OPTIMISING ENERGY CONSUMPTION USING GI

https://cs.adelaide.edu.au/~markus/
markus.wagner@adelaide.edu.au

Optimising energy consumption using GI

Project 1/2

Project 2/2

→ Two world-first presentations!!
GI to combat side-channel attacks

Project 1/2

ROSITA: Towards Automatic Elimination of Power-Analysis Leakage in Ciphers

Madura A. Shelton  
University of Adelaide  
madura.shelton@adelaide.edu.au

Niels Samwel  
Radboud University  
nsamwel@cs.ru.nl

Lejla Batina  
Radboud University  
lejla@cs.ru.nl

Francesco Regazzoni  
ALaRI – USI  
regazzoni@alari.ch

Markus Wagner  
University of Adelaide  
markus.wagner@adelaide.edu.au

Yuval Yarom  
University of Adelaide and Data61  
yval@cs.adelaide.edu.au

Side Channel Attacks
Our Goal

Leaky implementation → Automated Process “GI: Magic Sauce” → Leak-reduced implementation
A Computer Program

Instruction 1
Instruction 2
...
Instruction N

Execution
An Instruction

Instruction 1
Instruction 2
...
Instruction N

Diagram:

- $R_1$ to $t_1$
- $R_2$ to $t_2$
- $R_i$ to $R_n$
- $R_d$ to $t_2$
An Instruction

Instruction 1
Instruction 2
...
Instruction N

Diagram:
- \( R_1 \)
- \( R_2 \)
- \( t_1 \)
- \( t_2 \)
- \( R_i \)
- \( R_n \)
- \( R_d \)
An Instruction

Instruction 1

Instruction 2

...

Instruction N

Diagram with nodes t₁, t₂, R₁, R₂, Rᵢ, Rₙ, and R_d.
An Instruction

Instruction 1
Instruction 2
...
Instruction N

\[
\begin{align*}
R_1 & \rightarrow t_1 \\
R_2 & \rightarrow t_2 \\
R_i & \rightarrow R_n \\
R_d &
\end{align*}
\]
Power consumption of a register

Typical: the power depends on the Hamming weight of the value
Power consumption of a register - Ideally

Note: the line is horizontal to indicate the average over many repetitions.
Masking

Intermediate values are independent of key

Note: the lines are horizontal to indicate the average over many repetitions
Is masking alone sufficient?

\[ A \oplus R \rightarrow A' \]

Memory Bus
Is masking alone sufficient?

\[ A \oplus R \rightarrow A' \]
Is masking alone sufficient?

\[ A \oplus R \rightarrow A' \]
Is masking alone sufficient?

\[ B \oplus R \rightarrow B' \rightarrow A' \oplus B' \]
Is masking alone sufficient?

\[ B \oplus R \rightarrow B' \]

\[ A' \oplus B' = (A' \oplus R) \oplus (B' \oplus R) = A \oplus B \]
Measuring Power Consumption
Experimental setup
Evaluation - Test Vector Leakage Assessment (TVLA)

Test A - Fixed Input

Test B - Random Input

Can you spot the major difference at 4000-4500 samples?
Evaluation - Test Vector Leakage Assessment (TVLA)
Applying Countermeasures (industry standard)
Our Contribution (Rosita)

- Masked Implementation
- Leakage Detection using Simulation (ELMO*) [MOW17]
- Rule-based code rewrite (Rosita)
- Candidate Program
- Reduced-leakage Implementation
Rule-based code rewrite

At the moment: highly problem-specific.

But to begin with: when to apply which rule? We have extended the simulator to tell us where the leak occurs and due to which interaction.

Rules (very different from the GI-usual swap/copy/delete operators):

1. Operand interaction via the bus → \textit{movs} \ r7, \ r7 (we initialised the register r7 with a random value and the cipher is not allowed to use it)

2. Register reuse → overwrite the register with a random value first, e.g. \textit{movs} \ r3, \ r4 leaks → inserts \textit{movs} \ r3, \ r7 before this leaking instruction

3. Rotations: word masks and partial rotations

4. Memory interaction: complex, requires push/pop and other operations
Results

\[
\text{Slow down} = \frac{1430}{1293} = 1.11
\]

(a) Part of an AES implementation

(b) After applying code rewrites (1430 cycles)
Leakage as trace count increases (now: validated on hardware)

![Graph showing leakage as trace count increases]

ROSITA: Towards Automatic Elimination of Power-Analysis Leakage in Ciphers
https://arxiv.org/abs/1912.05183 (Section 5)
GI to combat side-channel attacks

Improve target code performance
Replacement code synthesis
Adapt to multiple architectures
Generalize limitations of code synthesis
Expand ELMO*’s simulation using ML

ROSITA: Towards Automatic Elimination of Power-Analysis Leakage in Ciphers
https://arxiv.org/abs/1912.05183 (Section 5)
GI to combat the energy hunger of apps

Project 2/2
What to do so that you can use GI to combat the energy hunger of apps

Project 2/2
What to do so that you can use GI to combat the energy hunger of apps and how to make sure that your results hold up.

This is Why You Should Rigorously Validate Non-functional Property/Energy Optimisation Experiments

Mahmoud A. Bokhari
Optimisation and Logistics, School of Computer Science,
The University of Adelaide, Australia
Computer Science Department, Taibah University,
Kingdom of Saudi Arabia
mahmoud.bokhari@adelaide.edu.au

Brad Alexander, Markus Wagner
Optimisation and Logistics, School of Computer Science,
The University of Adelaide, Australia
bradley.alexander@adelaide.edu.au
markus.wagner@adelaide.edu.au

To be submitted...
Why optimise the energy-consumption of apps?

Number of smartphone users >3 billion

Users expect

Reality
Why optimise the energy consumption of apps?

Users expect...

"4020 mAh" listed first!

Seen on 31/12/19 in a China Unicom store, Xi’dan, Beijing
Challenges for developers

Typical challenges
1. Developers lack understanding of the energy consumption
2. Different strategies for mobile devices and PCs
3. Balancing the trade-off between energy and performance for designers

Bonus challenges
1. Internal vs external sensors (noise)
2. Temperature sensitivity (noise)
3. Android debug bridge
4. An OS that keeps developing (read: it’s fighting us) + (noise)
5. Models are incomplete and quickly outdated
6. ... more noise.

I envy those of you who work in a noise-free environment!
Challenges for developers

Why all this lamenting?

Our observations and conjectures:
- There is little knowledge distributed across different domains on how to deal with these problems in isolation (read: one paper observing/mentioning/dealing with one aspect at a time, making it difficult to get a general overview)
- People avoid super-noisy problems.
- Phones 5 years ago were more deterministic platforms than they are now... and it’s just going to get a lot worse still (read: devices get more complex/efficient/dynamic/...)

Typical challenges
1. Developers lack understanding of the energy consumption
2. Different strategies for mobile devices and PCs
3. Balancing the trade-off between energy and performance for designers

Bonus challenges
1. Internal vs external sensors (noise)
2. Temperature sensitivity (noise)
3. Android debug bridge
4. An OS that keeps developing (read: it’s fighting us) +
5. Models are incomplete and quickly outdated
6. … more noise.
How do we validate our experimental results?

aka

How to know that your claims will hold up?
Fragmented Ecosystems

Mind the gap – a distributed framework for enabling energy optimisation on modern smart-phones in the presence of noise, drift, and statistical insignificance

Mahmoud A. Bokhari
1 Optimisation and Logistics
University of Adelaide, Australia
2 Computer Science Department
Taibah University
Kingdom of Saudi Arabia
mahmoud.bokhari@adelaide.edu.au

Lujun Weng, Markus Wagner, Bradley Alexander
Optimisation and Logistics
University of Adelaide, Australia
lujunweng@outlook.com
markus.wagner@adelaide.edu.au
bradley.alexander@adelaide.edu.au

Below: four different phone-OS combinations, orange/blue are two different test loads (but identical across all samples):
Mind the gap – a distributed framework for enabling energy optimisation on modern smart-phones in the presence of noise, drift, and statistical insignificance

Mahmoud A. Bokhari
1 Optimisation and Logistics
University of Adelaide, Australia
2 Computer Science Department
Taibah University
Kingdom of Saudi Arabia
mahmoud.bokhari@adelaide.edu.au

Lujun Weng, Markus Wagner, Bradley Alexander
Optimisation and Logistics
University of Adelaide, Australia
lujunweng@outlook.com
markus.wagner@adelaide.edu.au
bradley.alexander@adelaide.edu.au

Wait, it is even worse !!!
Individual runs of Rebound library (original configuration) in two experiments. The device was rebooted and recharged between the two experiments.
Issue: System States

- **System Memory Consumption**
  - Graph showing memory consumption over rebound samples for different experiments.
  - Experiments 1 to 8 are indicated.

- **Background Processes**
  - Graph showing the number of processes over rebound samples.
  - Highlighted peaks and troughs for each experiment.
Solution

Be fair and square
Solution

Be fair and square
Run solutions in similar conditions, i.e. system state(s)
Solution

Be fair and square
Run solutions in similar conditions, i.e. system state(s)
Solution

Be fair and square
Solution

Be fair and round
Solution

Be fair and round
Run solutions in a round robin fashion
Solution

Be fair and round
Run solutions in a round robin fashion
till a termination condition.
e.g.: battery level = 20%, or 10 runs per solution.

sol 1  sol 2  sol 3  sol 1  sol 2  sol 3  sol 1  sol 2  sol 3
Solution

Be fair and round
Run solutions in a round robin fashion
till a termination condition.
e.g.: battery level = 20%, or 10 runs per solution.

Maintenance: recharge/clean up
Solution

Be fair and round
Run solutions in a round robin fashion
till a termination condition.
 e.g.: battery level = 20%, or 10 runs per solution.
 Maintenance: recharge/clean up

Alternate between solution order
Solution

Be fair and round

Let’s try it on

... let’s discredit ourselves!
Solution

Conventional way: energy results

Expected: violins get lower and lower (as the energy consumption *should* drop)
Solution

Conventional way: system behaviour
The box contains 13 violins:
- 1 original configuration’s energy consumption
- 12 solutions forming a Pareto front (Mobiquitous’18 paper)

**Solution**

**Round Robin + rotate: energy results**

⇒ It’s not perfect yet, but at least we are trying harder.

Conjecture: maybe the Pareto front contained some dominated solutions after all.
(e.g., purple/loc10 is higher in both setups)
Solution

Round Robin + rotate: system behaviour
...make sure that your results hold up

Do you have a noisy system?
Do you have states?

➤ Be fair and square → round-robin + rotate your way!

While cute, it’s not perfect yet.

Todo: Find cheap, non-intrusive ways to incorporate the system state into the optimisation process.

This is Why You Should Rigorously Validate Non-functional Property/Energy Optimisation Experiments

Mahmoud A. Bokhari
Optimisation and Logistics, School of Computer Science, The University of Adelaide, Australia
Computer Science Department, Taibah University, Kingdom of Saudi Arabia
mahmoud.bokhari@adelaide.edu.au

Brad Alexander, Markus Arndt
Optimisation and Logistics, The University of Adelaide, Australia
brad.alexander@adelaide.edu.au, markus.arndt@adelaide.edu.au
OPTIMISING ENERGY CONSUMPTION USING GI

https://cs.adelaide.edu.au/~markus/
markus.wagner@adelaide.edu.au
