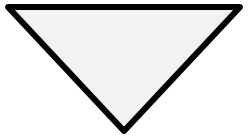
Perfect Guesses Lead to Immediate Secret Key Recovery

If all guesses for $y = 0$ are correct, the attacker can solve resulting equation system $z = Cs$ via Gaussian elimination.

$$z = \langle \mathbf{c} = \begin{pmatrix} 0 \\ \vdots \\ -1 \\ 1 \end{pmatrix}, \mathbf{s} = \begin{pmatrix} 1 \\ \vdots \\ 1 \\ 2 \end{pmatrix} \rangle + y \longrightarrow \begin{aligned} z_1 &= \langle \boldsymbol{c_1}, \boldsymbol{s} \rangle + y_1 \\ z_2 &= \langle \boldsymbol{c_2}, \boldsymbol{s} \rangle + y_2 \\ z_3 &= \langle \boldsymbol{c_3}, \boldsymbol{s} \rangle + y_3 \end{aligned}$$

Prior power side-channel and fault injection attacks on Dilithium exploit noisy a $y = 0$ oracle

Prior Power Side-Channel [UTMS24] or Fault Injection [UMB+23], can only obtain *noisy* samples $z = \langle \boldsymbol{c}, \boldsymbol{s} \rangle + y$



Can we still recover the secret key even if given only noisy information about nonce $\boldsymbol{y}$?

Side-Channel Can Reveal Information About y

$$z = \langle \boldsymbol{c} = \begin{pmatrix} 0 \\ \vdots \\ -1 \\ 1 \end{pmatrix}, \boldsymbol{s} = \begin{pmatrix} 1 \\ \vdots \\ 1 \\ 2 \end{pmatrix} \rangle + \boldsymbol{y}$$

Guess that is $\boldsymbol{y = 0}$ **correct**, attacker correctly guesses equation

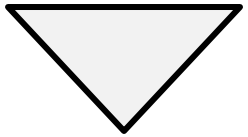
$$z_i = \langle c, s \rangle_i$$

Attacker guess that $\boldsymbol{y = 0}$ is **incorrect**, attacker adds equation

$$z_i = \langle \boldsymbol{c}, \boldsymbol{s} \rangle_i + y_i$$

Research Question: Under what conditions can side-channel attacks recover $\boldsymbol{s}$ from the resulting noisy equation system?

Research Question: Under what conditions can side-channel attacks can recover $\boldsymbol{s}$ from the resulting noisy equation system?



Contribution 1 – **Concealed ILWE**: A framework to model the result of noisy side-channel attacks on Fiat-Shamir With Abort Signature Schemes.

Summary - Contribution

1	***Concealed ILWE:*** A model for side-channel attacks on Fiat-Shamir With Abort Lattice-Based Signature Scheme.
2	Introducing **robust regression** to cryptanalysis, an algorithm to solve concealed ILWE instances not solvable by prior work.
3	A **novel power side-channel attack on first-order masked Dilithium, breaking all security levels.**

Recall: Integer Learning With Errors

Integer Learning With Errors: Given a set of ILWE samples $(z_i, \boldsymbol{c}_i) \in \mathbb{Z} \times \mathbb{Z}^n$ for $\boldsymbol{s} \in \mathbb{Z}^n$ with $z_i = \langle \boldsymbol{c}_i, \boldsymbol{s} \rangle + y_i$ with y_i drawn from some unknown distribution, recover $\boldsymbol{s}$.

We introduce the Concealed ILWE Problem, modelling side-channel attacks on Fiat-Shamir With Aborts

Concealed ILWE: Let the **secret** $\boldsymbol{s} \in \mathbb{Z}^n$, let the **concealment rate** $p \in [0,1]$.

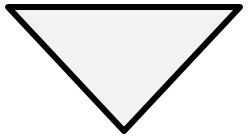
Given an oracle that returns samples $(z_i, \boldsymbol{c}_i) \in \mathbb{Z} \times \mathbb{Z}^n$ with $z_i = \boldsymbol{c}_i \boldsymbol{s} + \boldsymbol{y}_i$ with y_i:

- **Zero error**: $y_i = 0$ with probability $1 - p$
- **Zero-knowledge**: y_i is drawn such that it hides s information-theoretically with probability p

Also possible with our methods: Small, independent error (for other side-channel attacks)

Research Question: How can an attacker recover $\boldsymbol{s}$ from CILWE instances?

Research Question: How can an attacker recover s from CILWE instances?



Contribution 2 – **Robust regression**: Introducing methods novel to cryptanalysis to solve CILWE instances that were out of reach by prior work.

Existing methods to solve ILWE fall short for CILWE

Two methods exists to solve variants of ILWE:

Integer Linear Programming: No polynomial runtime guarantees, does not model small, independent errors [UMTS24].
Linear Least Squares regression: Does not extend to Concealed ILWE [BDE+18].

Generic Algorithm for ILWE [BDE+18]

High-Level Linear Regression Attack on ILWE

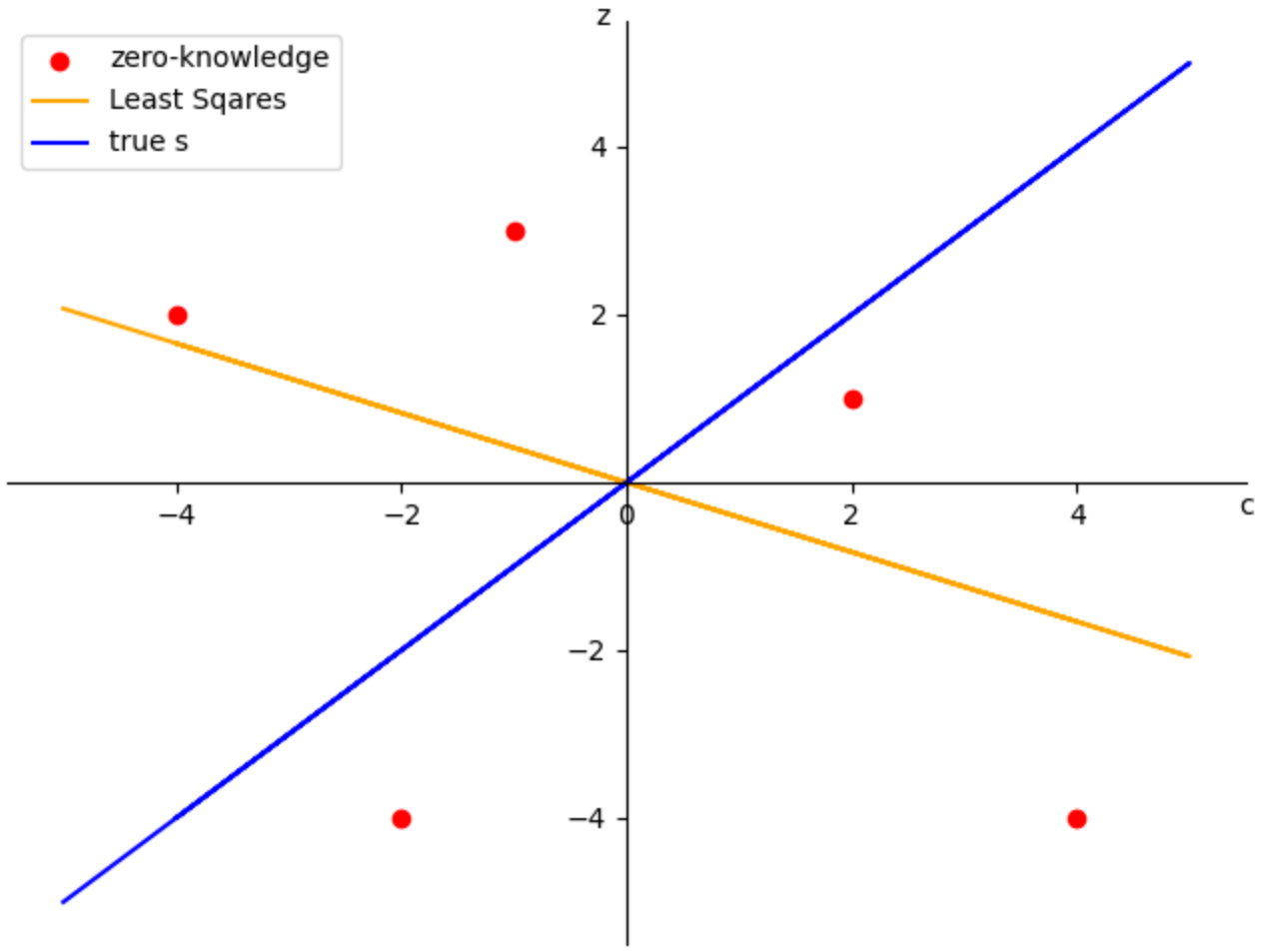
Given samples $(z_i, \boldsymbol{c}_i)$:
Formulate Regression Problem $\boldsymbol{z} = \boldsymbol{C}\boldsymbol{s} + \mathbf{y}$
1. Compute estimate $\hat{\boldsymbol{s}} \in \mathbb{R}^n$
2. Round to nearest integer, $\boldsymbol{s} = \lfloor \hat{\boldsymbol{s}} \rceil$

- Algorithm in the literature so far: Least Squares Regression [BDE+18]
  - Requires all errors y to be independent identically distributed (not applicable for CILWE)
- **Question**: Can we find a way to ignore all zero-knowledge samples?

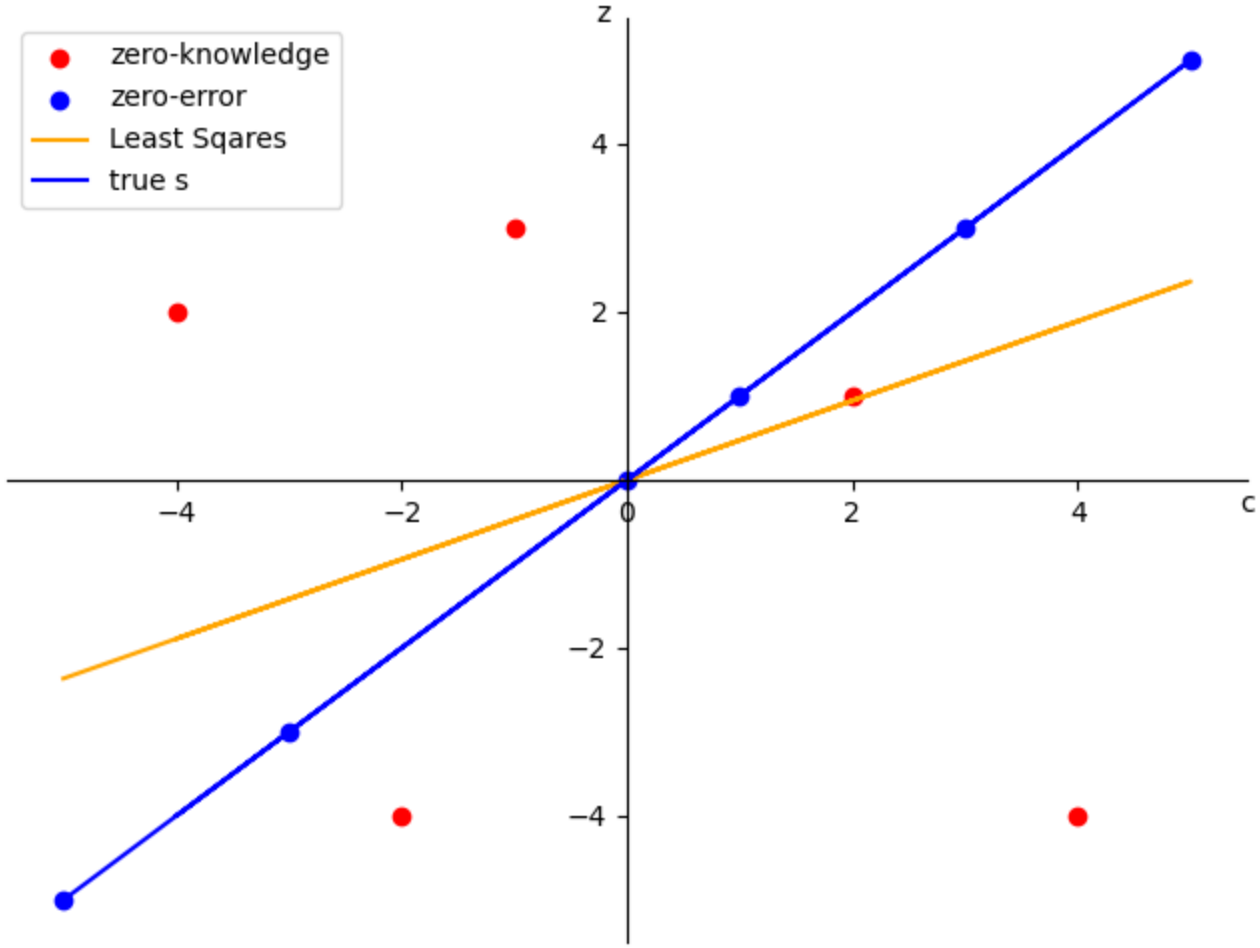
Summary - Contribution

1	***Concealed ILWE:*** A model for side-channel attacks on Fiat-Shamir With Abort Lattice-Based Signature Scheme.
2	Introducing **robust regression** to cryptanalysis, an algorithm to solve concealed ILWE instances not solvable by prior work.
3	A **novel power side-channel attack on first-order masked Dilithium, breaking all security levels.**

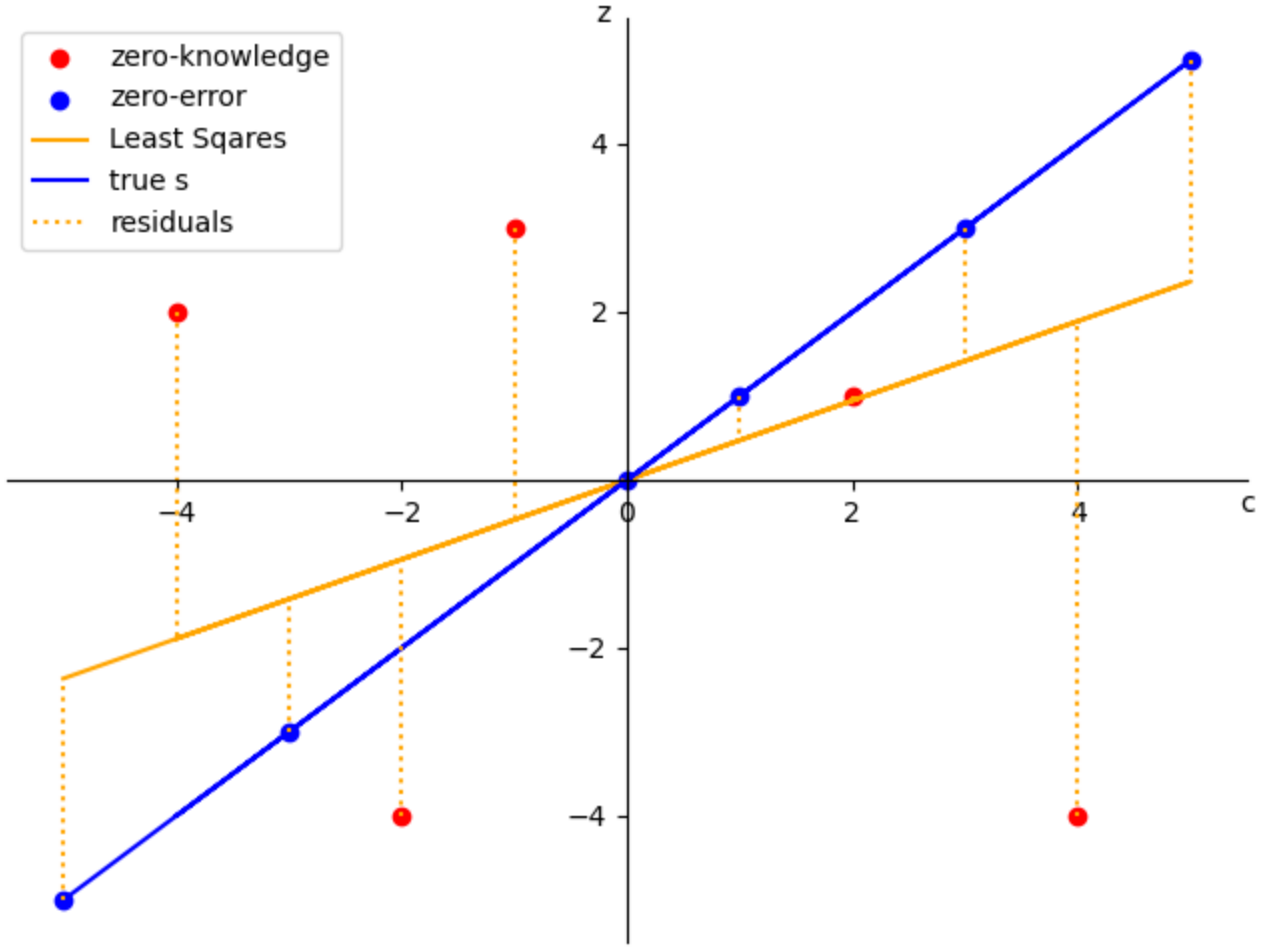
Key Recovery with Least Squares – only zero-knowledge samples



Key Recovery with Least Squares – Concealed ILWE samples



Residuals



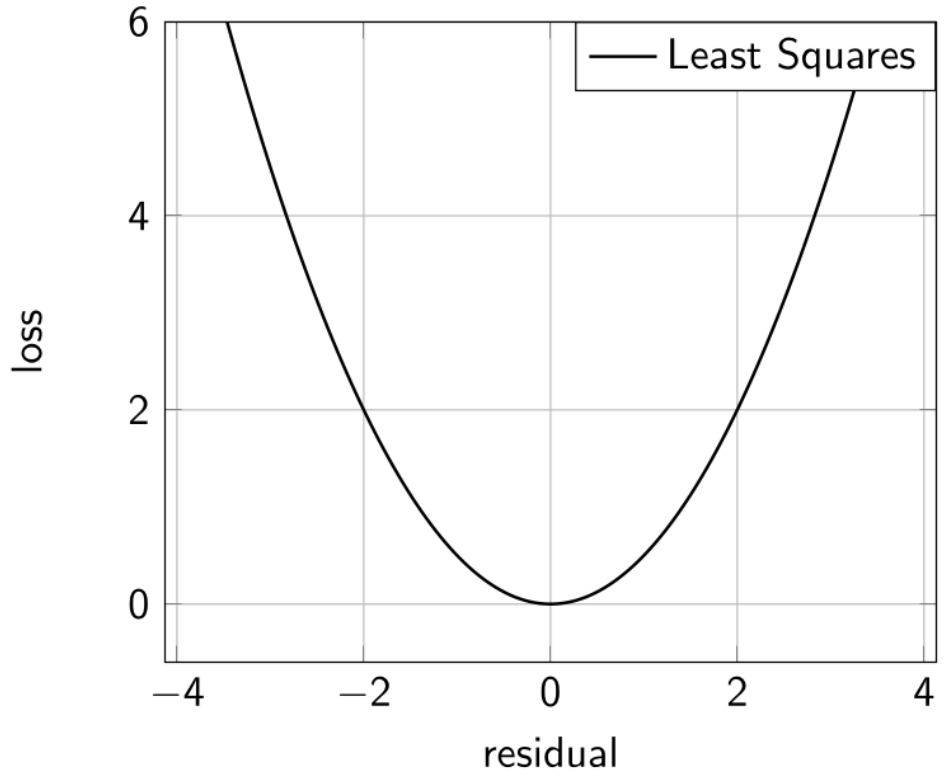
Let $\boldsymbol{s}' \in \mathbb{Z}^n$. A *residual* is the estimated error

$$r_i(\boldsymbol{s}_i) = z_i - < \boldsymbol{c}_i, \boldsymbol{s}' >$$

Robust Regression

Loss functions $\rho(r)$:

Least Squares: $\rho(r) = \frac{1}{2}r^2$

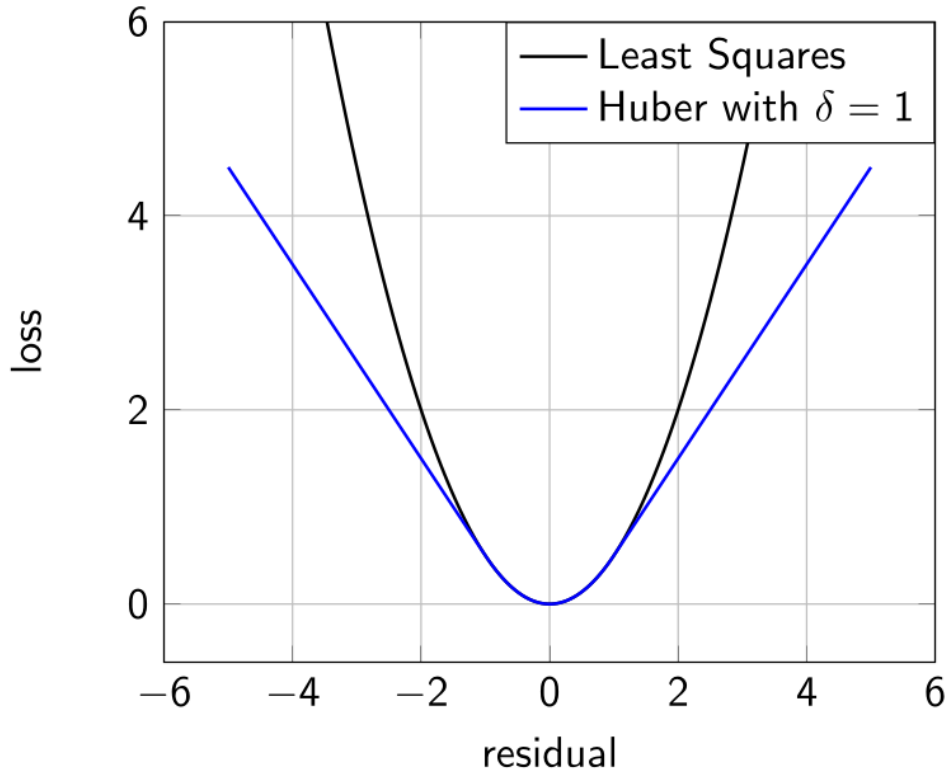


Robust Regression

Loss functions $\rho(r)$:

Least Squares: $\rho(r) = \frac{1}{2}r^2$

Huber: $\rho(r) = \begin{cases} \frac{1}{2}r^2, & r \leq \delta \\ \delta(|r| - \frac{1}{2}\delta), & r > \delta \end{cases}$



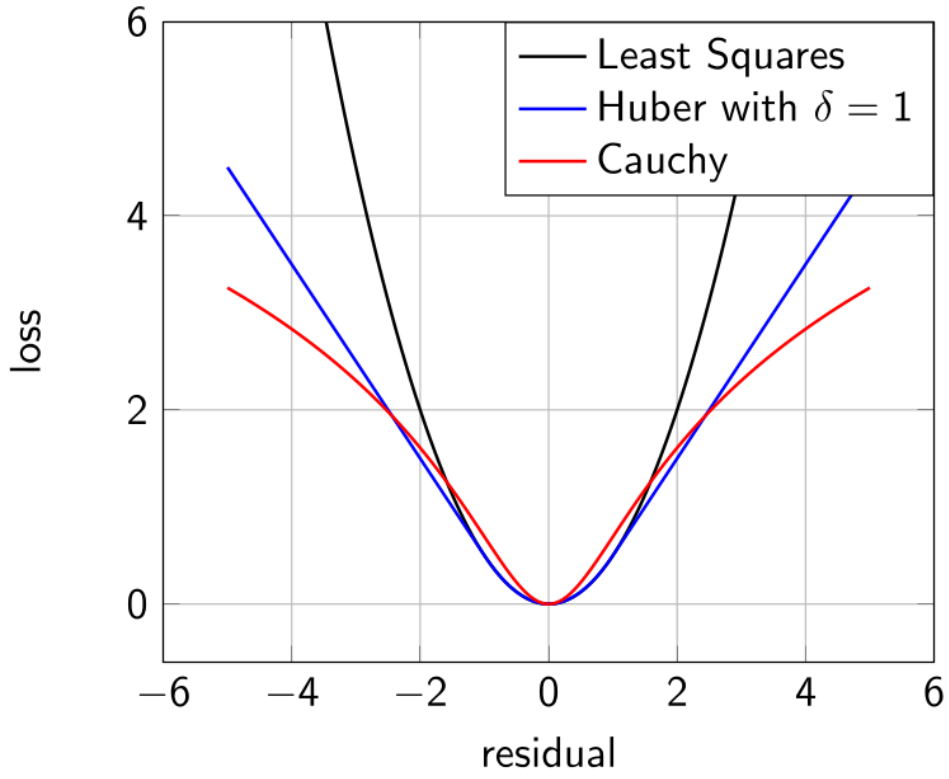
Robust Regression

Loss functions $\rho(r)$:

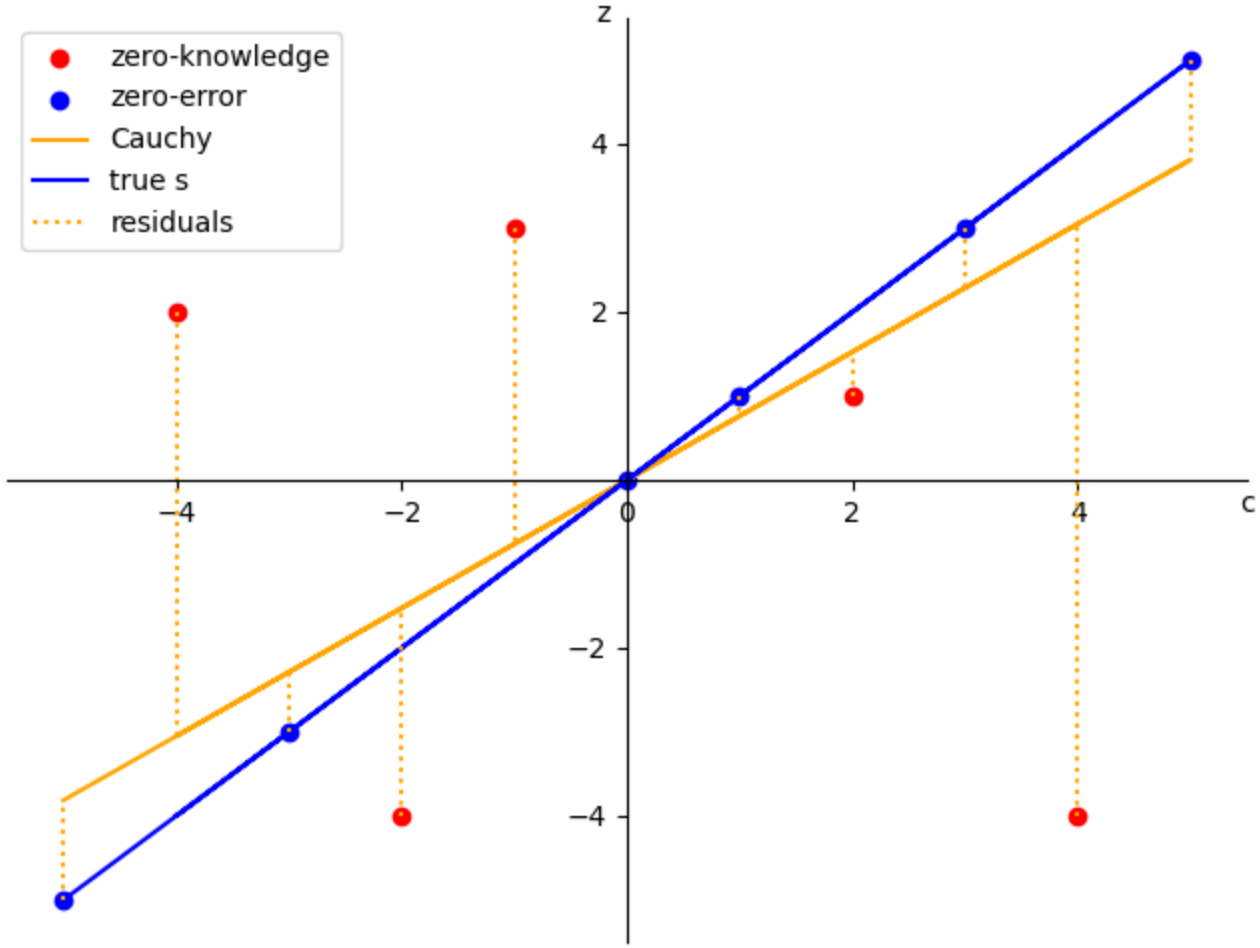
Least Squares: $\rho(r) = \frac{1}{2}r^2$

Huber: $\rho(r) = \begin{cases} \frac{1}{2}r^2, & r \leq \delta \\ \delta(|r| - \frac{1}{2}\delta), & r > \delta \end{cases}$

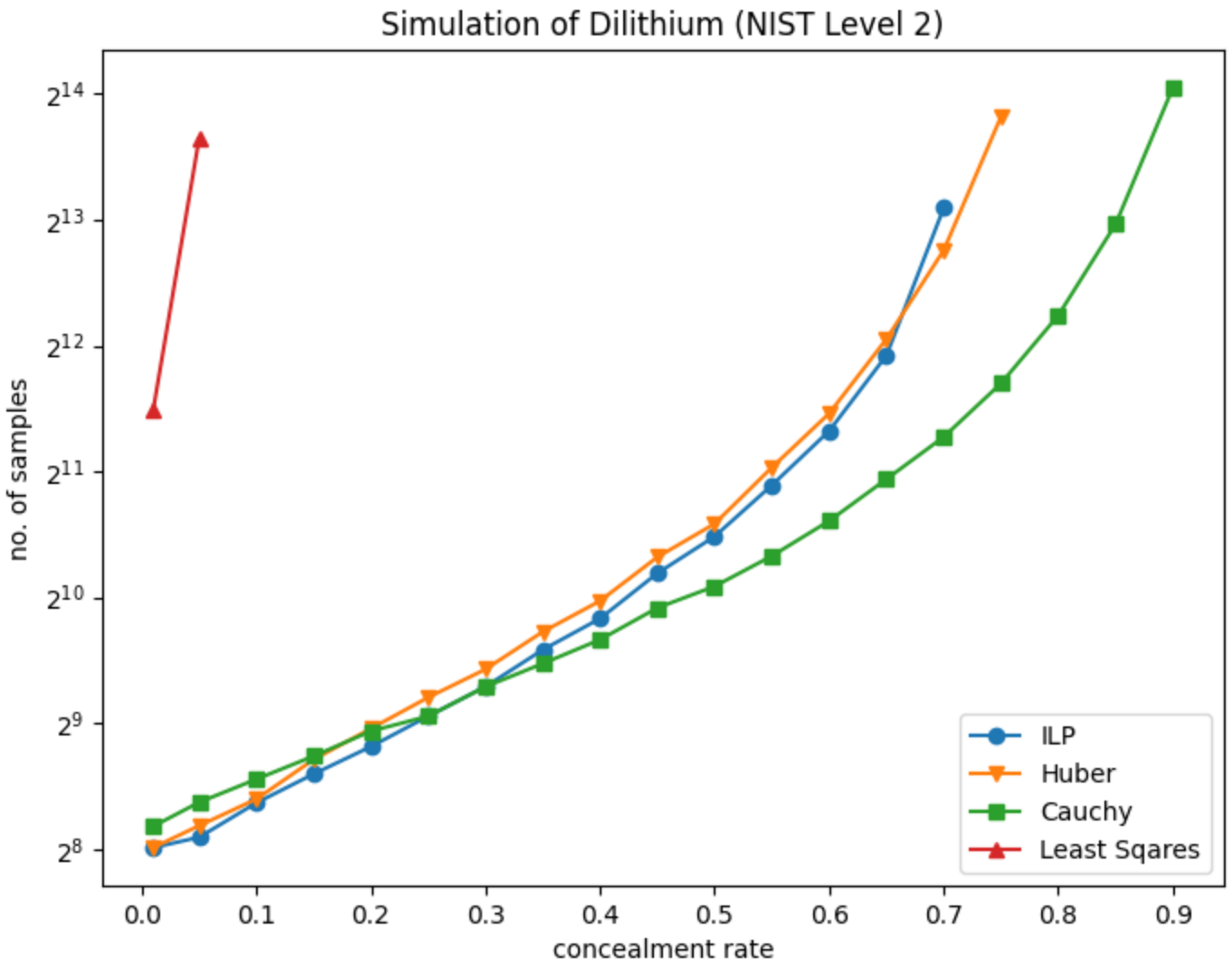
Cauchy: $\rho(r) = \ln(1 + r^2)$



Cauchy Regression on our Example



Performance of Robust Regression in Simulation

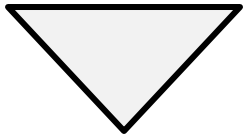


Results on CILWE instances:

- Least squares not suitable
- Cauchy very robust to high concealment
- Huber and ILP comparable, but Huber is faster

Research Question: Does the advantage of robust regression threaten security in practice?

Research Question: Does the advantage of robust regression threaten security in practice?



Contribution 3 – **Breaking Masked Dilithium**: A **novel power side-channel attack on first-order masked Dilithium, breaking all security levels.**

Summary - Contribution

1	***Concealed ILWE:*** A model for side-channel attacks on Fiat-Shamir With Abort Lattice-Based Signature Scheme.
2	Introducing **robust regression** to cryptanalysis, an algorithm to solve concealed ILWE instances not solvable by prior work.
3	A **novel power side-channel attack on first-order masked Dilithium, breaking all security levels.**

Robust regression enables attack on recently proposed masked Dilithium implementation

- Prior work at CHES'23 proposed a masked Dilithium implementation [CGTZ23]
  - Provably secure under t-probing model
- Recall: Masking splits secret y into independent shares $y = y_1 \oplus y_2$

Key Idea:

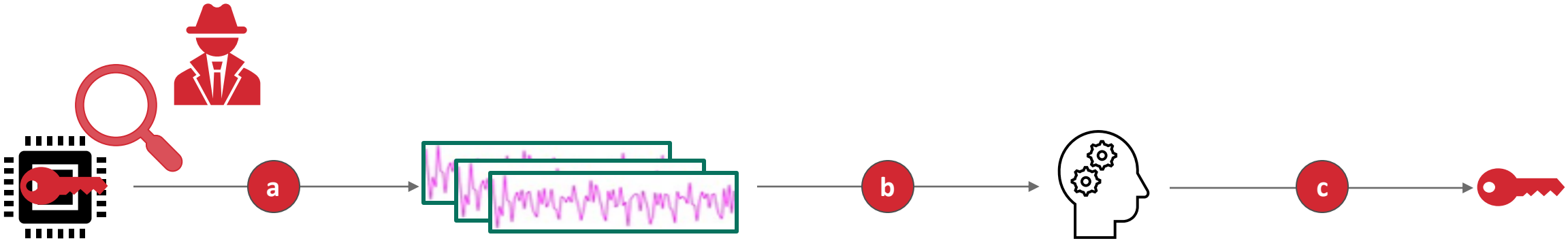
- Real-world implementation still leaks information when executed on ARM Cortex M4
  - which can be exploited through robust regression, but not with prior work

Improved Gadgets for the High-Order Masking of Dilithium

Jean-Sébastien Coron[1], François Gérard[1], Matthias Trannoy[1,2] and Rina Zeitoun[2]

[1] University of Luxembourg
jean-sebastien.coron@uni.lu,francois.gerard@uni.lu
[2] IDEMIA, Cryptography & Security Labs, Courbevoie, France
matthias.trannoy@idemia.com,rina.zeitoun@idemia.com

Power Side-Channel attacks exploit leakage through power traces



a Attacker measures power traces during signature computation

b Derives information from *power traces* through statistical analysis

c Recovers secret key from information

Threat Model:

- Attacker has physical access to victim device
- Can measure power during signature computation
- Can observe the signatures created by the device
- Attacker can trigger (multiple) signature generations

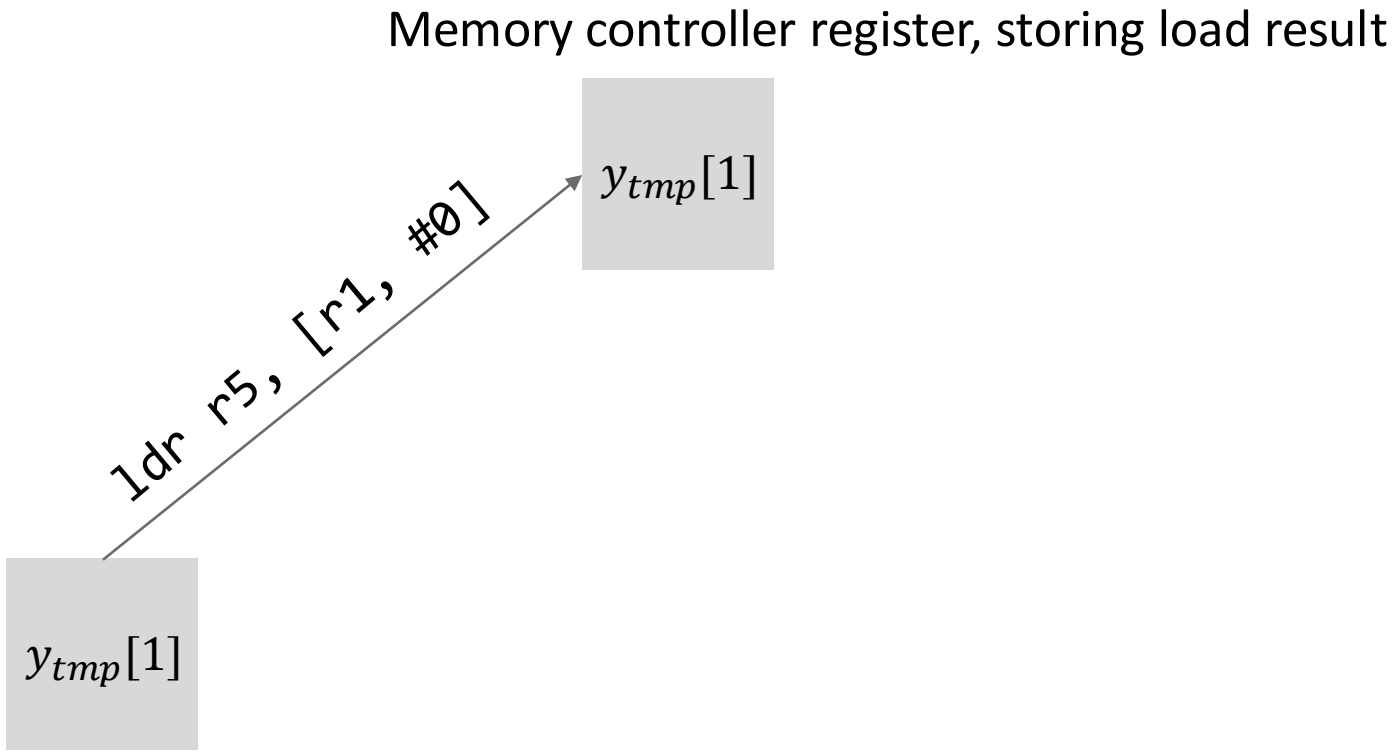
Robust regression enables a machine-learning based attack against first-order masked Dilithium implementation on STM32F4 micro-controller

1. We exploit a micro-architectural leakage to learn whether $y \geq 0$
2. This gives rise to CILWE instances with zero-knowledge, zero-error, and small, independent error samples
3. The resulting instance can be solved with Cauchy regression, whereas prior work falls short

Masking Prevent Secret Leakage By Processing Secrets in Shares

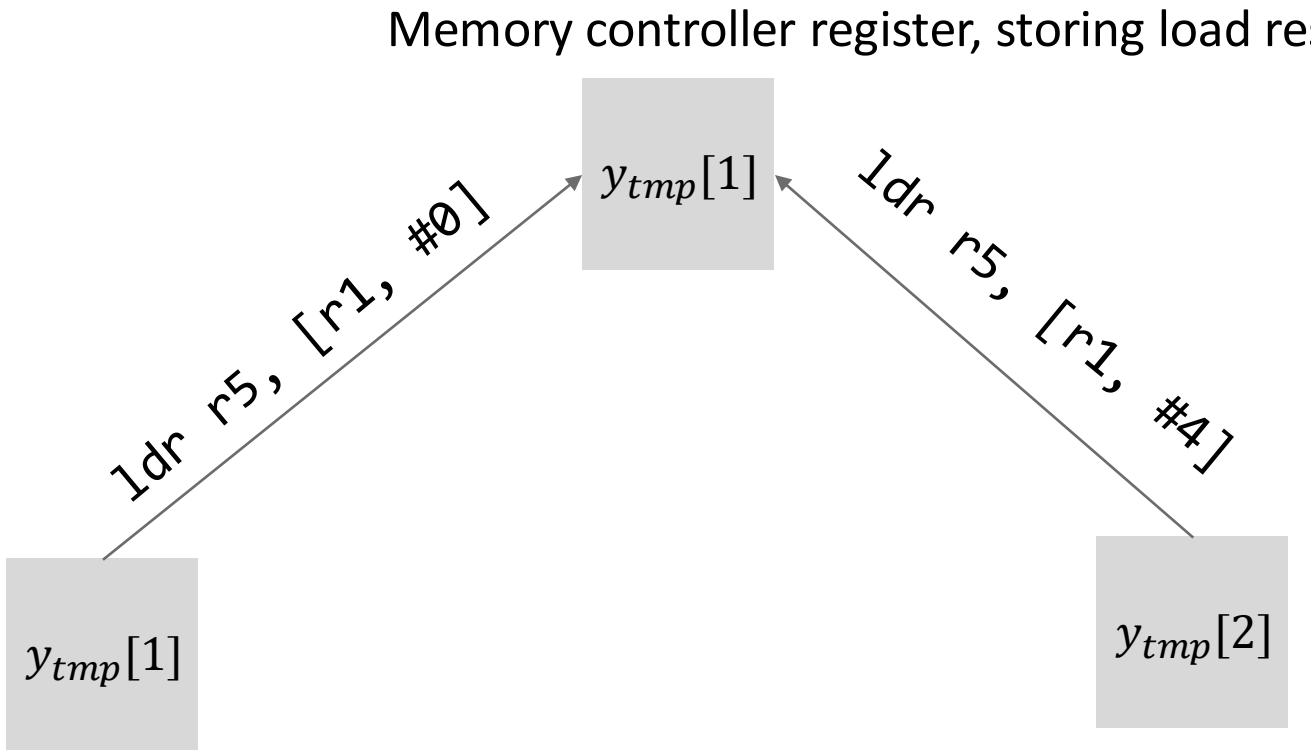
- The masked implementation samples y using Boolean shares and then converts to arithmetic masking using a B2A conversion method.
- This B2A loads shares consecutively from memory, leaking through the so-called Memory Remnant Effect.

A memory remnant effect allows to leak the hamming weight of y_i



- During boolean to arithmetic share conversion, y is implicitly stored in a masked representation y^{tmp}
- y_{tmp}‘s shares are loaded in two consecutively loads, which are buffered in an internal memory controller register

A memory remnant effect allows to leak the hamming weight of y_i



- y_{tmp} 's shares are loaded in two consecutively loads, which are buffered in an internal memory controller register
- **The second load reveals** $\boldsymbol{HW\left(y_{tmp}[1] \oplus y_{tmp}[2]\right) = HW(y)}$
- **Challenge:** The leakage is noisy

A memory remnant effect allows to leak the hamming weight of y

We can combine an imperfect machine-learning classifier with Cauchy regression to elevate this leakage into secret key recovery.

We can exploit the leakage through a profiling power side-channel attack

- **Profiling Phase**: We train, on a clone device, a machine-learning classifier to distinguish whether $y \geq 0$ during B2A conversion
  - Training set of 60k traces is sufficient
- **Attack Phase**:
  - Collect traces during signature generation on the victim device, and use our machine-learning classifier to create an CILWE instance
  - Use Cauchy Regression to recover the secret key from the resulting instances

Cauchy Regression Allows For Key Recovery On All NIST Security Levels

We can tolerate >10% zero-knowledge samples (higher is harder.)

NIST Security Level	2	3	5
Required Signatures	~600k	~2.5 Million	~2.6 Million
Zero-Knowledge	**12%**	**15%**	**14%**
Small, Independent Error	85%	83%	84%
Zero-Error	2%	1.1%	1.2%
Runtime	< 2 minutes	< 2 minutes	< 2 minutes

Cauchy regression allows for key recovery on all NIST Security Levels in minutes, while prior algorithms (least squares, ILP) fall short.

Cauchy Regression Allows For Key Recovery On All NIST Security Levels

NIST Security Level	2	3	5
Required Signatures	~600k	~2.5 Million	~2.6 Million
Zero-Knowledge	**12%**	**15%**	**14%**
Small, Independent Error	85%	83%	84%
Zero-Error	2%	1.1%	1.2%
Runtime	< 2 minutes	< 2 minutes	< 2 minutes

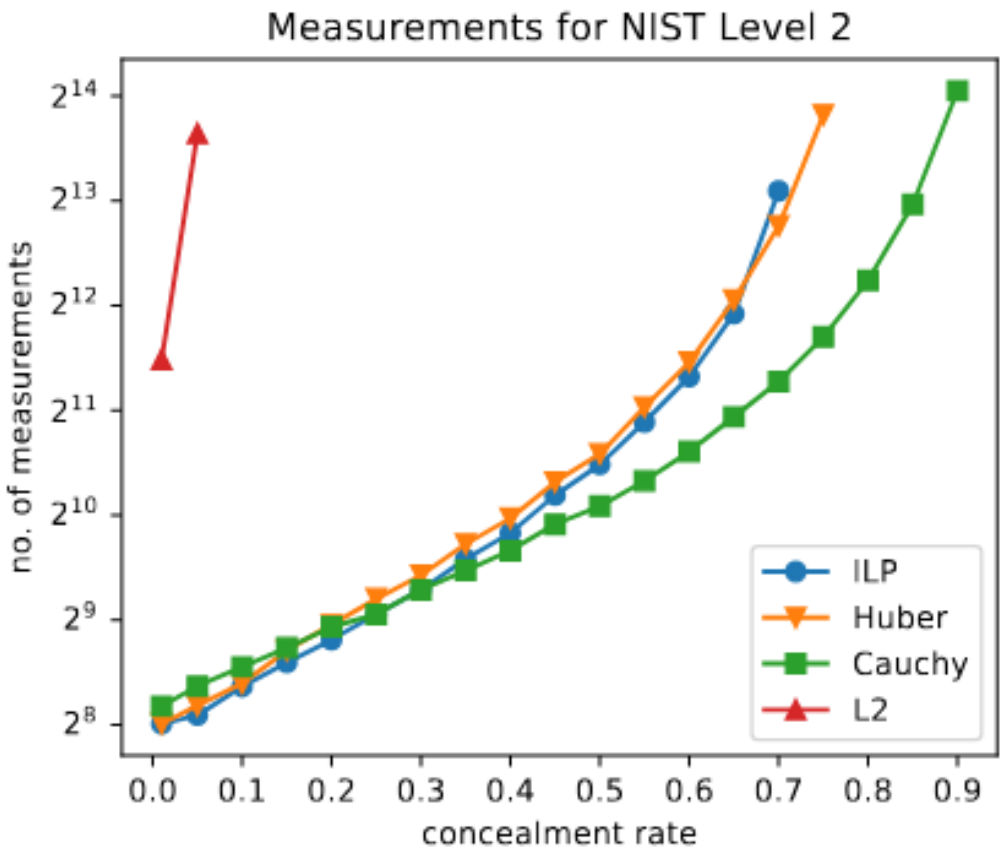
- Cauchy regression allows for key recovery on all NIST Security Levels in minutes, while prior algorithms (least squares, ILP) fall short.
- Prior attacks were only demonstrated on security level 2 [HNP25].

Conclusion – Thank you for your attention!

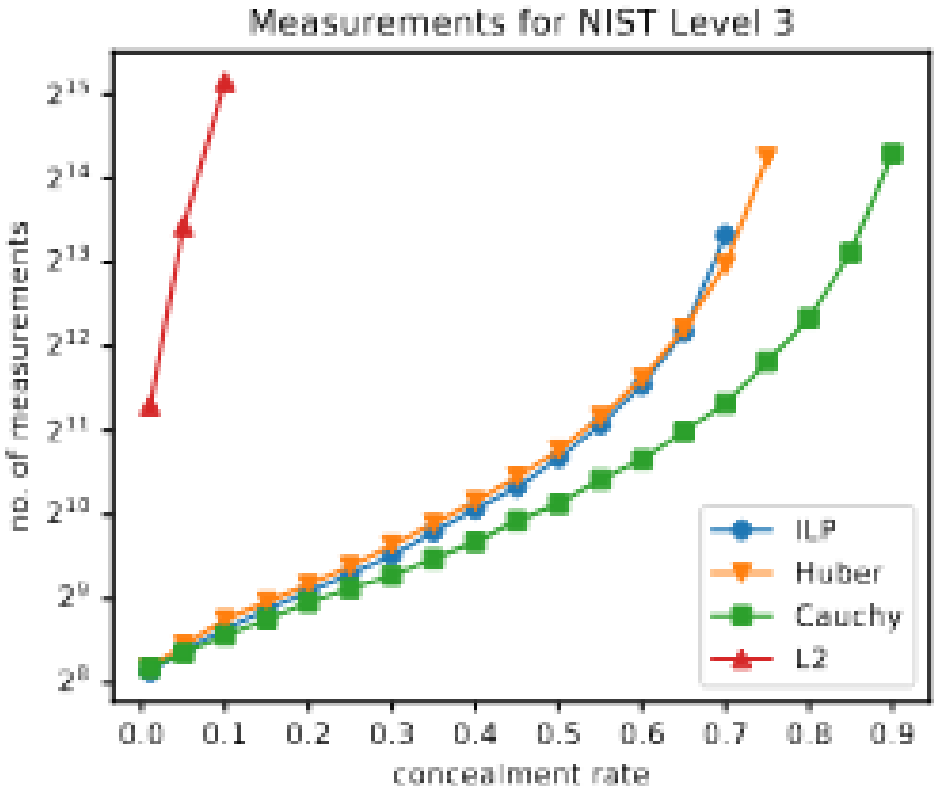
More Details in the paper:

- Sample bounds for Huber regression on Dilithium.
- Detailed analysis of micro-architectural leakage.
- More extensive comparision to prior work.

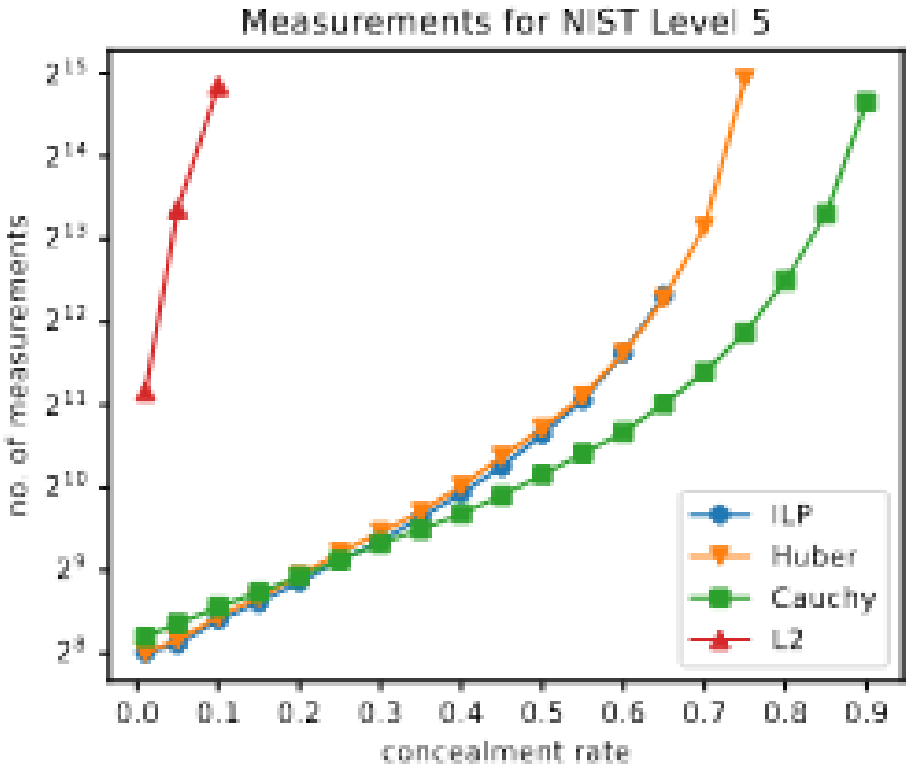
https://github.com/fgsect/concealed_ilwe


Measurements for NIST Level 2
no. of measurements
2^{14}
2^{13}
2^{12}
2^{11}
2^{10}
2^{9}
2^{8}
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
concealment rate
ILP
Huber
Cauchy
L2

(a) NIST-2


Measurements for NIST Level 3
no. of measurements
2^{15}
2^{14}
2^{13}
2^{12}
2^{11}
2^{10}
2^{9}
2^{8}
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
concealment rate
ILP
Huber
Cauchy
L2

(b) NIST-3

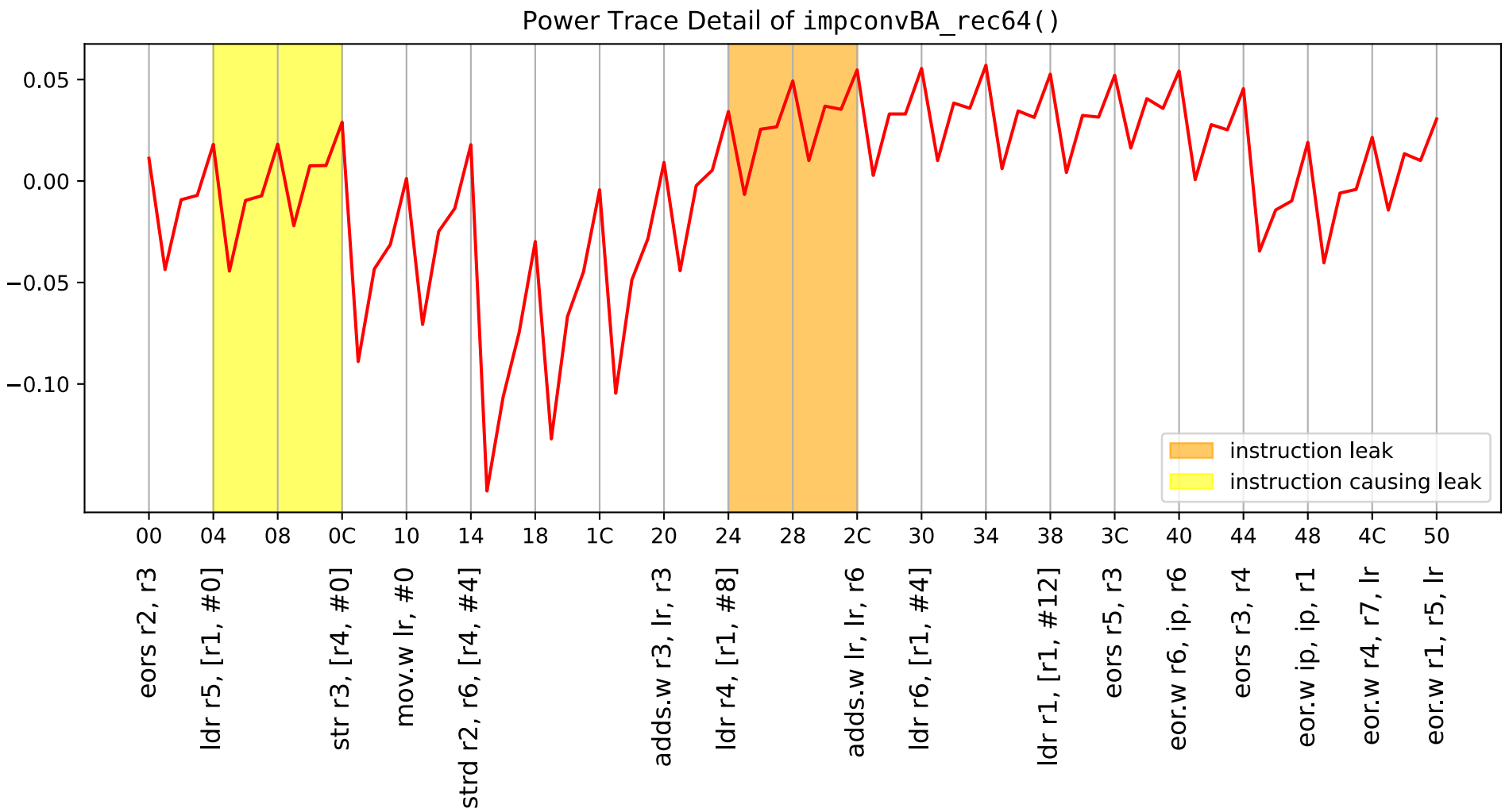

Measurements for NIST Level 5
no. of measurements
2^{15}
2^{14}
2^{13}
2^{12}
2^{11}
2^{10}
2^{9}
2^{8}
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
concealment rate
ILP
Huber
Cauchy
L2

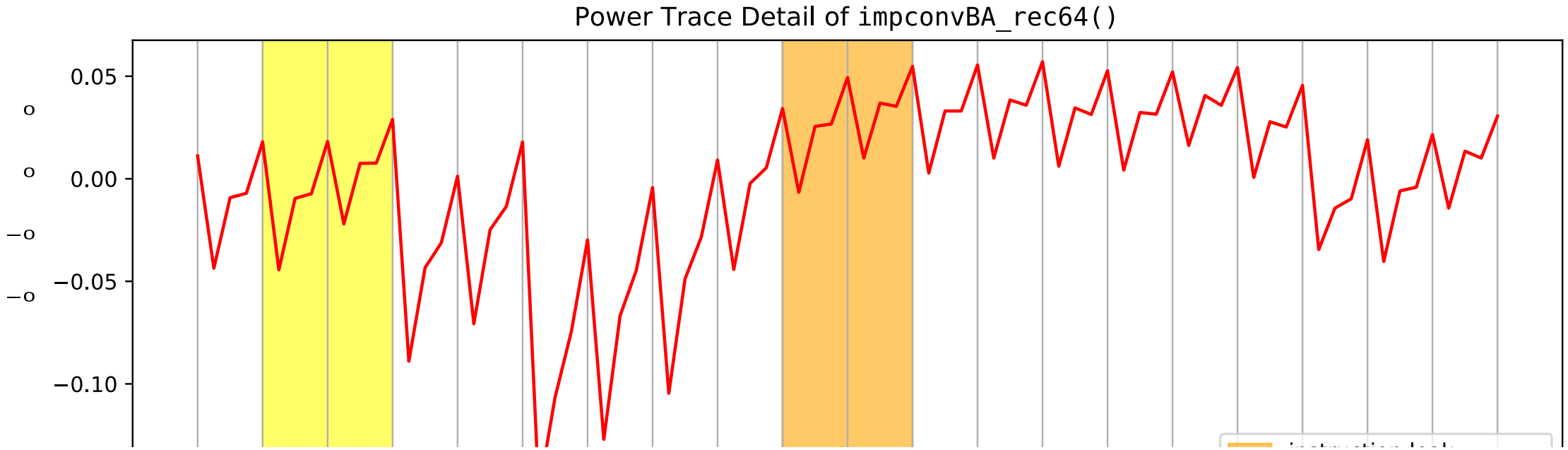
(c) NIST-5

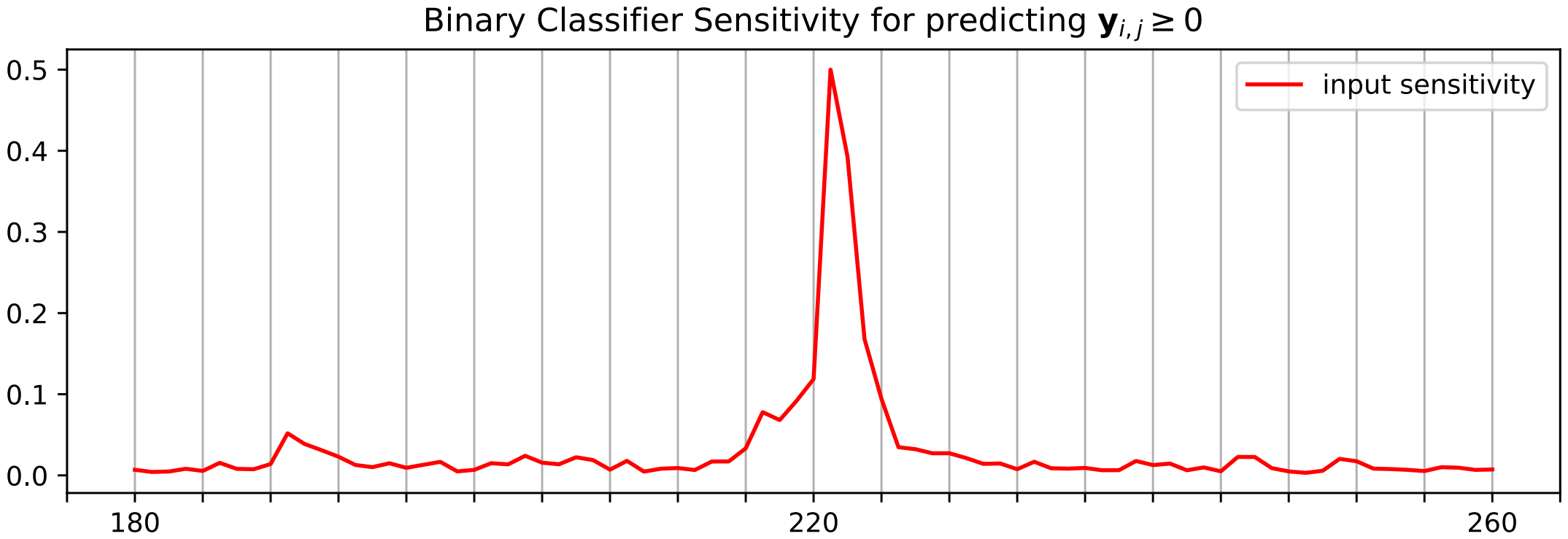
Machine Learning Classifier Statistics

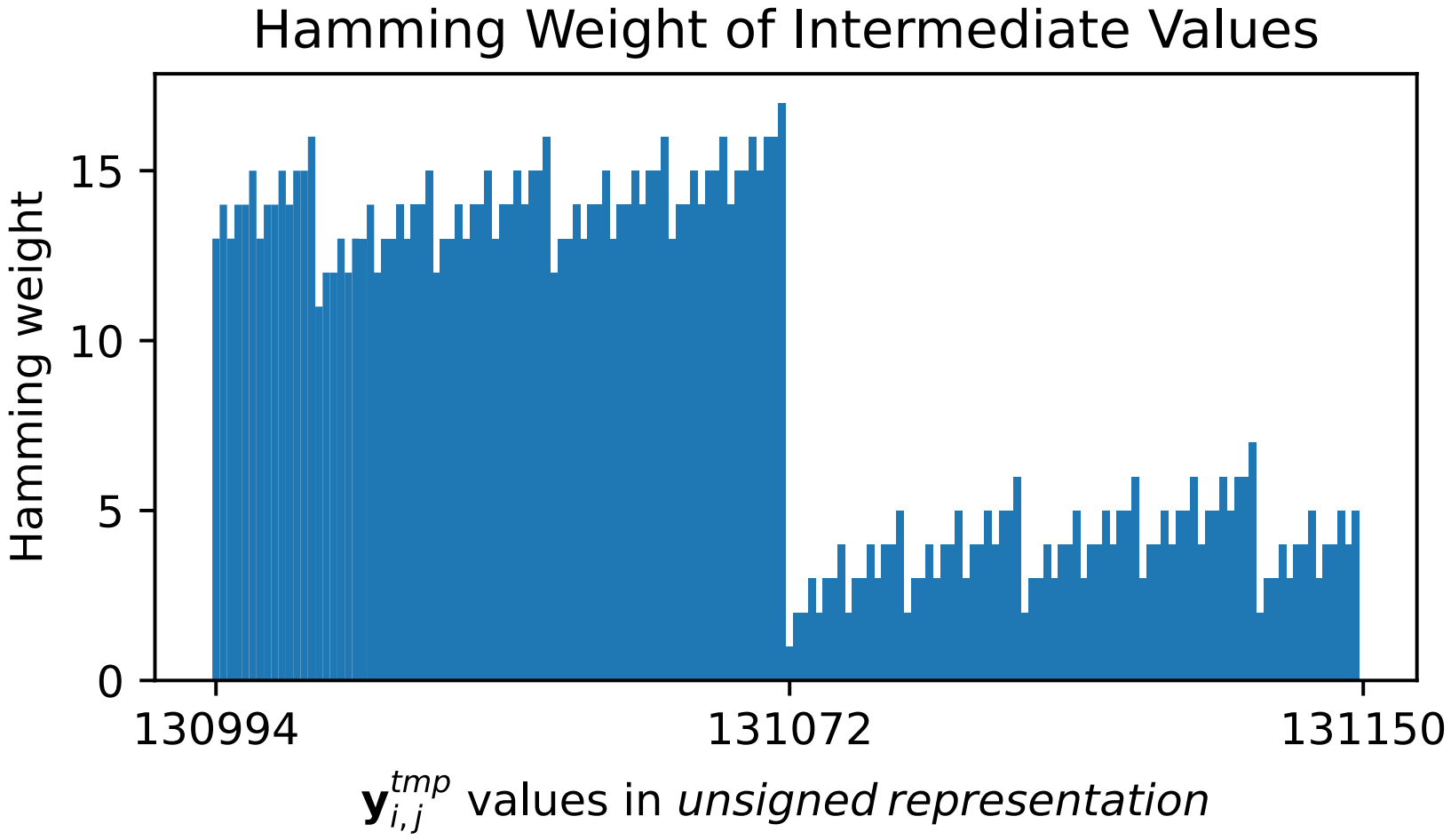
Metric	Training	Attack
Accuracy	90.77	89.76
Precision	94.30	87.75
TPR	86.79	92.59
FPR	5.25	13.13

Table 4: Binary classifier profiling (training) and attack characteristics (in %) for a predicition threshold of 80%. Precision is the fraction of samples correctly classifed as $\boldsymbol{y}_{i,j} \geq 0$ (out of all samples classifed as $\boldsymbol{y}_{i,j} \geq 0$) (see Section 2.4).


Power Trace Detail of impconvBA_rec64()
0.05
0.00
−0.05
−0.10
instruction leak
instruction causing leak
00 04 08 0C 10 14 18 1C 20 24 28 2C 30 34 38 3C 40 44 48 4C 50
eors r2, r3
ldr r5, [r1, #0]
str r3, [r4, #0]
mov.w lr, #0
strd r2, r6, [r4, #4]
adds.w r3, lr, r3
ldr r4, [r1, #8]
adds.w lr, lr, r6
ldr r6, [r1, #4]
ldr r1, [r1, #12]
eors r5, r3
eor.w r6, ip, r6
eors r3, r4
eor.w ip, ip, r1
eor.w r4, r7, lr
eor.w r1, r5, lr


Power Trace Detail of impconvBA_rec64()
0.05
0.00
−0.05
−0.10


Binary Classifier Sensitivity for predicting $\mathbf{y}_{i,j} \geq 0$
input sensitivity
0.5
0.4
0.3
0.2
0.1
0.0
180
220
260


Hamming Weight of Intermediate Values
Hamming weight
0
5
10
15
130994
131072
131150
$\mathbf{y}_{i,j}^{tmp}$ values in *unsigned representation*

Motivation: Side-Channel Attacks Break Fiat-Shamir With Aborts Zero Knowledge Property

Input: $\boldsymbol{s} \in \mathbb{Z}^n$, message $\boldsymbol{m} \in \{0,1\}^*$
Output: Zero-knowledge ILWE sample $(z, \boldsymbol{c})$

1 **repeat**
2 | Sample $y \in \mathbb{Z}$ at random uniformly from a bounded internval
3 | Compute $\boldsymbol{c} = H(\boldsymbol{m}, y) \in \mathbb{Z}^n$
4 | Compute $z = \langle \boldsymbol{s}, \boldsymbol{c} \rangle + y$
5 **until** $(z, \boldsymbol{c})$ *is zero-knowledge*

Algorithm 1: High-Level Description: Fiat Shamir with Aborts Signature

Motivation: Side-Channel Attacks Break Fiat-Shamir With Aborts Zero Knowledge Property

Input: $\boldsymbol{s} \in \mathbb{Z}^n$, message $\boldsymbol{m} \in \{0,1\}^*$
Output: Zero-knowledge ILWE sample $(z, \boldsymbol{c})$

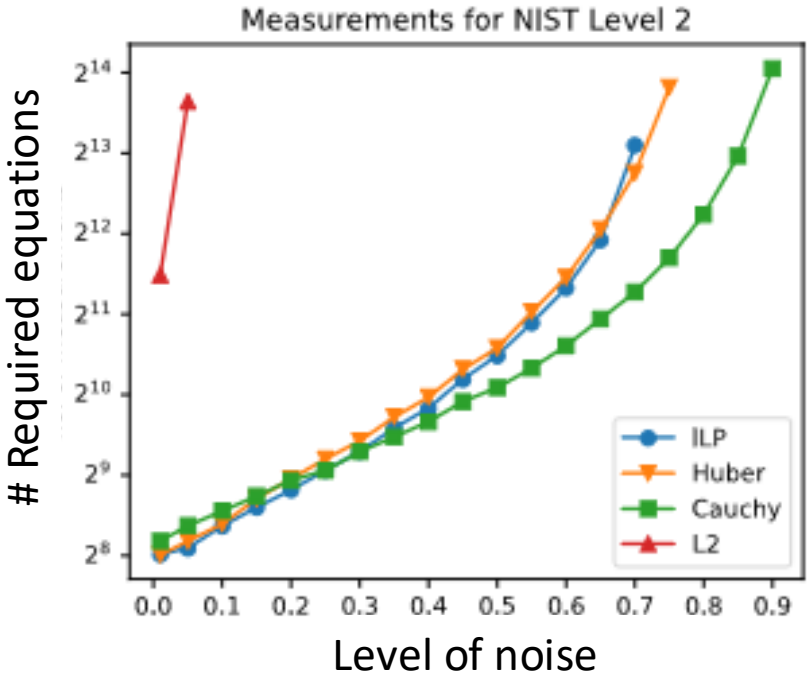
1 **repeat**
2 | Sample $y \in \mathbb{Z}$ at random uniformly from a bounded internval
3 | Compute $\boldsymbol{c} = H(\boldsymbol{m}, y) \in \mathbb{Z}^n$
4 | Compute $z = \langle \boldsymbol{s}, \boldsymbol{c} \rangle + y$
5 **until** $(z, \boldsymbol{c})$ *is zero-knowledge*

Algorithm 1: High-Level Description: Fiat Shamir with Aborts Signature

- Side-channel and fault-injection leak information about the nonce y
- This breaks the zero-knowledge property of Fiat-Shamir With Aborts Schemes

Algorithms enable key recovery with only a minor leakage on the nonce

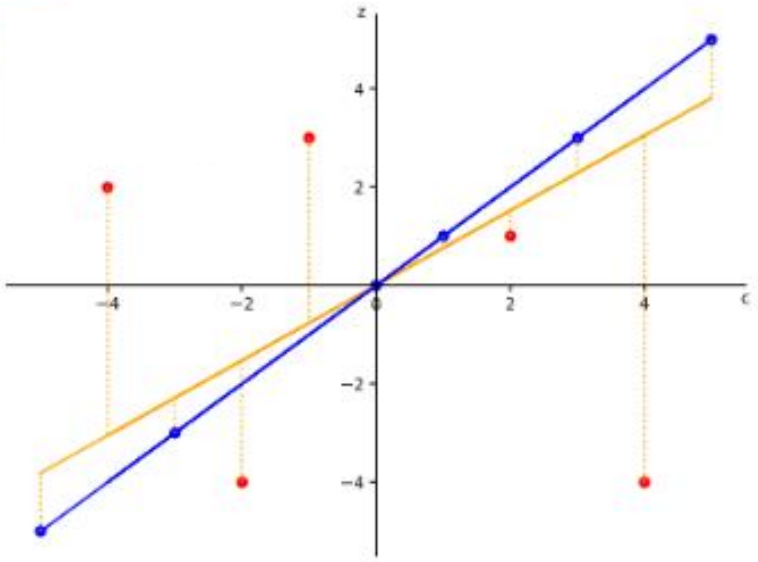
- The introduced algorithms allow for key recovery even with very noisy samples
- Practical Evaluation:
  - ILP can handle up to 80% noise rate
  - Robust Regression can handle up to 90% noise rate for Dilithium
- Result: These algorithms enable key recovery with only a minor, noisy leakage in the implementation!
- Thesis presents additional optimizations

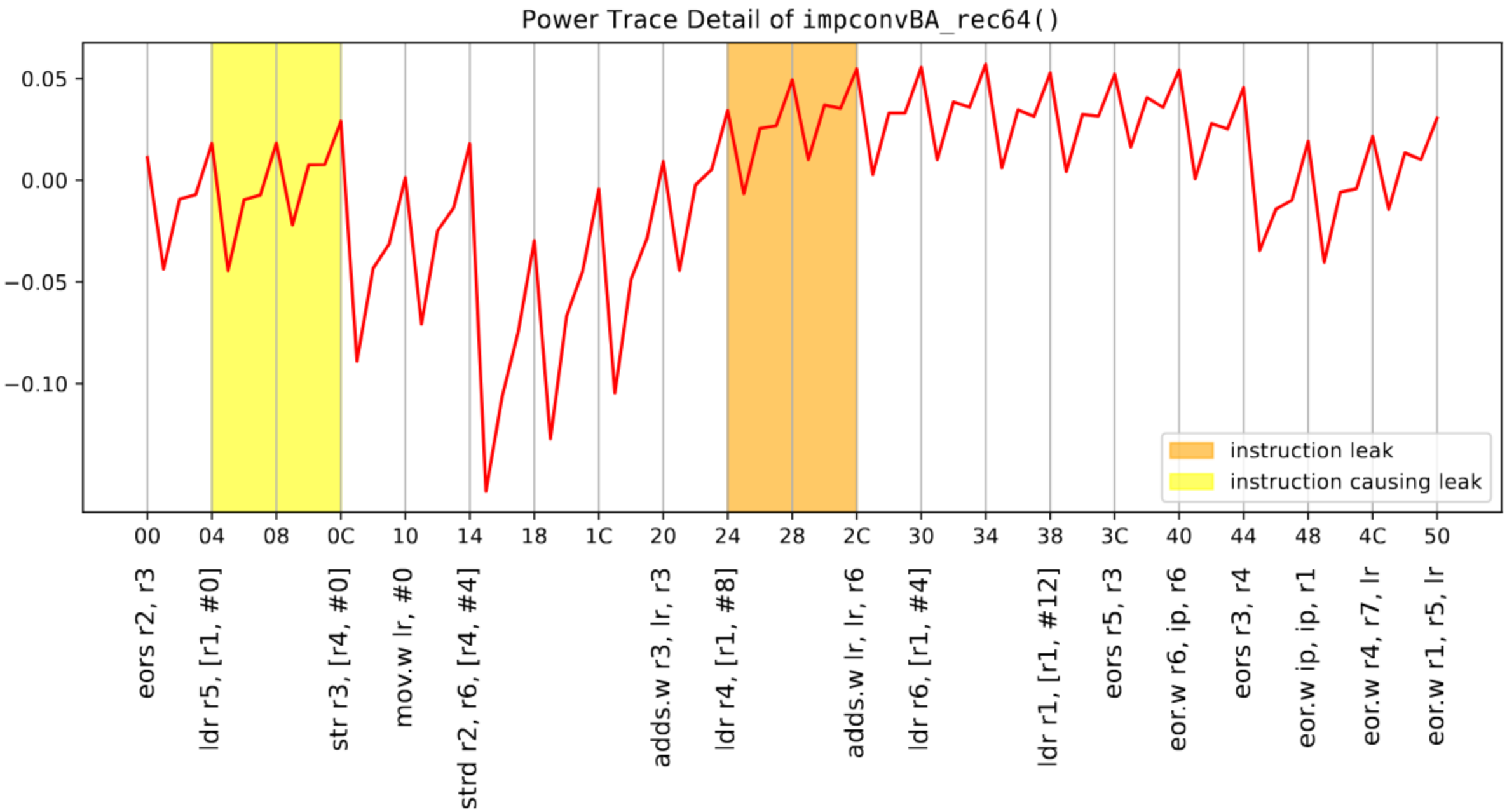


Thesis Paper: Solving Concealed ILWE and Its Application for Breaking Masked Dilithium, incl. ***VQ Ulitzsch****, in submission to Asiacrypt 25*
Thesis Paper: Profiling Side-Channel Attacks on Dilithium, ***VQ Ulitzsch*** *et. al., SAC'22*
Thesis Paper: Defeating Fault Countermeasures in Lattice Signatures with ILP, ***VQ Ulitzsch*** *et. al., TCHES'23*

Robust Regression Allows For Efficient Key Recovery

- Recovering s_1 from a set of samples $\{(z = cs_1 + \hat{y}, c)\}$ can be viewed as a regression problem, with $\hat{y}$ being the noise
- However: Standard least-squares regression methods require the noise $\hat{y}$ to be independent of the secret s_1

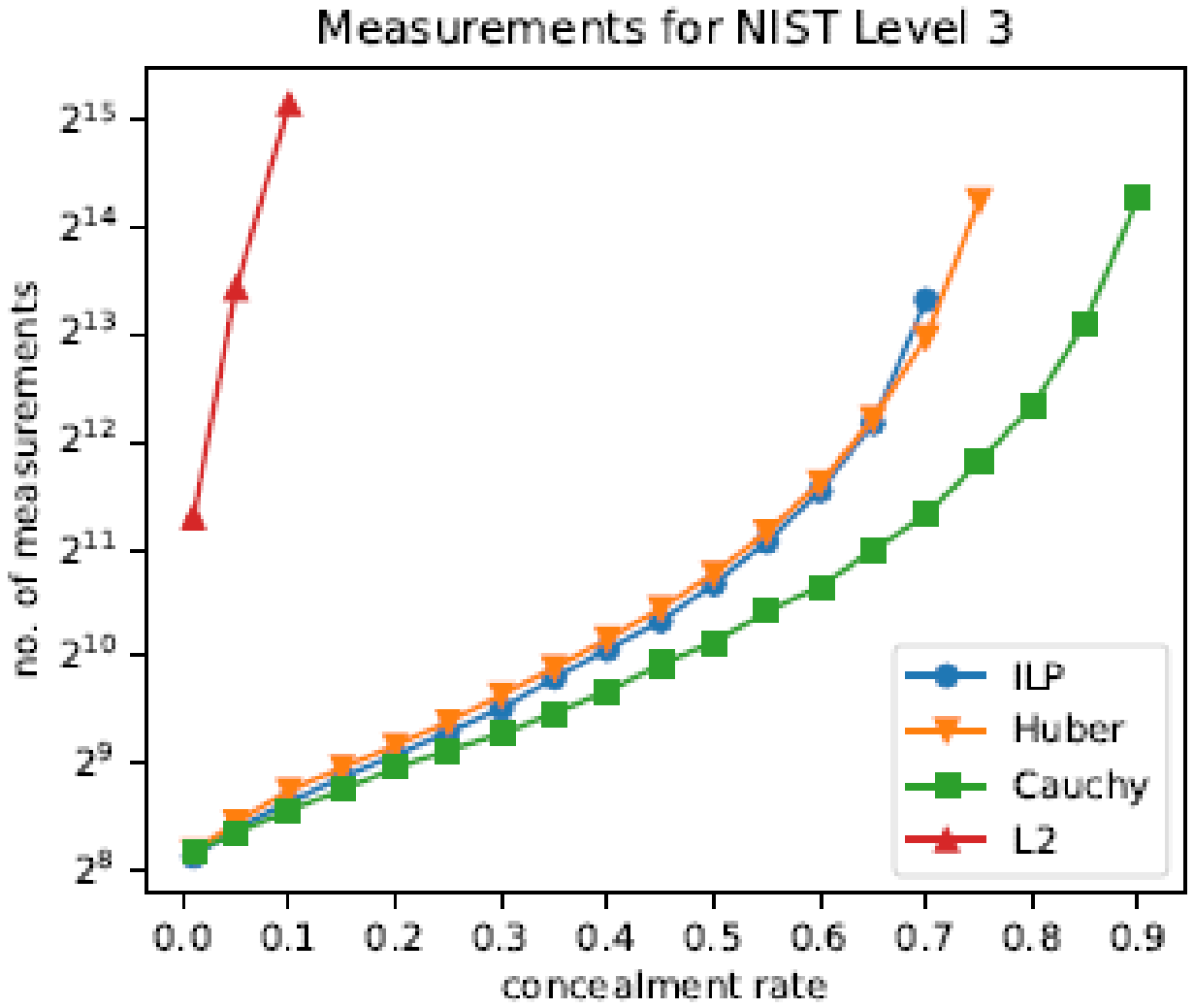



Power Trace Detail of impconvBA_rec64()
0.05
0.00
−0.05
−0.10
instruction leak
instruction causing leak
00
04
08
0C
10
14
18
1C
20
24
28
2C
30
34
38
3C
40
44
48
4C
50
eors r2, r3
ldr r5, [r1, #0]
str r3, [r4, #0]
mov.w lr, #0
strd r2, r6, [r4, #4]
adds.w r3, lr, r3
ldr r4, [r1, #8]
adds.w lr, lr, r6
ldr r6, [r1, #4]
ldr r1, [r1, #12]
eors r5, r3
eor.w r6, ip, r6
eors r3, r4
eor.w ip, ip, r1
eor.w r4, r7, lr
eor.w r1, r5, lr

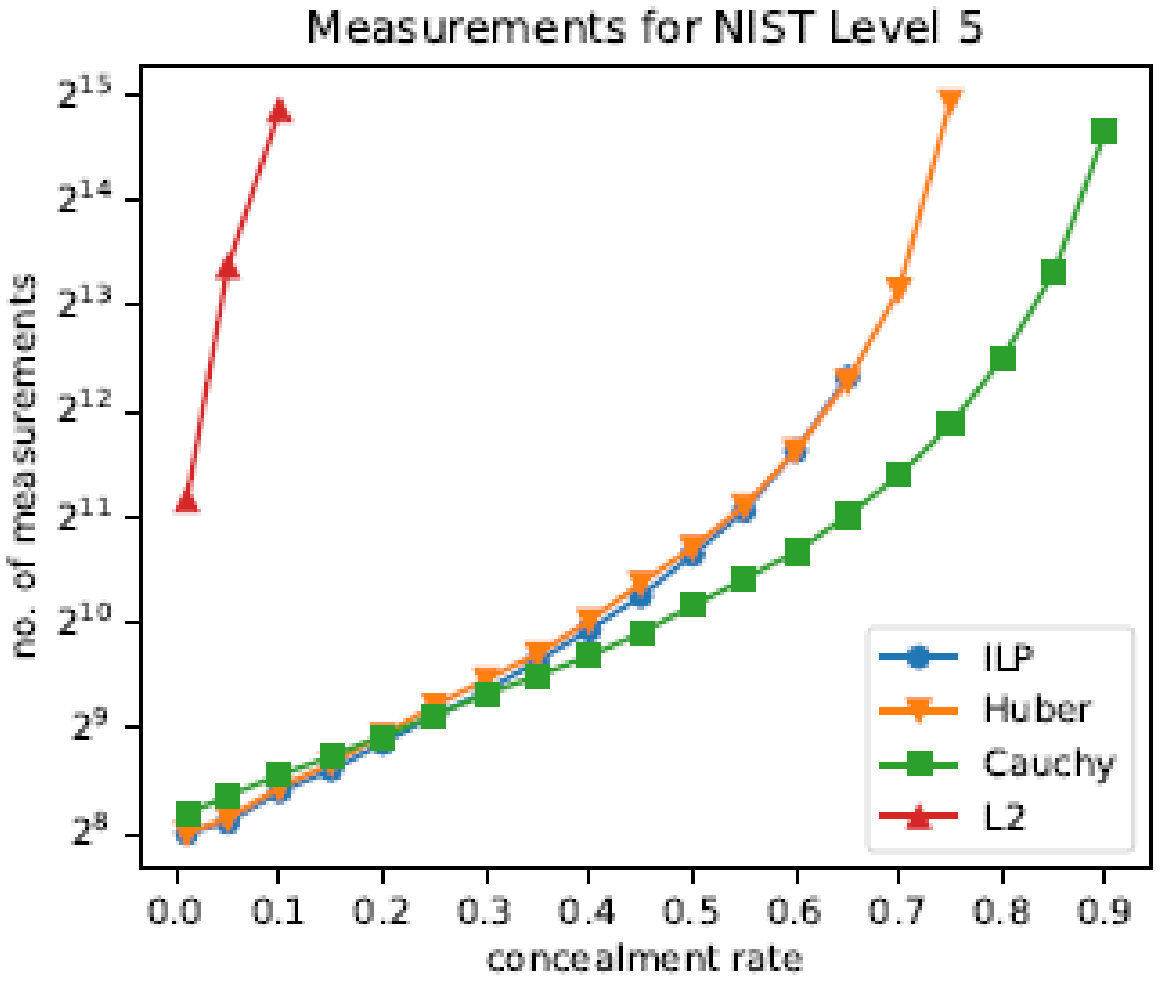
Robust Regression Allows For Efficient Key Recovery

- Recovering s_1 from a set of samples $\{(z = cs_1 + \hat{y}, c)\}$ can be viewed as a regression problem, with $\hat{y}$ being the noise
- However: Standard least-squares regression methods require the noise $\hat{y}$ to be independent of the secret s_1
- But: Samples of the form $z = cs_1 + \hat{y}$ give rise to a mixture distribution where
  - Incorrect guesses are correlated with s_1 (outlier distribution)
  - Correct guesses (where $\hat{y} = 0$) are independent of s_1
- **Key Idea**: Use outlier-resilient regression methods, such as Huber and Cauchy Regression.
  - Method's loss functions place lower weight on outliers, allowing for secret key recovery

- Recovering s_1 from a set of samples $\{(z = cs_1 + \hat{y}, c)\}$ can be viewed as a regression problem, with $\hat{y}$ being the noise
- How ıst-squares regression methods require the nois dent of the secret s_1
- But: Samples of the form $z = cs_1 + \hat{y}$ give rise to a mixture distribution where
  - Incorrect guesses are correlated with s_1 (outlier distribution)
  - Correct guesses (where $\hat{y} = 0$) are independent of s_1
- **Key Idea**: Use outlier-resilient regression methods, such as Huber and Cauchy Regression.
  - Method's loss functions place lower weight on outliers, allowing for secret key recovery

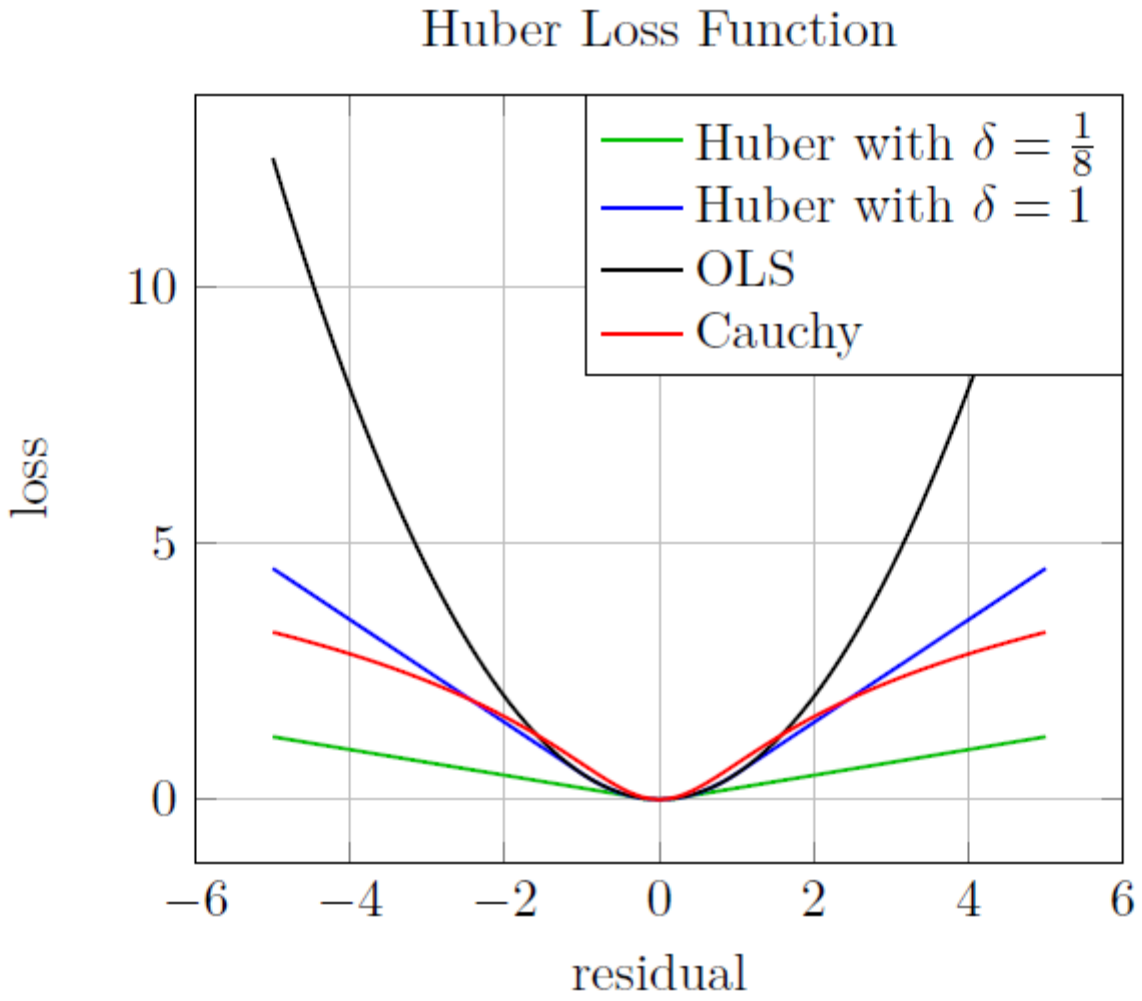

Measurements for NIST Level 3
no. of measurements
2^{15}
2^{14}
2^{13}
2^{12}
2^{11}
2^{10}
2^{9}
2^{8}
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
concealment rate
ILP
Huber
Cauchy
L2

(B) NIST-3

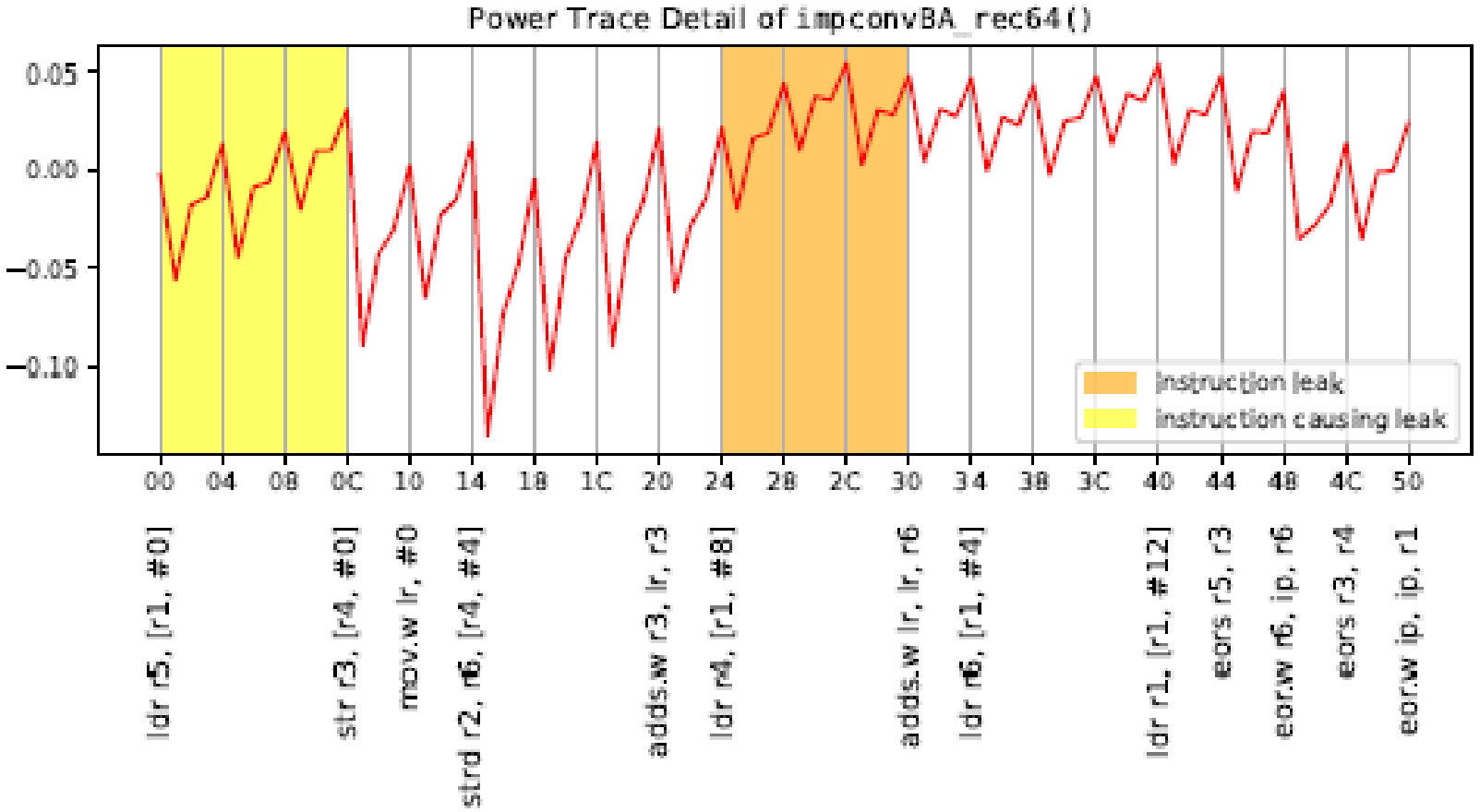

Measurements for NIST Level 5
no. of measurements
2^{15}
2^{14}
2^{13}
2^{12}
2^{11}
2^{10}
2^{9}
2^{8}
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
concealment rate
ILP
Huber
Cauchy
L2

(c) NIST-5

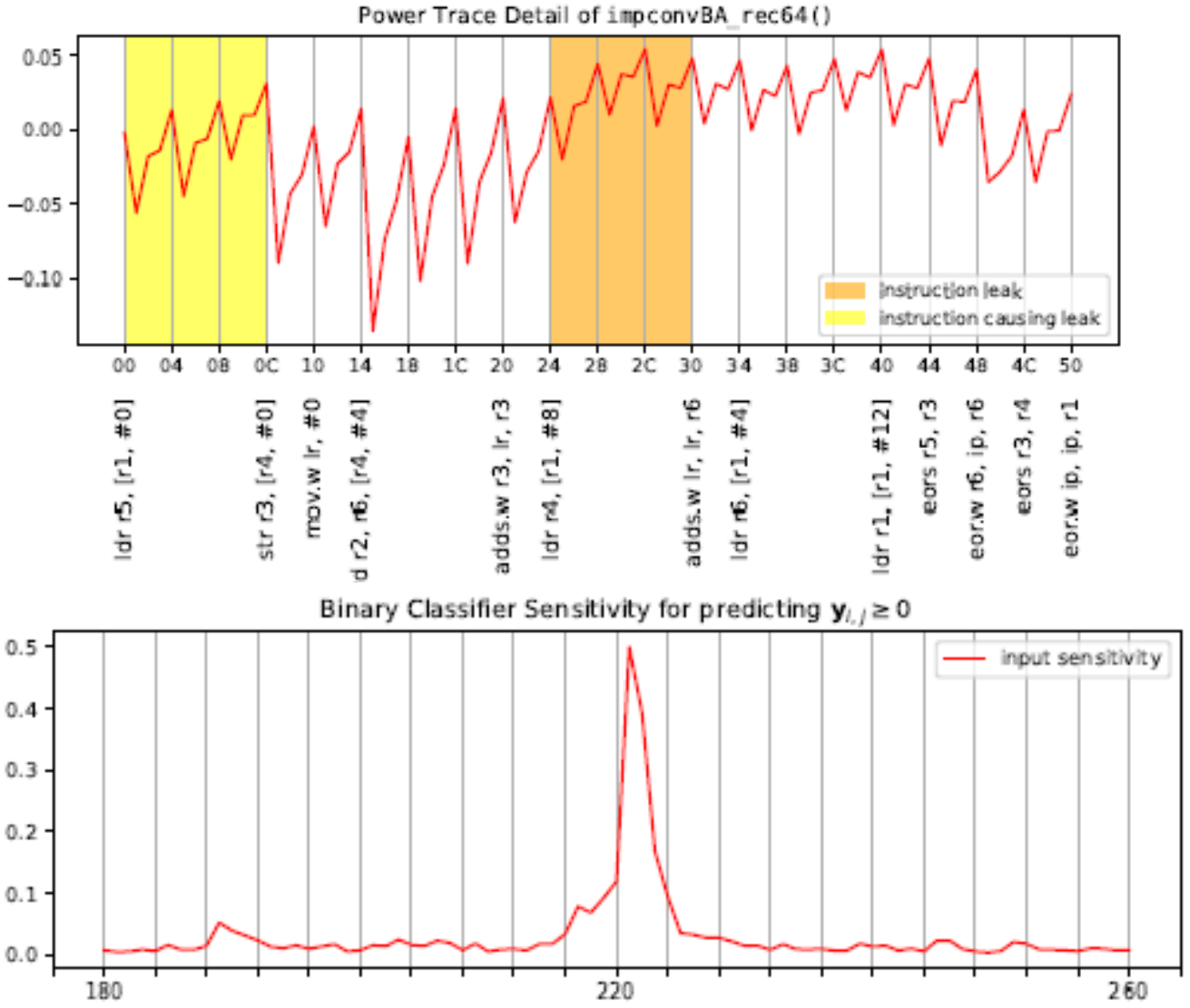
Huber and Cauchy regression place less emphasis on bigger loss



Leakage analysis reveals second load as responsible instruction

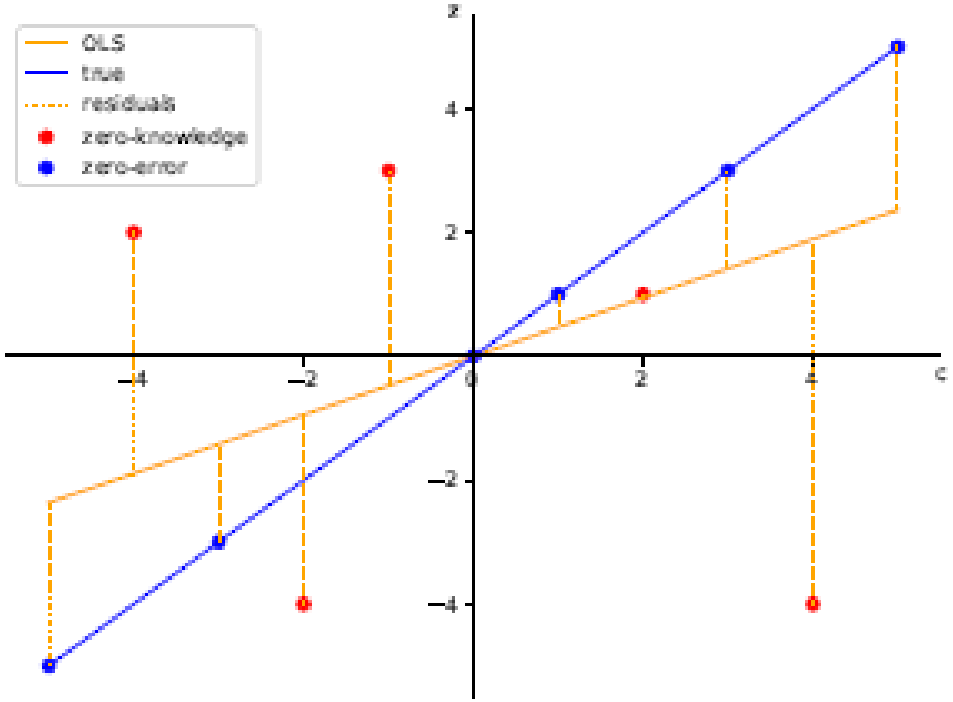


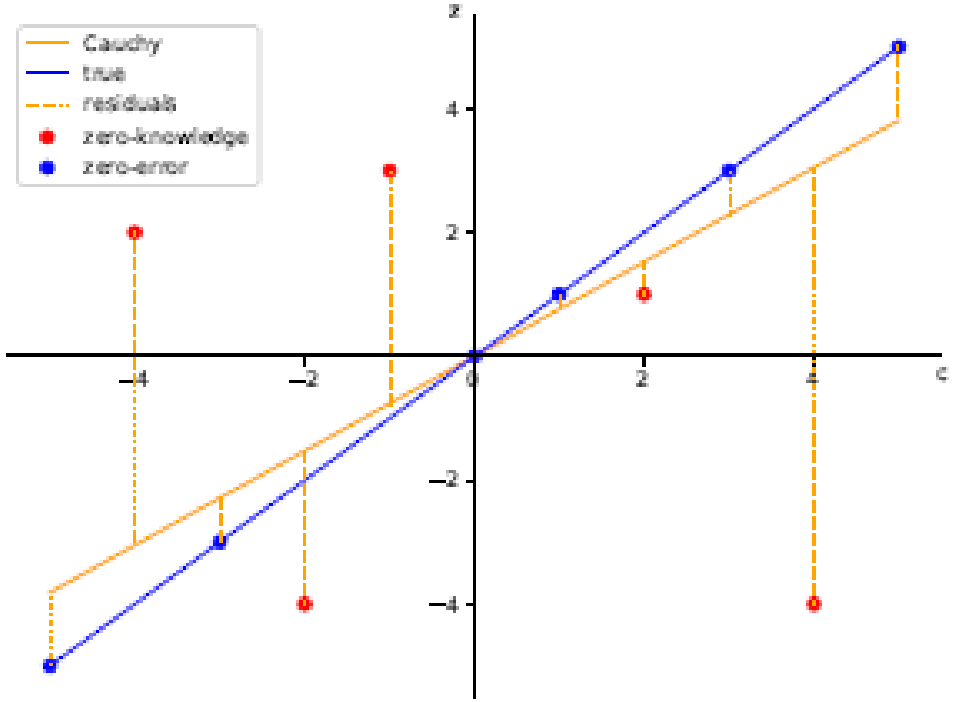
Leakage analysis reveals second load as responsible instruction



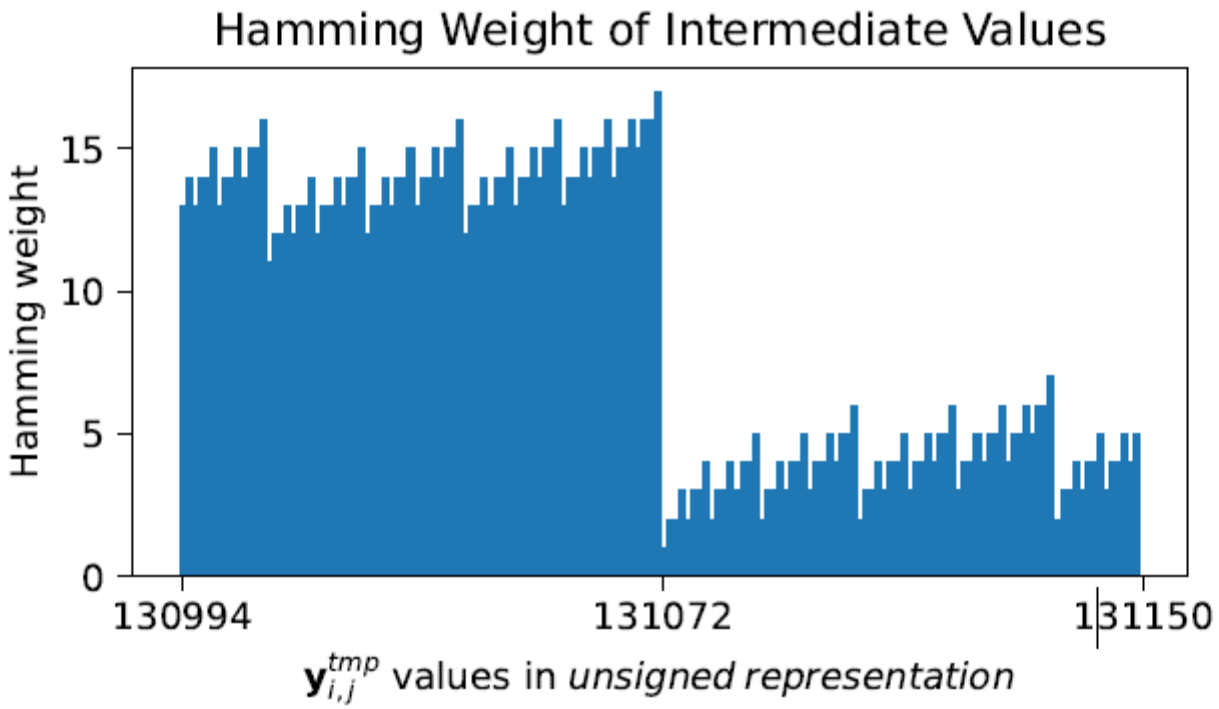
(c) Input sensitivity analysis trace of the profiling stage binary classifier trained to predict coefficients $\mathbf{y}_{i,j} \geq 0$.

Huber and Cauchy regression place less emphasis on bigger loss





Masked Implementation Uses Boolean-To-Arithmetic Share Conversion



- During boolean to arithmetic share conversion, y is implicitly stored in a y^{tmp} unsigned representation (through two arithmetic shares)
- y^{tmp} 's shares are loaded in two consecutively loads, which are most likely "buffered" in some internal pipeline register
- The second loads reveals Hamming Weight (HW) of the xor of the two shares - which is the HW of y^{tmp}