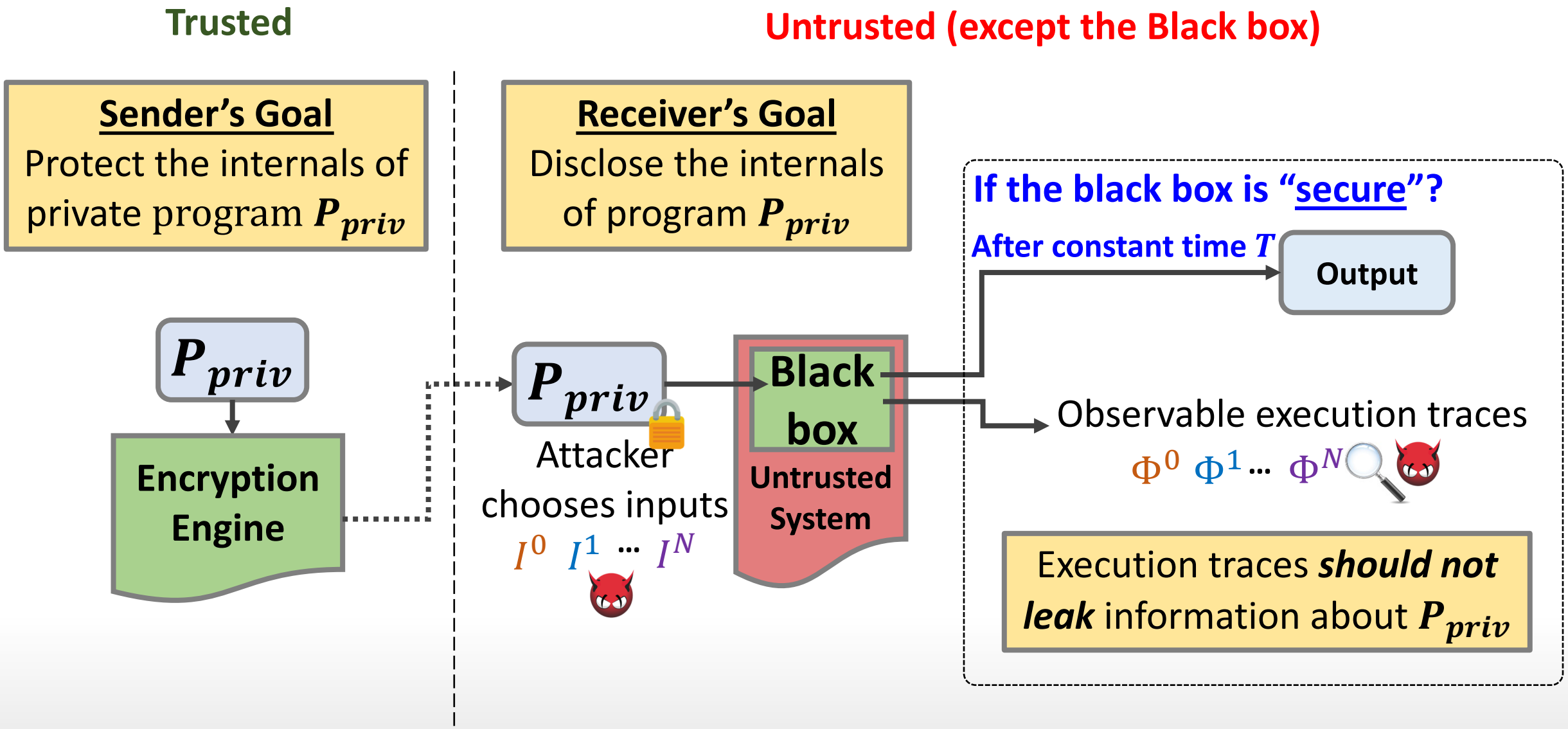
OBFUSCURO: A Commodity Obfuscation Engine for Intel SGX

Adil Ahmad*, Byunggill Joe*, Yuan Xiao

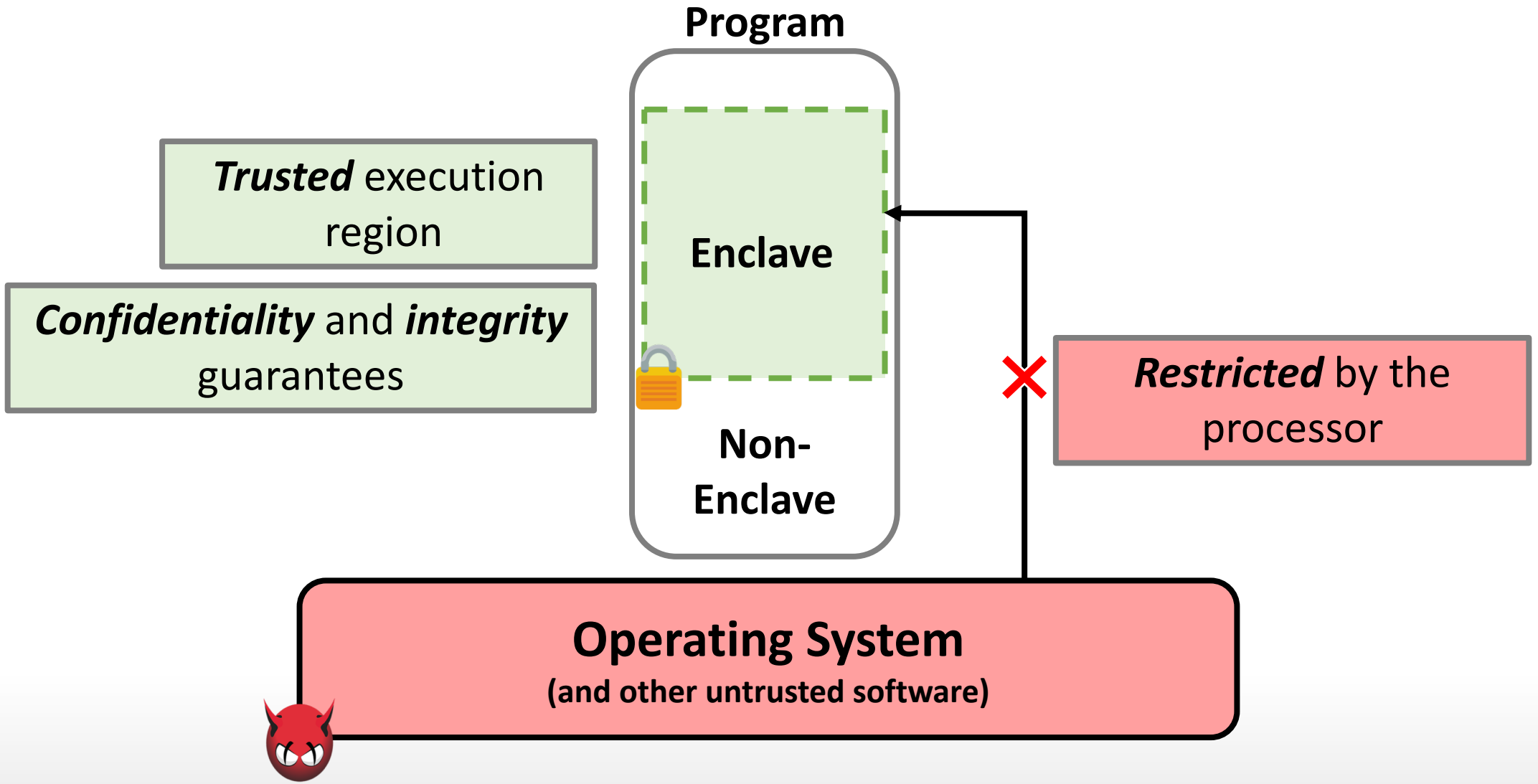
Yinqian Zhang, Insik Shin, Byoungyoung Lee

(* denotes equal contribution)

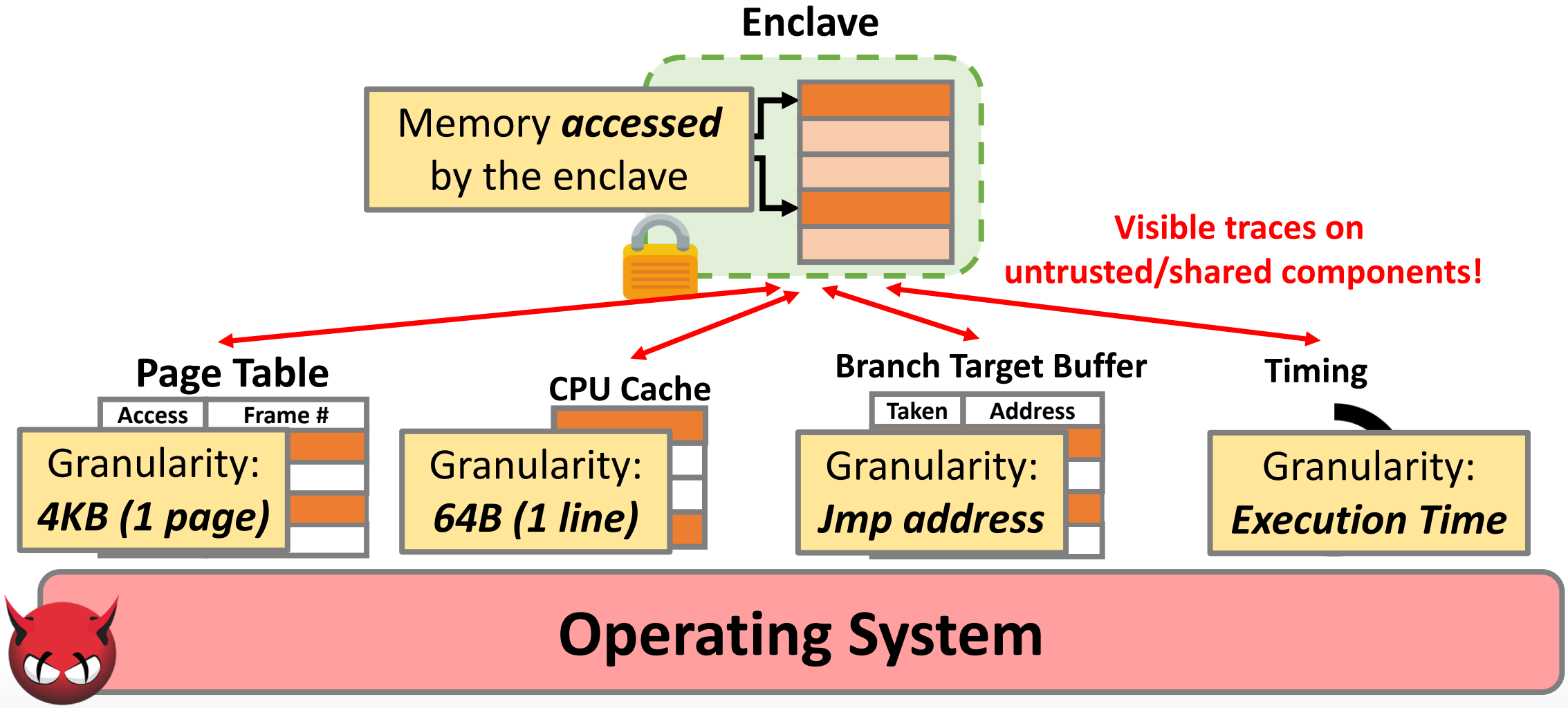
Program Obfuscation



Wait, isn't that what Intel SGX does?



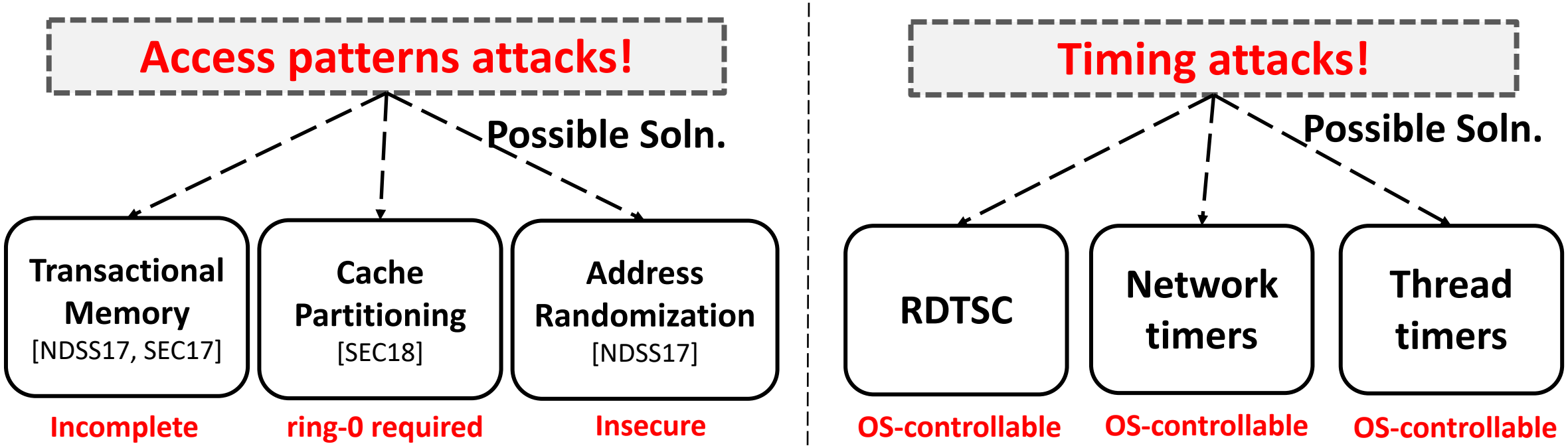
Intel SGX is not perfect!



***Paging*, Branch-prediction** and **Cache attacks!**

[S&P14, SEC17, ASPLOS18, DIMVA17, WOOT17]

Learning from existing solutions!



Lesson #1

Ring-3 enclaves cannot hide access patterns through side-channels!

Lesson #2

Unreliable timers for SGX enclaves!

Our approach

- **Indistinguishable enclave program(s)**
  - A code block executed **N times** on C-Pad, and data block accessed from D-Pad
  - C-Pad and D-Pad are one cache-line (64B) in size!

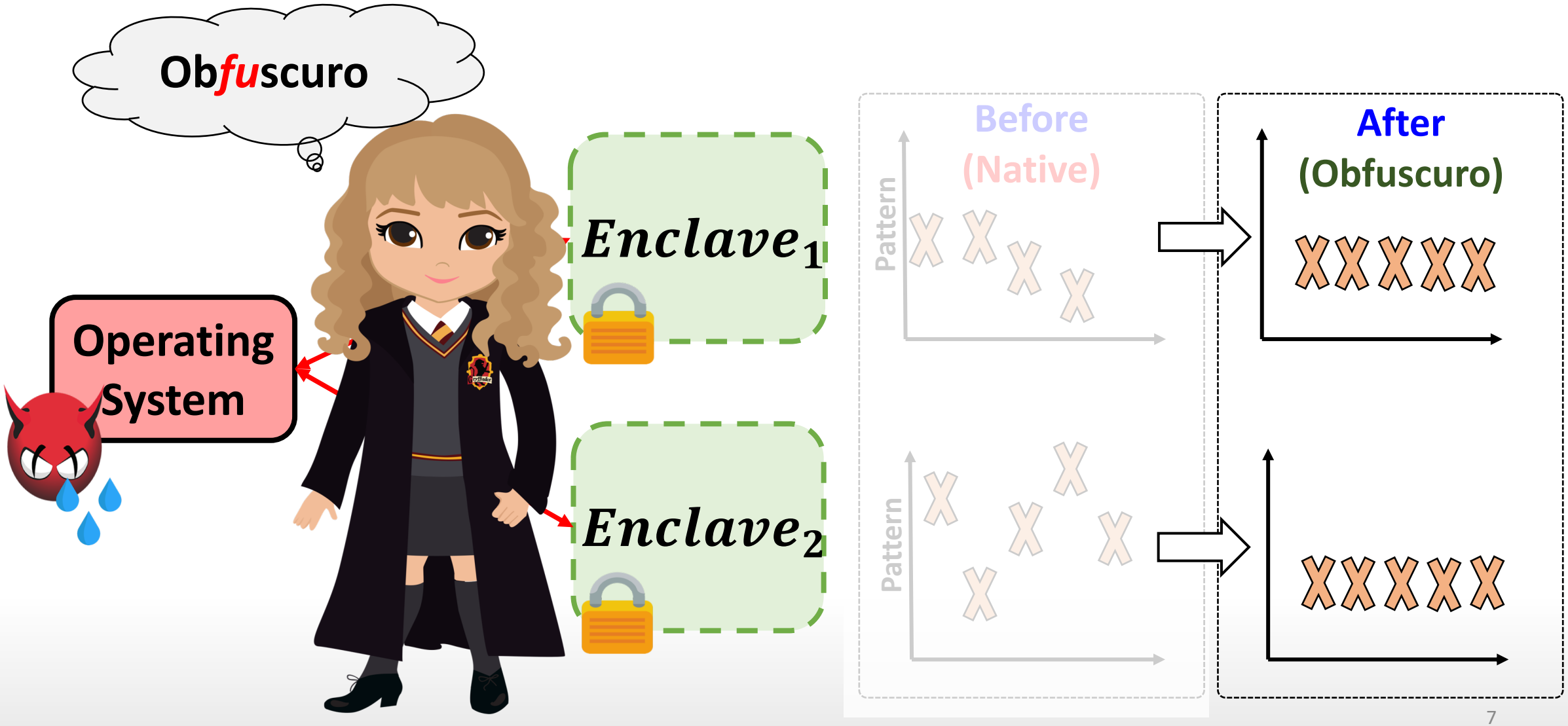
Instead of ***trying to hide*** traces,
all enclaves should leak ***the same*** traces!

Cache Attack: Same cache-lines

Branch Attack: Same branch

Timing Attack: Same time to execute N code blocks

Let Hermione explain!

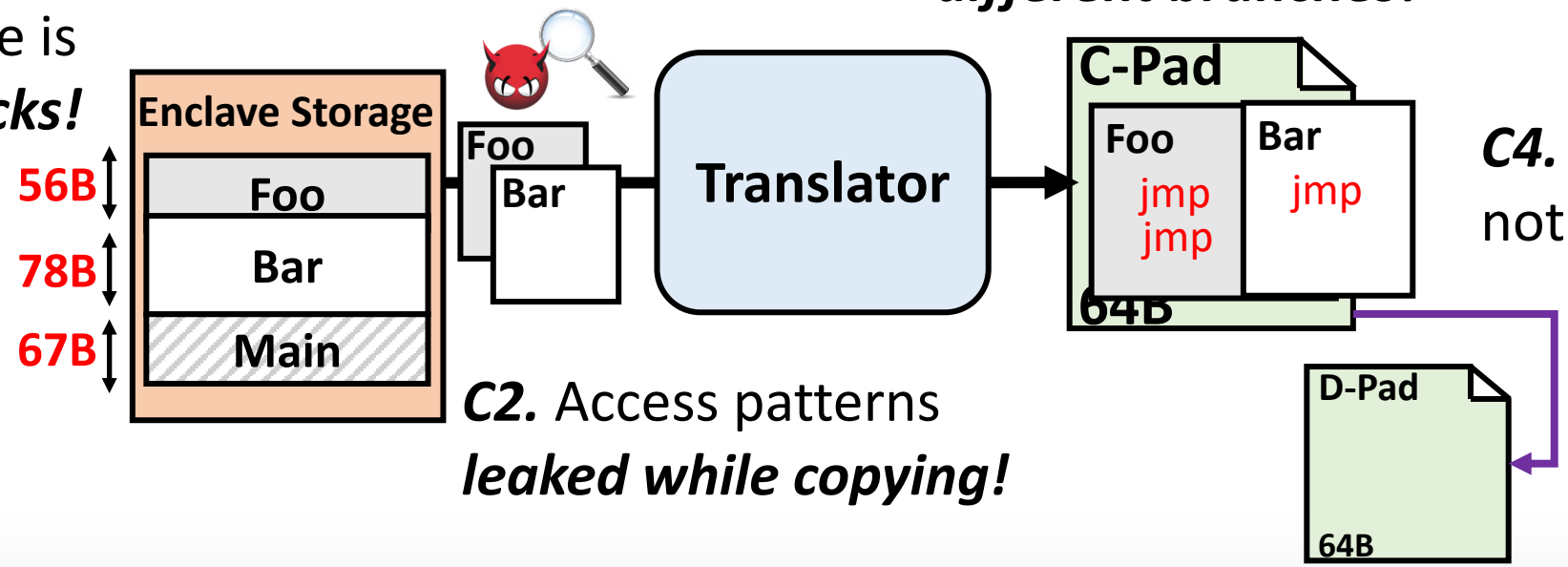


Cool, what's the challenge?

- **Naïve solution**
  - Use a software-translator to copy all code and data onto C/D-Pad

C1. Native code is ***not in 64B blocks!***

C3. Code can have ***different branches!***

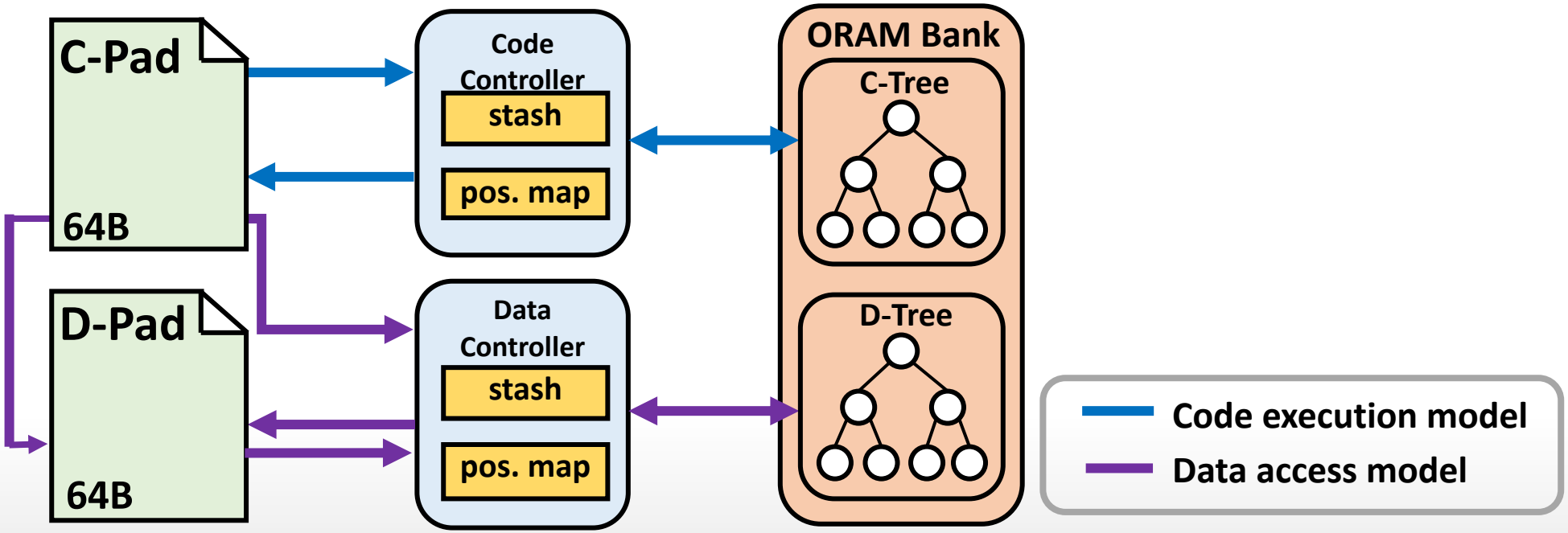


C2. Access patterns ***leaked while copying!***

C4. Timing issues not even discussed!

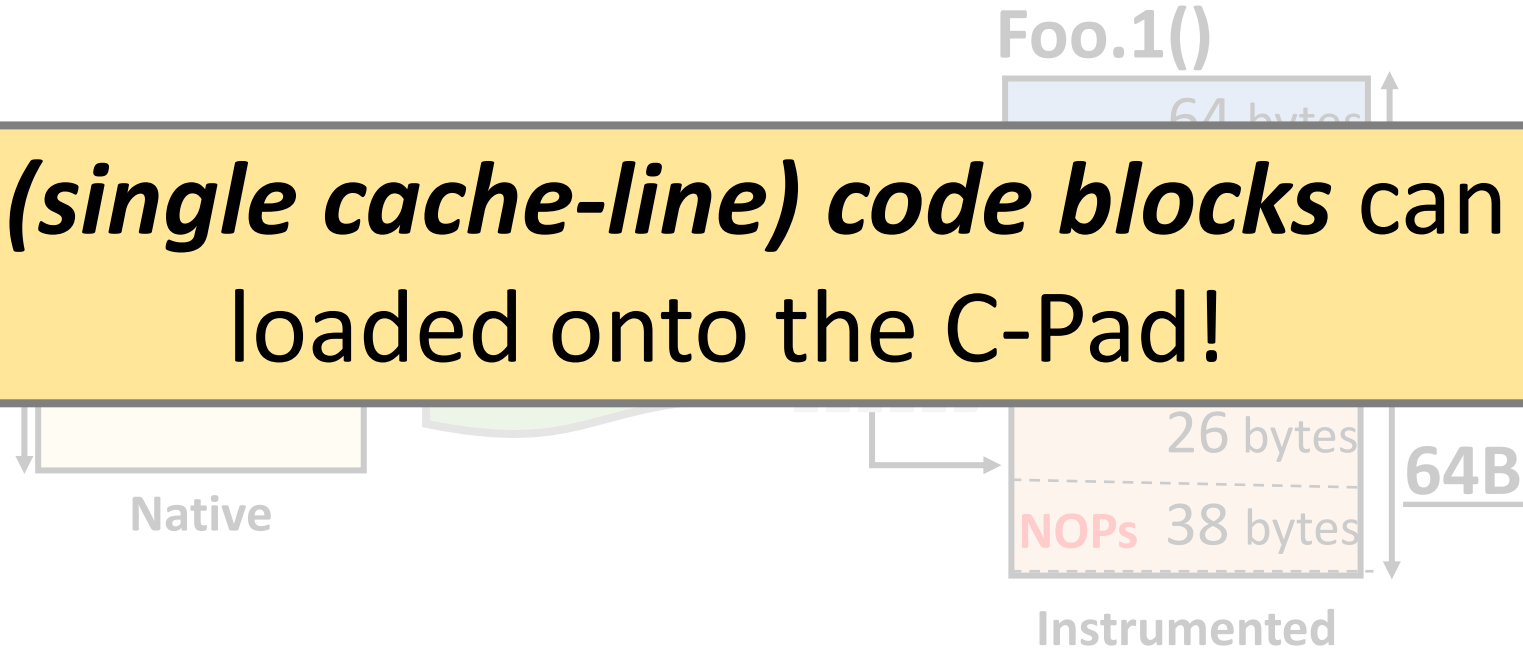
Obfuscuro

- **Program obfuscation on Intel SGX**
  - All programs should exhibit same patterns irrespective of logic/input.
  - Adapted from Harry Potter spell "Obscuro" **(translation :> Darkness)**



C1. Enforce code blocks of identical sizes

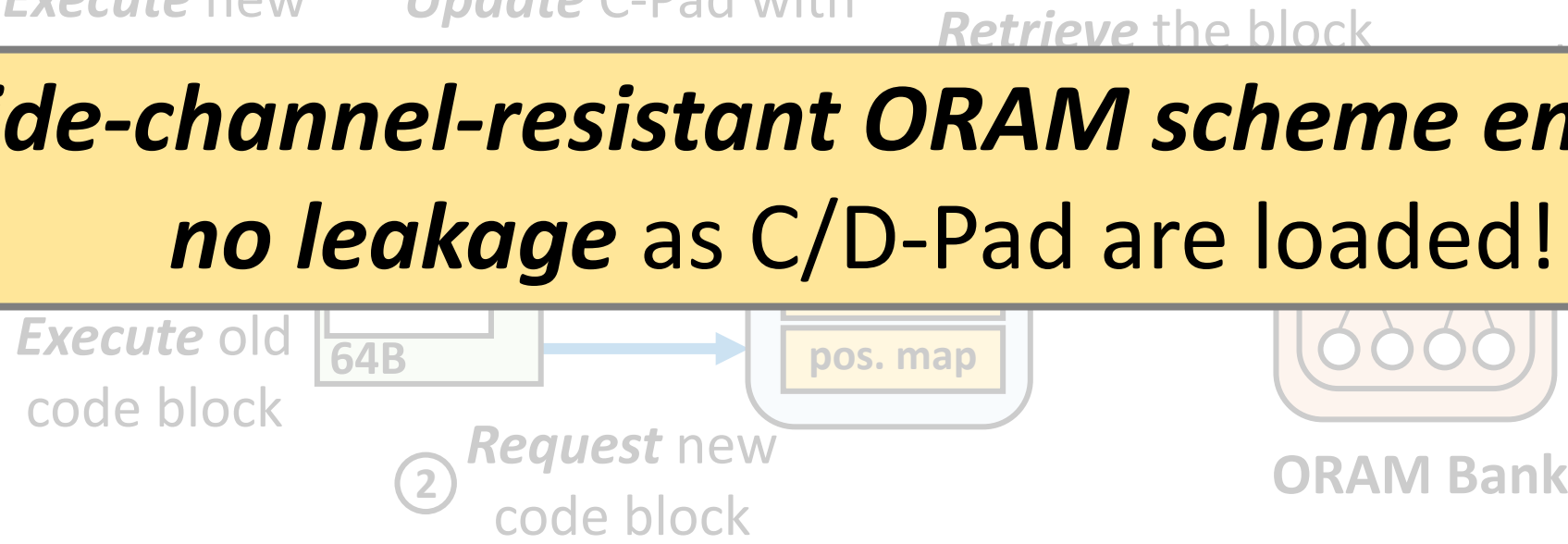
- Break code blocks into 64 bytes and pad using nop



64B (single cache-line) code blocks can be loaded onto the C-Pad!

C2. Securely loading C/D-Pad

- Fetch code and data using Oblivious RAM (ORAM)
  - The code and data is fetched onto C-Pad and D-Pad resp.

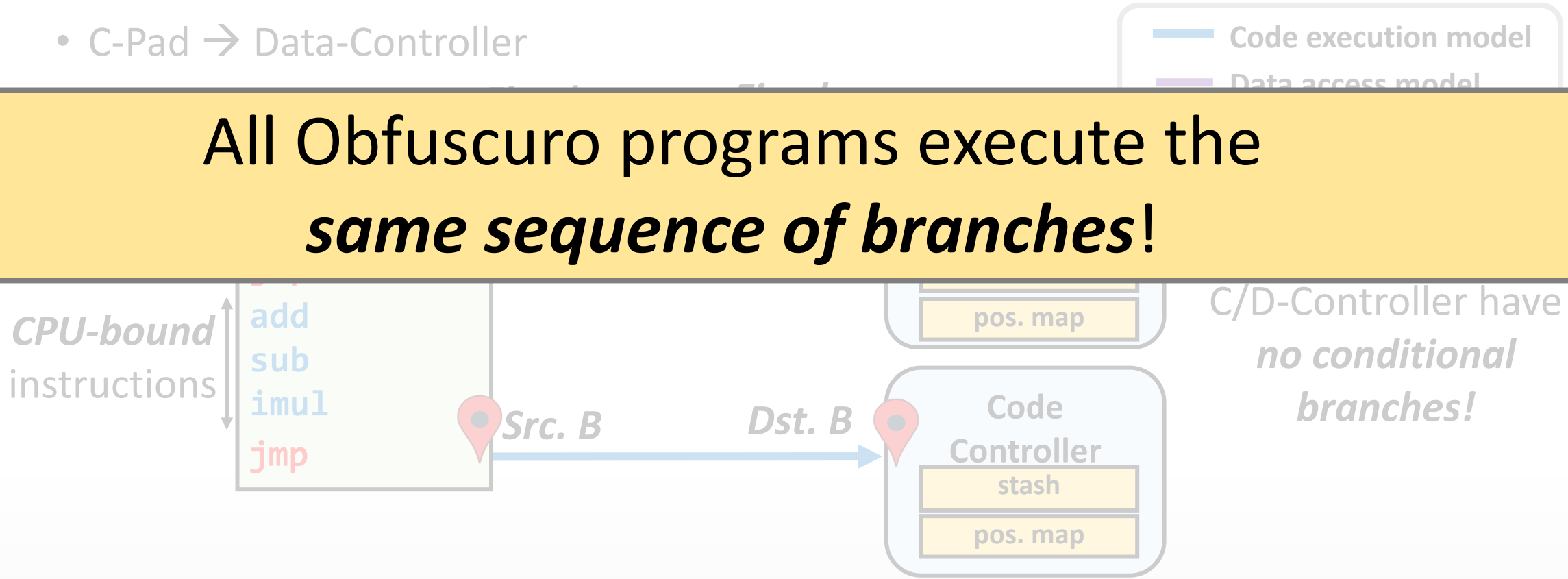


Side-channel-resistant ORAM scheme ensures no leakage as C/D-Pad are loaded!

C3. Align branches to/from C-Pad

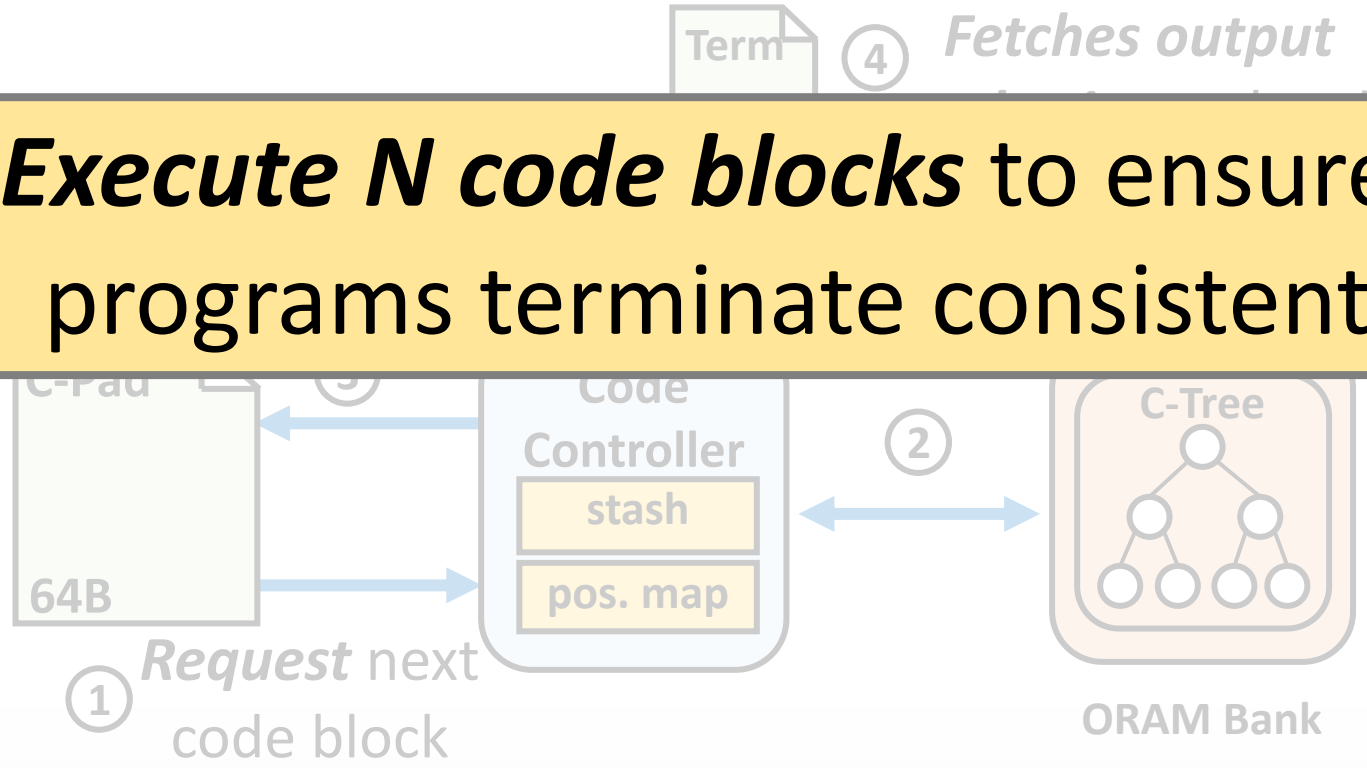
- Each instrumented code block has two branches to fixed locations
  - C-Pad → Code-Controller
  - C-Pad → Data-Controller

All Obfuscuro programs execute the ***same sequence of branches***!



C4. Ensuring execution time consistency

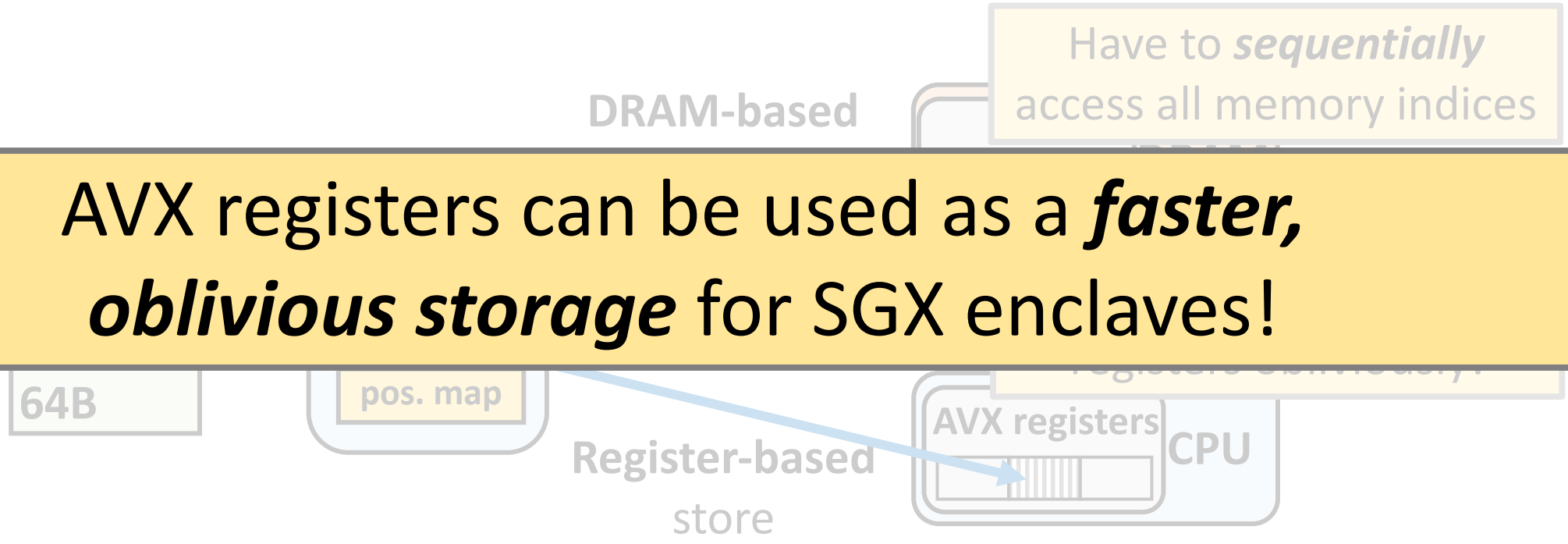
- The program executes fixed number of code blocks



Execute N code blocks to ensure all programs terminate consistently!

Faster memory store for enclaves

- Use AVX registers as store instead of ”Oblivious” store



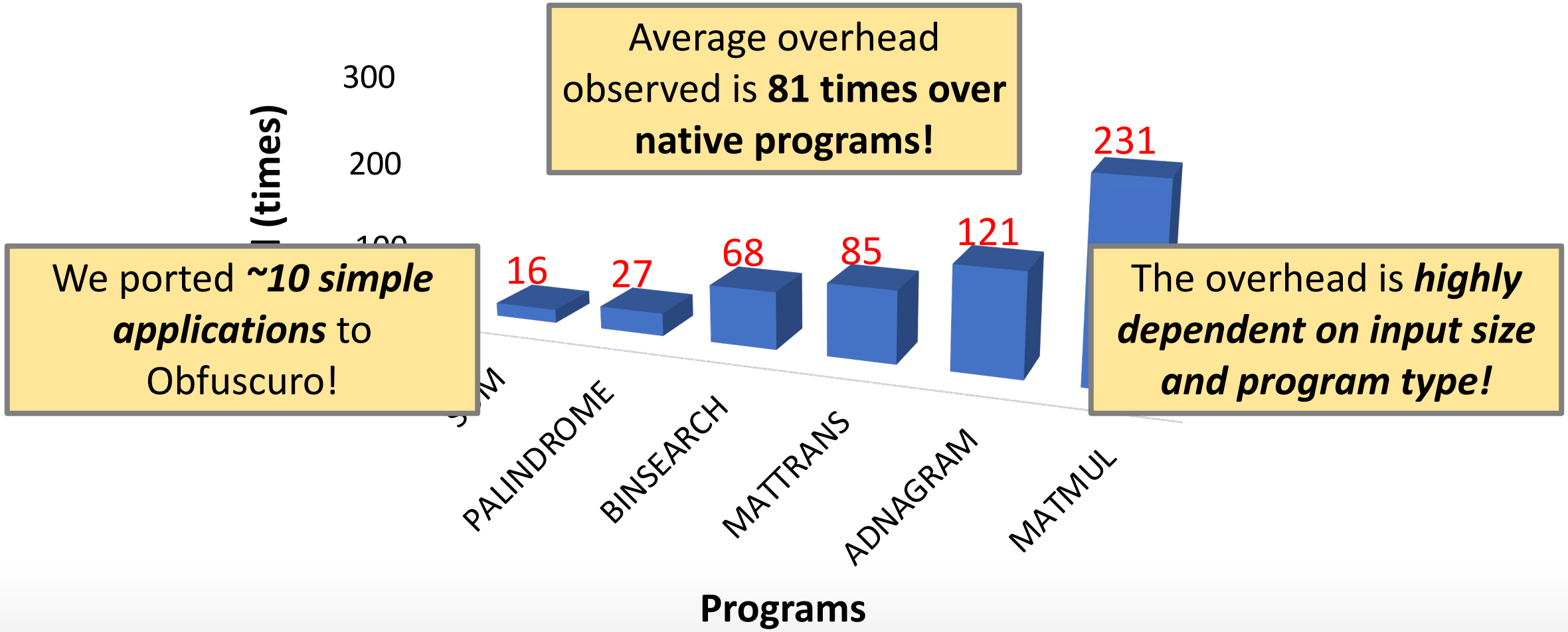
AVX registers can be used as a ***faster, oblivious storage*** for SGX enclaves!

Implementation

- **LLVM compiler suite (3117 LoC)**
  - Breaks all code into similar blocks **(C1)**
  - Instrument and align all control and data-flow instructions **(C3)**
- **Runtime library (2179 LoC)**
  - Initializes ORAM trees and performs secure ORAM operations **(C2)**
  - Terminate program and fetch output **(C4)**
- **Intel SGX SDK (25 LoC)**
  - Assign memory regions for C/D-Pad **(support)**

Performance Evaluation

Average overhead observed is **81 times over native programs!**



We ported ***~10 simple applications*** to Obfuscuro!

The overhead is ***highly dependent on input size and program type!***

Ending Remarks!

1. Program obfuscation is a ***remarkable dream*** to achieve
2. Various software/hardware limitations ***hinder*** the realization of program obfuscation on Intel SGX
3. Existing solutions have a ***limited approach*** towards side-channel mitigation in Intel SGX
4. Obfuscuro is compiler-based scheme which addresses this issue by ensuring all programs leak ***same access patterns***

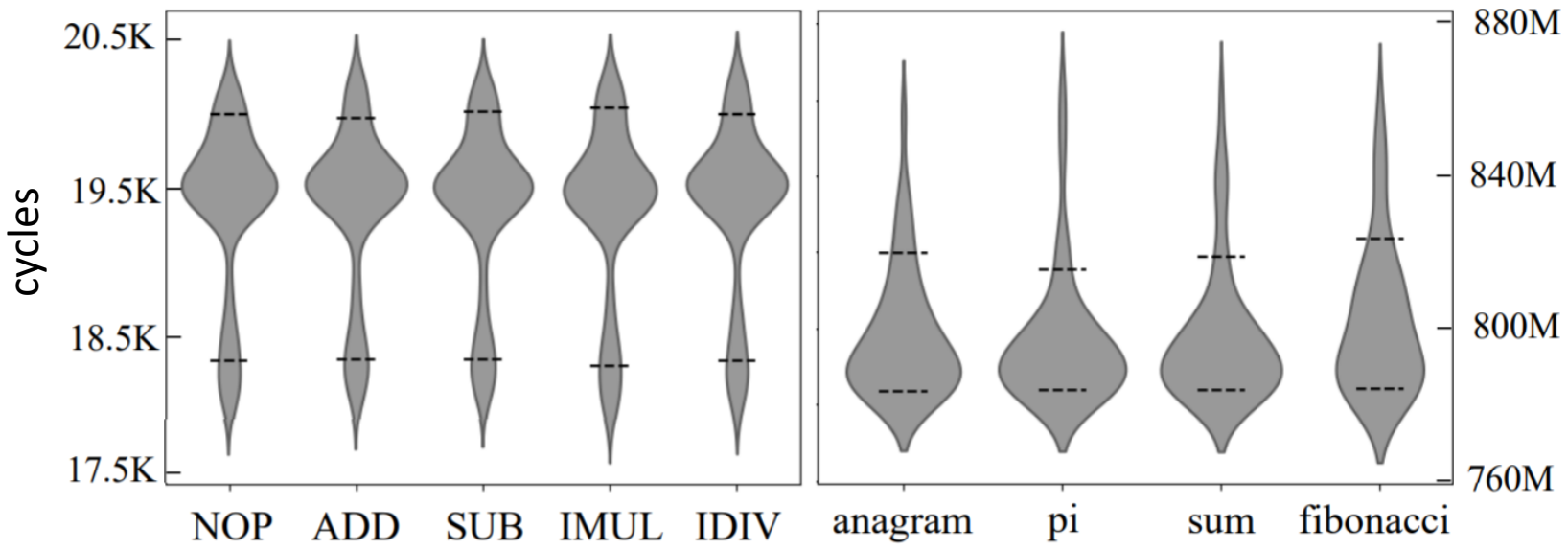
Adil Ahmad

Contact: ahmad37@purdue.edu

감사합니다

(Translation ~ Thanks!) ;)

Execution Time Evaluation



Code block with instructions of each type

General programs

ORAM access time dominates the time of code block execution!