Compass: Navigating the Design Space of Taint Schemes for RTL Security Verification

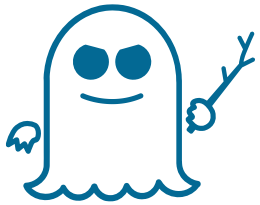
Yuheng Yang* (MIT), Qinhan Tan* (Princeton)

Thomas Bourgeat (EPFL), Sharad Malik (Princeton), Mengjia Yan (MIT)

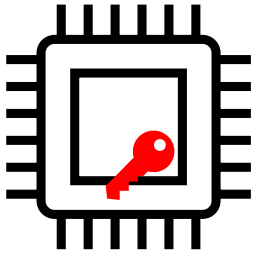
**Co-first Authors*

Hardware Security Problems

- Side channel attacks
- Spectre attacks



- TEEs root-of-trust key leakage

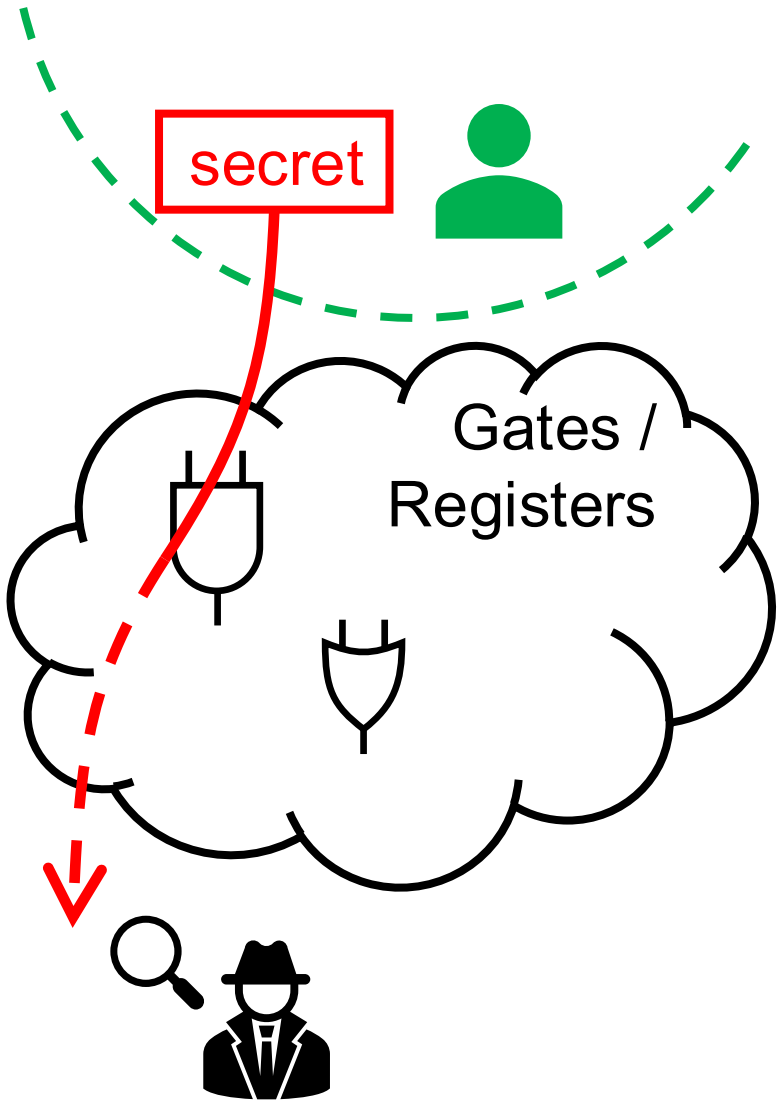


- ……

Violations of **Information Flow Property**

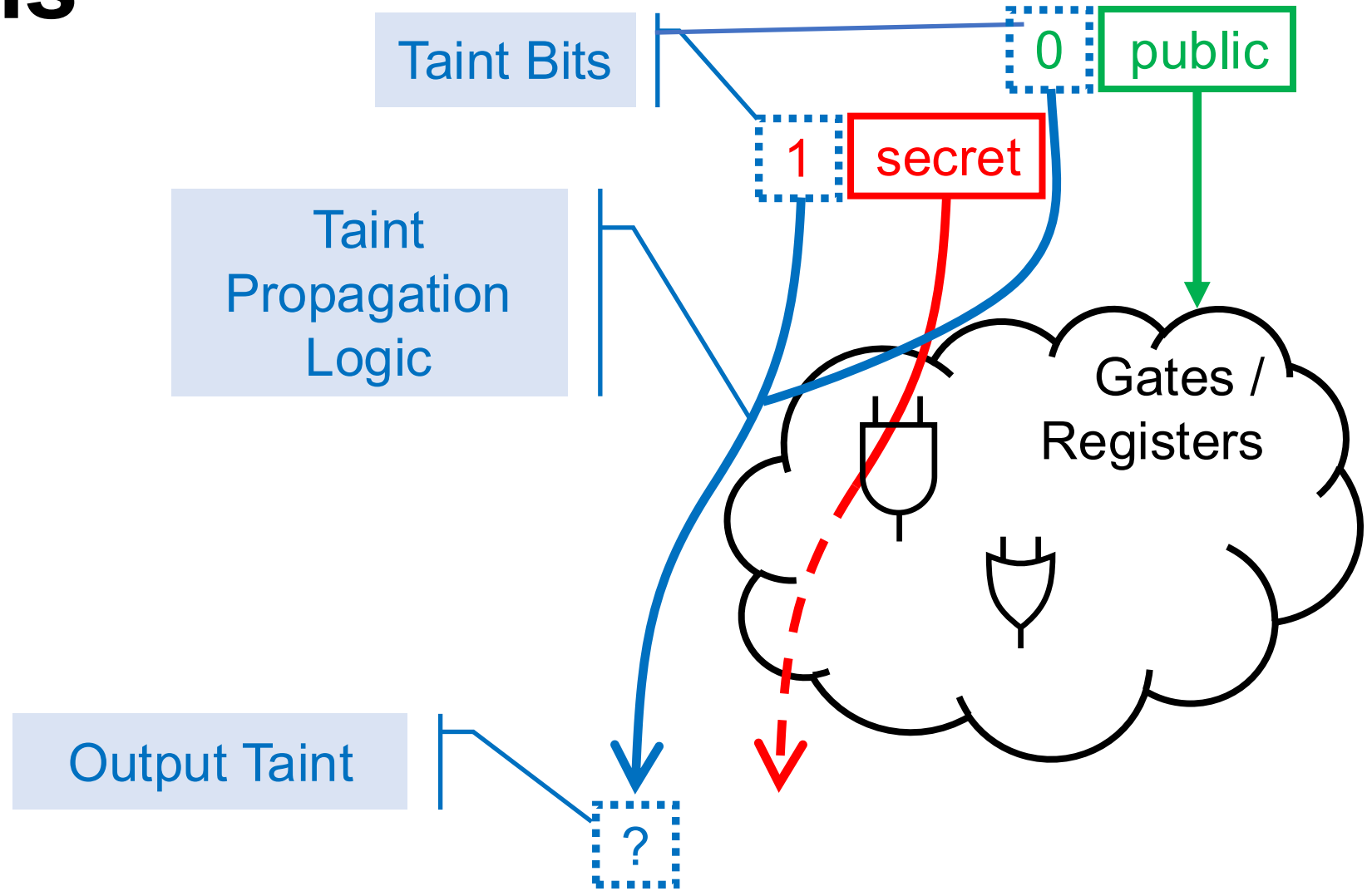
Information Flow Property

Information flow property checks whether a secret can influence the value of attacker observable signals



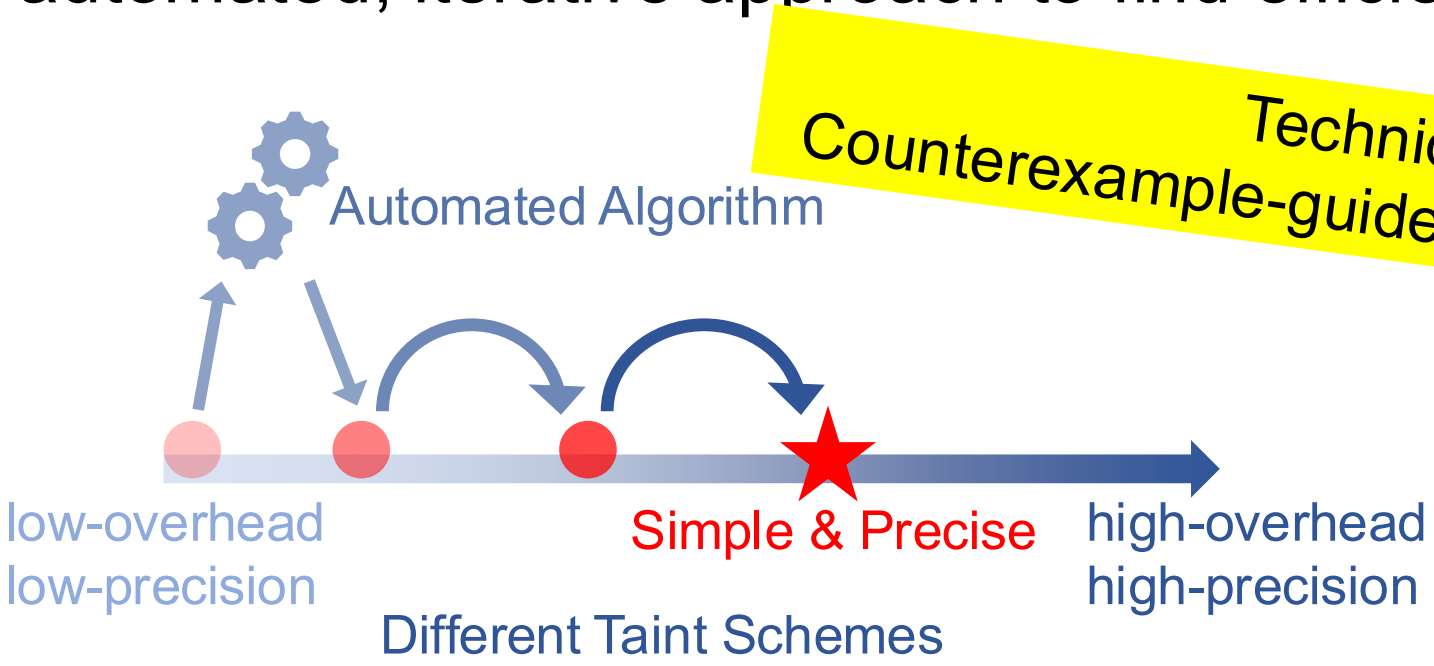
Taint Analysis

Taint analysis over-approximates information flow

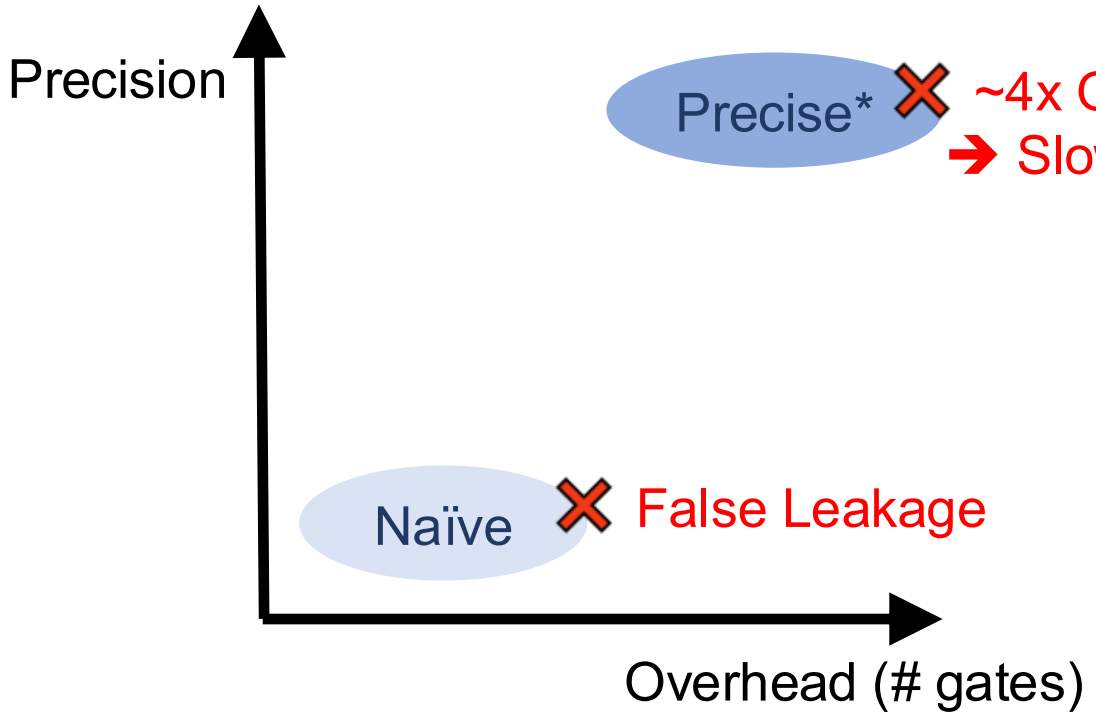


Overview

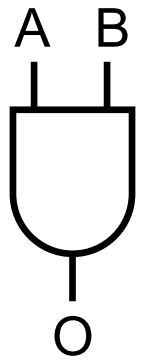
- Observation: There is a large design space of taint schemes
- Goal: Find taint schemes that are simple and precise
- Solution: An automated, iterative approach to find efficient taint schemes



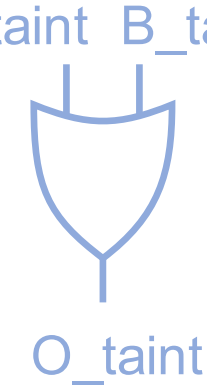
Design Taint Schemes



AND Gate



Naïve Taint Logic

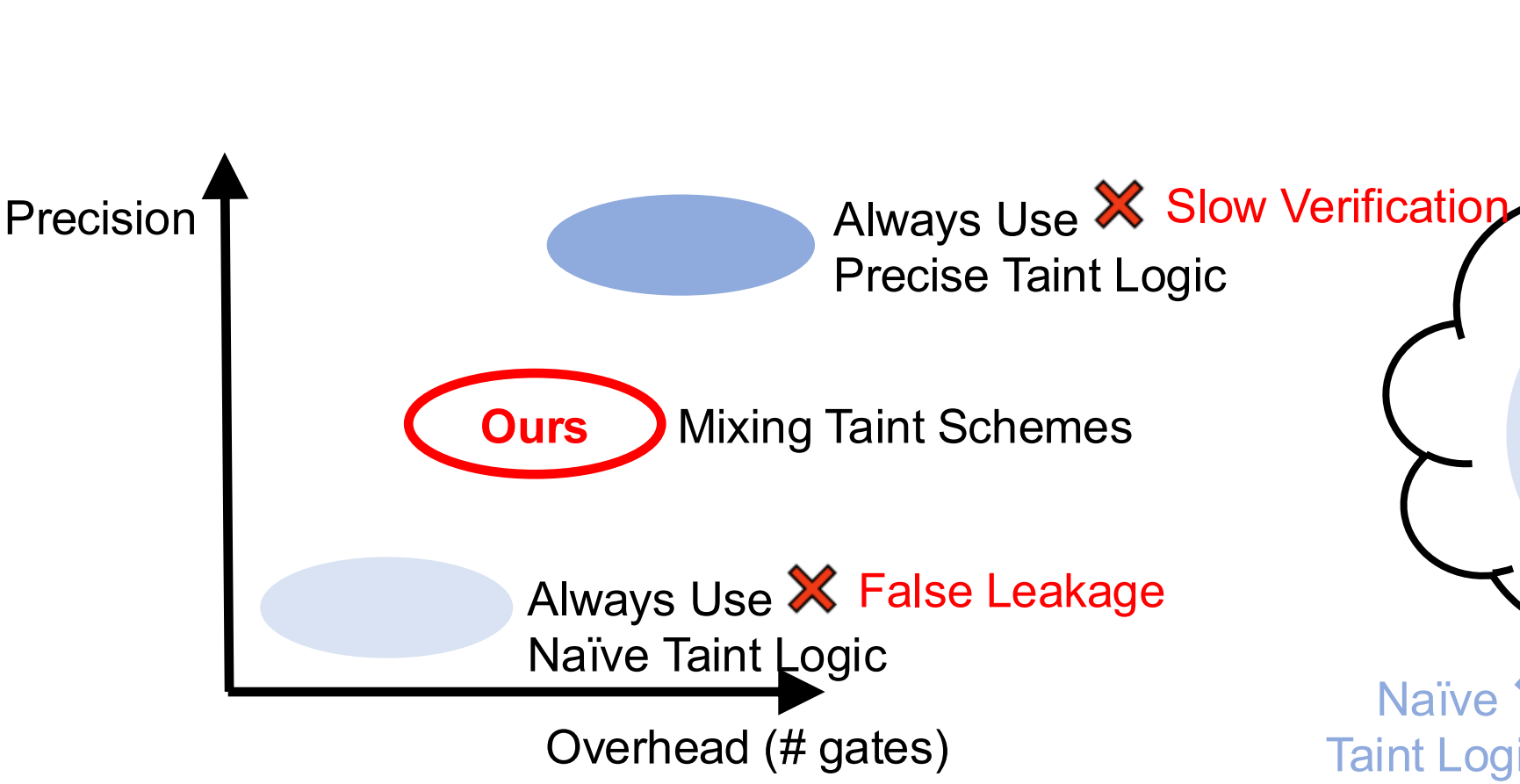


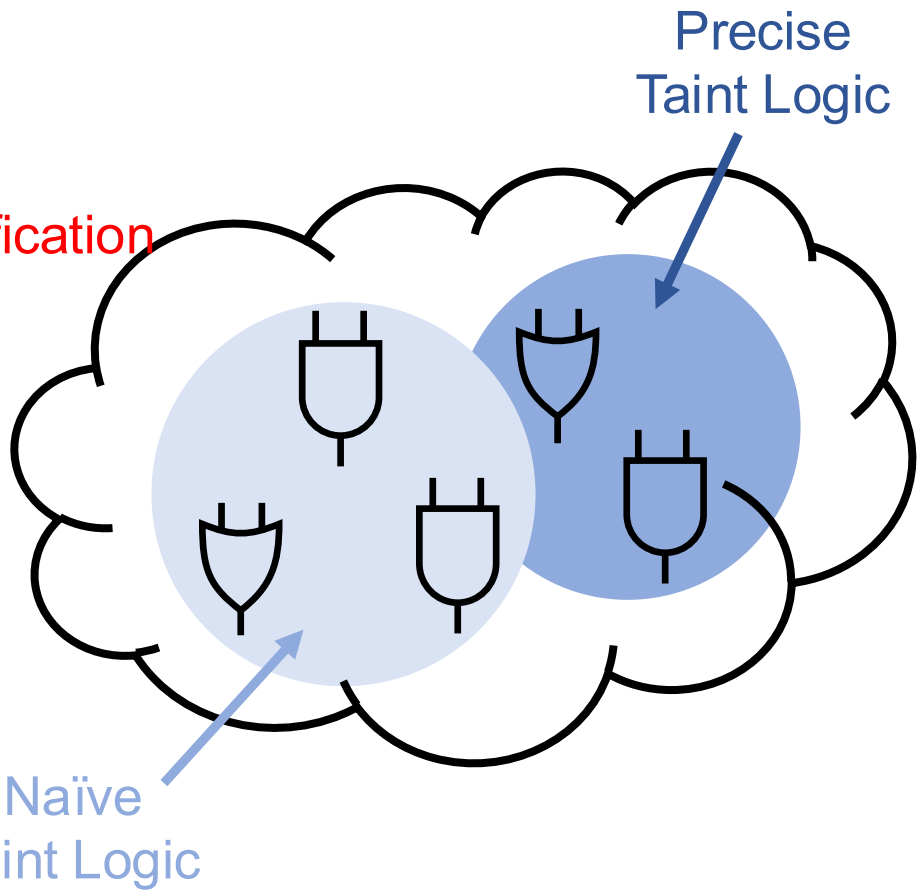
Naïve taint logic introduces false flow

A	B	O	Leakage ?	O_taint
0	0			
1	0			

*Complete Information Flow Tracking from the Gates Up. Mohit Tiwari, et al. 2009.

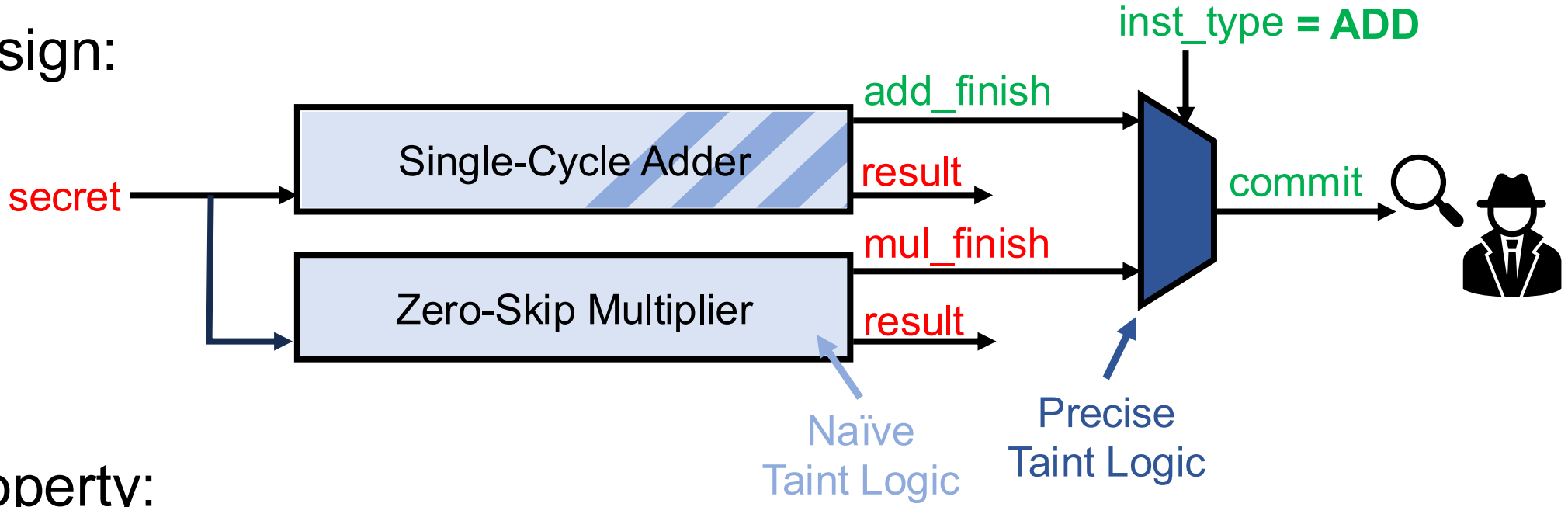
Design Taint Schemes





Advantage of Mixing Taint Schemes

- Design:

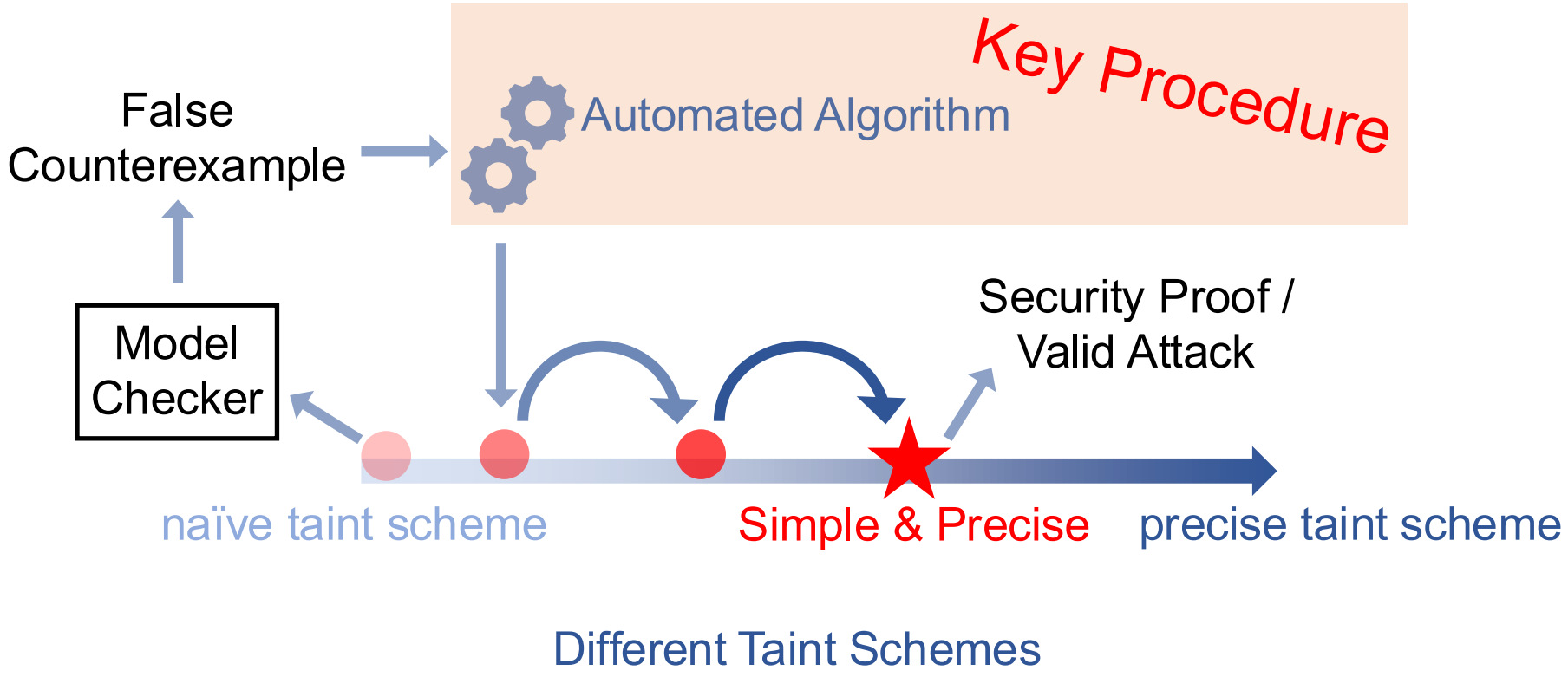


- Property:
  - Prove that if **`inst_type==ADD`**, then **`commit_taint==0`**

Challenge: How to automatically discover gates that need precise taint logic?

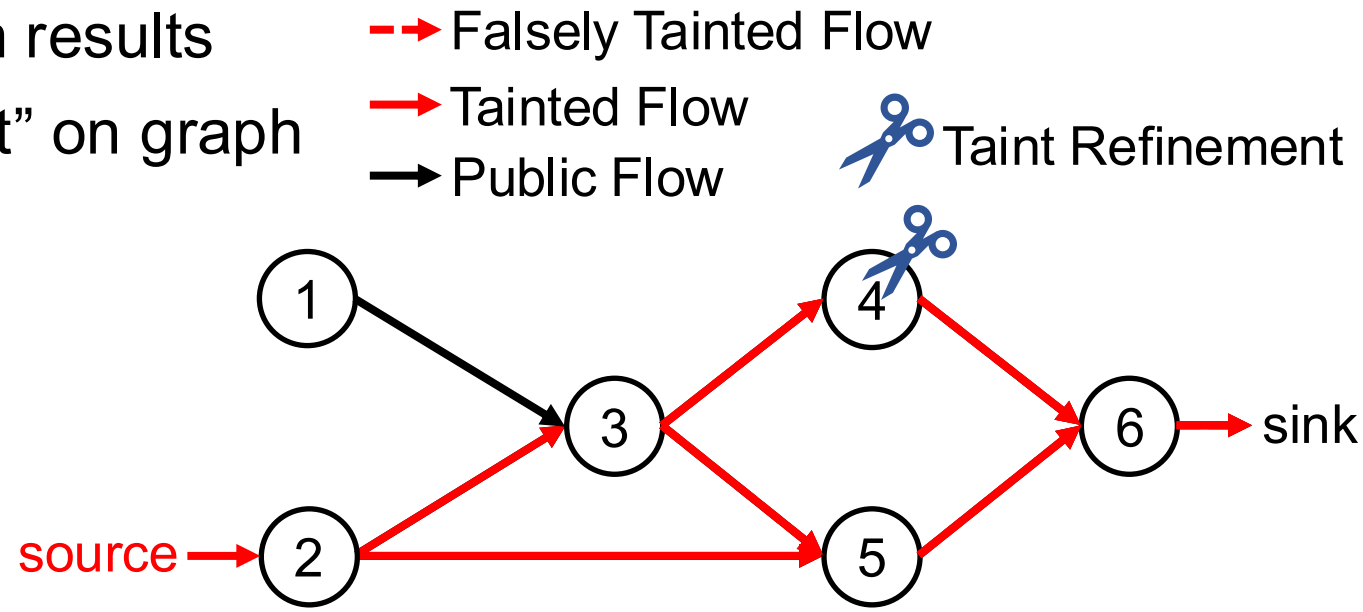
Compass Framework

Automated Counterexample-Guided Taint Refinement



Refinement Problem Setup

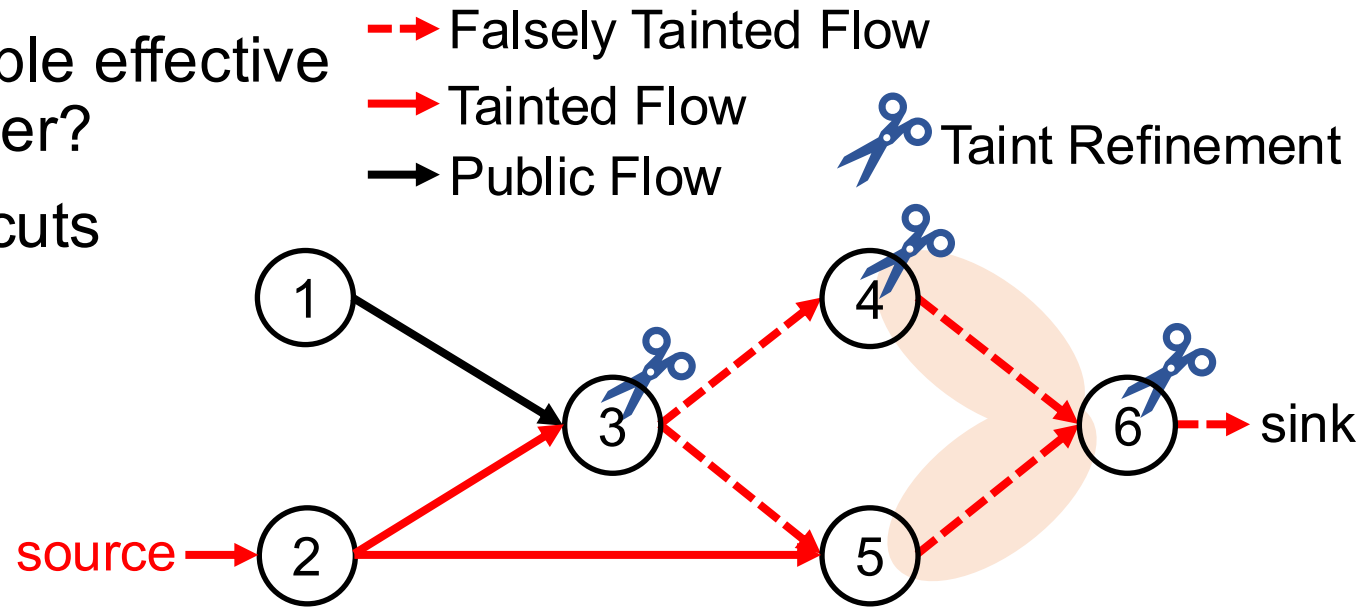
- View circuit as a graph
- Color edges based on taint propagation results
- View taint refinement of a gate as a “cut” on graph



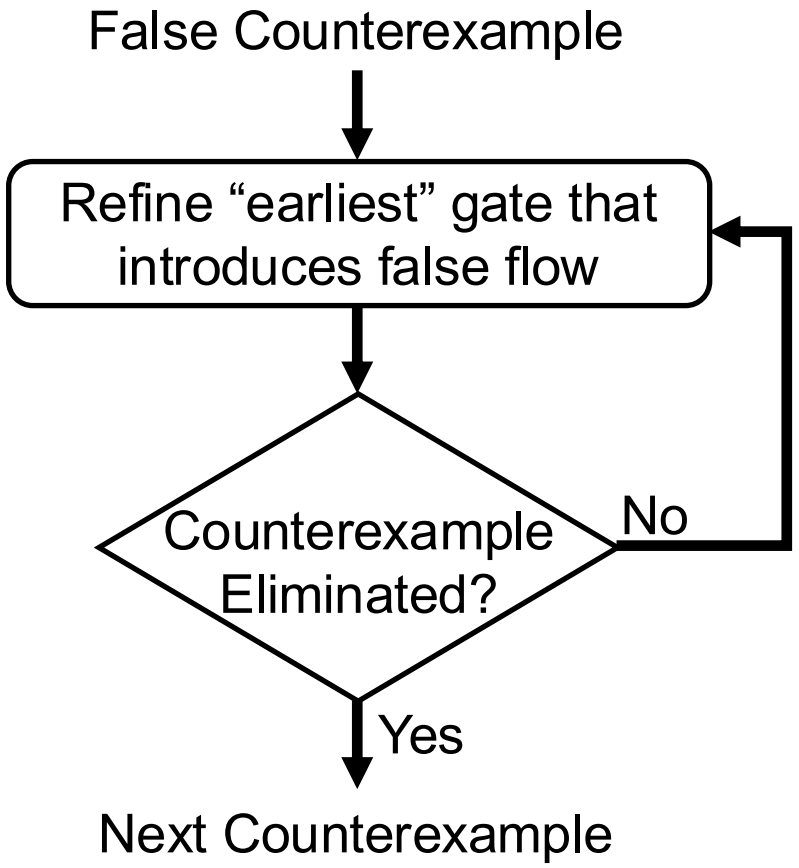
Eliminate false counterexample = Find cuts on graph

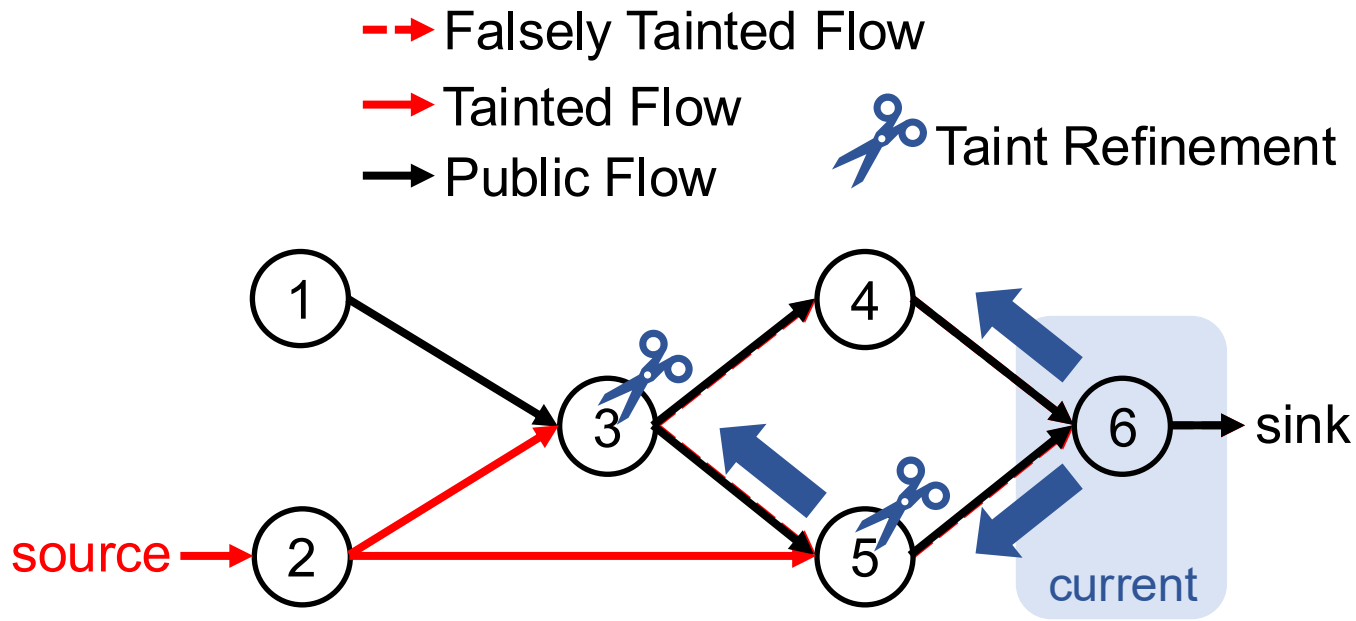
Some Complications…

- Not all refinements can effectively remove false flows
- Given a false flow, there can exist multiple effective cuts to remove it. Which one works better?
- Sometimes, we need to make multiple cuts together to remove a false taint



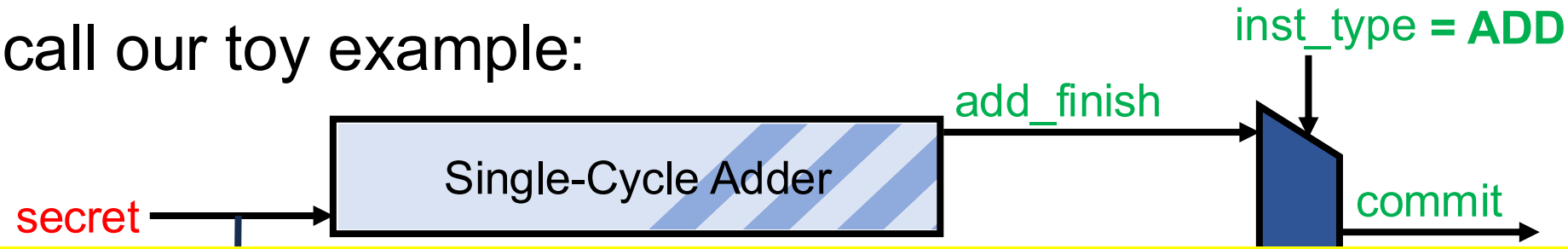
Our Iterative Algorithm





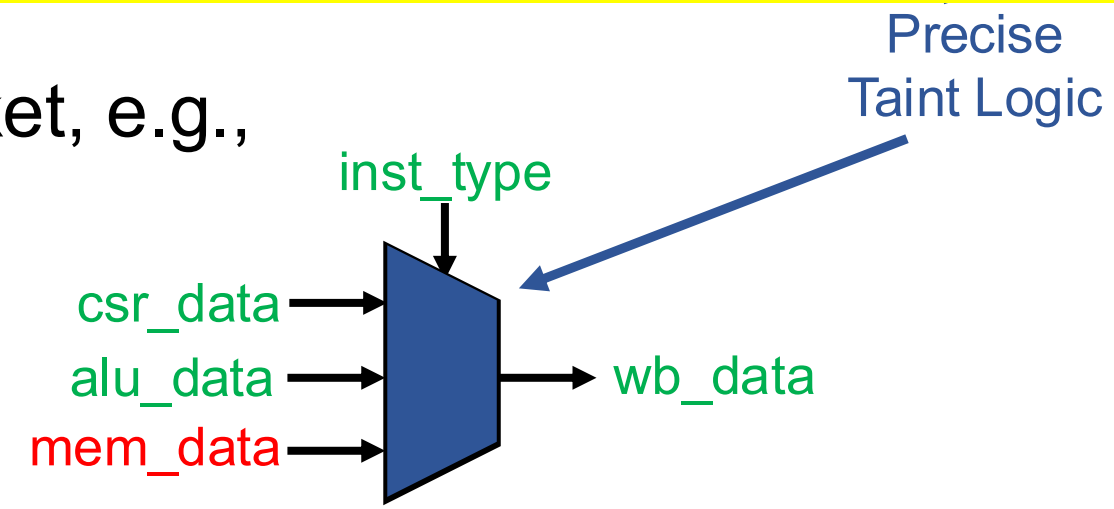
Analysis of Refinement Results

- Recall our toy example:

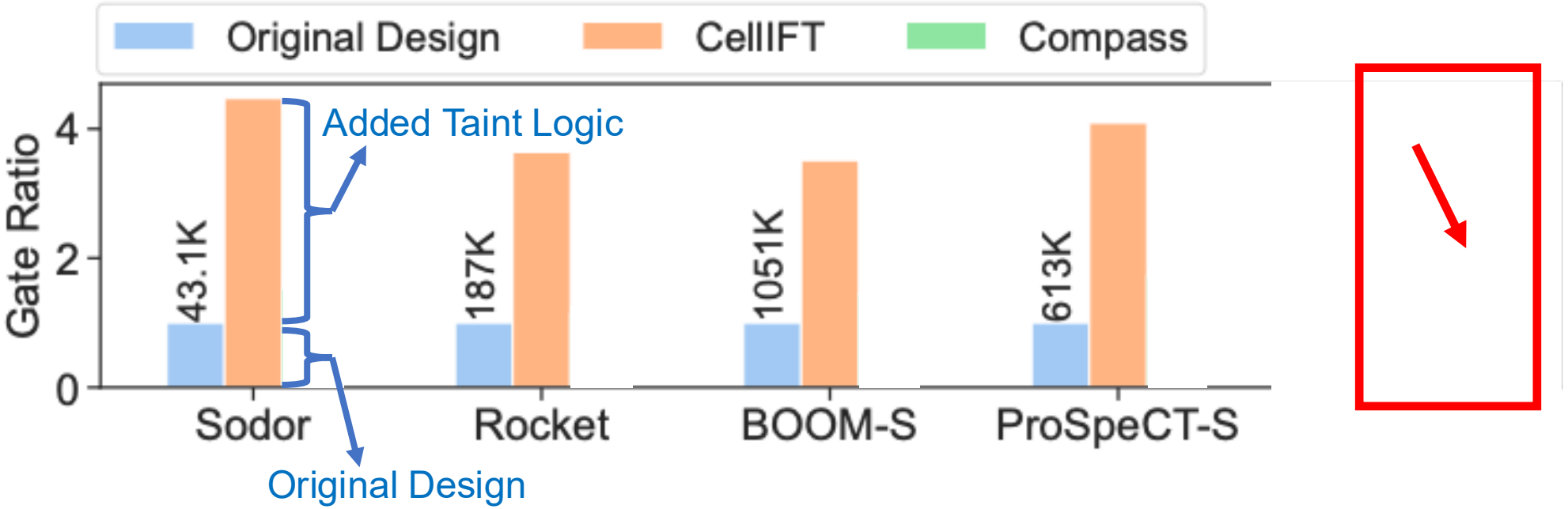


Finding: Most taint refinements target gates that operate at the boundary between secret & public signals

- It is common in Rocket, e.g.,



Result 1: Lower Taint Overhead

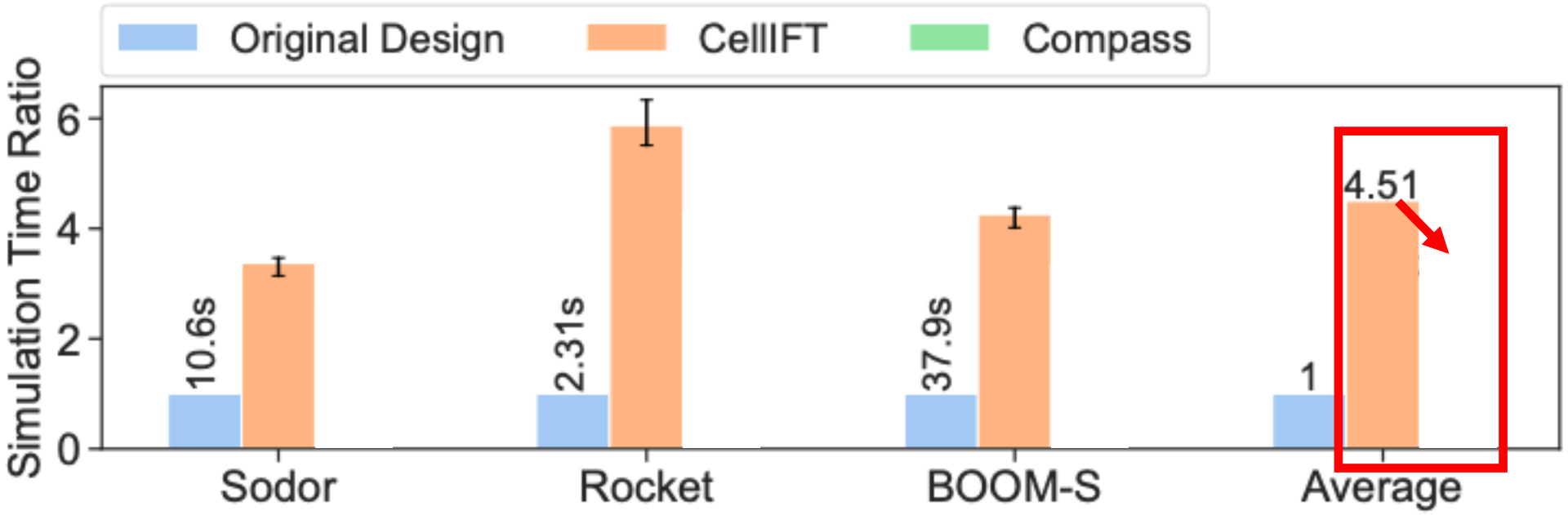


84% fewer taint gates

Security Property: Secure speculation[1]
Baseline: CellIFT[2]

[1]Hardware-Software Contracts for Secure Speculation. Marco Guarnieri, et al. 2021.
[2]CellIFT: Leveraging Cells for Scalable and Precise Dynamic Information Flow Tracking in RTL. Flavien Solt, et al. 2022.

Result 2: Faster Simulation



32% less simulation time

Benchmarks: median, rsort, qsort, matrix_mul, rsa

Result 3: Faster Model Checking

Unbounded Proof

	CellIFT	Compass	
	Proof Time	**Proof Time**	**Time to Find Taint Scheme**
Sodor	2 h	10 s	5 min

Faster Proof

Bounded Proof

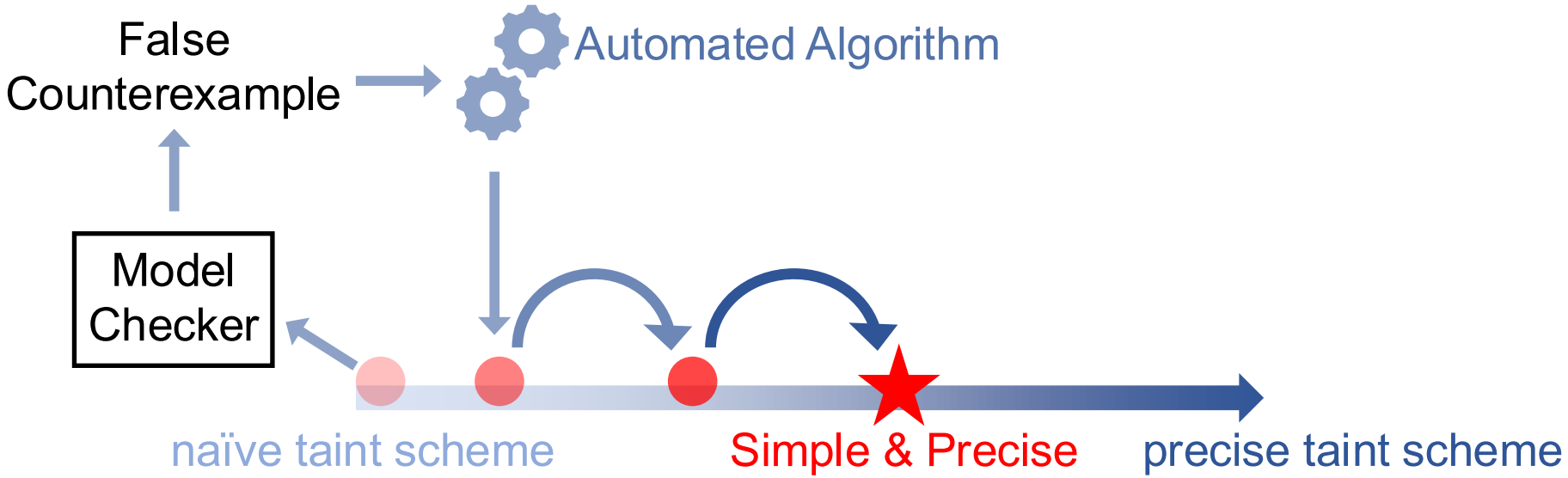
	CellIFT	Compass	
	Bound with 7 days	**Bound with 24 h**	**Time to Find Taint Scheme**
Rocket	41	159	1 h
BOOM	26	28	31 h
ProSpeCT	29	29	35 h

Check More Cycles

https://github.com/MATCHA-MIT/Compass

Conclusion

- Compass:



Compass finds simple yet precise taint schemes, enabling faster security verification