Alcatraz: Secure Remote Computation via Sequestered Encryption in Hardware Security Module

Albert Lu

Mentors:
Jules Drean and Sacha Servan-Schreiber

October 19th, 2025
MIT PRIMES October Conference

Side-Channels Attacks

Code

Shared Resources
(e.g. Cache)

Attacker

Attacker can observe

“SECRETS”

One program can exploit shared resources to spy on another

Example of Side Channels

Family using the internet

access

You watching a movie

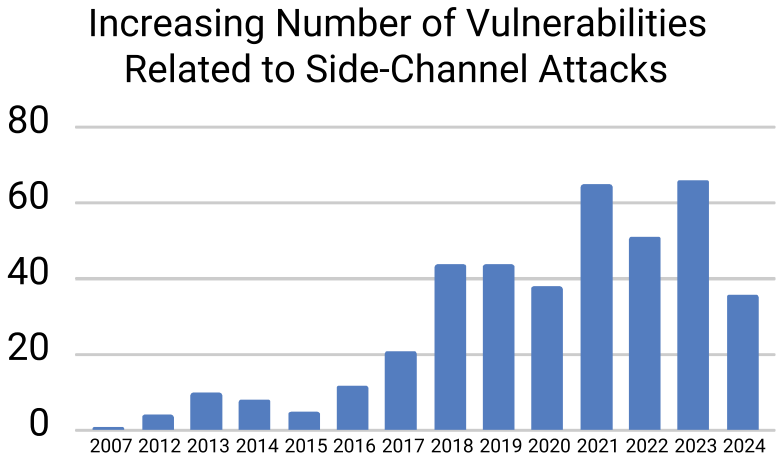
access

Shared Resources
Wifi

You observe “movie lags a lot” INFER “family is also using wifi”

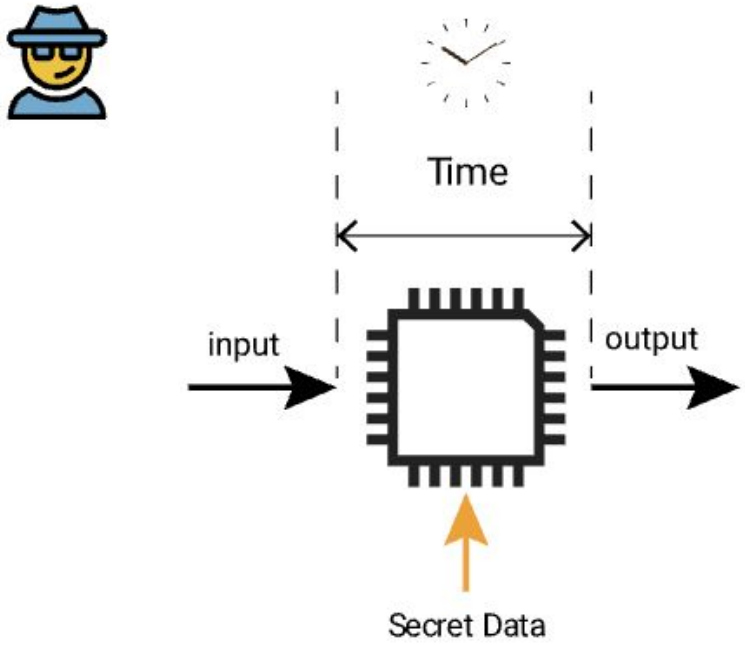
Importance of Side-Channel Attacks

- Side-channel attackers
  - Attack hardware, not just software
  - Extract cryptographic keys
- Growing number of vulnerabilities



Data Source: Common Vulnerabilities and Exposures (https://cve.org).

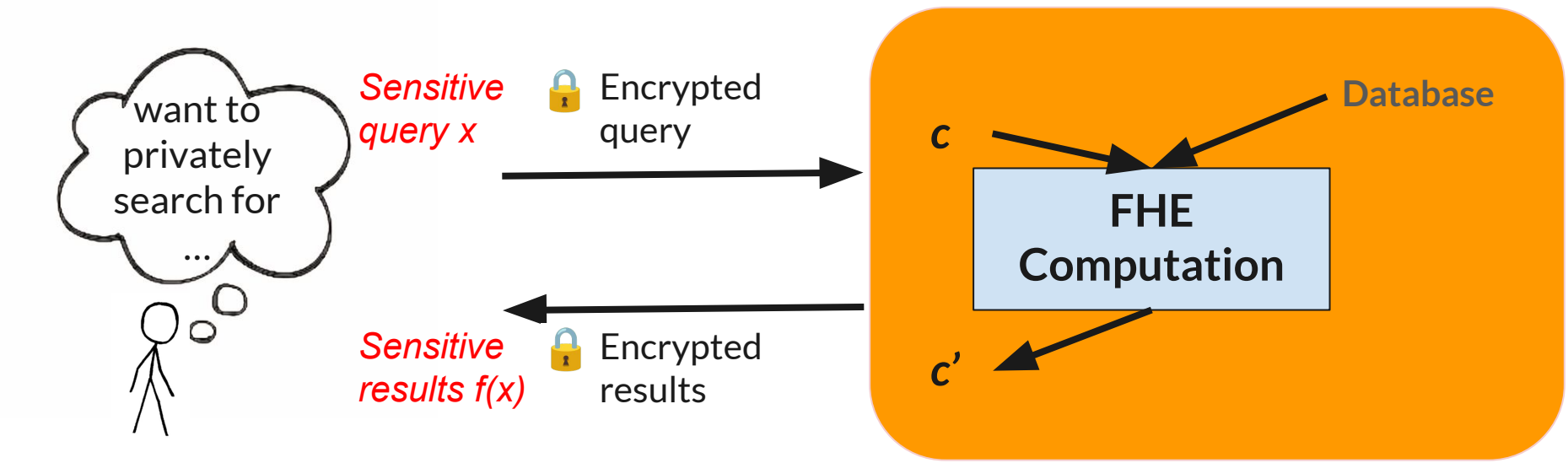
Timing Based Side Channel Attacks



Secure Remote Computation

	Ideal Solution: Fully Homomorphic Encryption (**FHE**)	More Practical: Trusted Execution Environment (**TEE**)	Our Solution: **Alcatraz** (inspired by both)
Security	Based on strong cryptographic assumption	Based on empirical hardware security; Vulnerable to side-channel attacks	Minimal trusted hardware; Protected against side-channels
Efficiency	Slow	Fast	Fast

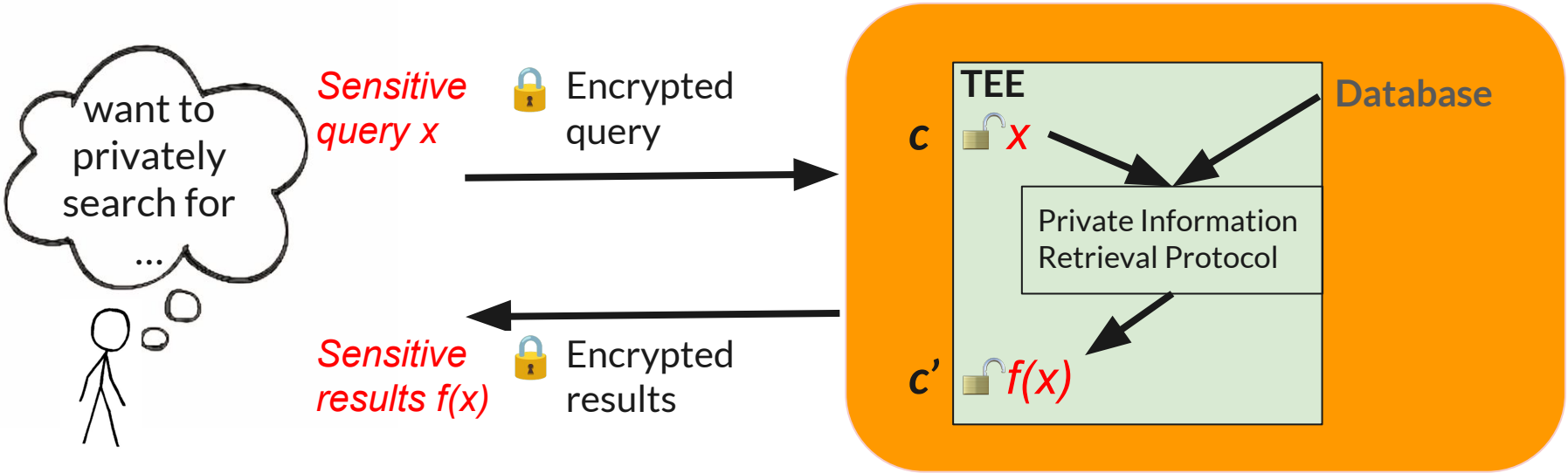
Fully Homomorphic Encryption (FHE)



FHE:

- Computes directly on encrypted data (*ciphertext c*) → Slow
- Never exposes sensitive data

Trusted Execution Environment (TEE)



TEE

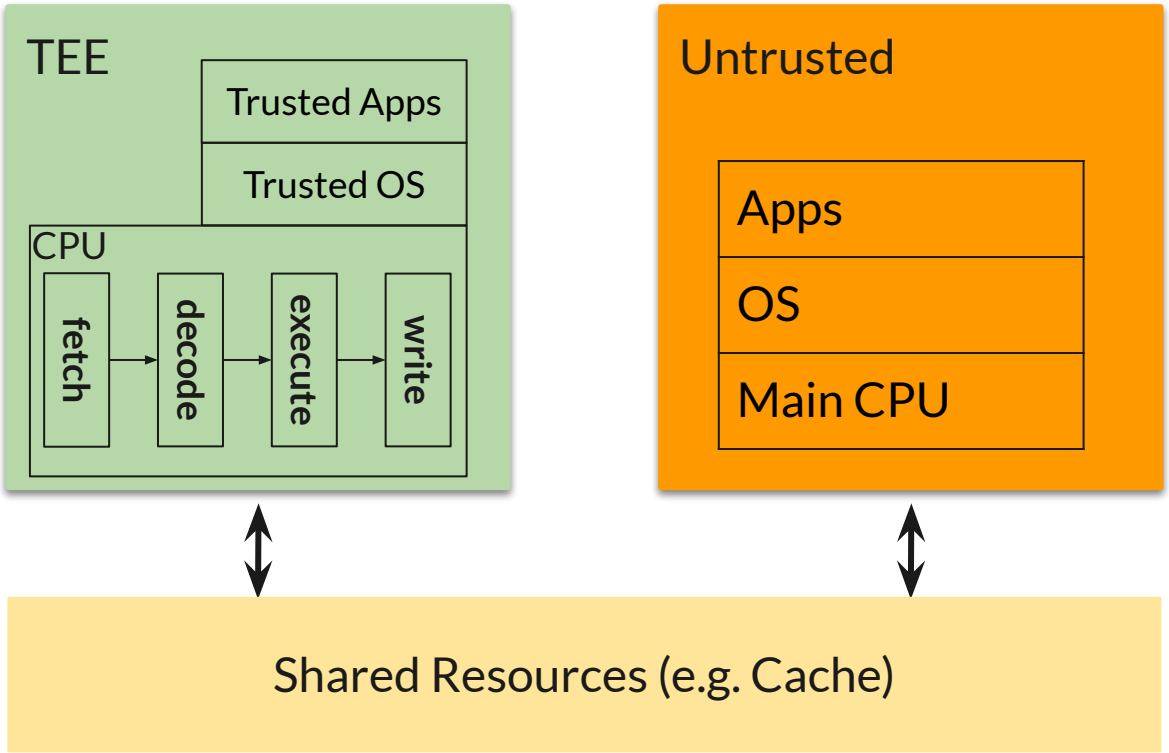
- Trusts TEE so operation is done on unencrypted data → Fast
- Problem: leads to large attack surface, subject to side-channel attacks

Our Solution to Secure Remote Computation

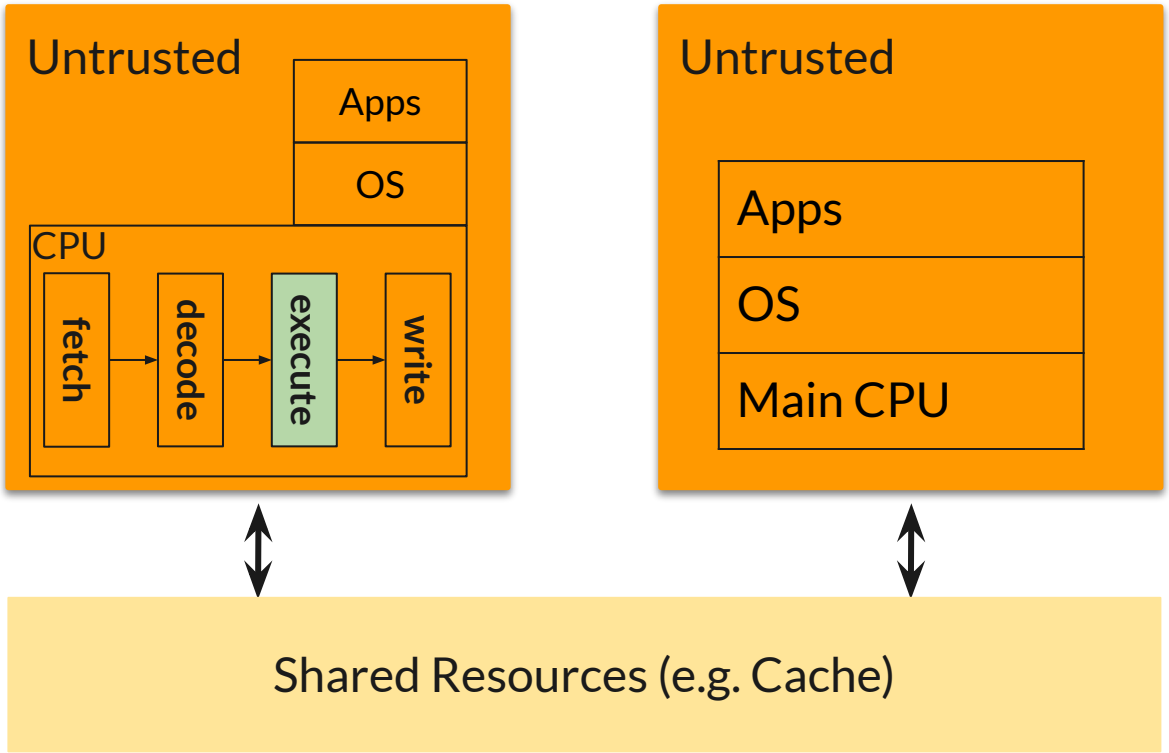
- Take inspiration from Fully Homomorphic Encryption (FHE) and Trusted Execution Environment (TEE)
- Based on trusted hardware
- BUT reduce our “trusted area” as much as possible
- Result: **mitigate side channel attacks**

Our Objective:
Reducing Area of Trust

Trusted Execution Environment and Shared Resources



Minimize Trusted Hardware within Execute Stage



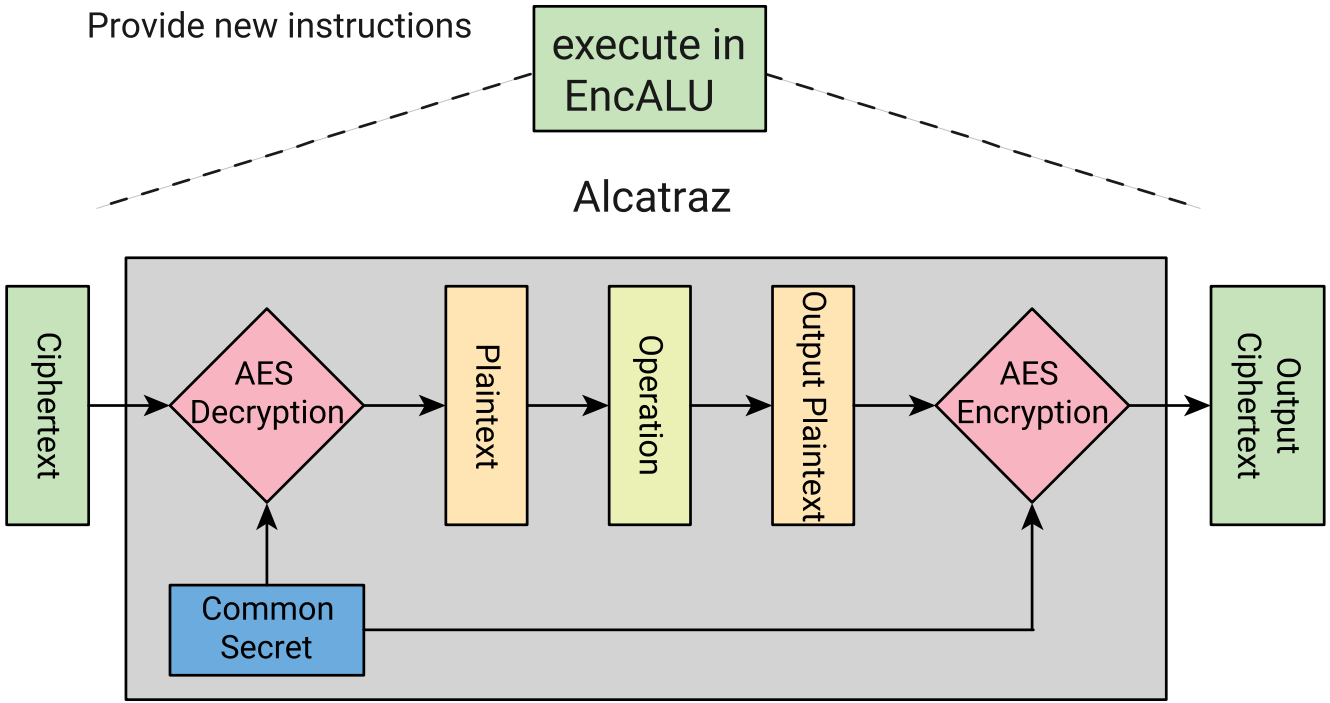
Research Questions

- RQ1: How to create an architecture that minimizes the attack surface?
  - Design Alcatraz, an encrypted Arithmetic Logic Unit (ALU) gated by Sequestered Encryption

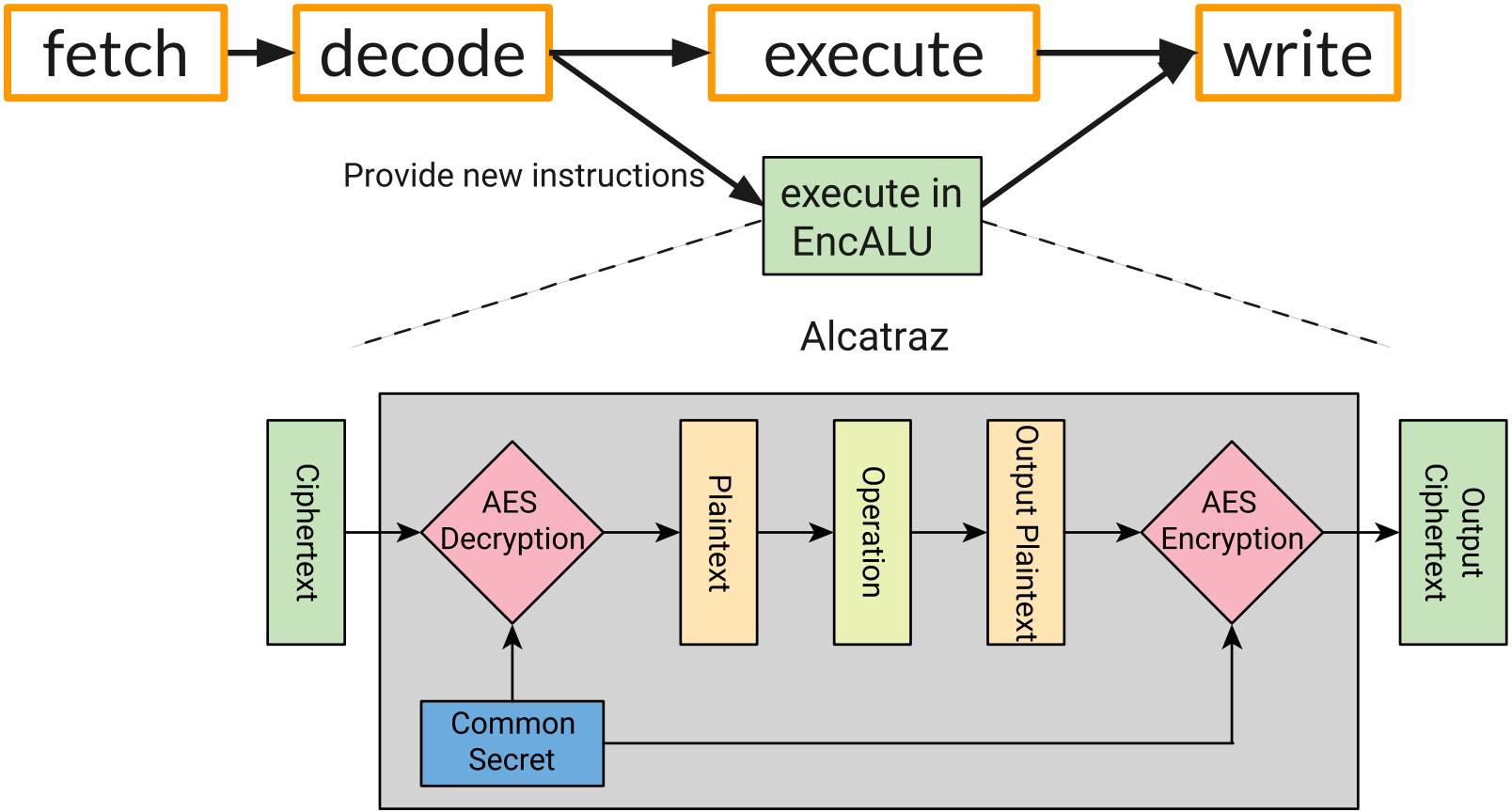
- RQ2: How to verify our architecture is correct and secure?
  - Apply the Knox framework to prove security

- RQ3: Apply Alcatraz to PIR and compare with existing state-of-the-art FHE approaches
  - 7-21x faster

Our Design (Alcatraz): an Encrypted ALU

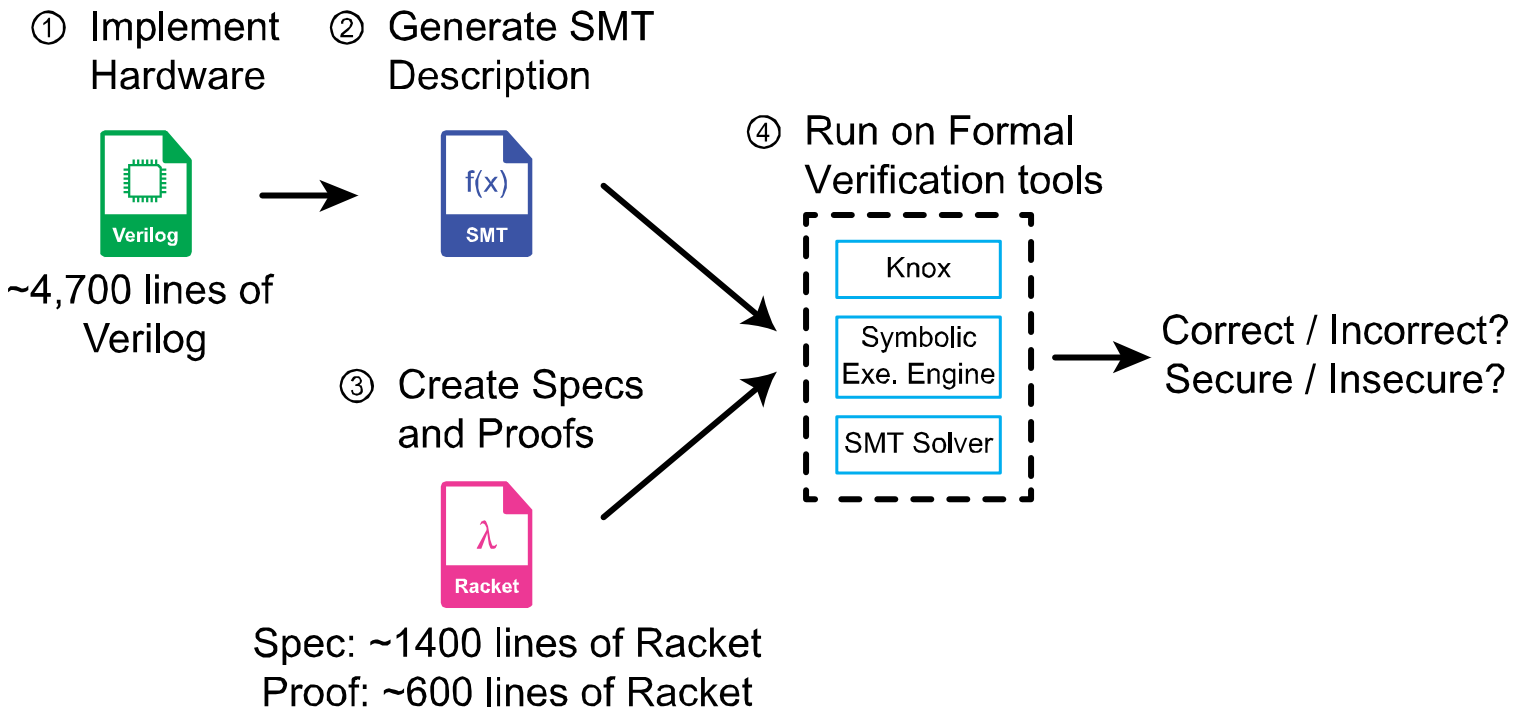


Our Design (Alcatraz): an Encrypted ALU



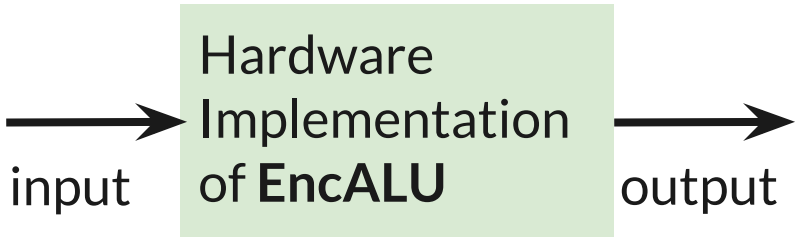
Proving Security of Encrypted ALU Against Timing-Based Side Channel Attacks

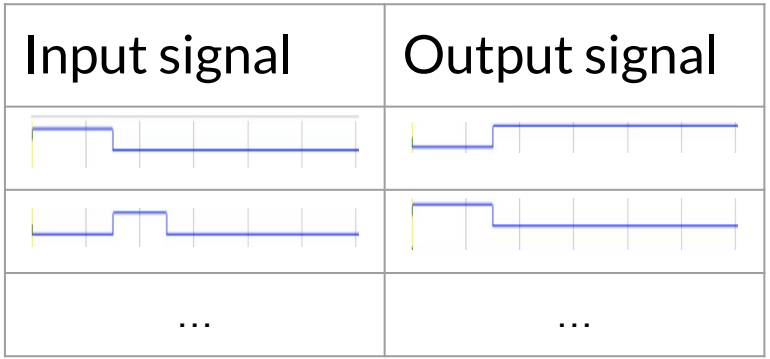
Verification Overview



Formal Verification

- Goal: prove our hardware module is secure against **all** possible inputs
- Challenge: **Infeasible** to try all types of input signals one by one





- Solution: Knox Framework

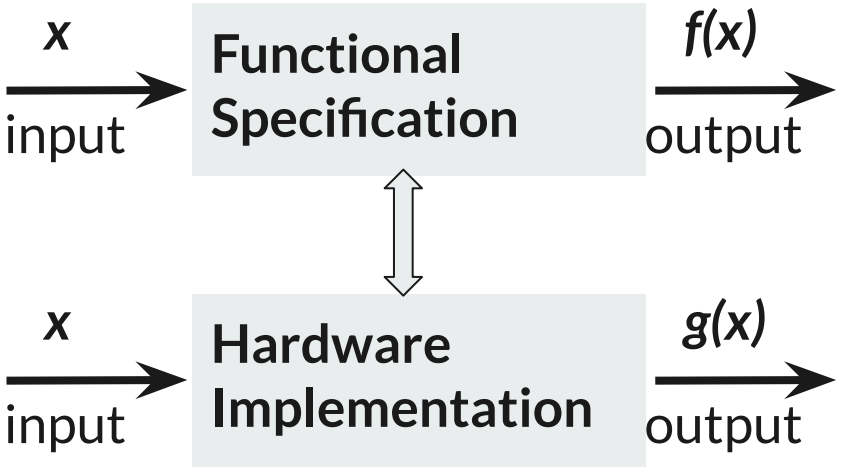
Knox Framework: Prove "*f(x) ≠ g(x)*" is Unsatisfiable

- Use "symbols" to represent the input signals
- ***f(x)***: symbolic execution result following the **functional specification**
- ***g(x)***: symbolic execution result following the **hardware implementation**

Ideal world (correct and secure)

We want to prove these two are indistinguishable

Real world



Use **Satisfiability Modulo Theories** (SMT) solvers to prove the formula

Athalye et al. "Verifying Hardware Security Modules with Information-Preserving Refinement" (OSDI 2022)

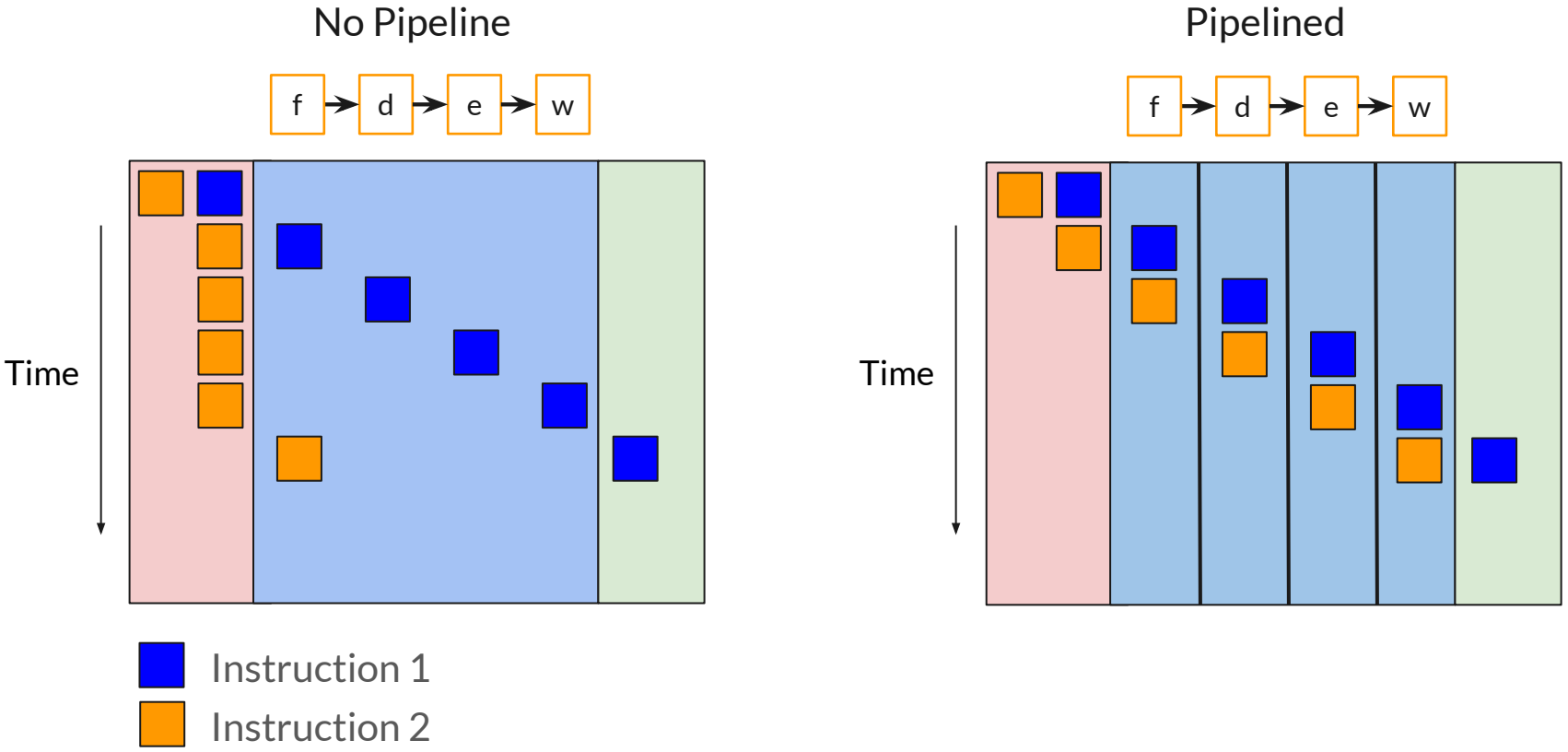
Applying Alcatraz to Private Information Retrieval

Overview of Implementation

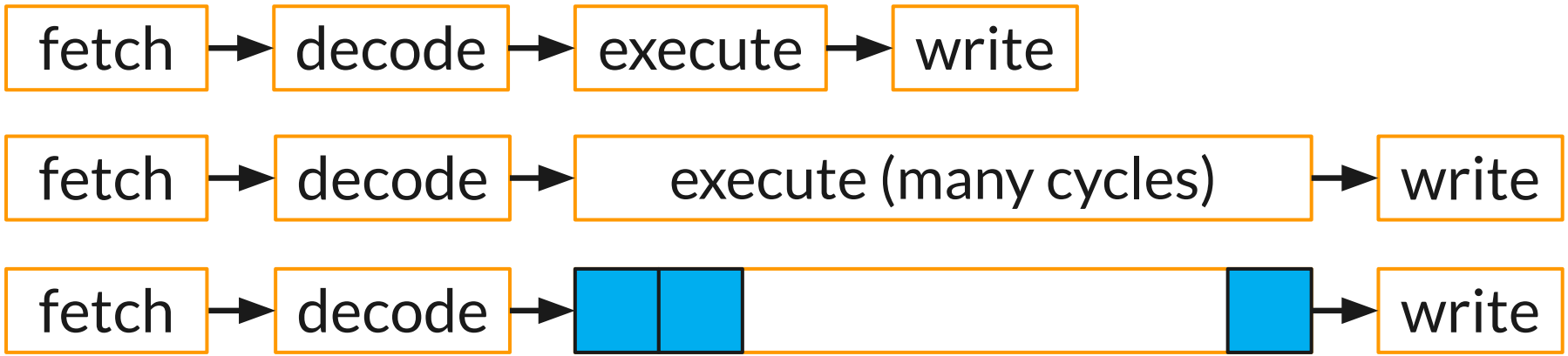
- We implemented our encrypted ALU in Verilog to extend an open source RISC-V core (Ibex) and vector coprocessor (Vicuna)
- Simulated using Verilator
- Encoded the customized instructions using inline assembly
- Compared performance of Private Information Retrieval

https://ibex-core.readthedocs.io/en/latest/
https://vicuna.readthedocs.io/en/latest/

Pipelining Stages: **f**etch, **d**ecode, **e**xecute, **w**rite



Superpipelining Execute Stage

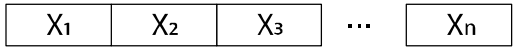


Superpipelining execute stage

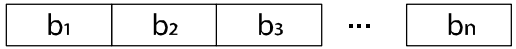
- The execute stage of Alcatraz takes many cycles.
- With basic pipelining, it is still not efficient.
- Our solution: **superpipelining execute stage**

Utilize Data Parallelism

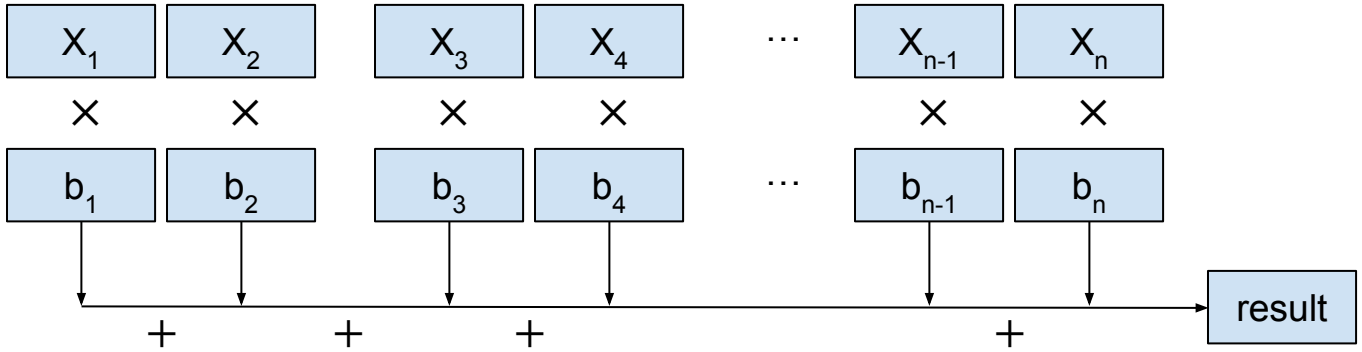
- Private Information Retrieval:
  - Remote server hosts a database of n entries



  - Client sends a query: “retrieve the k-th entry”

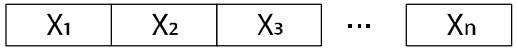


  - Computation on Server:

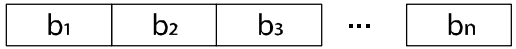


Utilize Data Parallelism

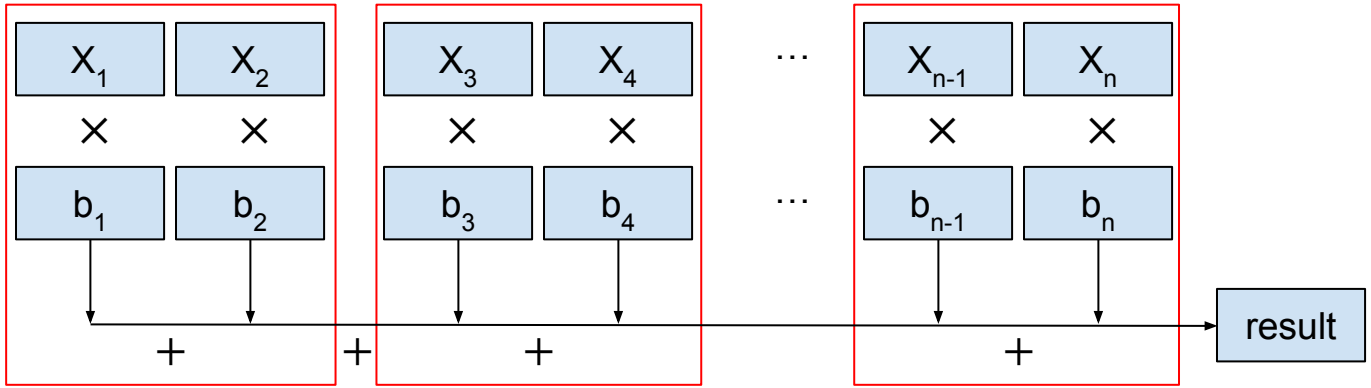
- Private Information Retrieval:
  - Remote server hosts a database of n entries



  - Client sends a query: “retrieve the k-th entry”

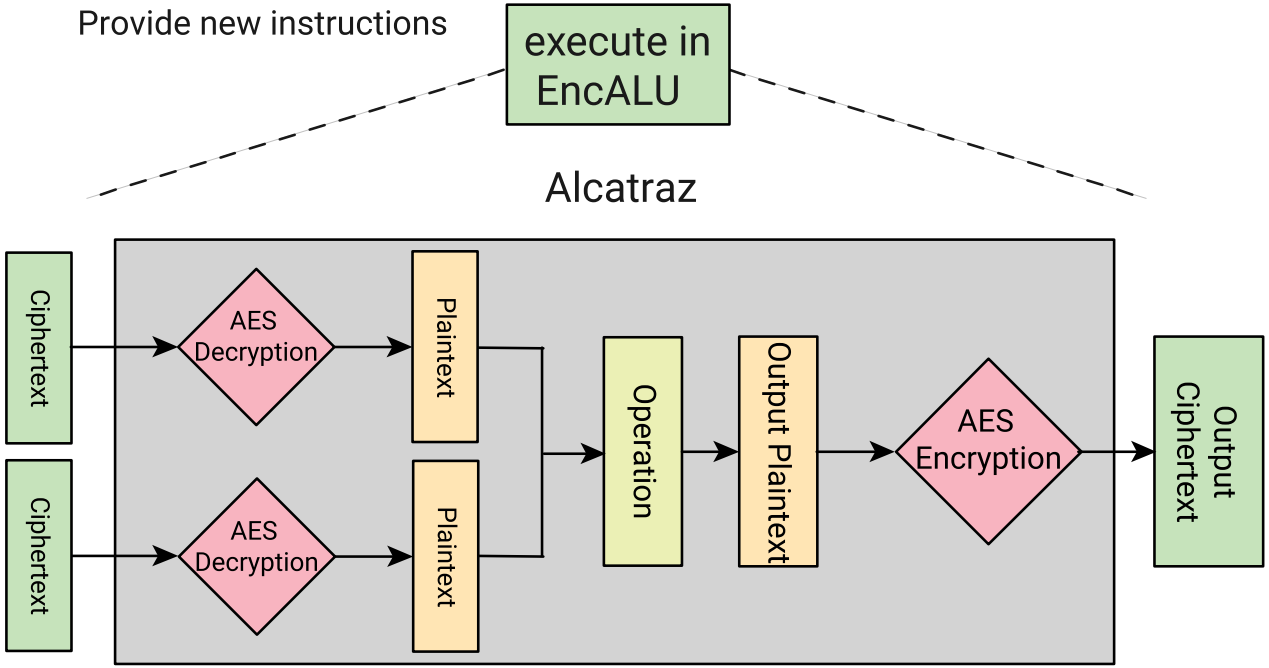


  - Computation on Server:



Utilize Data Parallelism via SIMD

- We take advantage of data parallelism to process multiple encrypted data blocks simultaneously in one instruction (“Single Instruction Multiple Data”)



Performance Comparison: Single Query PIR

- Compared with one of the SOTA methods (SpiralPIR), Alcatraz is estimated to be 7-21 times faster.
- Alcatraz's query size is 16 Bytes while that of Spiral is 8-15 MB.

	Database Size (entries)	Entry Size	Computation Time (s)		Speedup
			SpiralPIR	Alcatraz (Estimated)	
Dataset 1	2^{20}	256B	0.85	**0.040**	21.3×
Dataset 2	2^{18}	30KB	8.99	**1.174**	7.66×
Dataset 3	2^{14}	100KB	2.38	**0.245**	9.71×

S. J. Menon and D. J. Wu, "SPIRAL: Fast, High-Rate Single-Server PIR via FHE Composition," 2022 IEEE Symposium on Security and Privacy (SP)

Acknowledgements

My Mentors

Dr, Jules Drean Dr. Sacha Servan-Schreiber

Author of Knox (PRIMES Alum)

Dr. Anish Athalye

Prof. Srini Devadas, Dr. Slava Gerovitch, and MIT PRIMES for making this possible!

References

- *Athalye et al. “Verifying Hardware Security Modules with Information-Preserving Refinement” (OSDI 2022)*

- S. J. Menon and D. J. Wu, "SPIRAL: Fast, High-Rate Single-Server PIR via FHE Composition," 2022 IEEE Symposium on Security and Privacy (SP), San Francisco, CA, USA, 2022, pp. 930-947, doi: 10.1109/SP46214.2022.9833700.

- Biernacki, et. al. “Sequestered Encryption: A Hardware Technique for Comprehensive Data Privacy”, 2022

Thank you!