

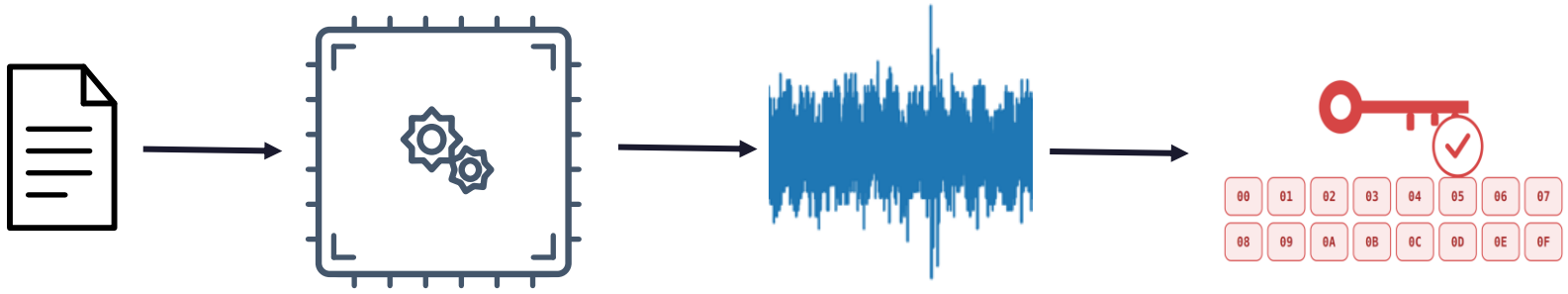

microSCALE: Static Analysis for Microarchitectural Side-channel Leakage Evaluation

Akshay Kumar E, Pranav Krishna N, Annapurna Valiveti, Pallavi Borkar, Aditi Roy, Chester Rebeiro

Indian Institute of Technology Madras

35th Usenix Security Symposium

Power Side Channel Analysis



Crypto Program **Processor** **Power Trace** **Secret**

Power Analysis – The Hidden Threat


BLEEPINGCOMPUTER

New Collide+Power side-channel attack impacts almost all CPUs

By Bill Toulas

August 2, 2023 01:37 PM 1

Power Analysis – The Hidden Threat

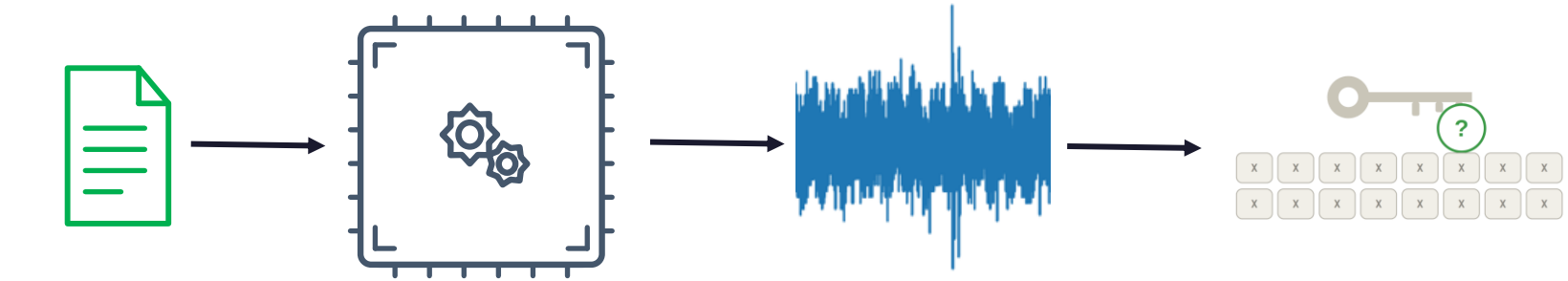








Counter Measure at Program Level



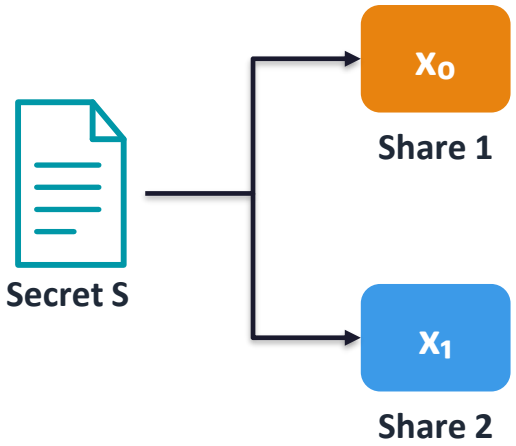
Masking – A Popular Countermeasure



Secret S

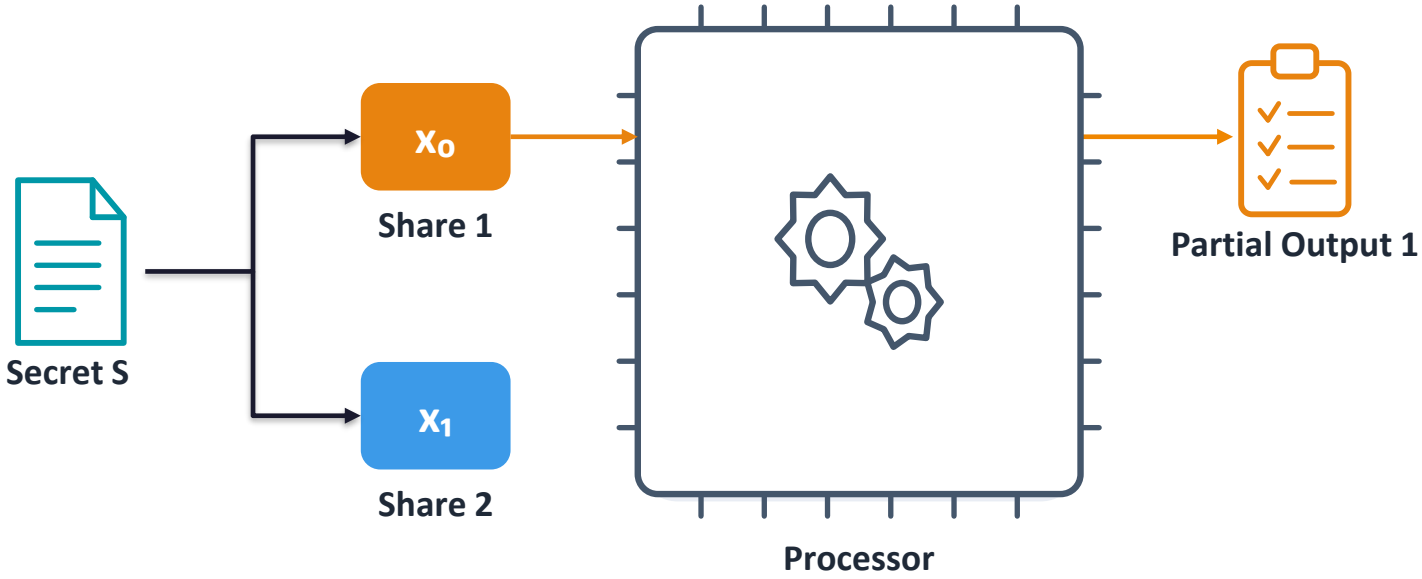
Masking – A Popular Countermeasure

Step 1: Split the secret into two shares



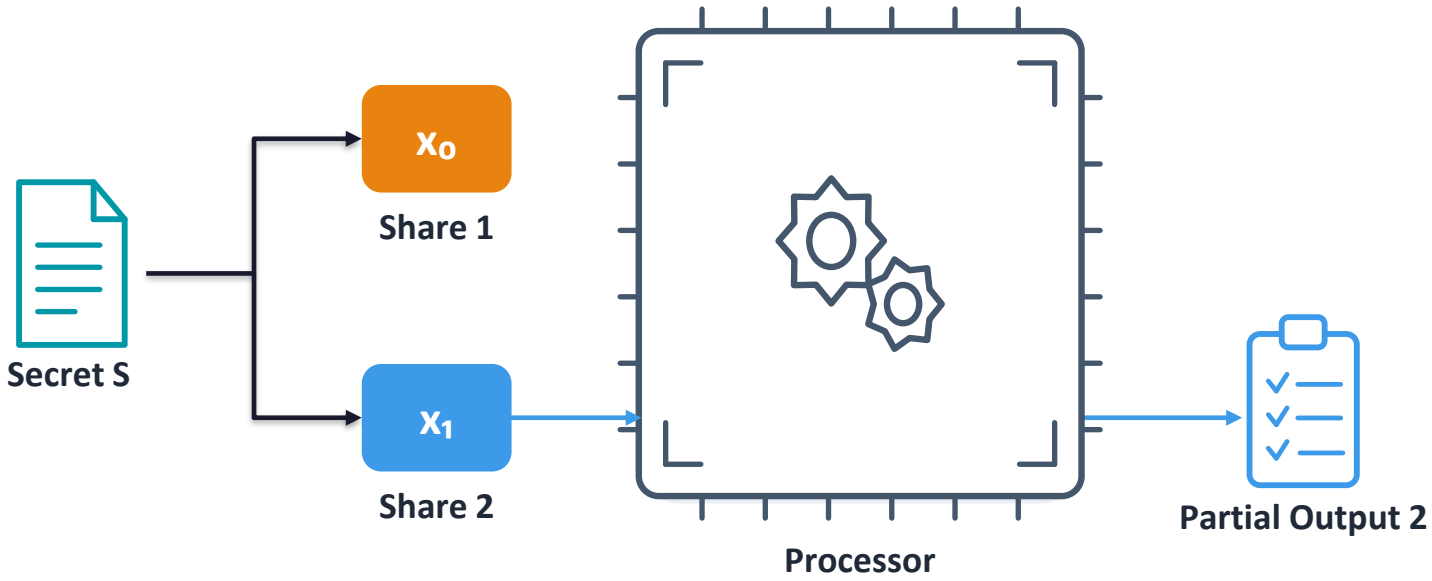
Masking – A Popular Countermeasure

Step 2: Process Share 1 to get Partial Output 1



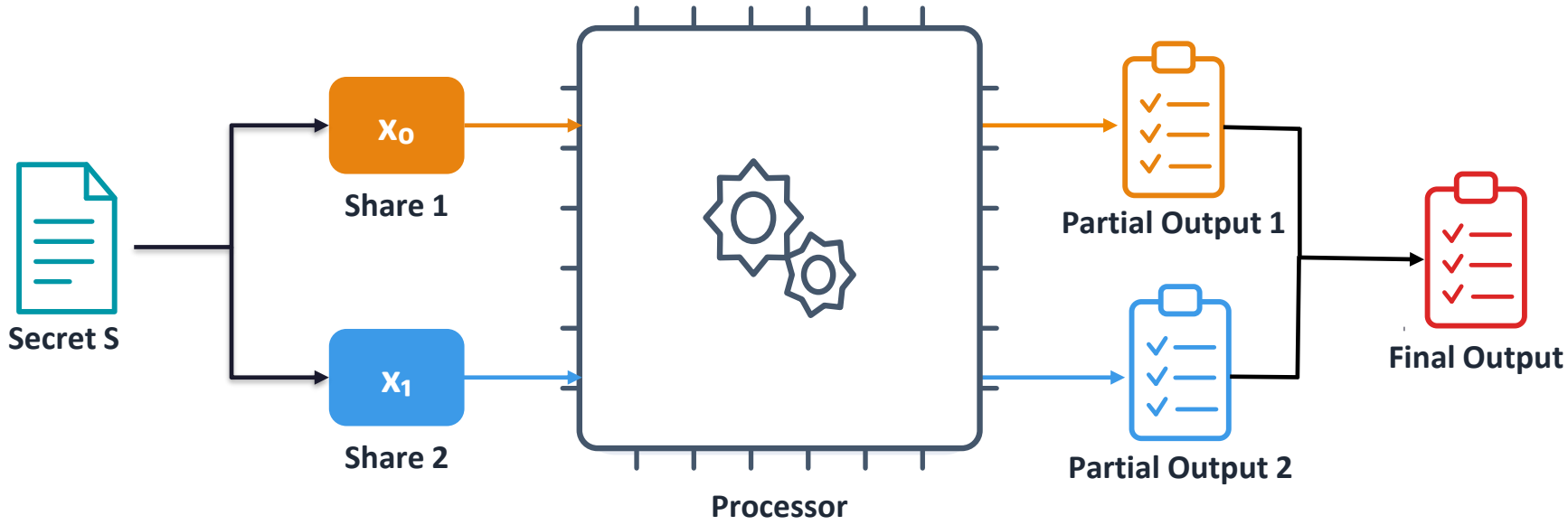
Masking – A Popular Countermeasure

Step 3: Process Share 2 to get Partial Output 2



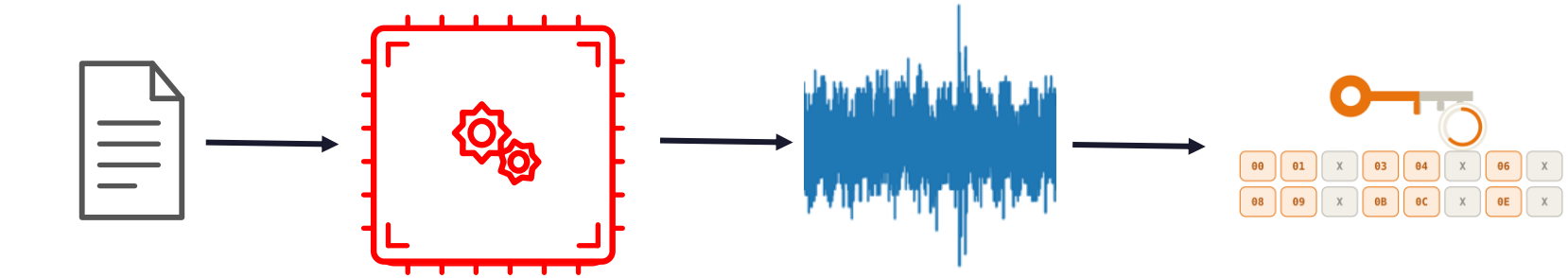
Masking – A Popular Countermeasure

Step 4: Construction of the final output



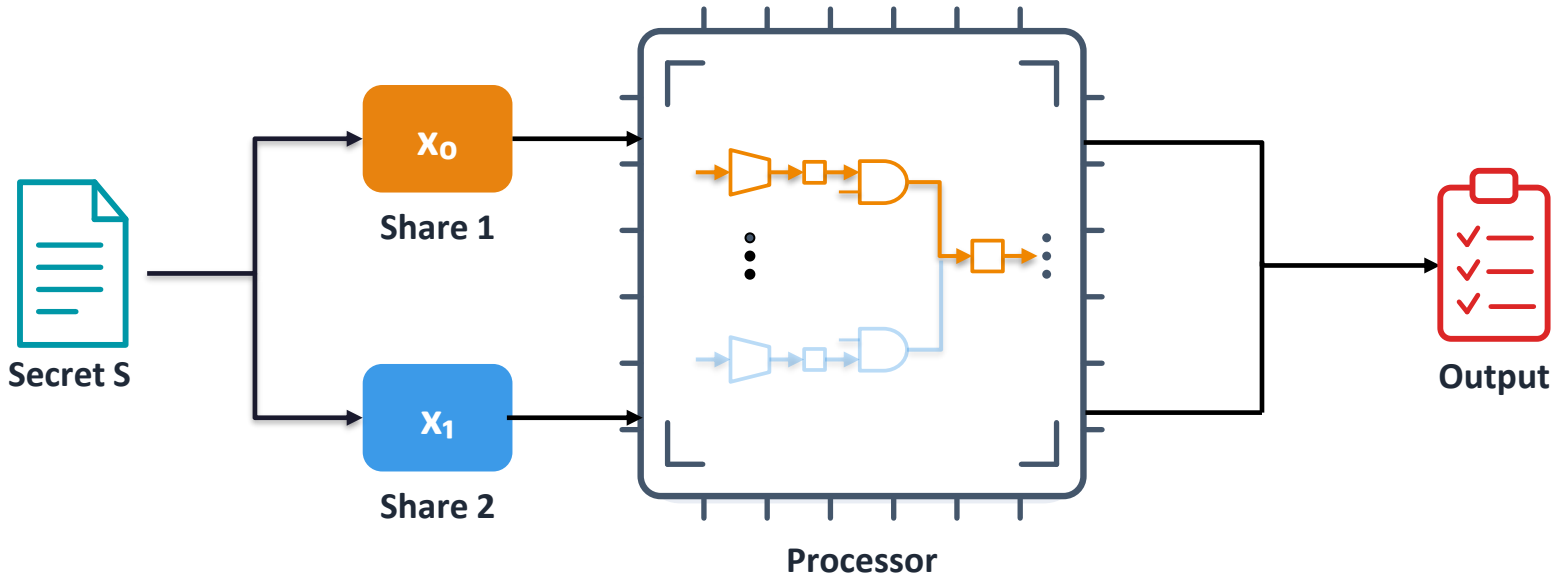
Protected Implementation — Still Leaks!

Secret leaks in practice



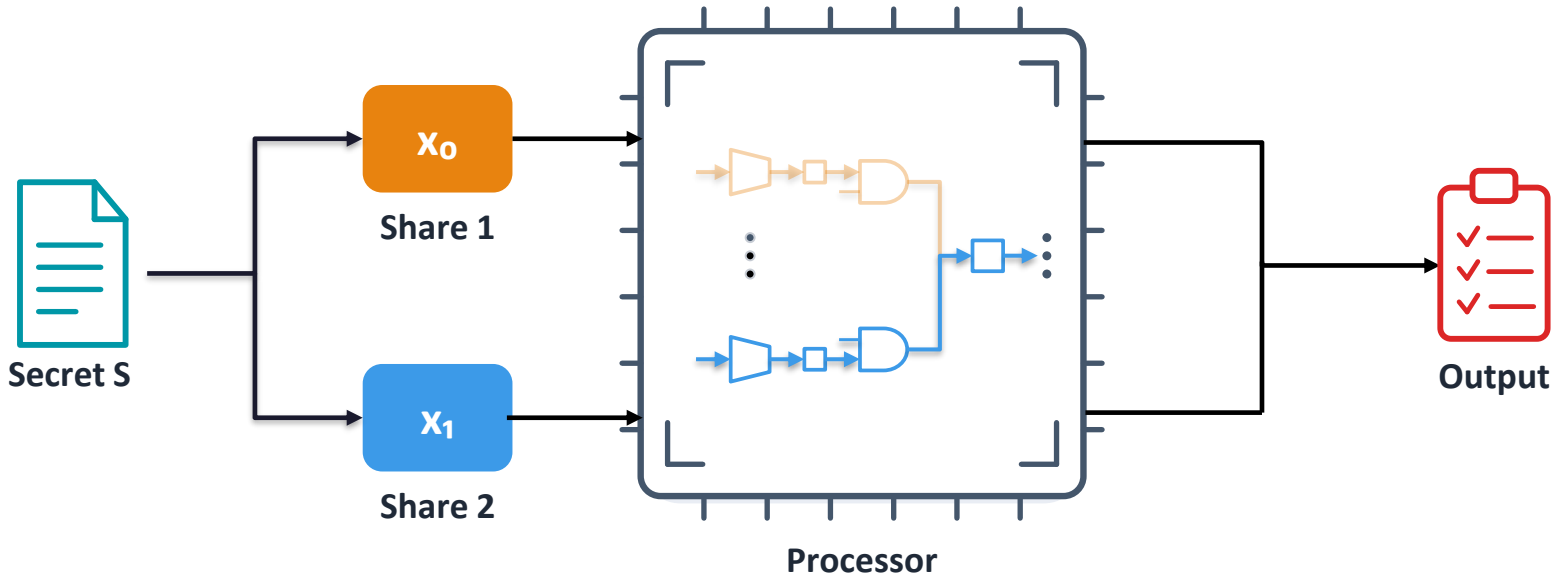
Microarchitecture : The Hidden Culprit

microarchitectural interactions can leak the secret



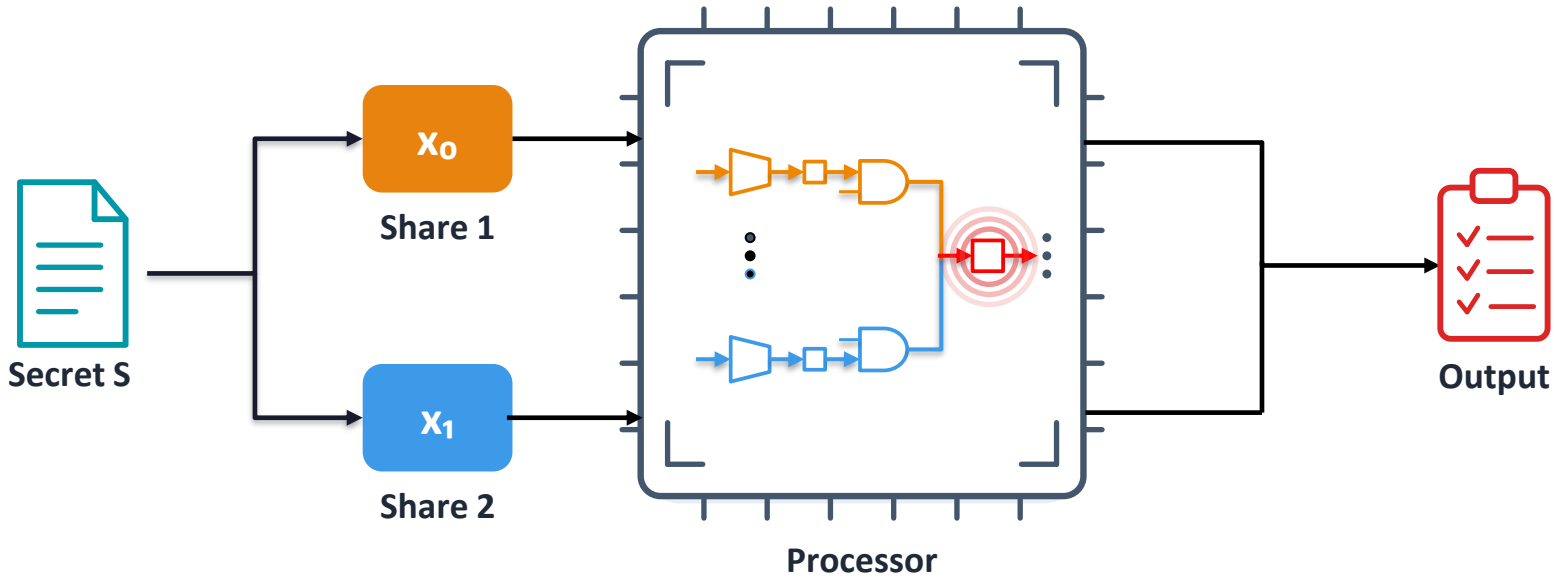
Microarchitecture : The Hidden Culprit

microarchitectural interactions can leak the secret



Microarchitecture : The Hidden Culprit

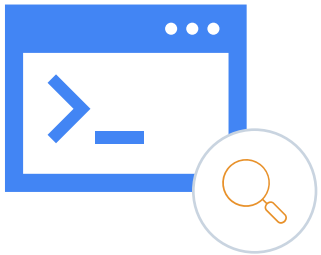
microarchitectural interactions can leak the secret



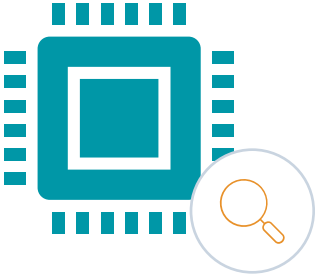
Locating the Source of Leakage



To fix a leakage in the microarchitecture,
we must first see **where** it leaks and **why** it leaks.



Software



Hardware

How Do We Locate the Leak?

Existing Approaches



Simulation-Based

- Slow
- Limited coverage
- ELMO [1], ROSITA [2]



Formal Methods

- State explosion
- Requires High level of expertise
- Power Contract [3], COCO [4]

Our Contributions



Modeling

CHALLENGE

How to model complex, tightly-coupled RTL structures?



Detection

CHALLENGE

Which interactions actually leak?



Localization

CHALLENGE

How to pinpoint It to hardware and software?

Our Contributions



Modeling

CHALLENGE

How to model complex, tightly-coupled RTL structures?

Microarchitectural Graph captures every path



Detection

CHALLENGE

Which interactions actually leak?

Leakage aware rules flag recombination



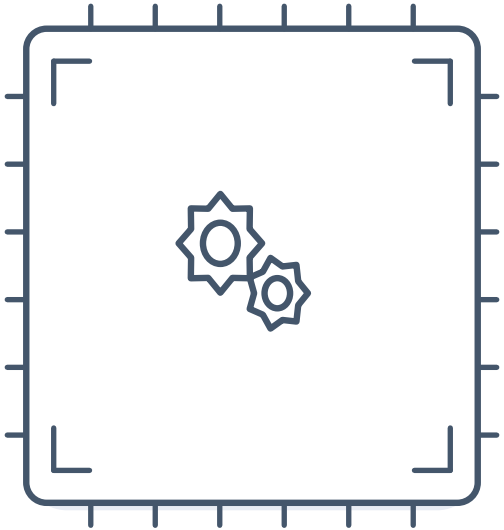
Localization

CHALLENGE

How to pinpoint It to hardware and software?

Every leakage is mapped to hardware and software

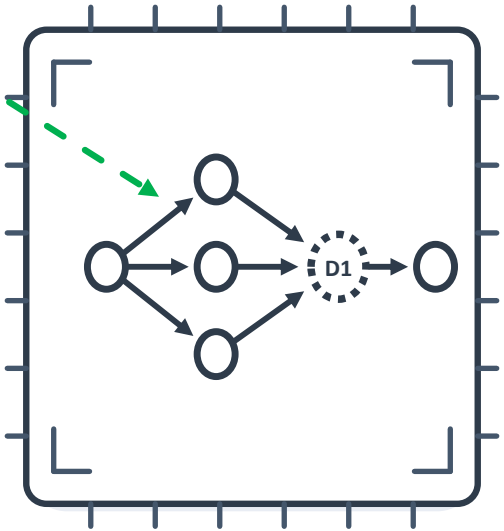
microSCALE – High Level Idea



Processor

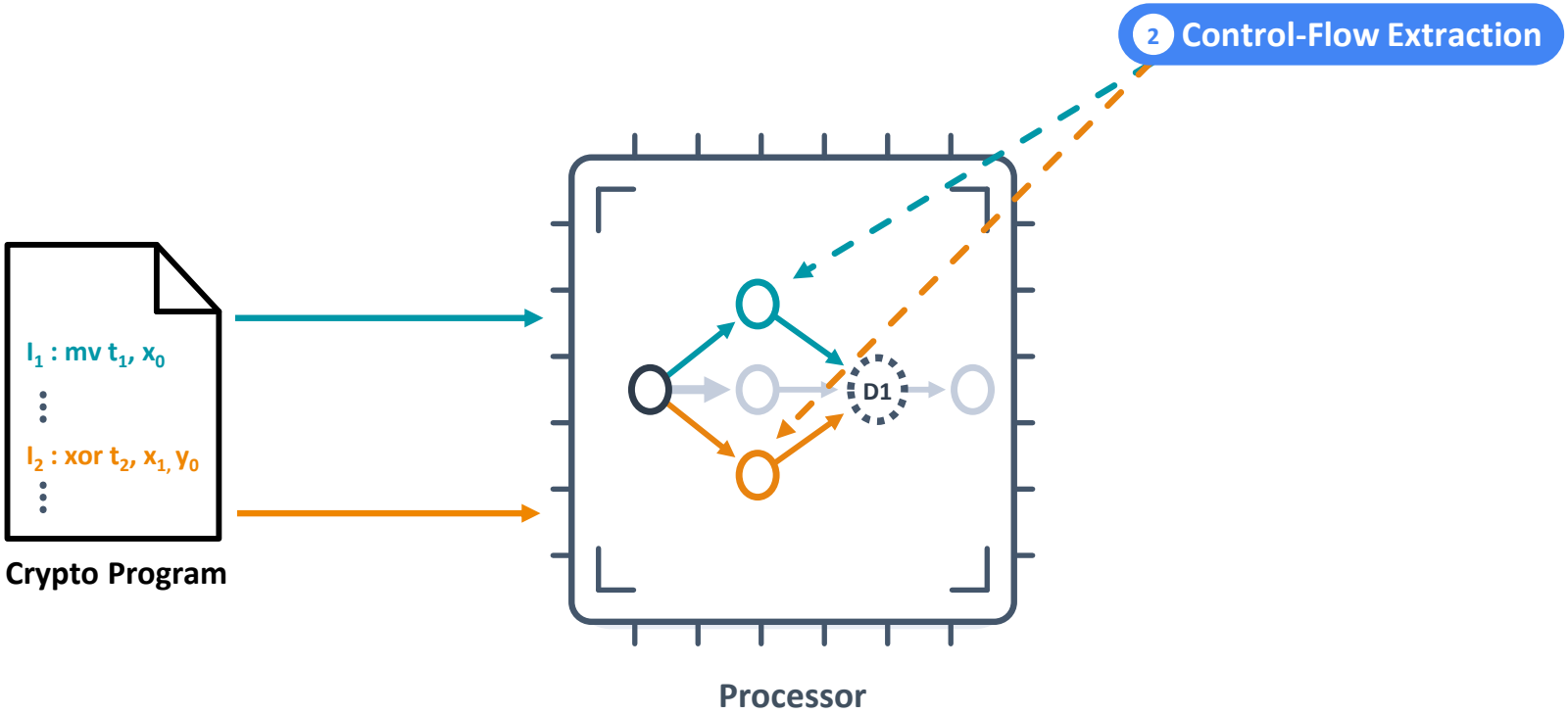
microSCALE – High Level Idea

1 Microarchitectural Graph Generation

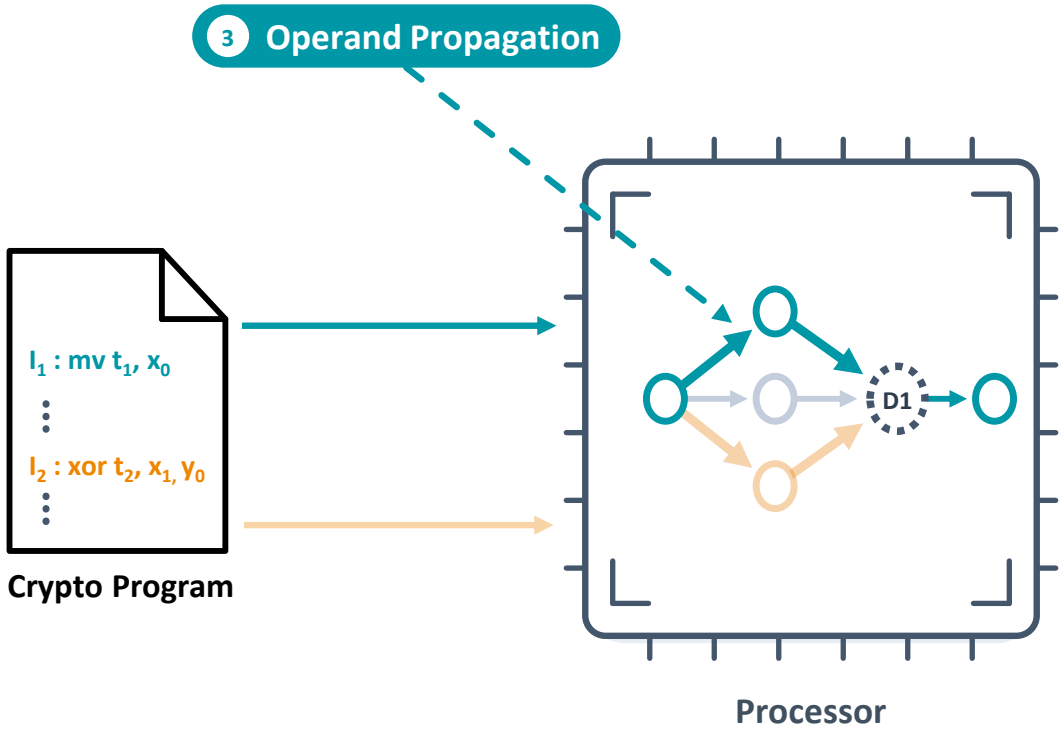


Processor

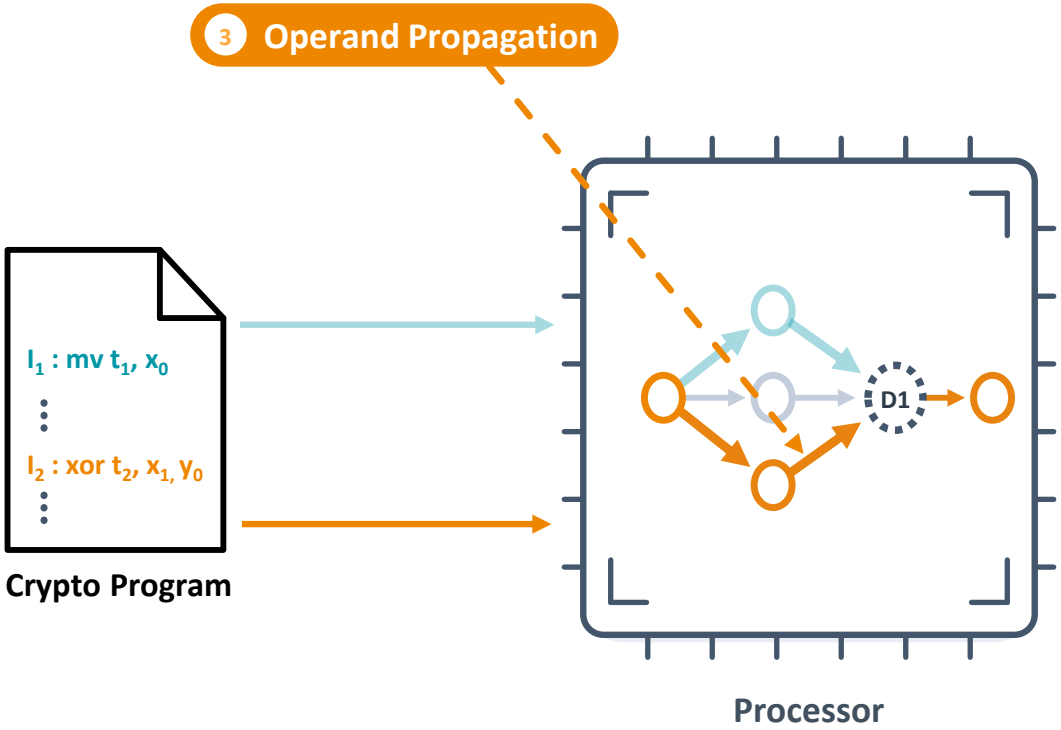
microSCALE – High Level Idea



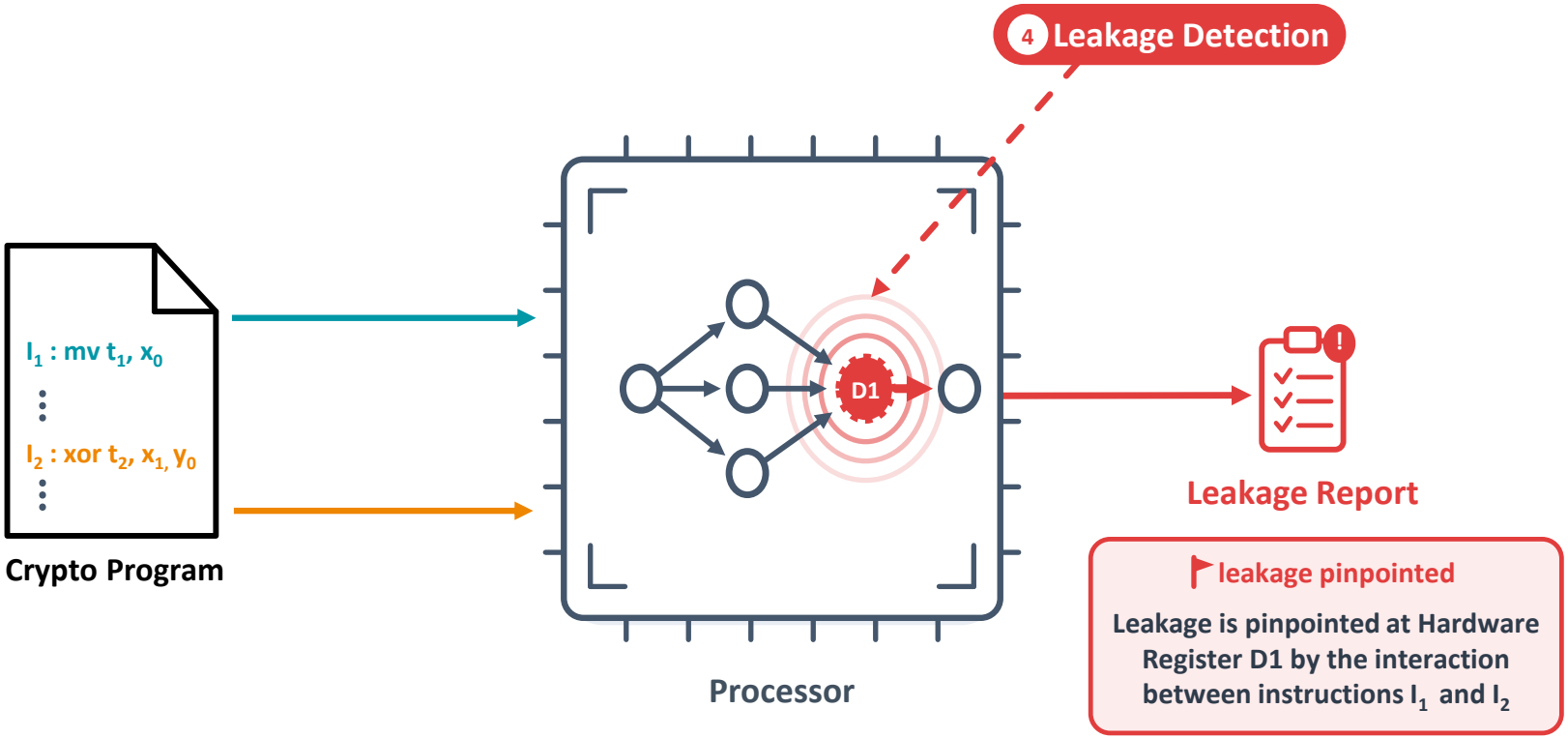
microSCALE – High Level Idea



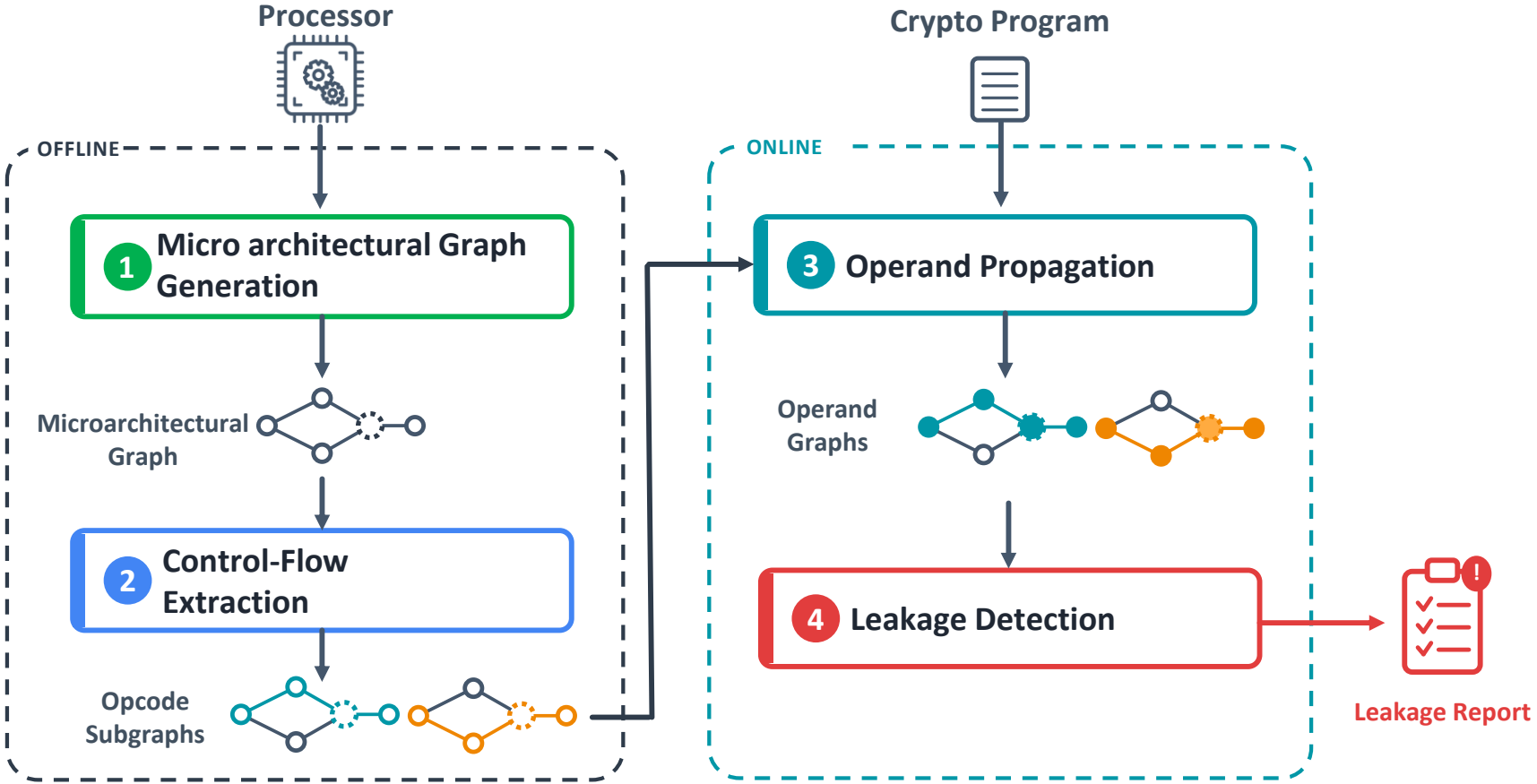
microSCALE – High Level Idea



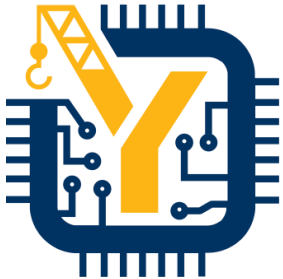
microSCALE – High Level Idea



microSCALE Framework



Putting microSCALE to the Test



RocketChip RISC-V Processor



GIFT



ASCON

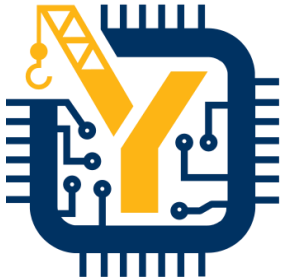


PRESENT



AES

Putting microSCALE to the Test



RocketChip RISC-V Processor



GIFT



ASCON



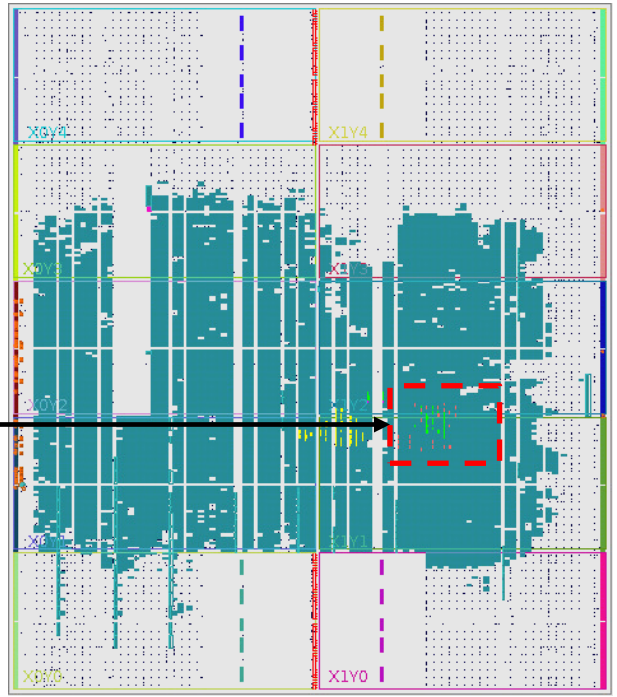
PRESENT



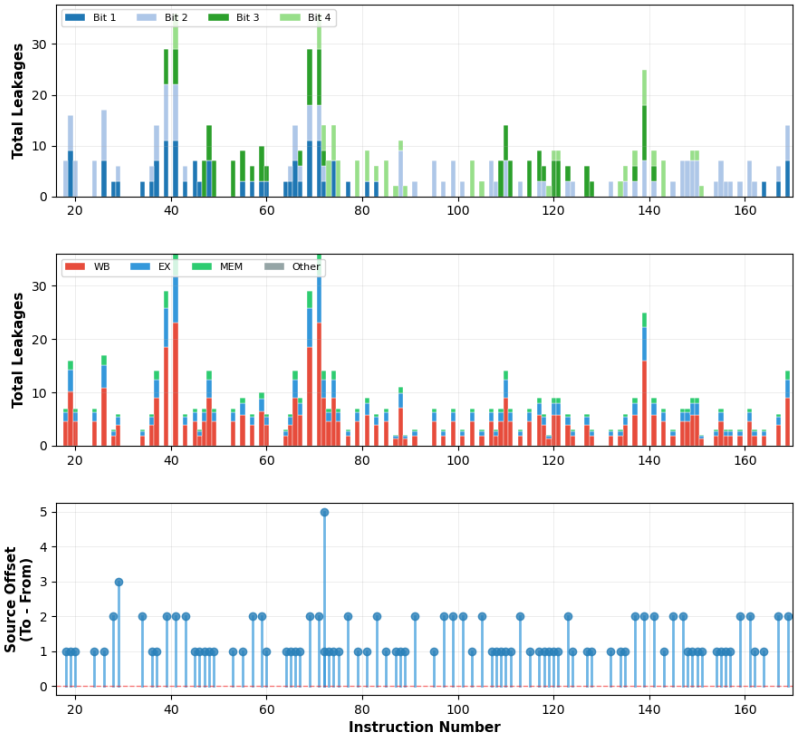
AES

Leakage Localization in Rocket Microarchitecture

```
....
xor s3, s3, a7
la s10, tempOut21
sh s2, 0(s10)
la s10, tempOut22
sh s3, 0(s10)
and s11, a0, a3
....
```



Analysis of GIFT S-box



Bit-level leakage

Leakage from individual secret-share bit positions.

Pipeline-stage leakage

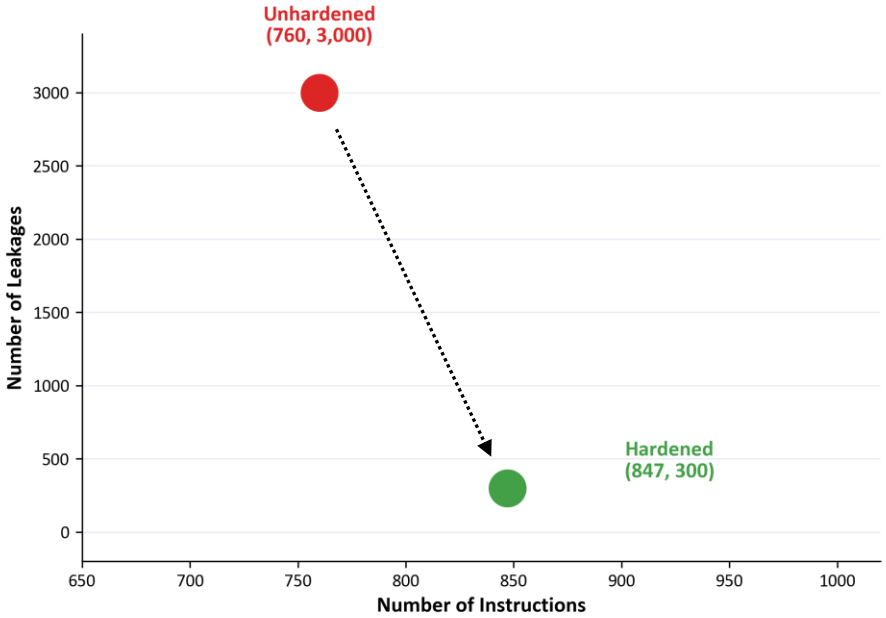
Leakage across the processor's pipeline stages.

Instruction-pair separation

Source–destination separation in leaking instruction pairs.

Protecting GIFT implementation using microSCALE

GIFT one-round - TVLA on 1M traces



90.3%

reduction in TVLA leakage points

87 dummy instructions (≈11% overhead) — Hardened based on microSCALE's report

What microSCALE Delivers?



Automated static RTL analysis



Graph-based representation of RTL



Leakage aware rules to track propagation



Validated on various ciphers (RISC-V Rocket), TVLA results prove the efficacy



Government of India
Ministry of Electronics and Information Technology



Thank You!

References

[1] David McCann, Elisabeth Oswald, and Carolyn Whit-nall. Towards Practical Tools for Side Channel AwareSoftware Engineering: 'Grey Box' Modelling for In-struction Leakages. In Engin Kirda and ThomasRistenpart, editors, 26th USENIX Security Symposium,USENIX Security 2017, Vancouver, BC, Canada, Au-gust 16-18, 2017, pages 199–216. USENIX Association,2017.

[2] Madura A. Shelton, Lukasz Chmielewski, Niels Samwel,Markus Wagner, Lejla Batina, and Yuval Yarom.Rosita++: Automatic Higher-Order Leakage Elimina-tion from Cryptographic Code. In Yongdae Kim, JongKim, Giovanni Vigna, and Elaine Shi, editors, CCS '21:2021 ACM SIGSAC Conference on Computer and Com-munications Security, Virtual Event, Republic of Korea,November 15 - 19, 2021, pages 685–699. ACM, 2021.

[3] Roderick Bloem, Barbara Gigerl, Marc Gourjon, VedadHadzic, Stefan Mangard, and Robert Primas. PowerContracts: Provably Complete Power Leakage Mod-els for Processors. In Heng Yin, Angelos Stavrou,Cas Cremers, and Elaine Shi, editors, Proceedings ofthe 2022 ACM SIGSAC Conference on Computer andCommunications Security, CCS 2022, Los Angeles, CA,USA, November 7-11, 2022, pages 381–395. ACM, 2022

[4] Barbara Gigerl, Vedad Hadzic, Robert Primas, StefanMangard, and Roderick Bloem. Coco: Co-Design andCo-Verification of Masked Software Implementationson CPUs. In Michael D. Bailey and Rachel Greenstadt,editors, 30th USENIX Security Symposium, USENIXSecurity 2021, August 11-13, 2021, pages 1469–1468.USENIX Association, 2021