OBFUSCURO: A Commodity Obfuscation Engine for Intel SGX

Adil Ahmad*, Byunggill Joe*, Yuan Xiao

Yinqian Zhang, Insik Shin, Byoungyoung Lee

(* denotes equal contribution)

PURDUE UNIVERSITY

KAIST

THE OHIO STATE UNIVERSITY

SEOUL NATIONAL UNIVERSITY

Program Obfuscation

Program Obfuscation

Trusted

Untrusted (except the Black box)

Sender's Goal
Protect the internals of private program P_{priv}

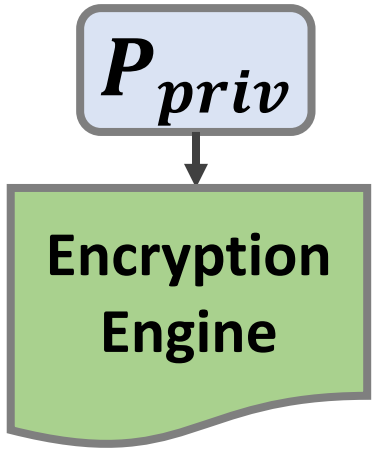
P_{priv}

Program Obfuscation

Trusted

Untrusted (except the Black box)

Sender's Goal
Protect the internals of private program P_{priv}

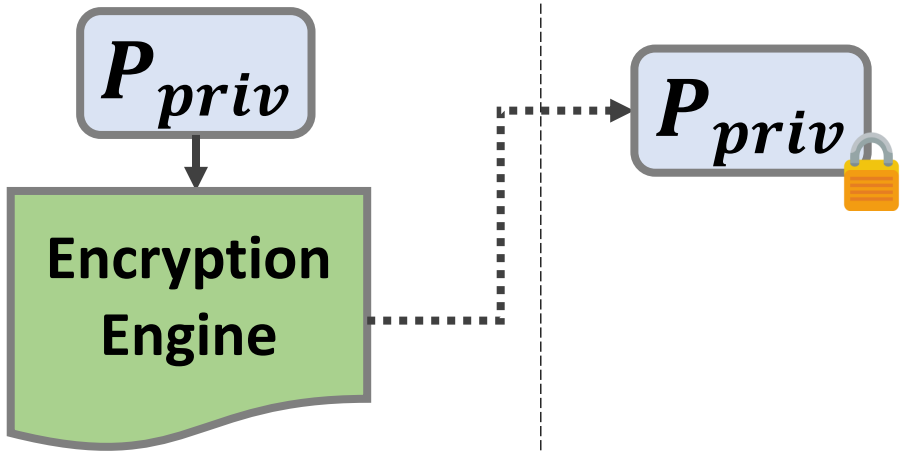


Program Obfuscation

Trusted

Untrusted (except the Black box)

Sender's Goal
Protect the internals of private program P_{priv}

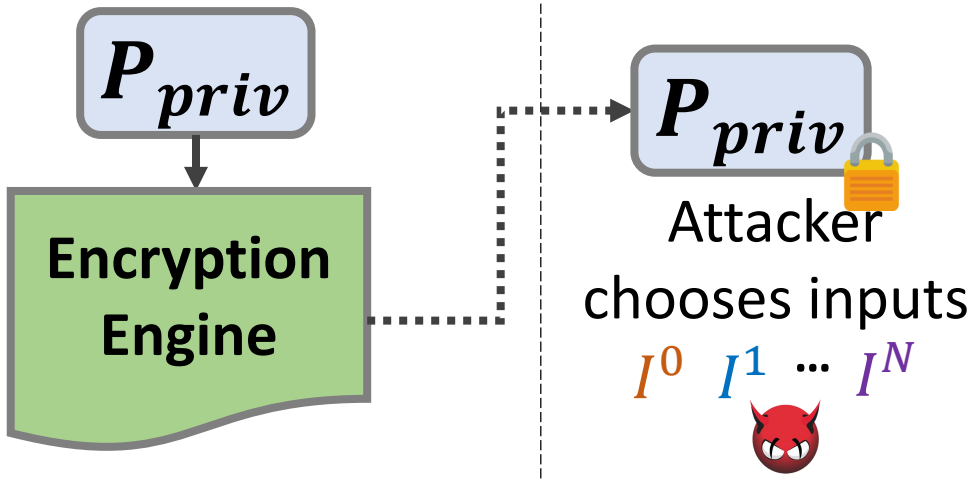


Program Obfuscation

Trusted

Untrusted (except the Black box)

Sender's Goal
Protect the internals of private program P_{priv}

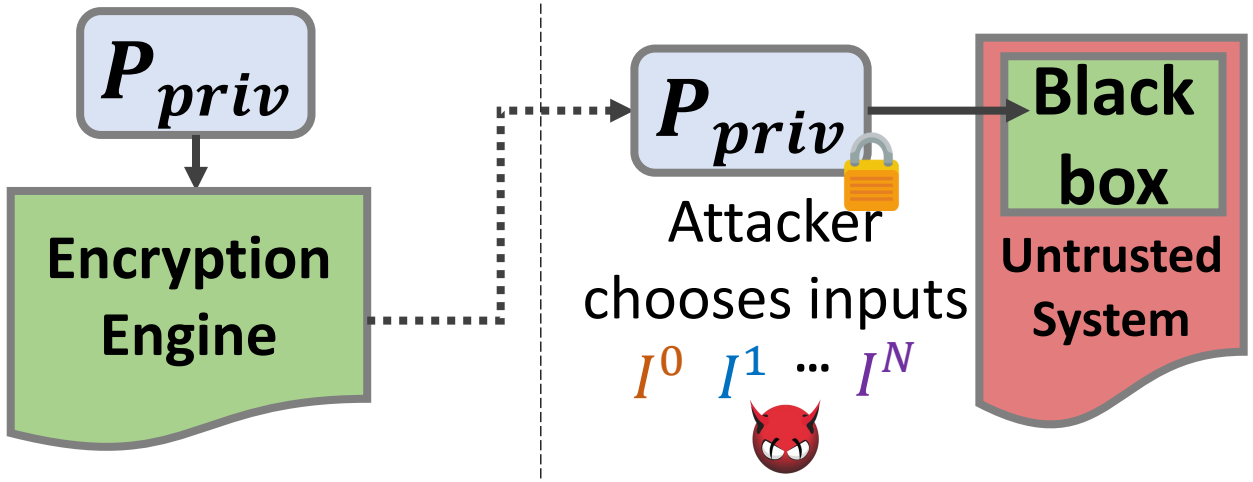


Program Obfuscation

Trusted

Untrusted (except the Black box)

Sender's Goal
Protect the internals of private program P_{priv}



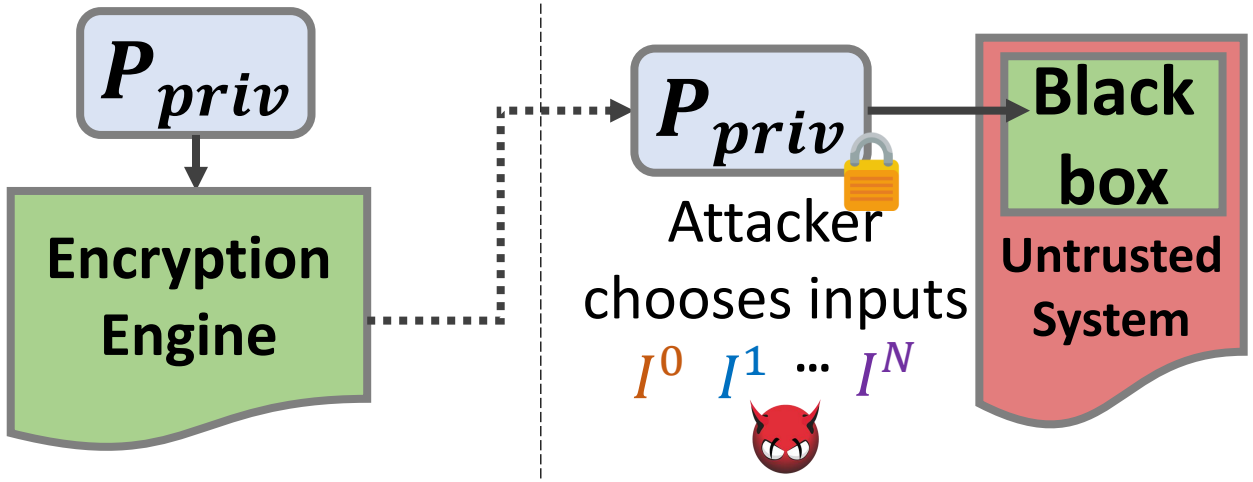
Program Obfuscation

Trusted

Untrusted (except the Black box)

Sender's Goal
Protect the internals of private program P_{priv}

Receiver's Goal
Disclose the internals of program P_{priv}



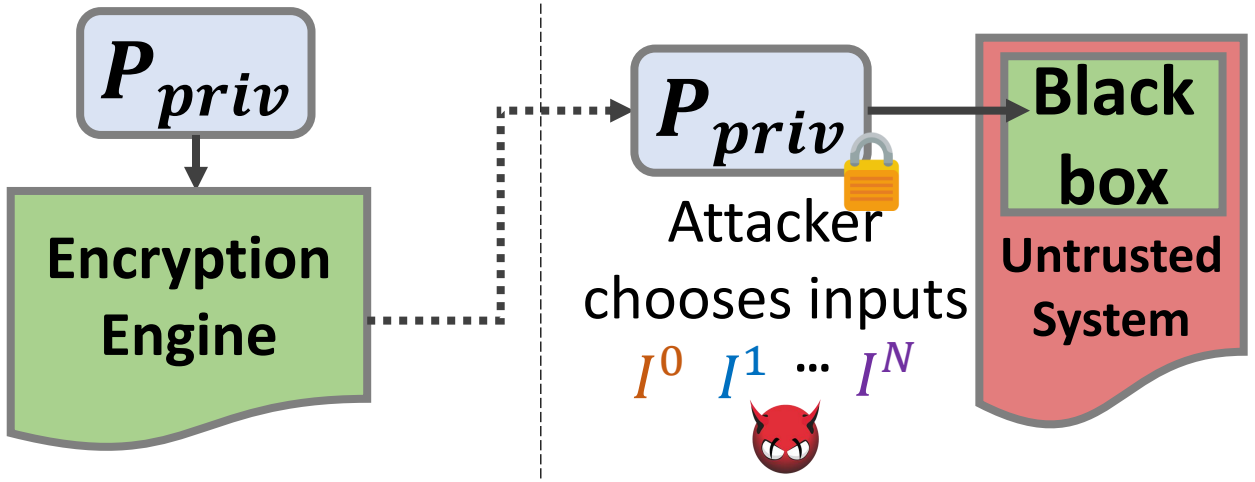
Program Obfuscation

Trusted

Untrusted (except the Black box)

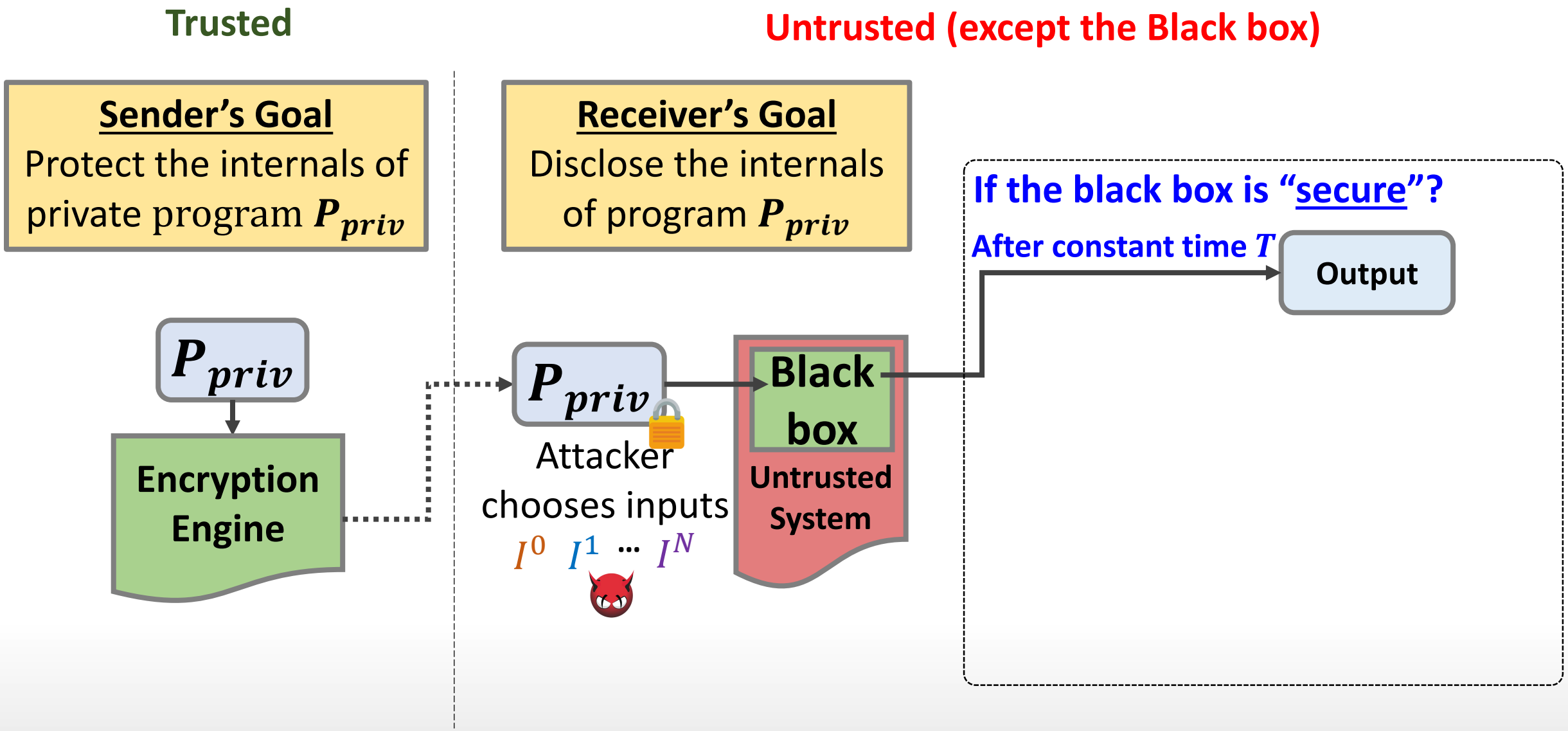
Sender's Goal
Protect the internals of private program P_{priv}

Receiver's Goal
Disclose the internals of program P_{priv}



If the black box is "secure"?

Program Obfuscation



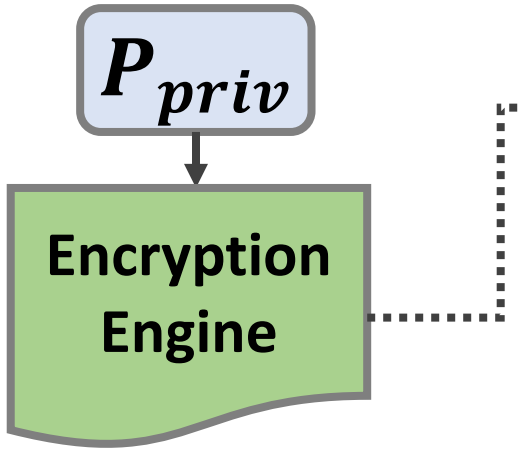
Program Obfuscation

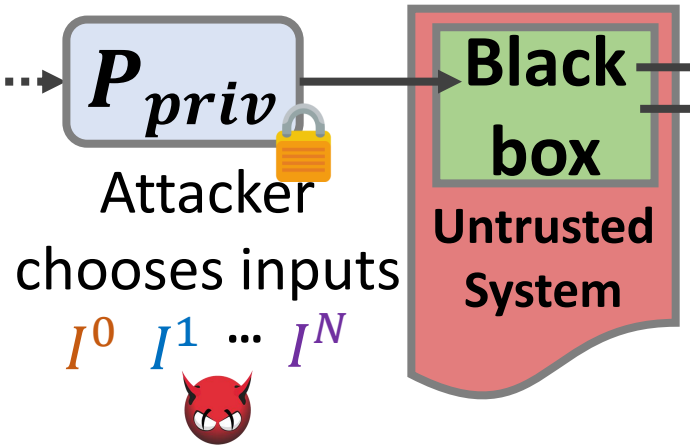
Trusted

Untrusted (except the Black box)

Sender's Goal
Protect the internals of private program P_{priv}

Receiver's Goal
Disclose the internals of program P_{priv}





If the black box is "secure"?

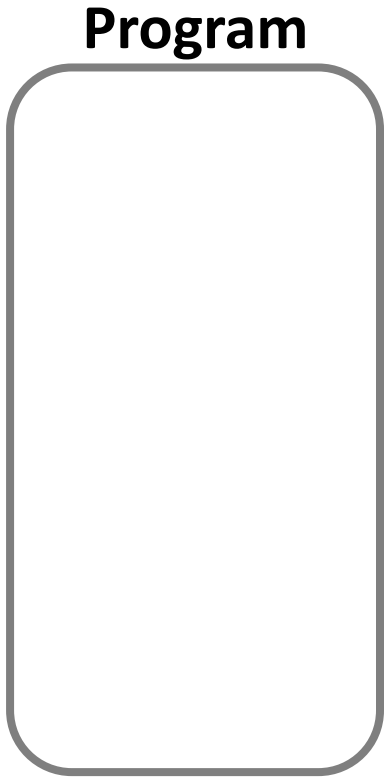
After constant time T → **Output**

Observable execution traces
Φ^0 Φ^1 … Φ^N

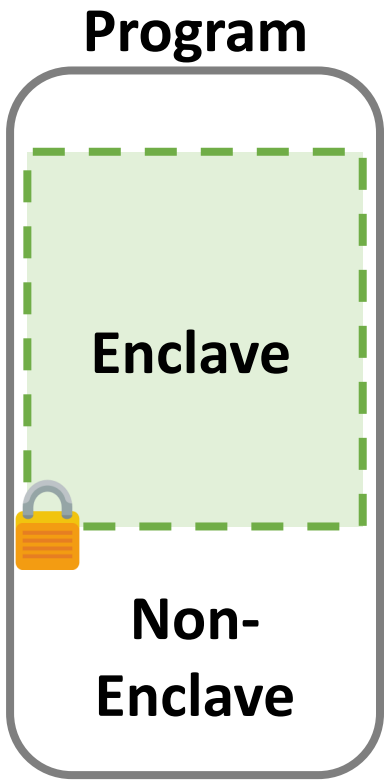
Execution traces ***should not leak*** information about P_{priv}

Wait, isn't that what Intel SGX does?

Wait, isn’t that what Intel SGX does?



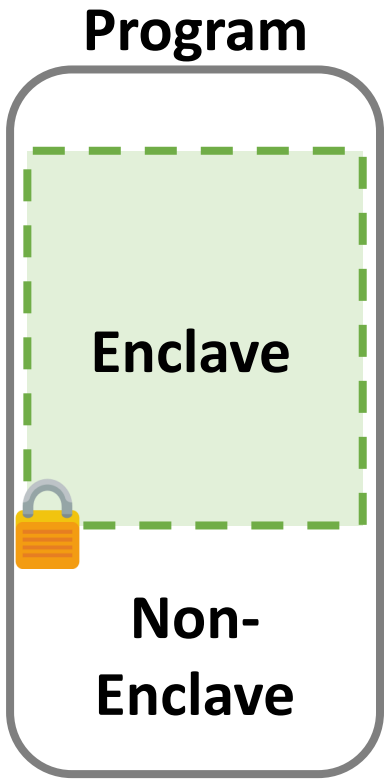
Wait, isn't that what Intel SGX does?



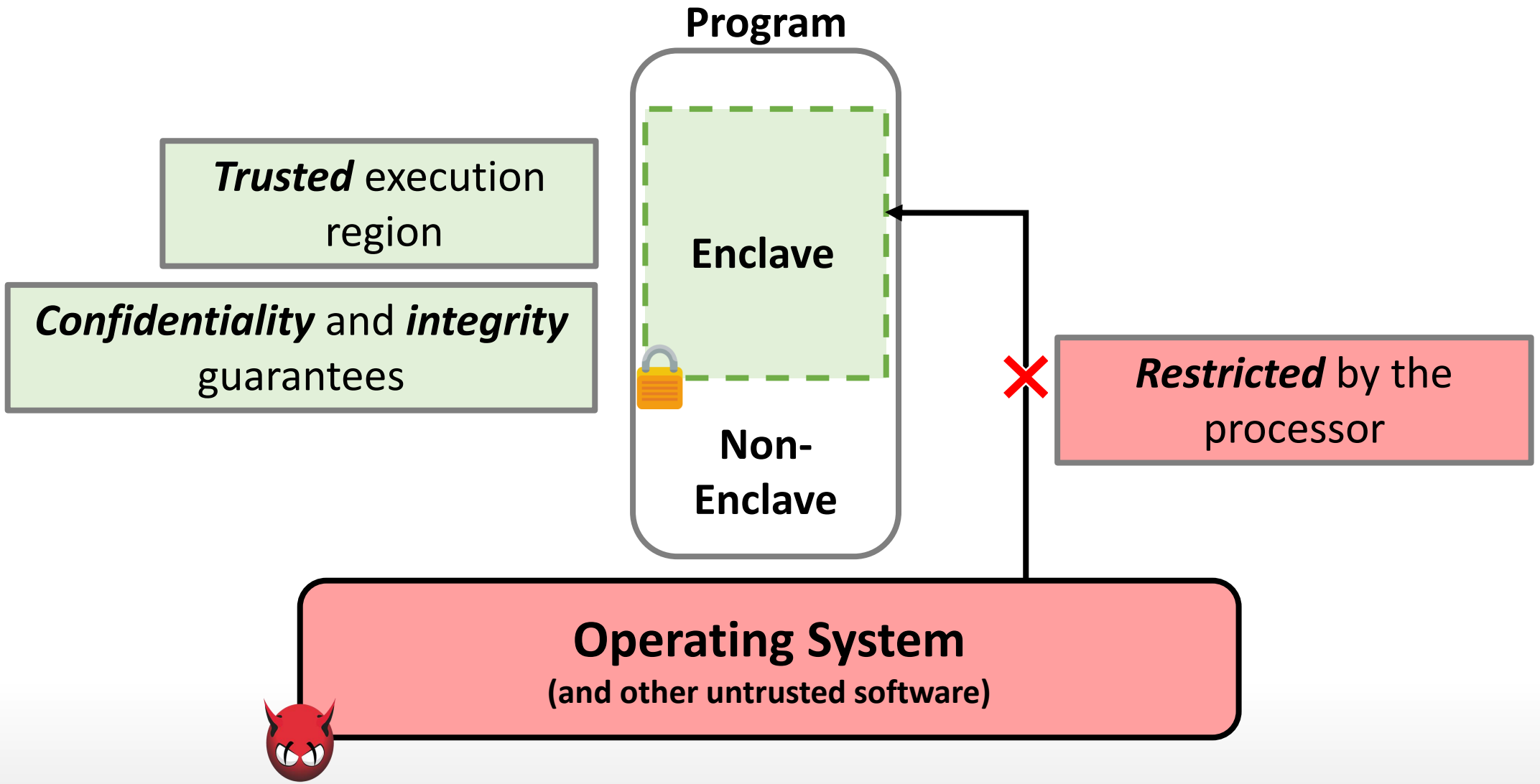
Wait, isn't that what Intel SGX does?

Trusted execution region

Confidentiality and ***integrity*** guarantees

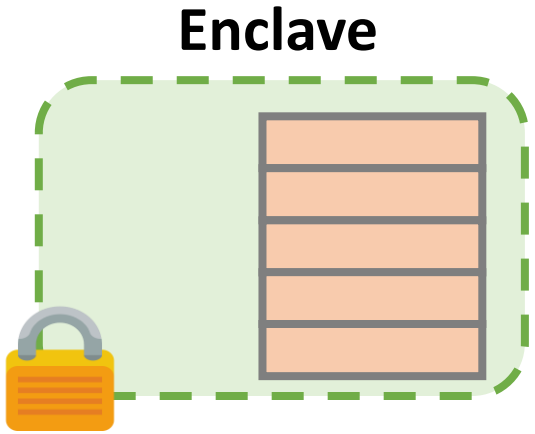


Wait, isn't that what Intel SGX does?

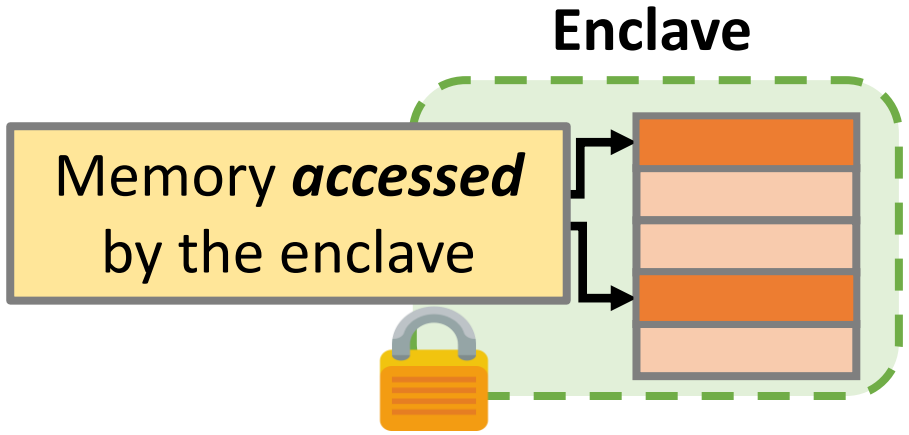


Intel SGX is not perfect!

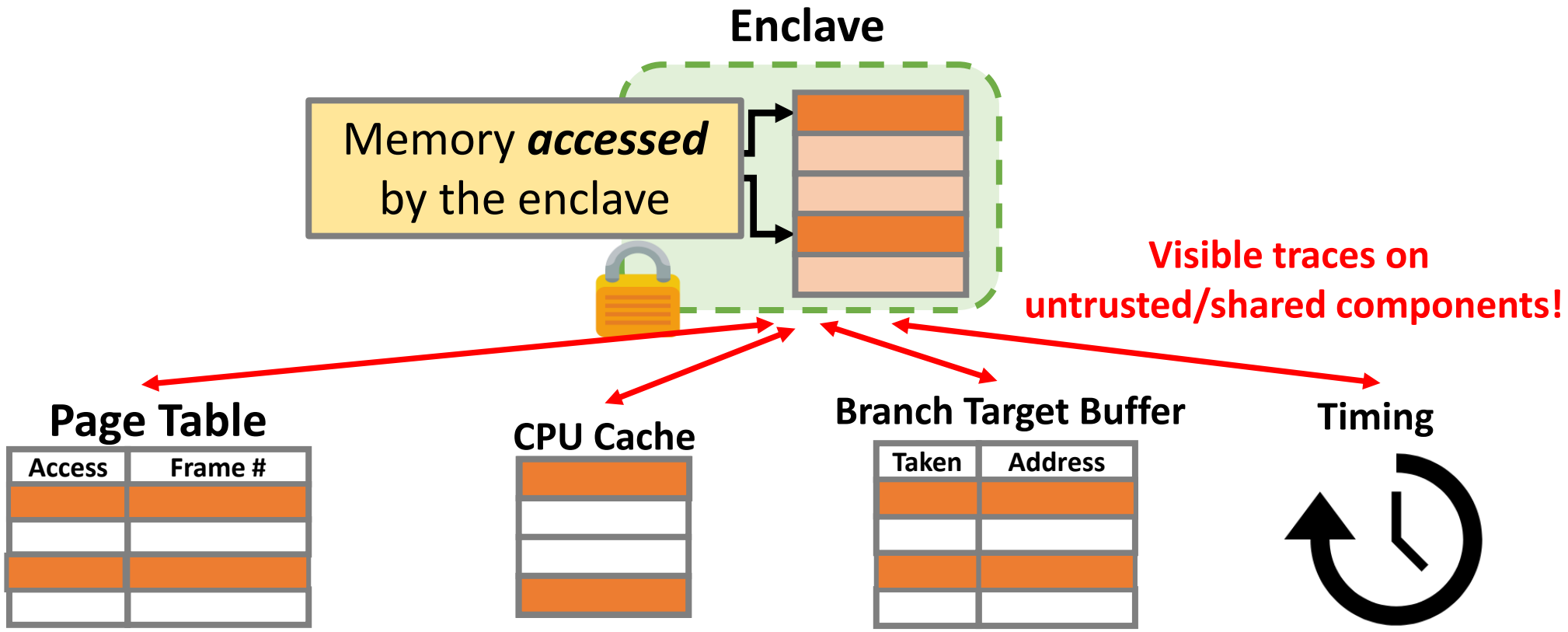
Intel SGX is not perfect!



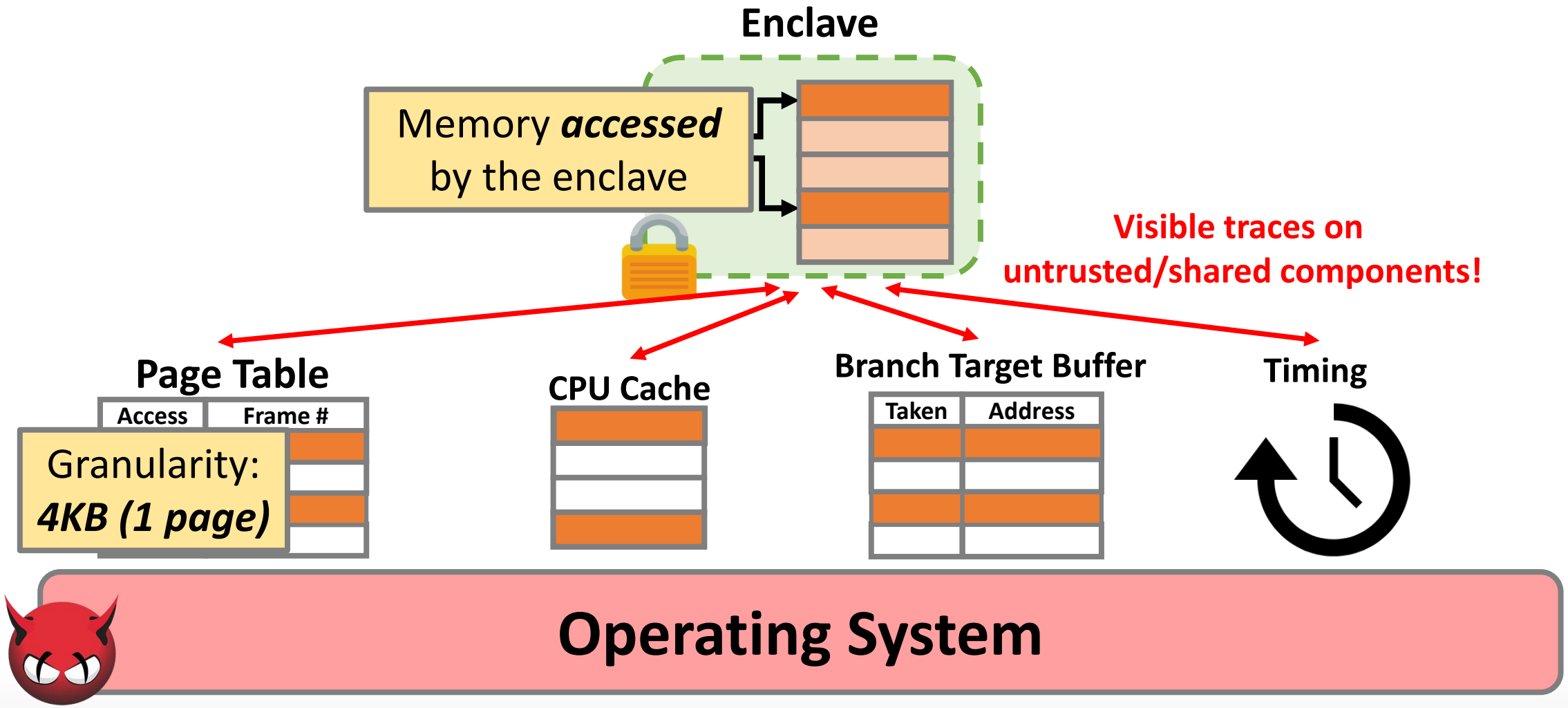
Intel SGX is not perfect!



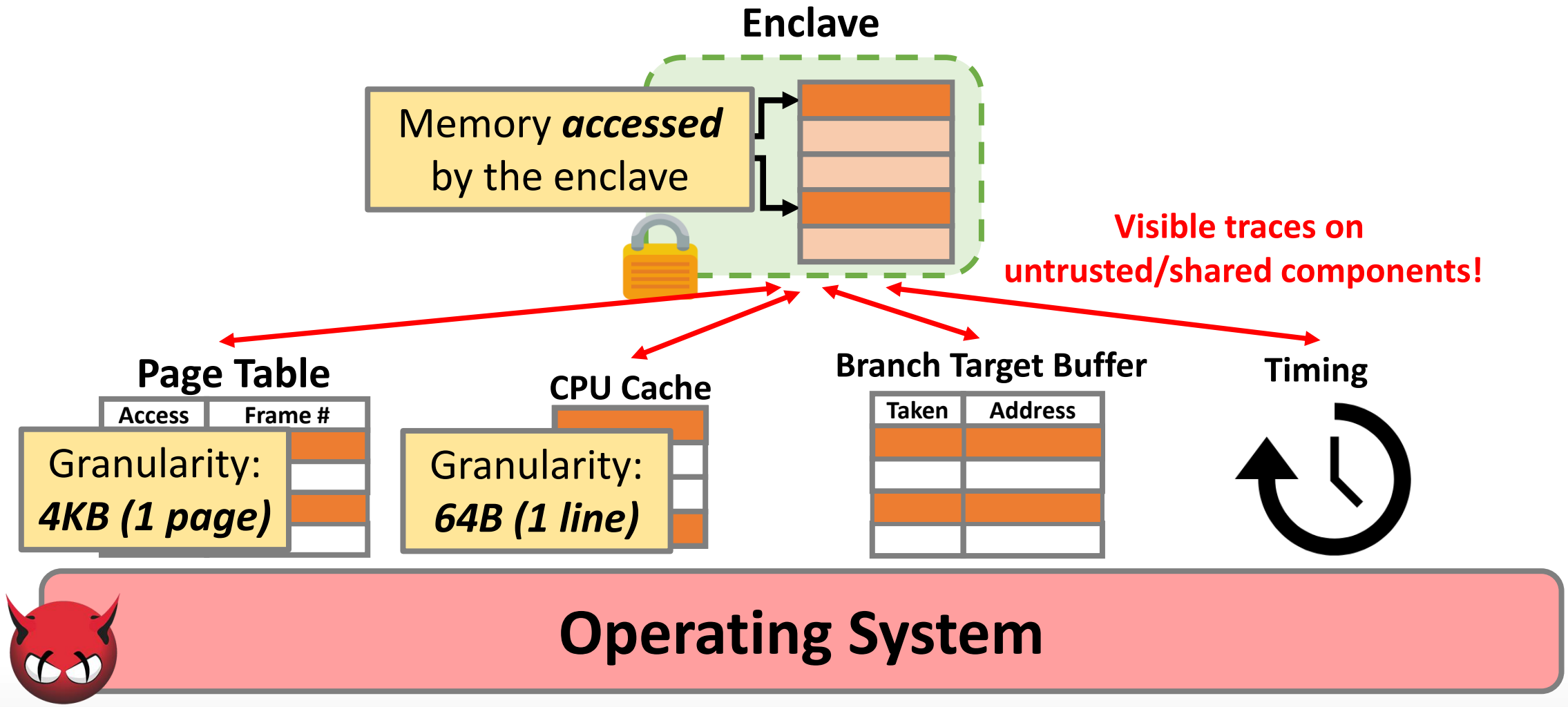
Intel SGX is not perfect!



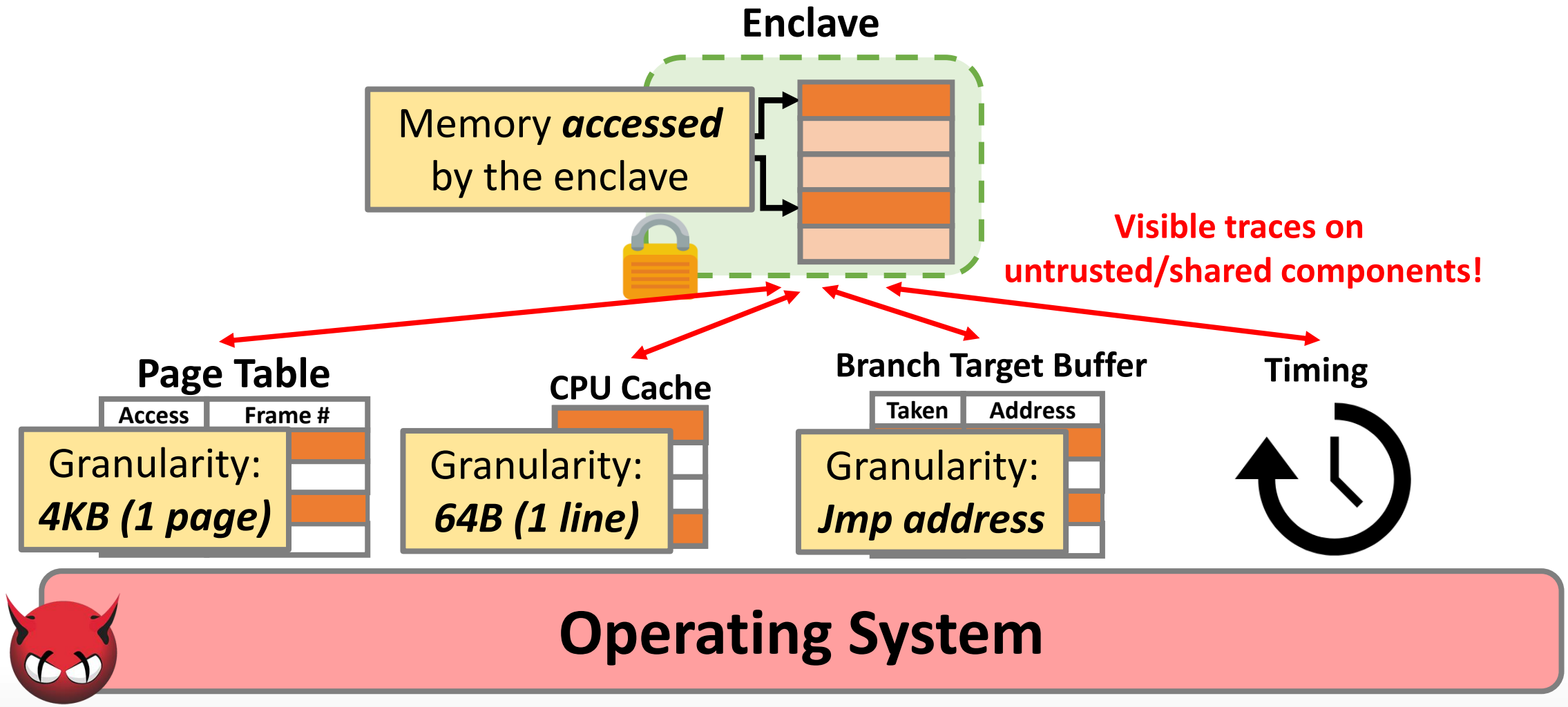
Intel SGX is not perfect!



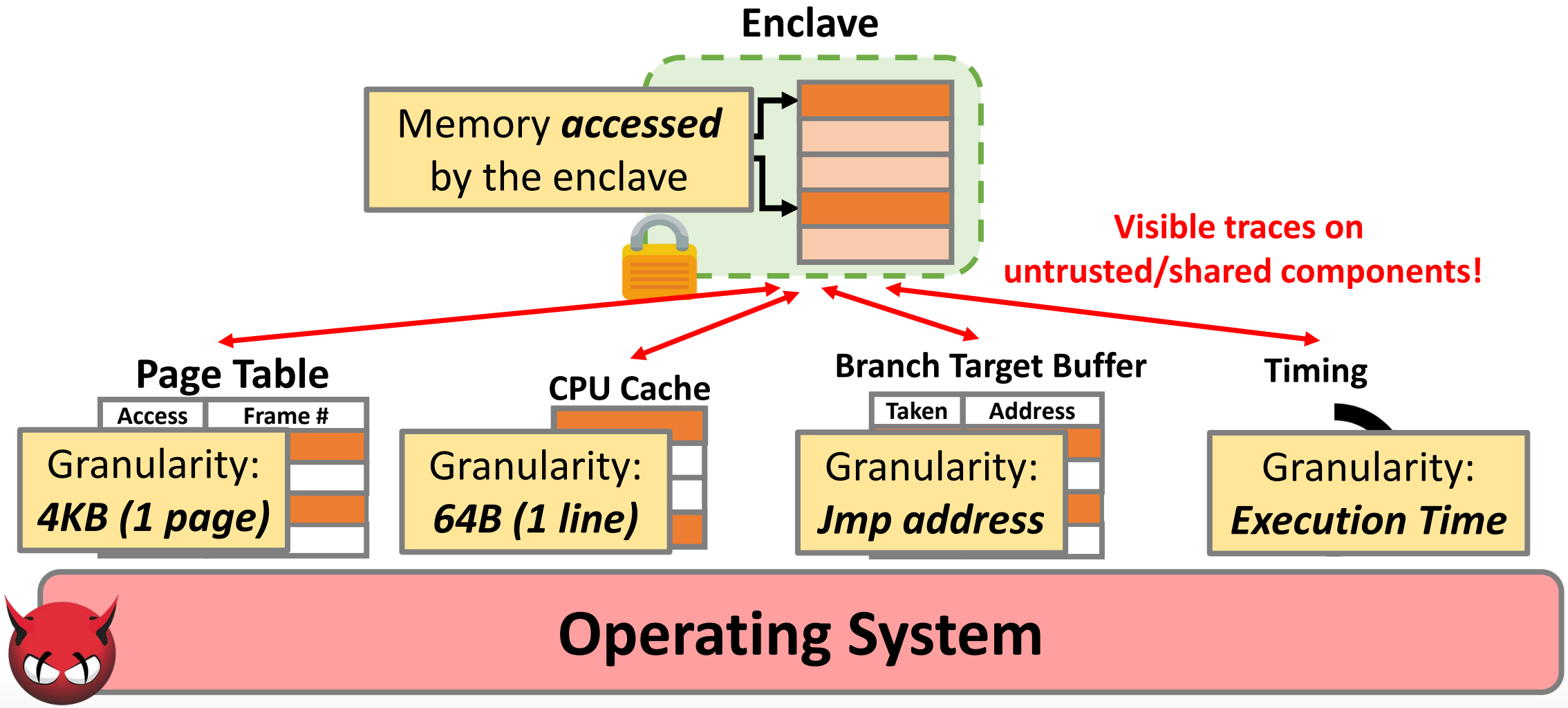
Intel SGX is not perfect!



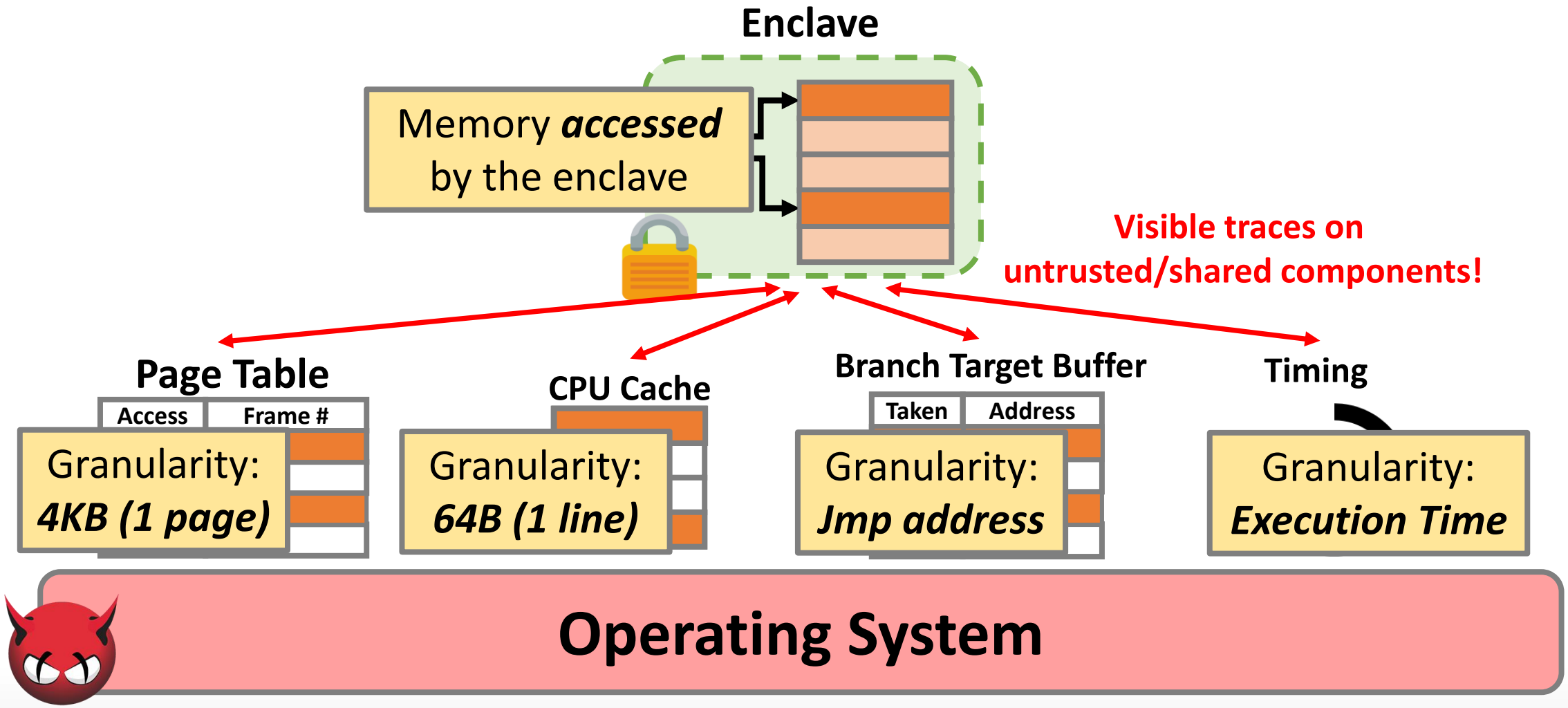
Intel SGX is not perfect!



Intel SGX is not perfect!



Intel SGX is not perfect!



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

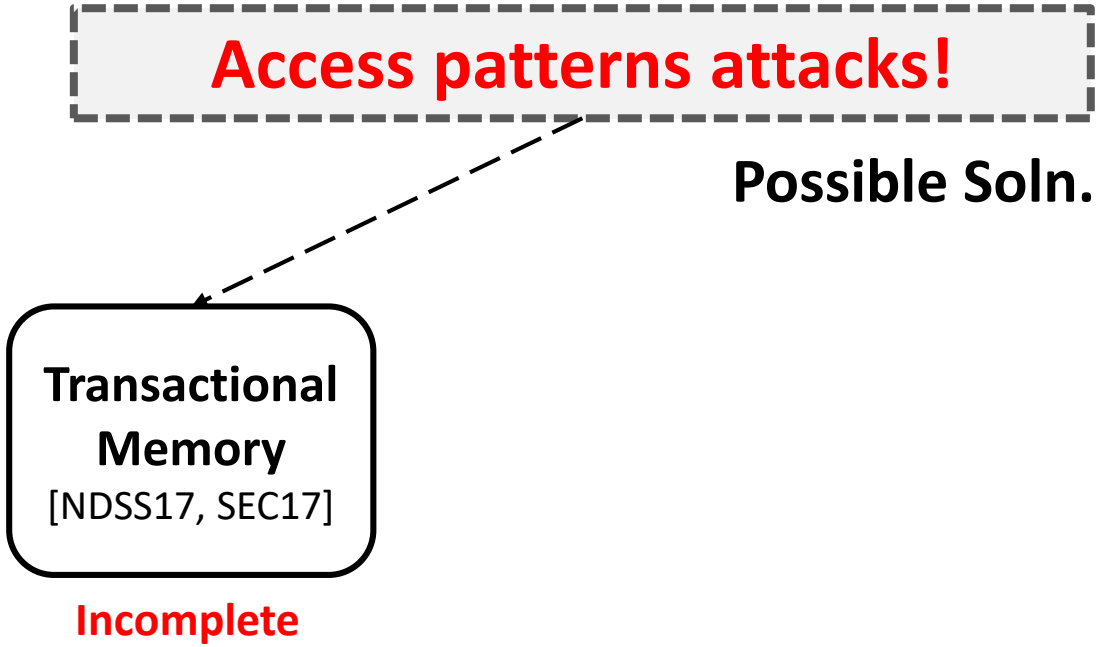
[S&P14, SEC17, ASPLOS18, DIMVA17, WOOT17]

Learning from existing solutions!

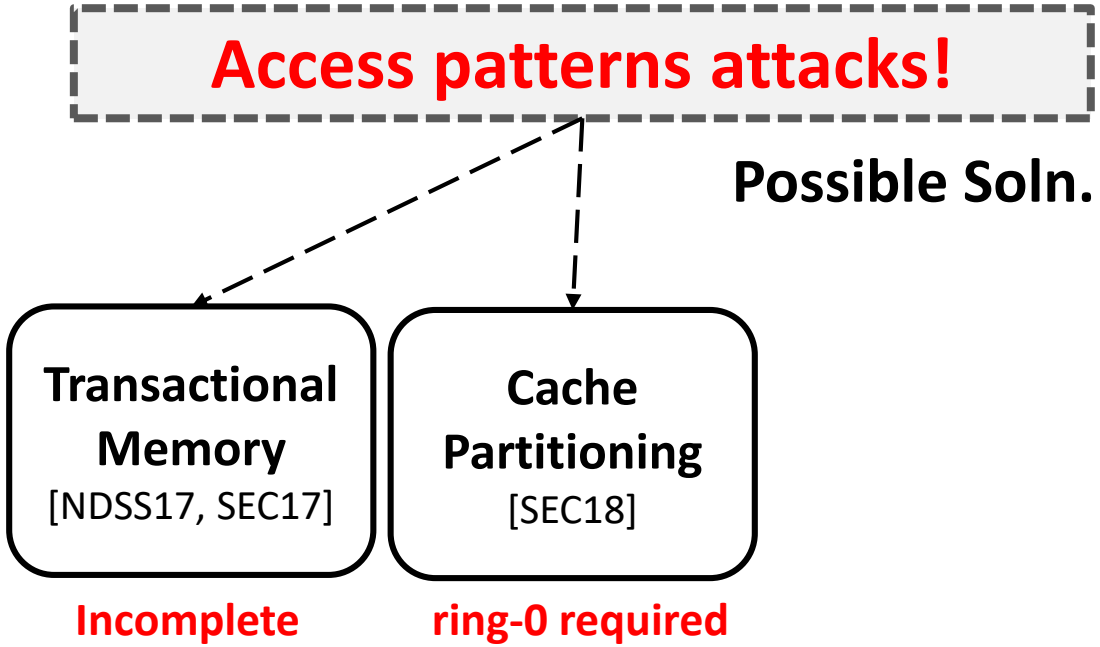
Learning from existing solutions!

Access patterns attacks!

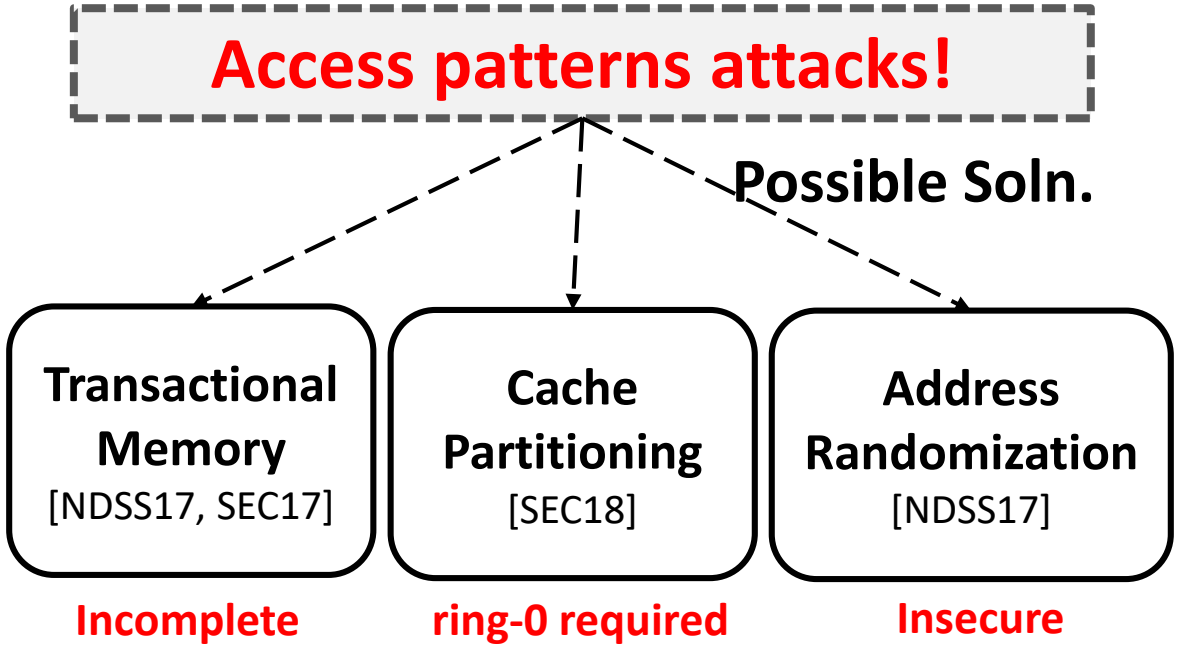
Learning from existing solutions!



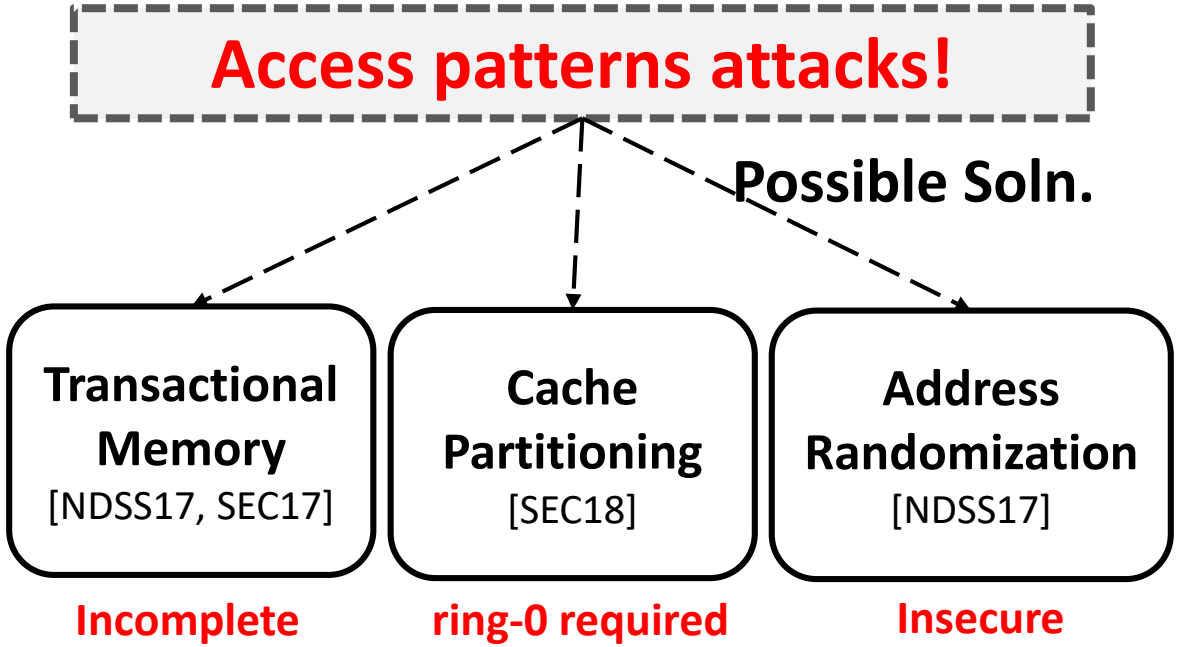
Learning from existing solutions!



Learning from existing solutions!



Learning from existing solutions!

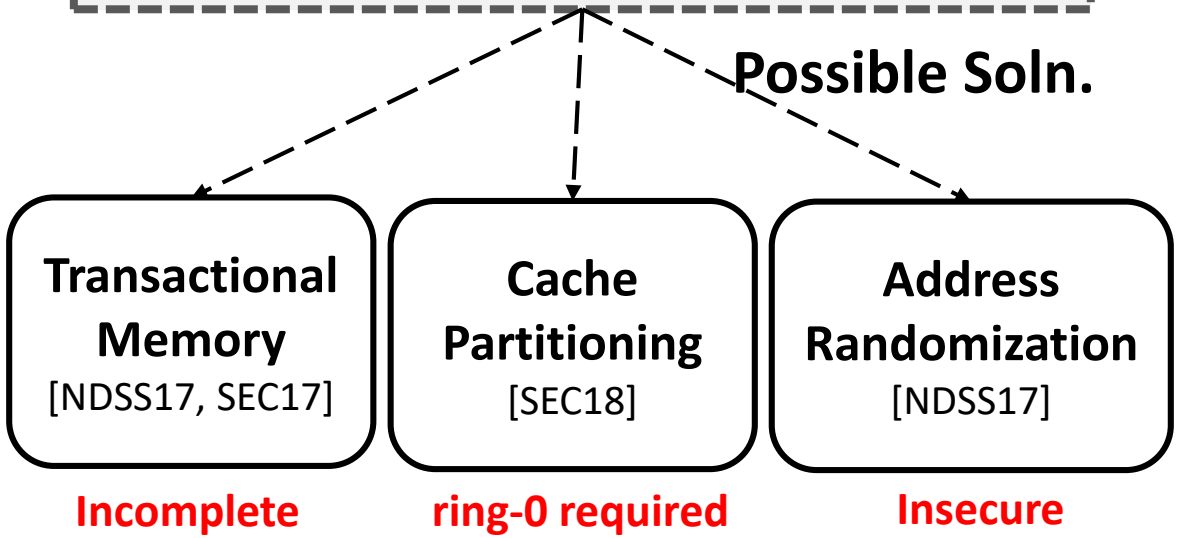


Lesson #1

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

Learning from existing solutions!

Access patterns attacks!

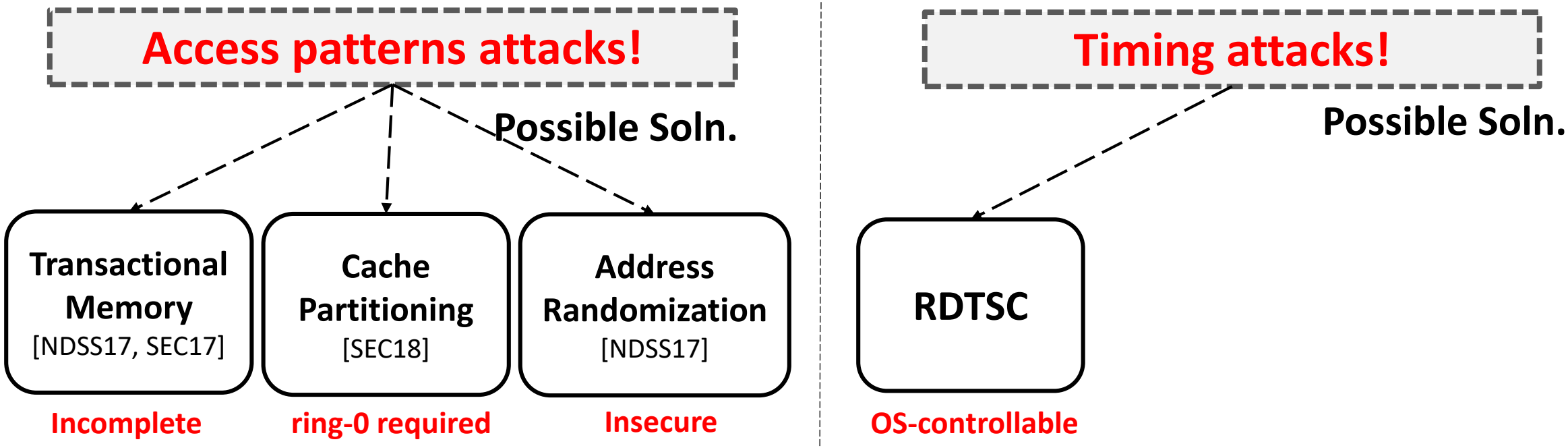


Lesson #1

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

Timing attacks!

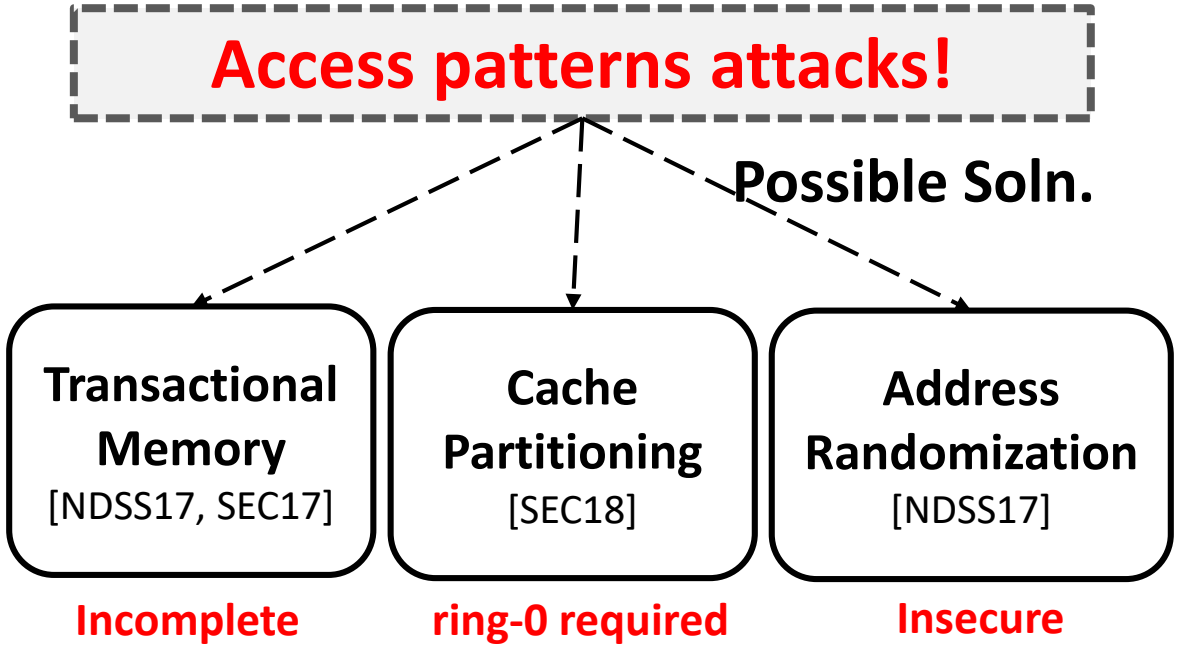
Learning from existing solutions!

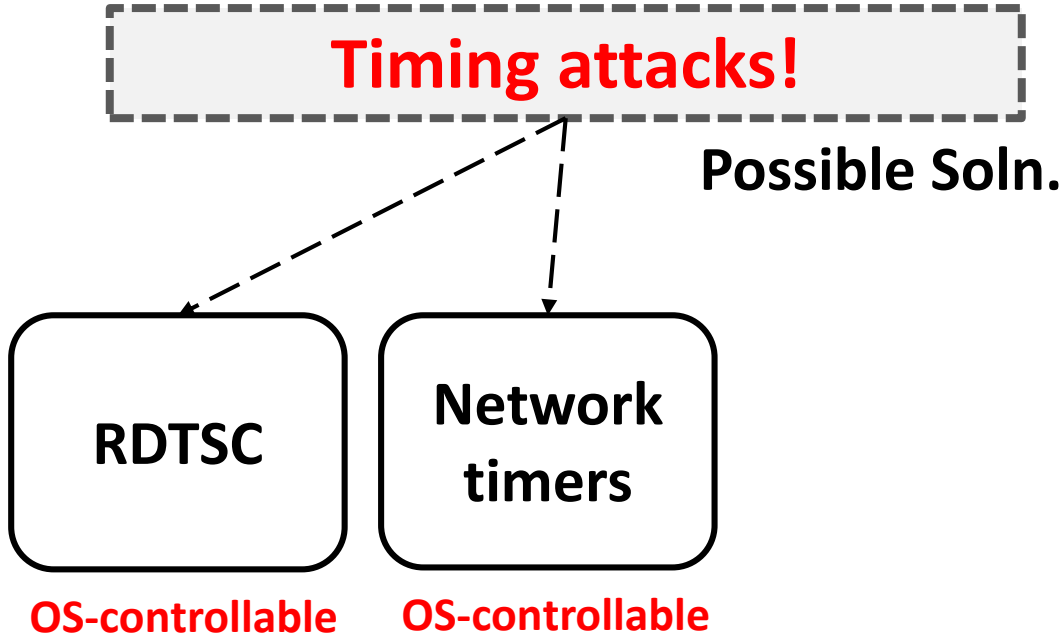


Lesson #1

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

Learning from existing solutions!

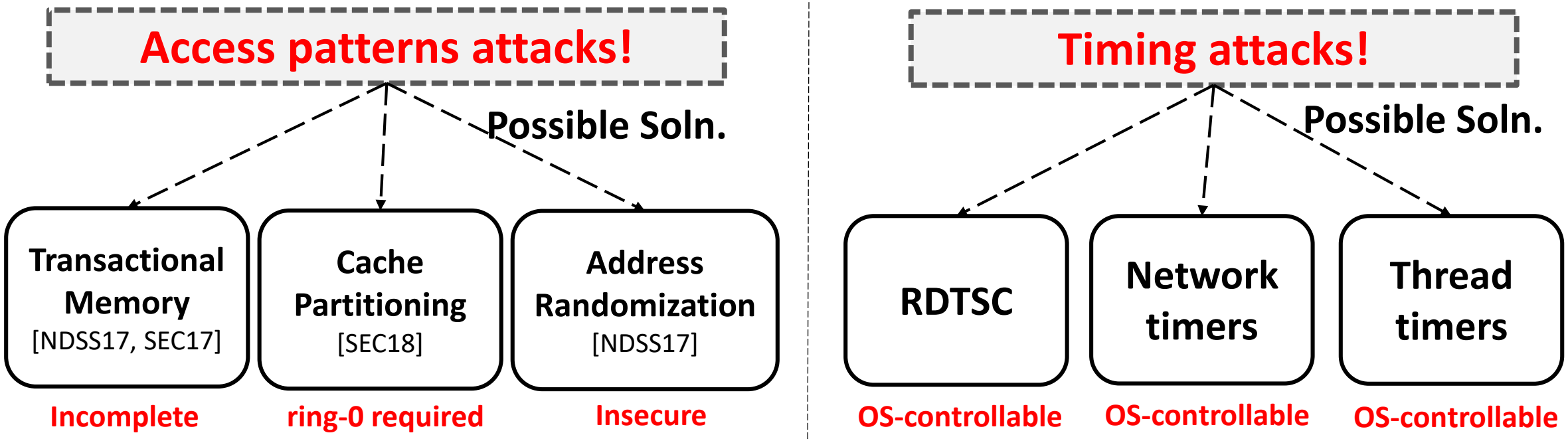




Lesson #1

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

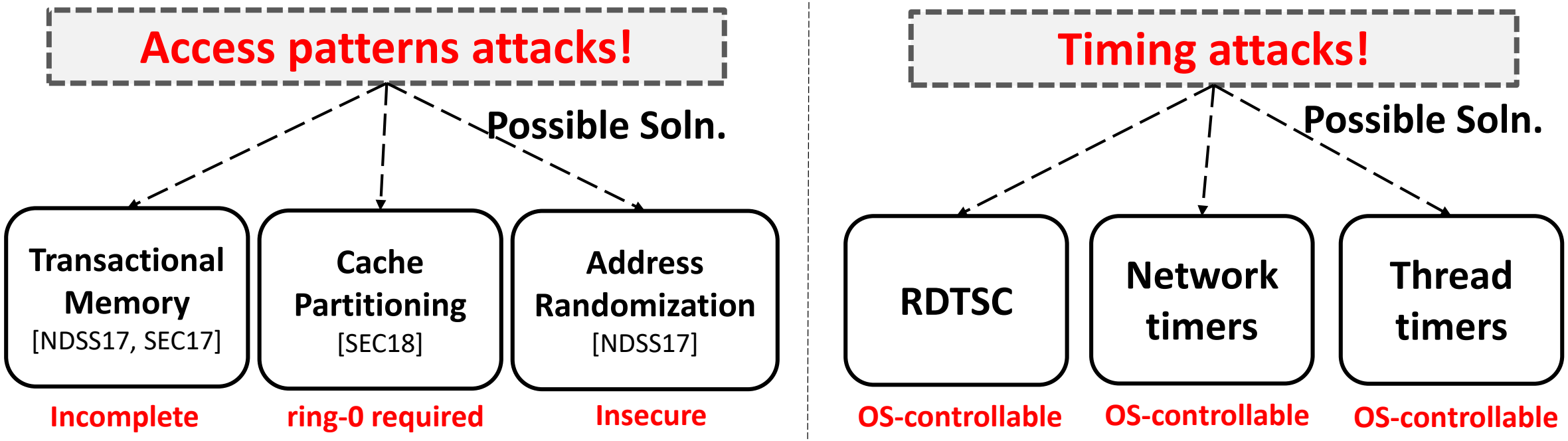
Learning from existing solutions!



Lesson #1

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

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

Our approach

- **Indistinguishable enclave program(s)**

Our approach

- **Indistinguishable enclave program(s)**
  - A code block executed **N times** on C-Pad, and data block accessed from D-Pad

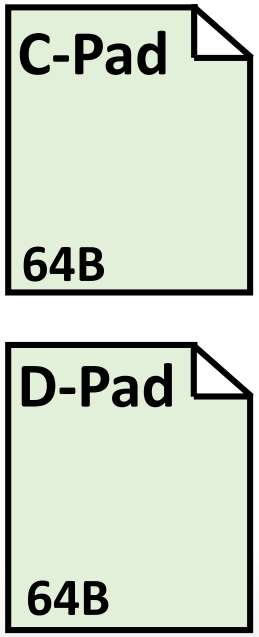
Our approach

- **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!

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!

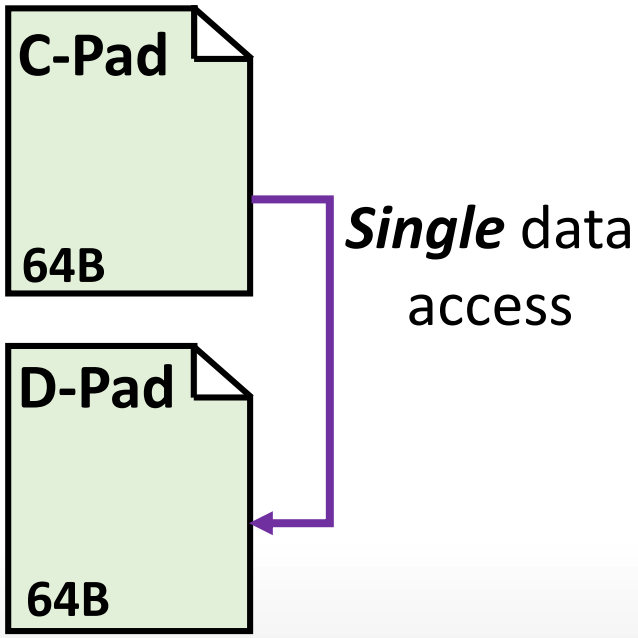
# of executions: 0



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!

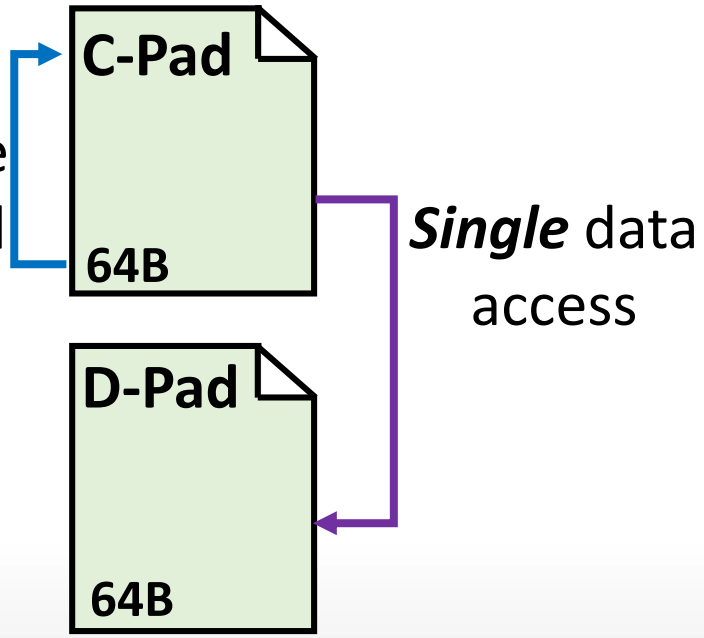
# of executions: 0



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!

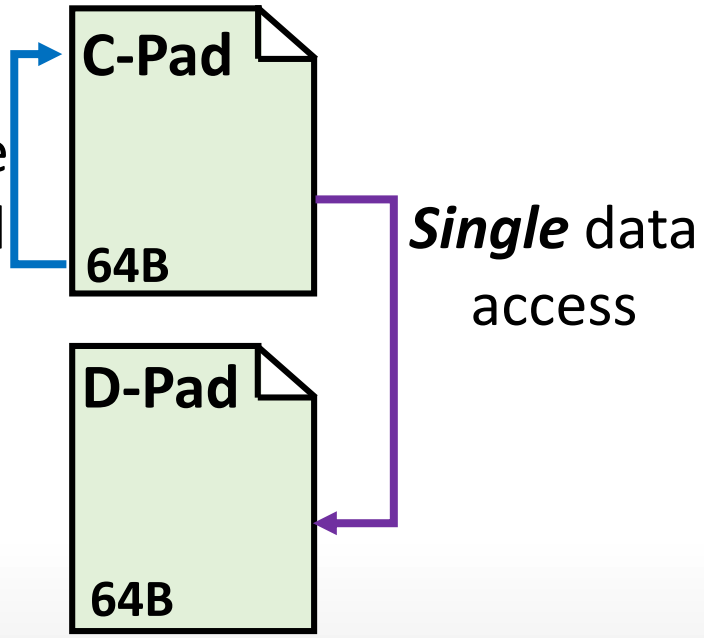
# of executions: ***1***



Our approach

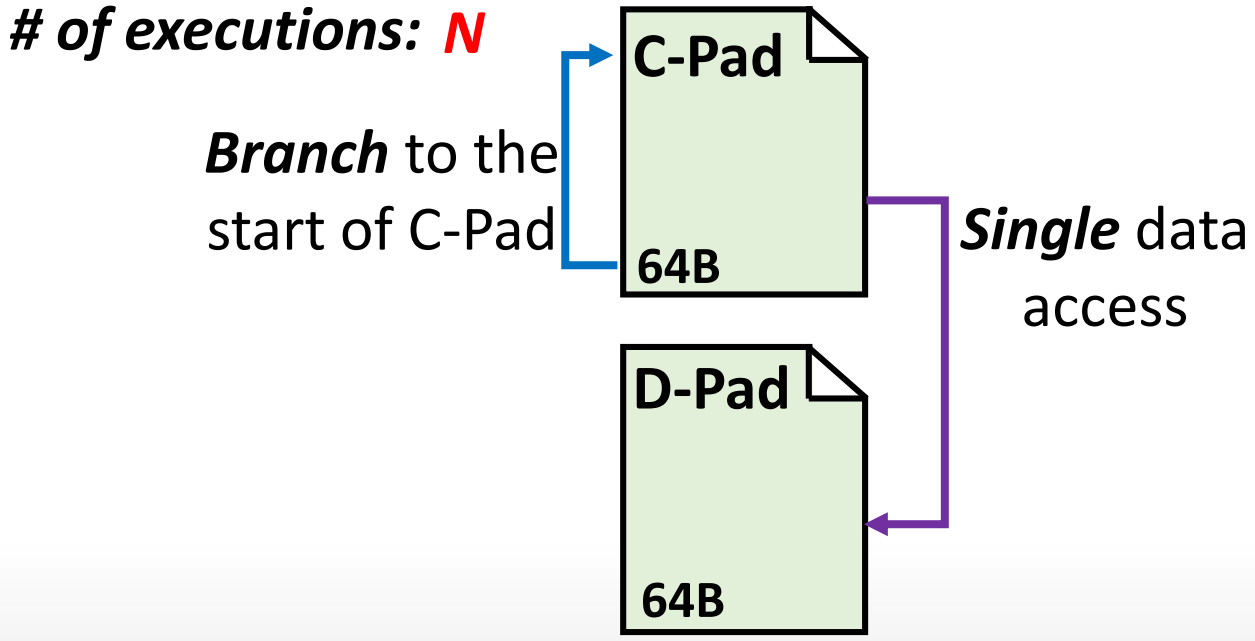
- **Indistinguishable enclave program(s)**
  - A code block executed **N times** on C-Pad, and data block accessed from D-Pad
  - C-Pad and D-Pad are one cache-line (64B) in size!

# of executions: ***1***



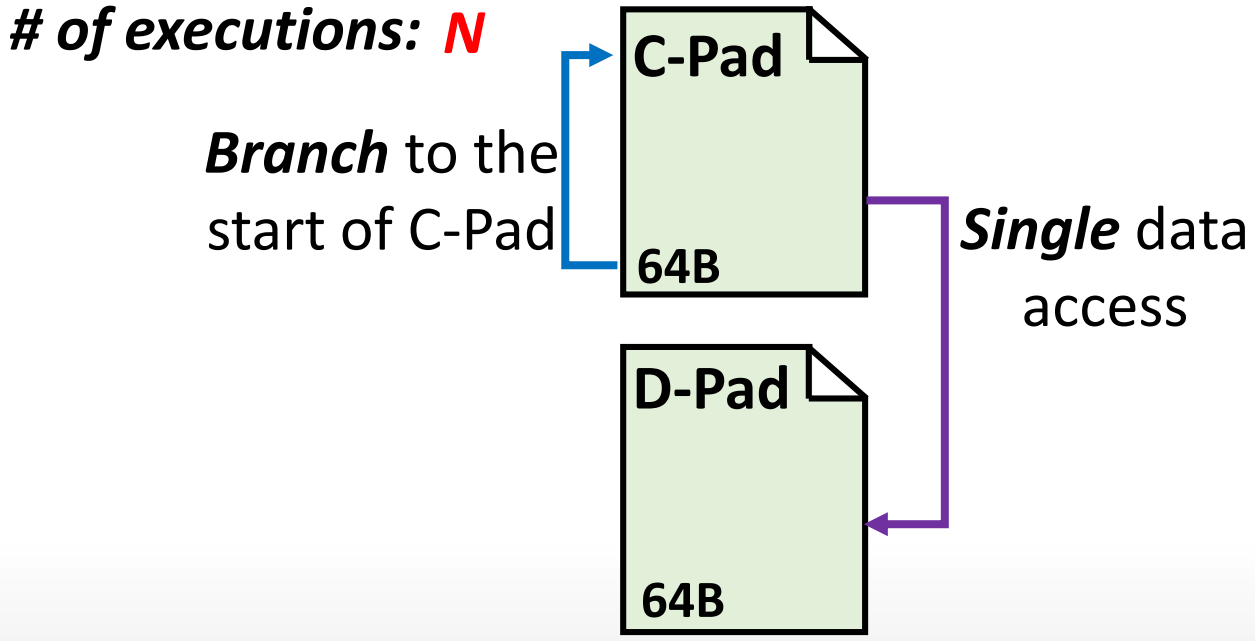
Our approach

- **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!



Our approach

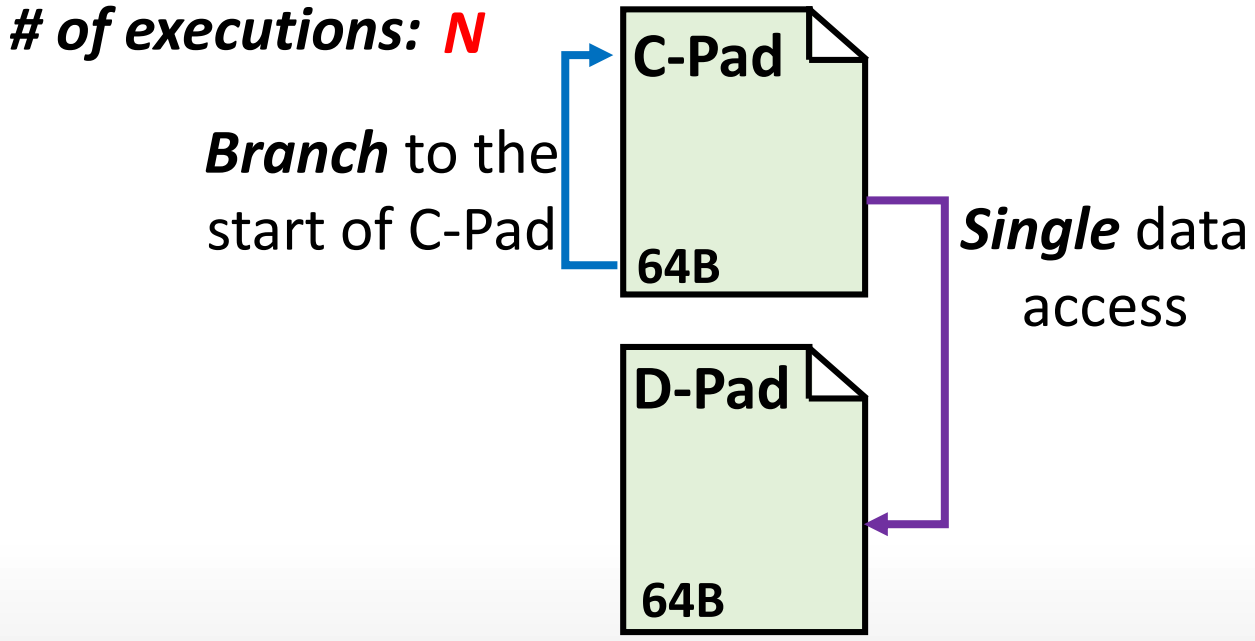
- **Indistinguishable enclave program(s)**
  - A code block executed **N times** on C-Pad, and data block accessed from D-Pad
  - C-Pad and D-Pad are one cache-line (64B) in size!



What do the attacks reveal?

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!

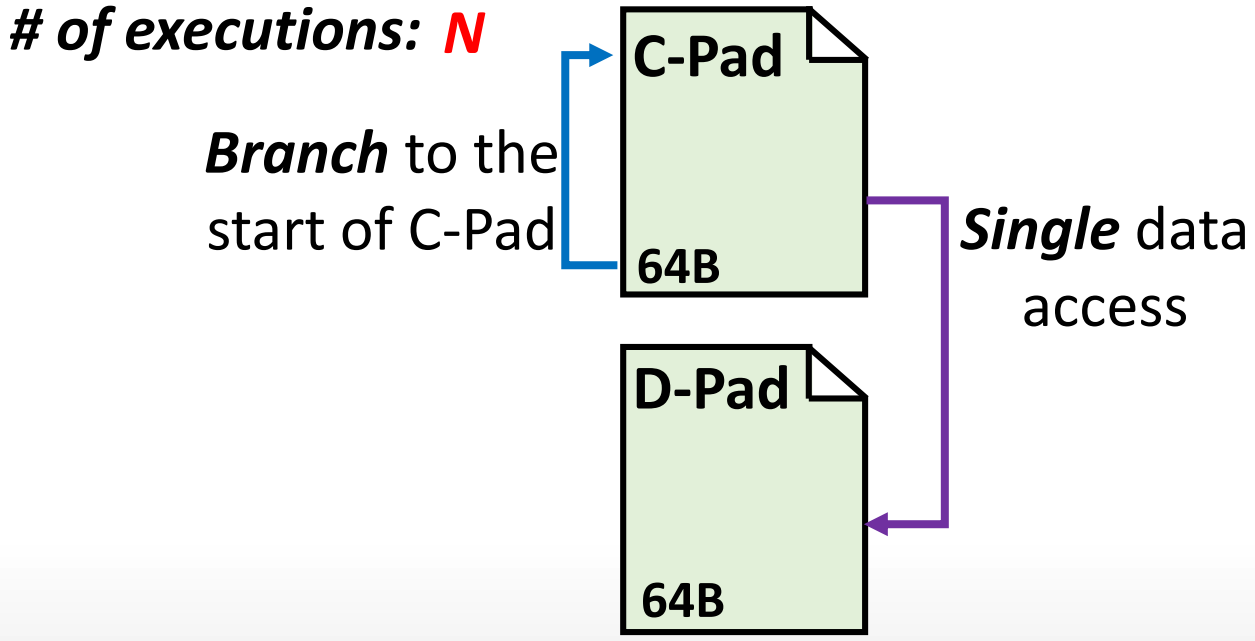


What do the attacks reveal?

Paging Attack: Same page

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!



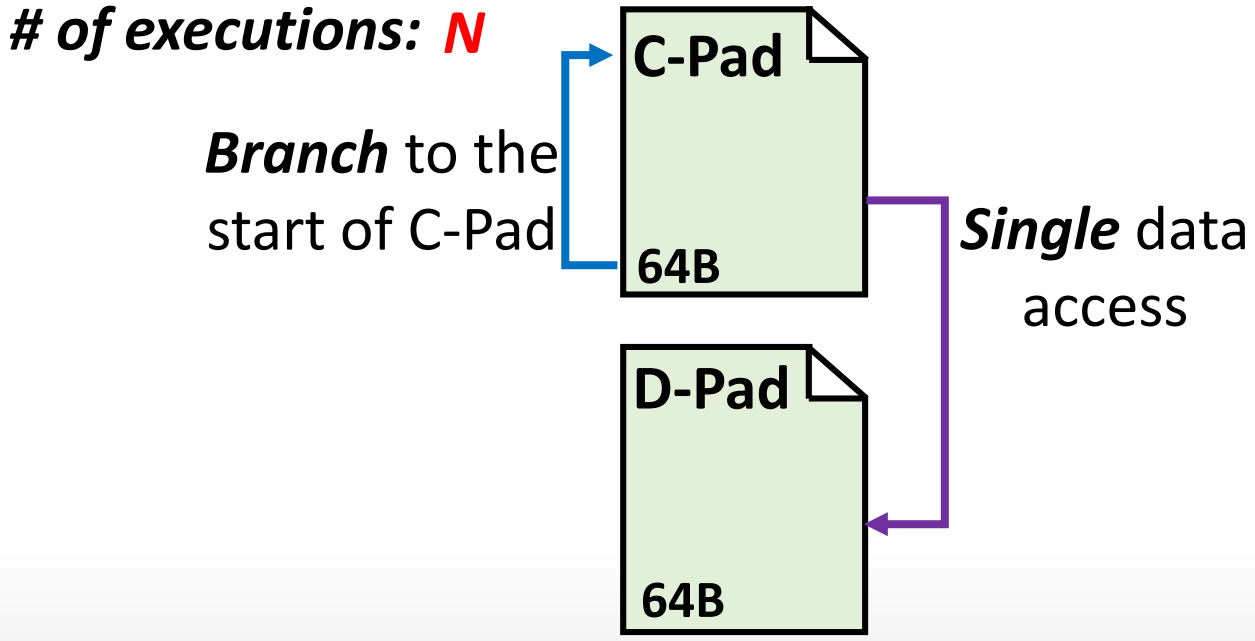
What do the attacks reveal?

Paging Attack: Same page

Cache Attack: Same cache-lines

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!



What do the attacks reveal?

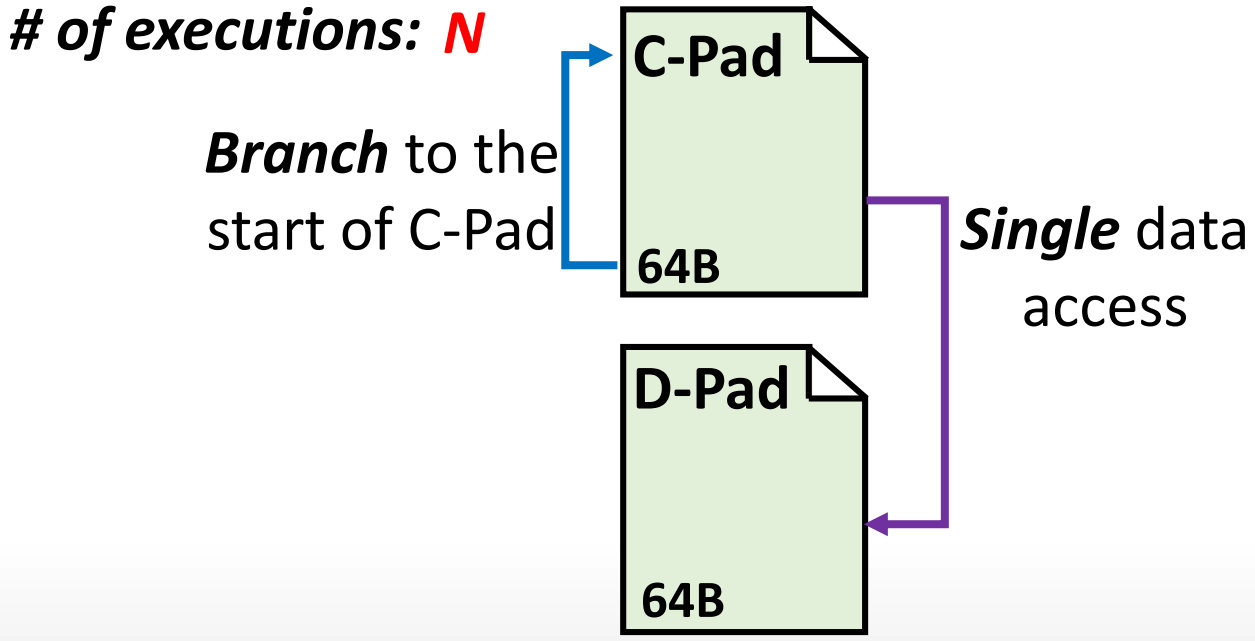
Paging Attack: Same page

Cache Attack: Same cache-lines

Branch Attack: Same branch

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!



What do the attacks reveal?

Paging Attack: Same page

Cache Attack: Same cache-lines

Branch Attack: Same branch

Timing Attack: Same time to execute N code blocks

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!

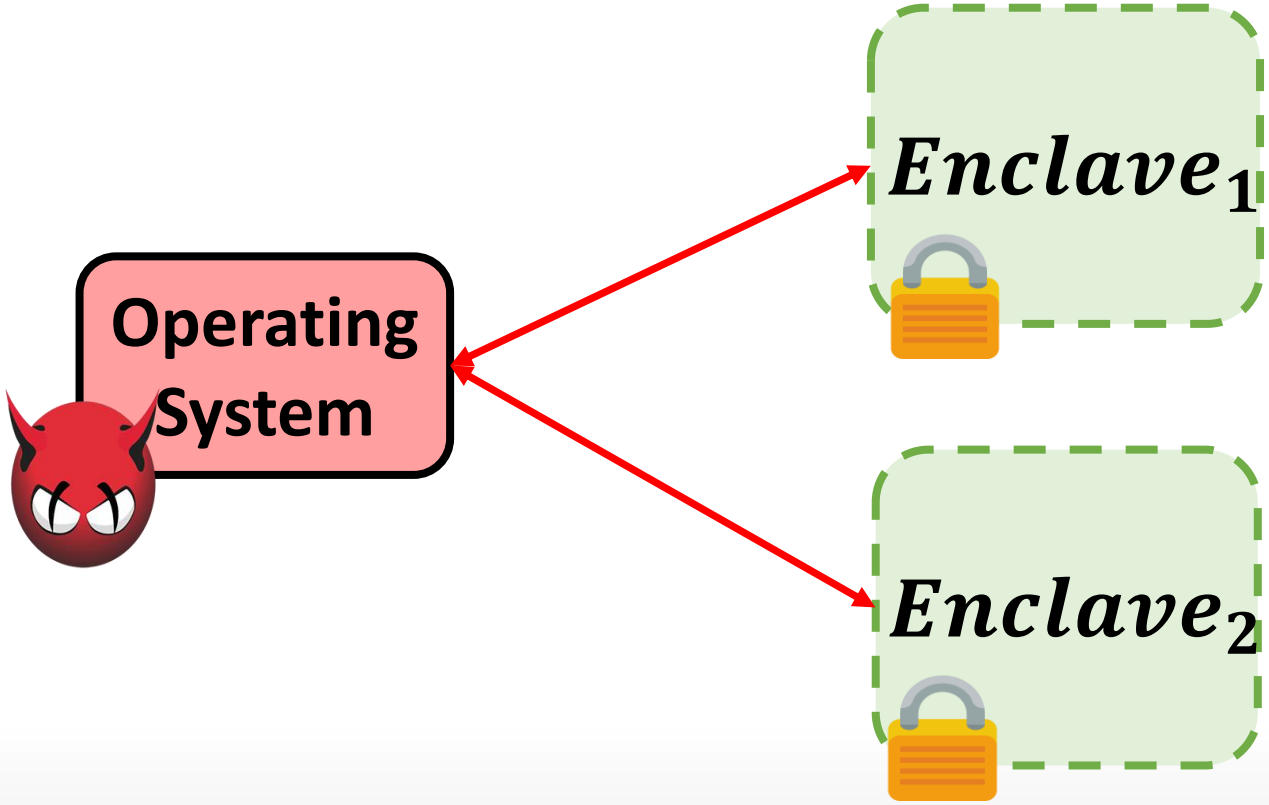
Cache Attack: Same cache-lines

Branch Attack: Same branch

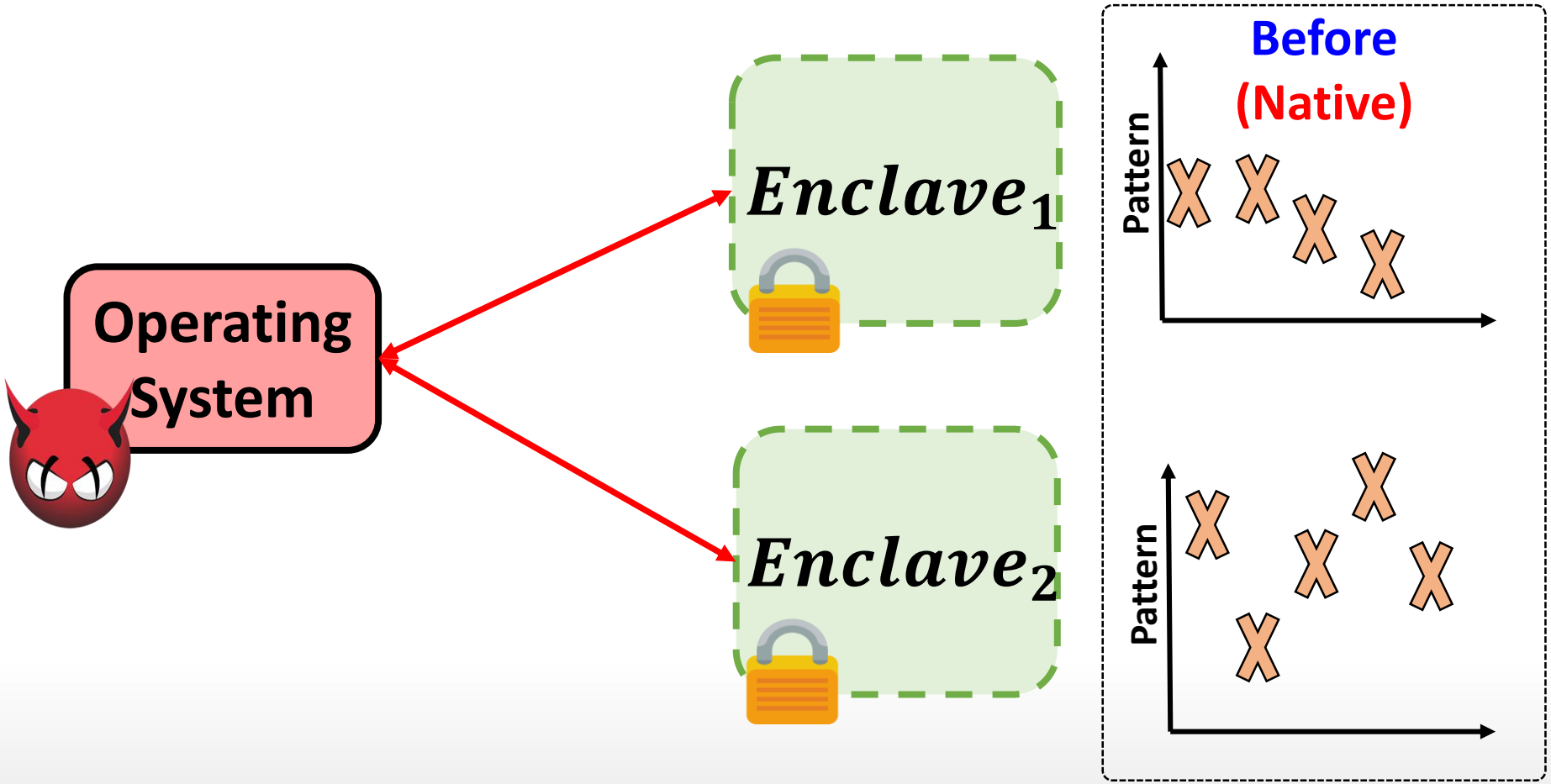
Timing Attack: Same time to execute N code blocks

Let Hermione explain!

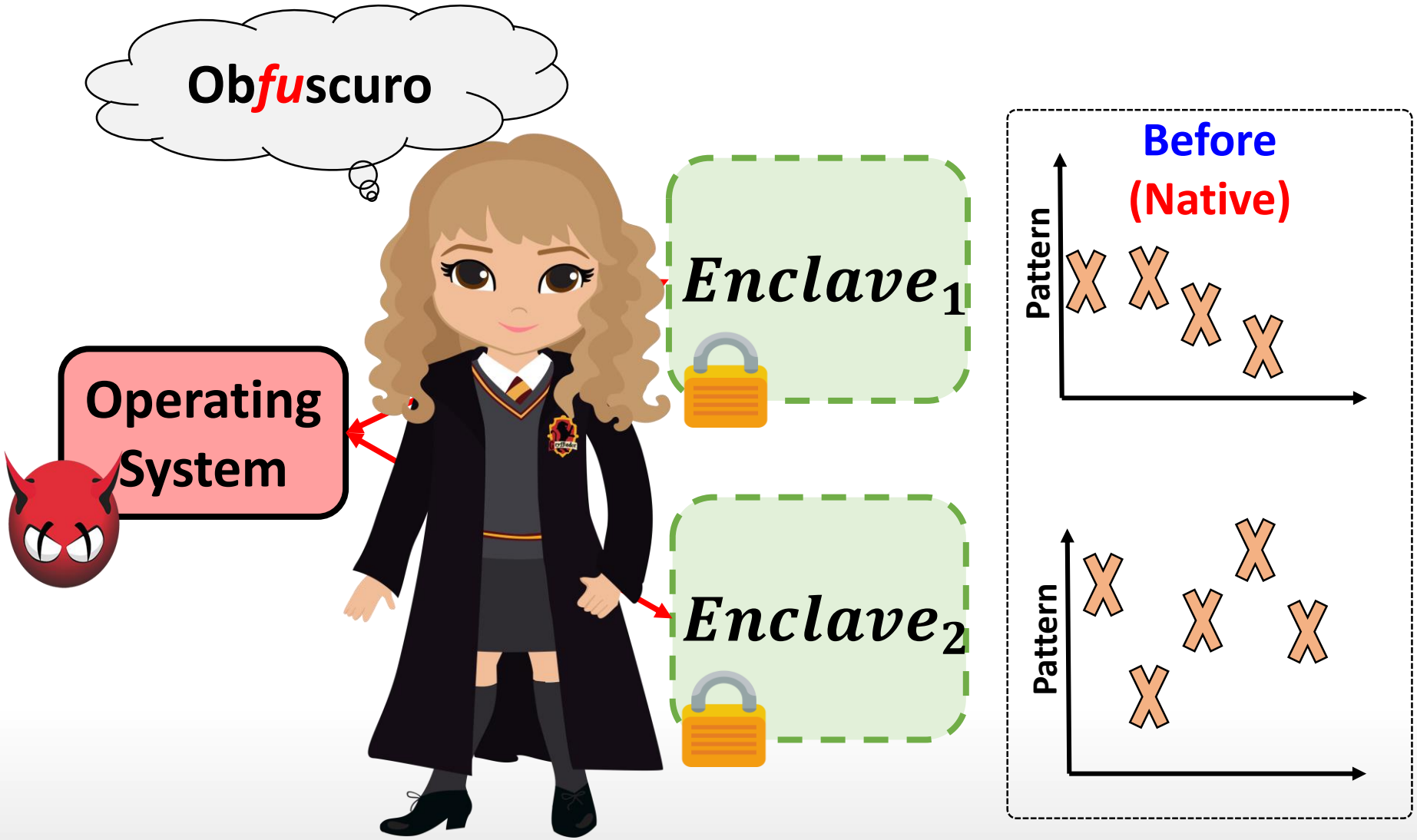
Let Hermione explain!



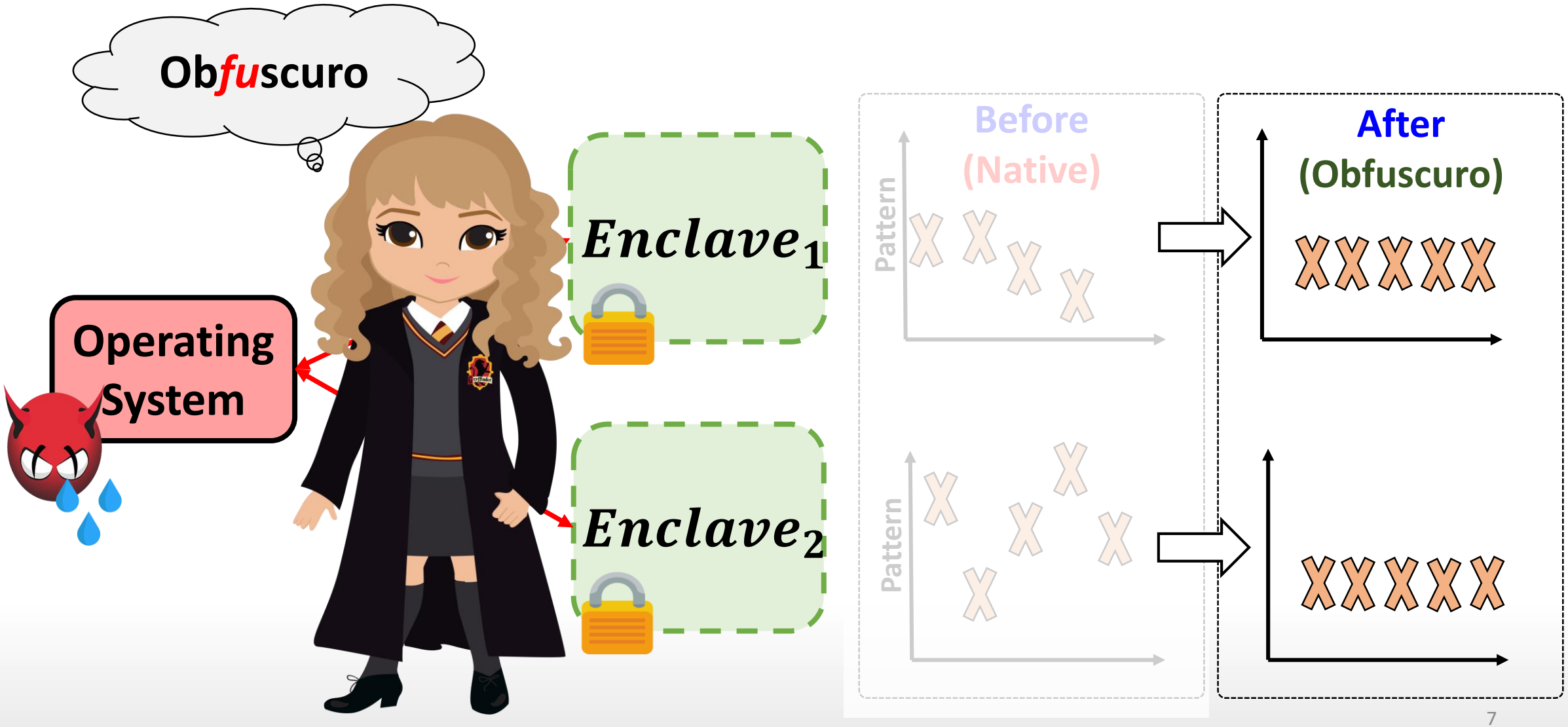
Let Hermione explain!



Let Hermione explain!



Let Hermione explain!



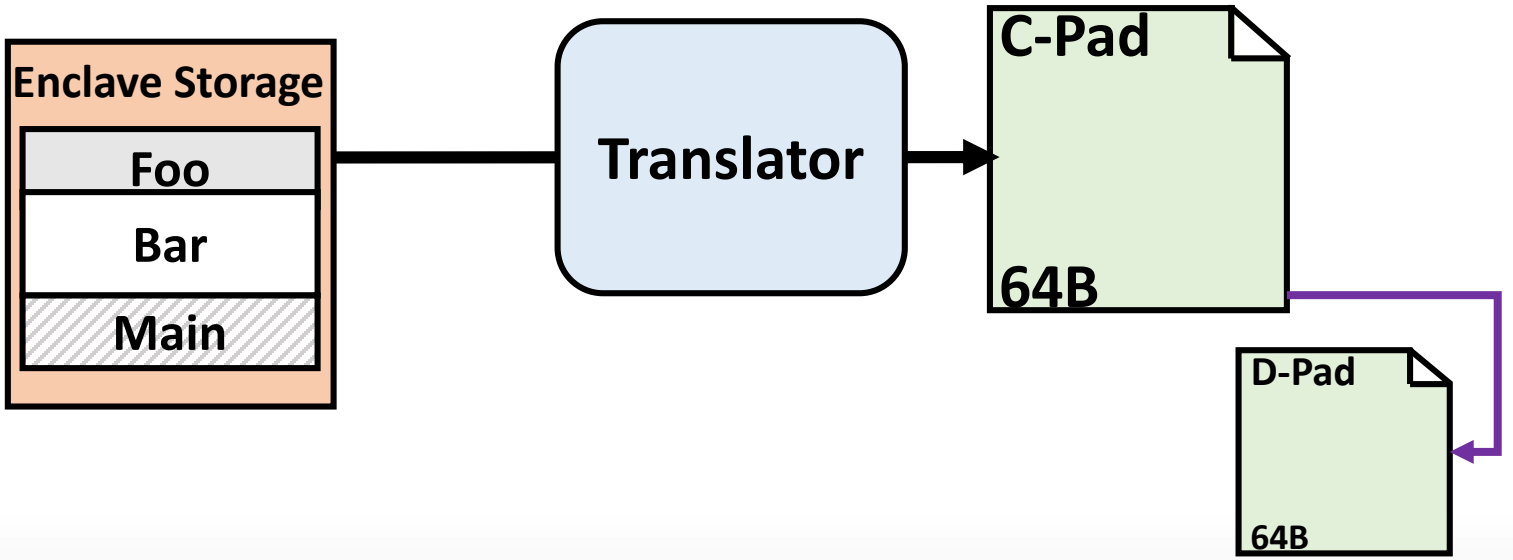
Cool, what's the challenge?

Cool, what's the challenge?

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

Cool, what's the challenge?

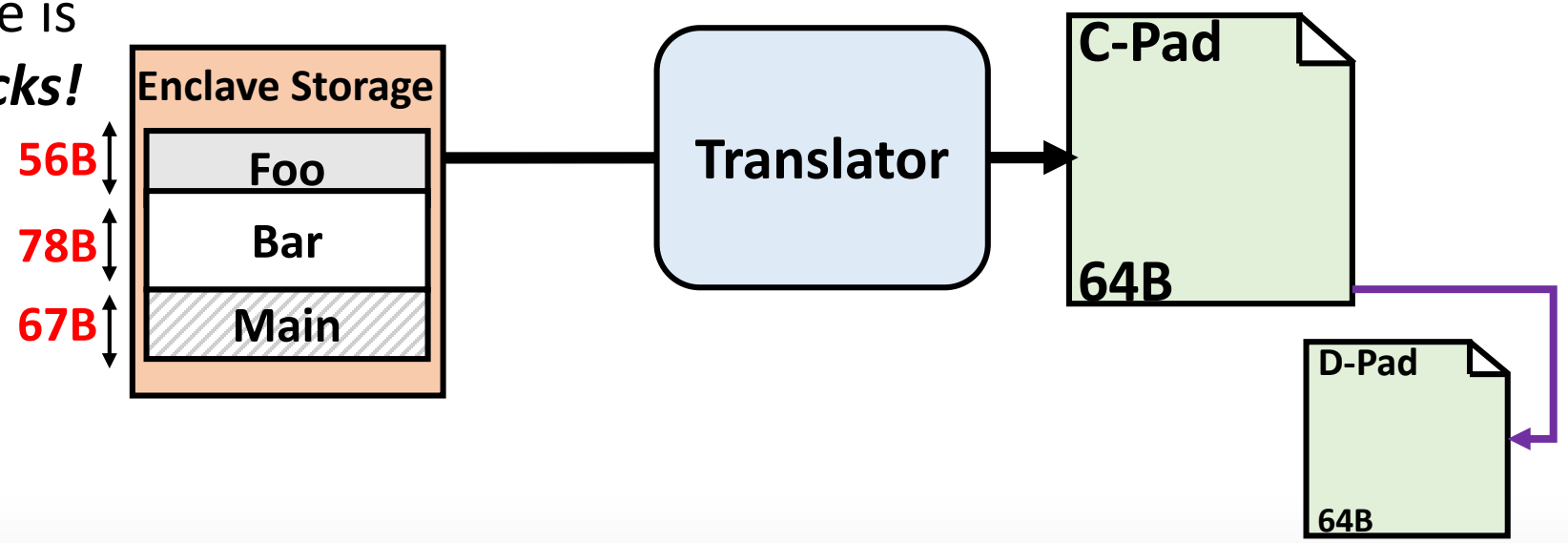
- **Naïve solution**
  - Use a software-translator to copy all code and data onto C/D-Pad



Cool, what's the challenge?

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

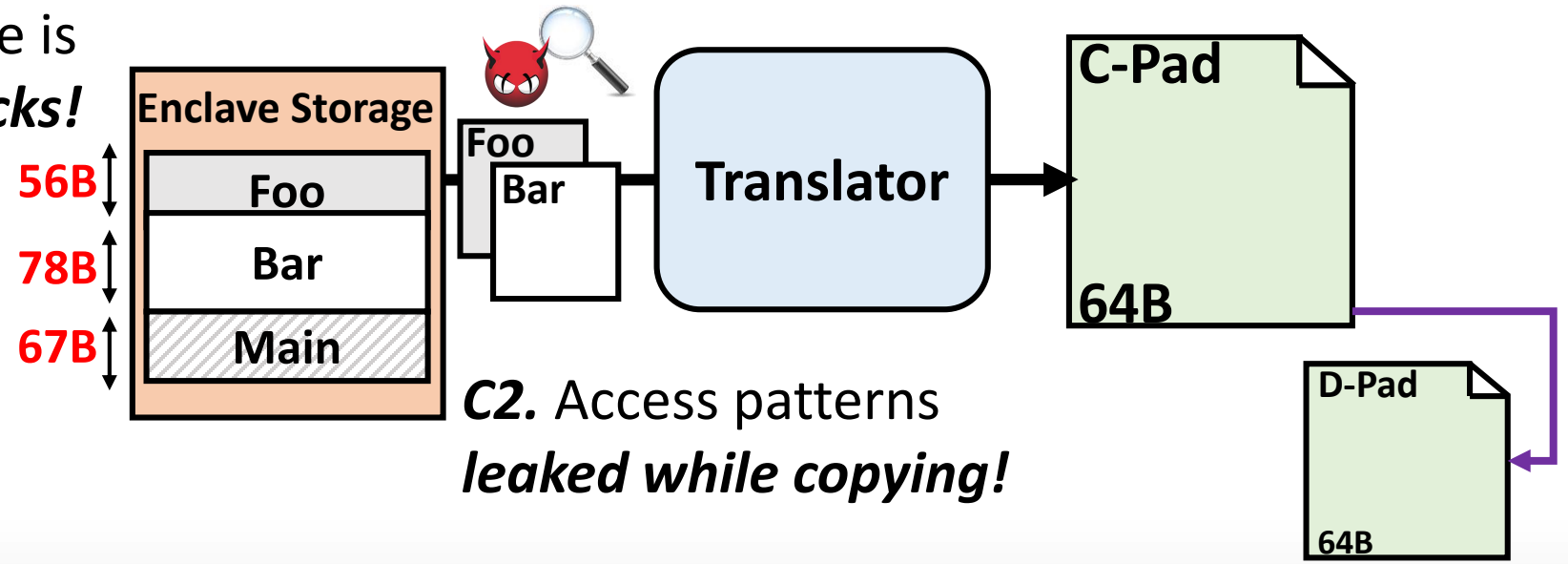
C1. Native code is ***not in 64B blocks!***



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!***



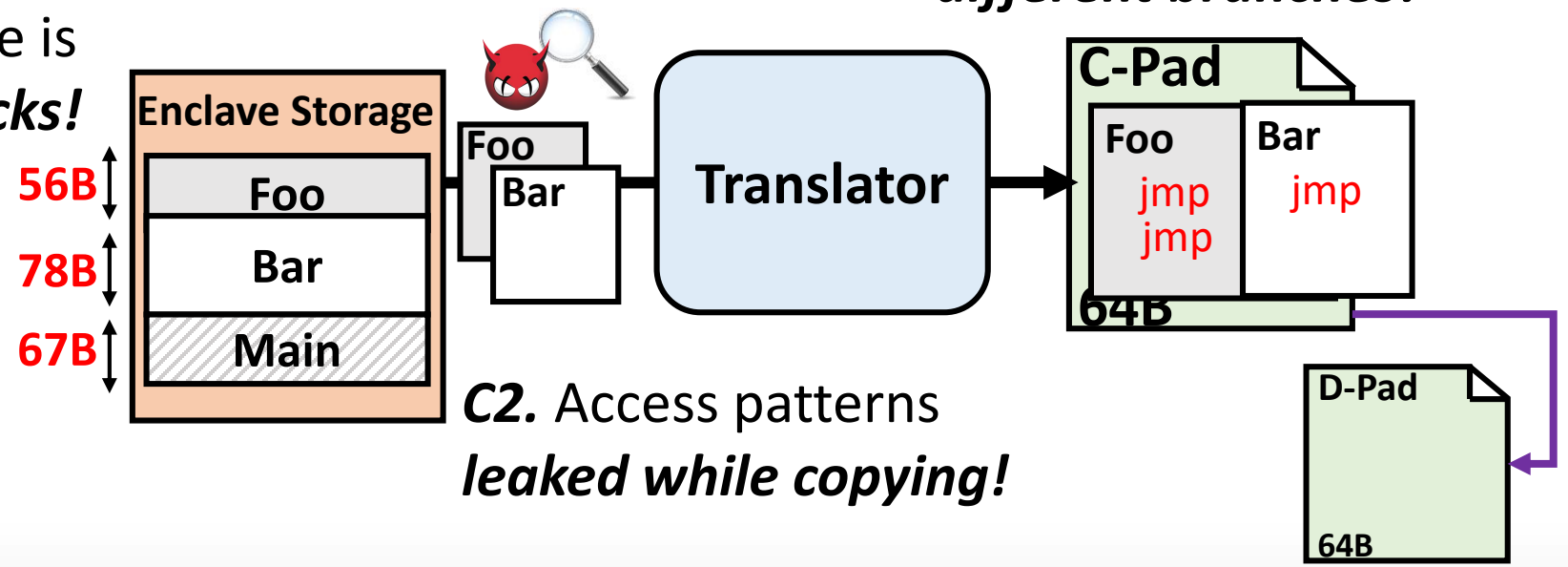
C2. Access patterns ***leaked while copying!***

Cool, what's the challenge?

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

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

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



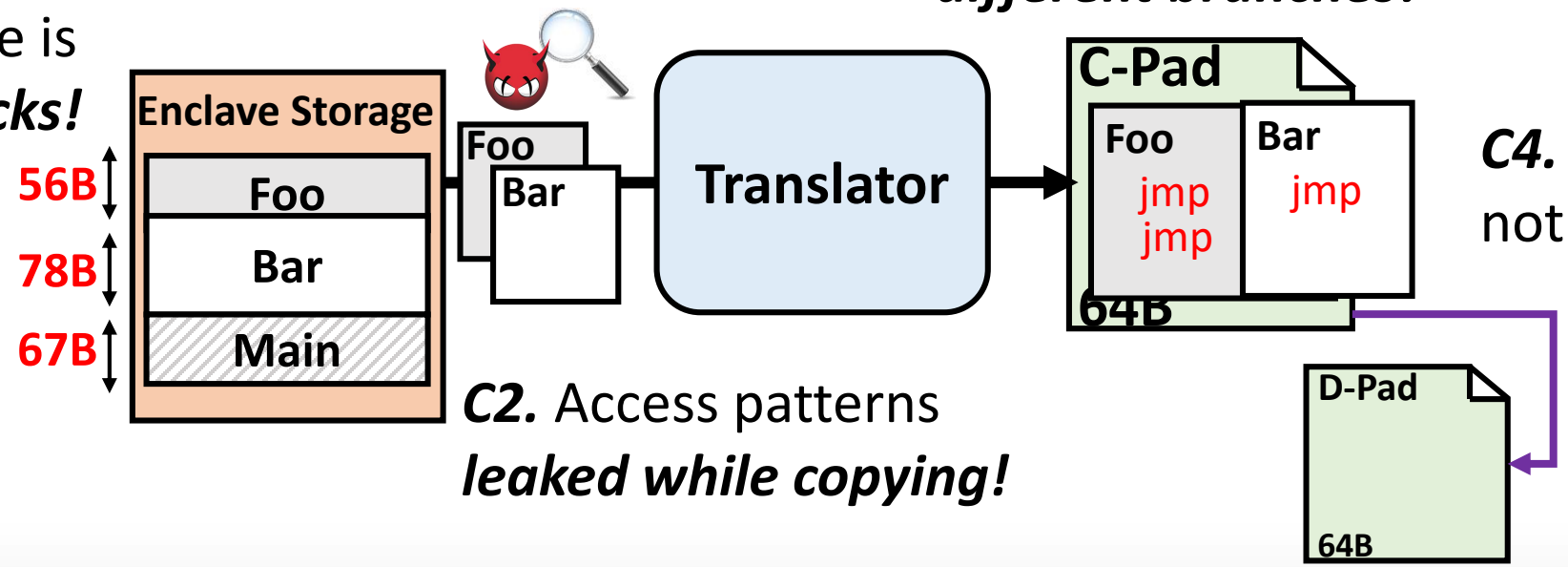
C2. Access patterns ***leaked while copying!***

Cool, what's the challenge?

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

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

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

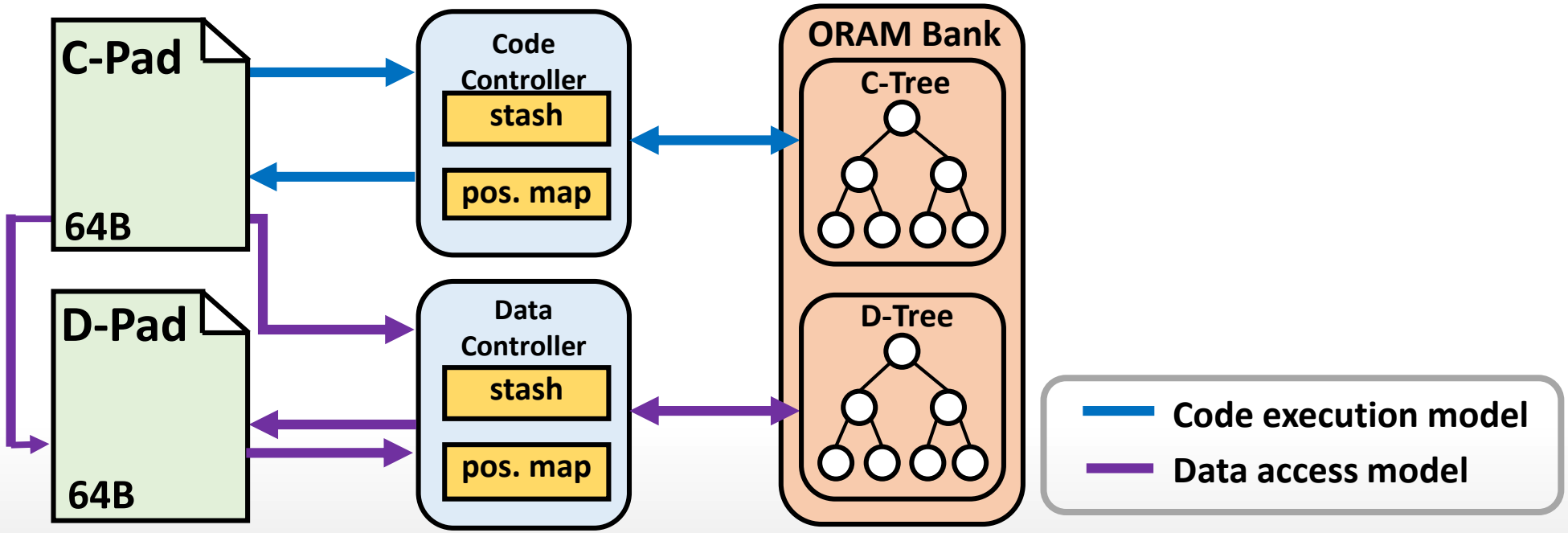


C2. Access patterns ***leaked while copying!***

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)**



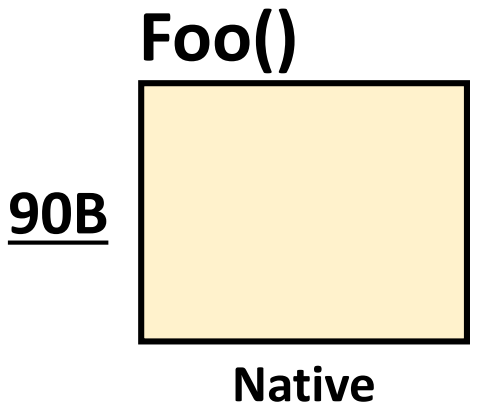
C1. Enforce code blocks of identical sizes

C1. Enforce code blocks of identical sizes

- Break code blocks into 64 bytes and pad using nop

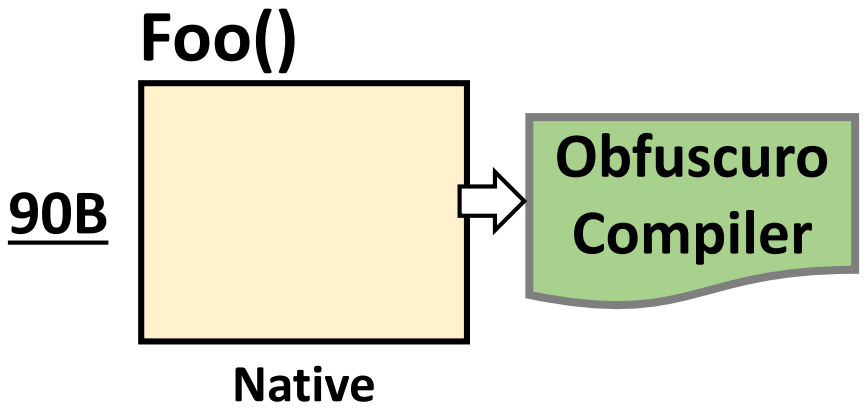
C1. Enforce code blocks of identical sizes

- Break code blocks into 64 bytes and pad using nop



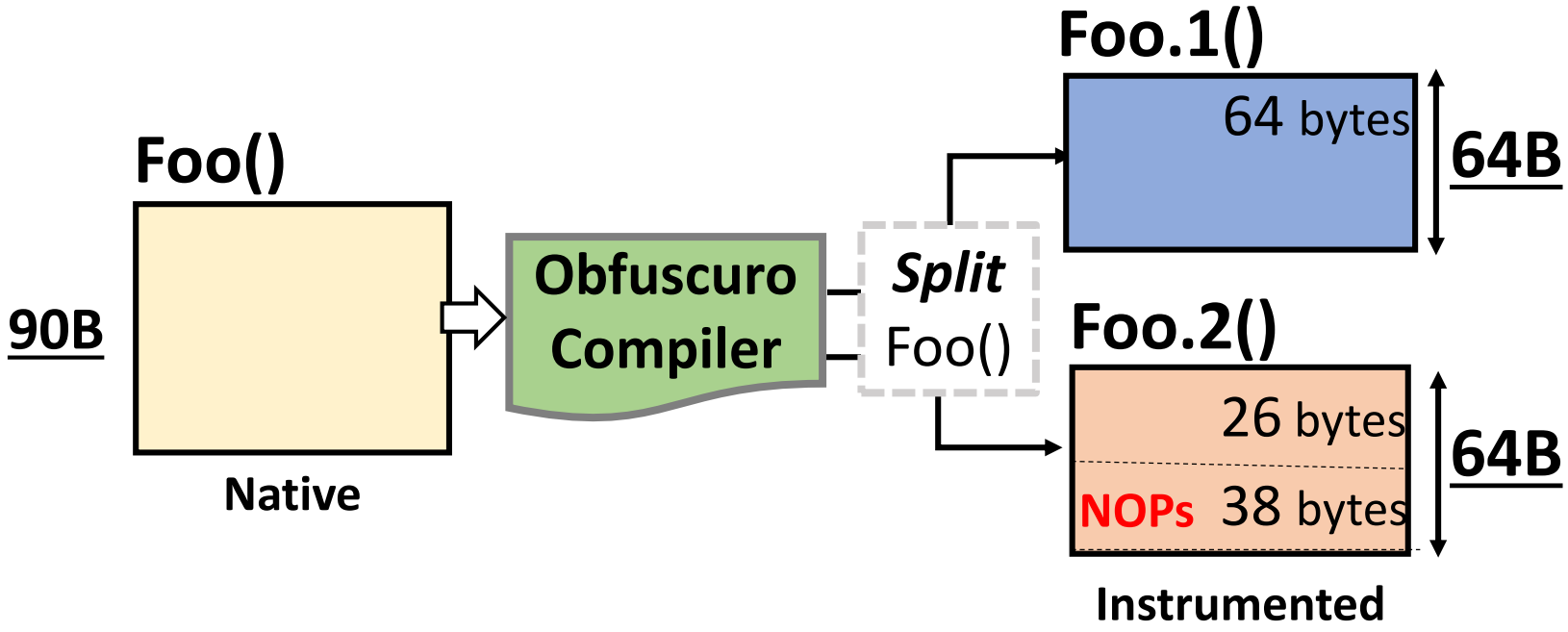
C1. Enforce code blocks of identical sizes

- Break code blocks into 64 bytes and pad using nop



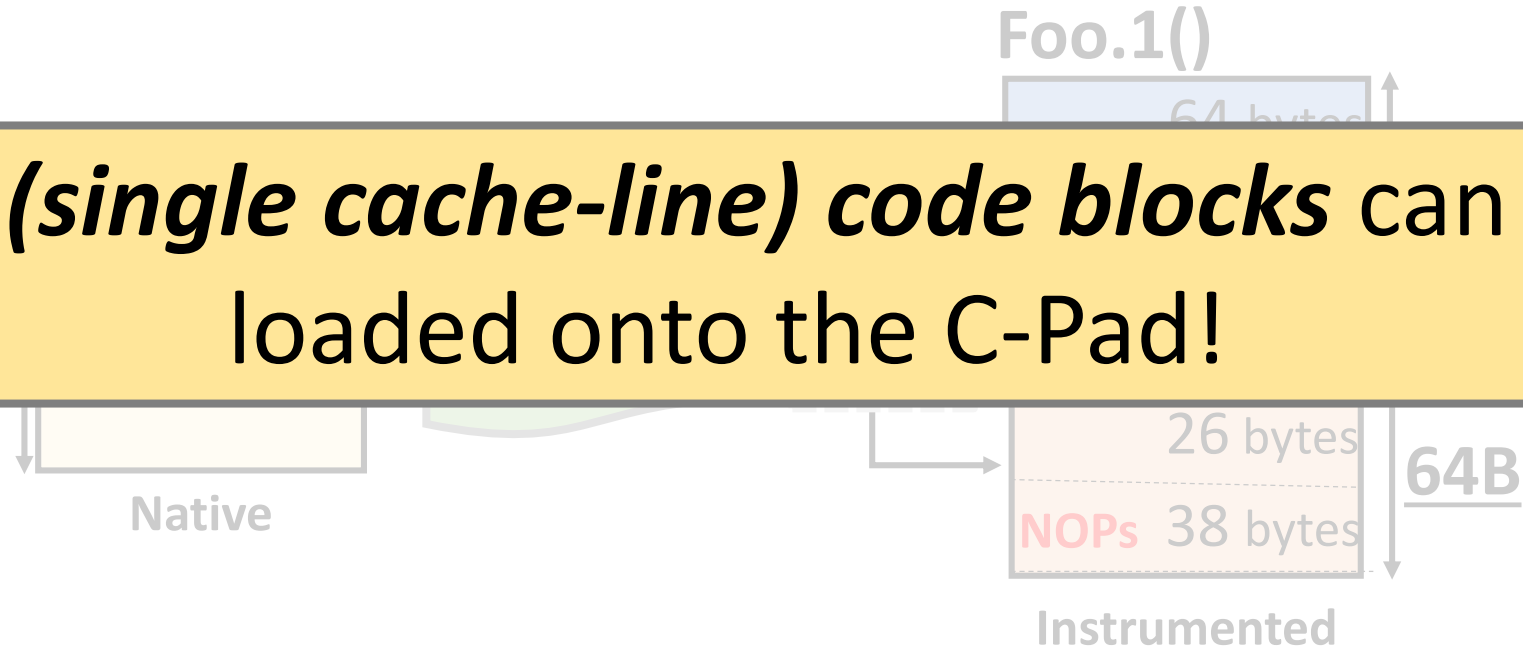
C1. Enforce code blocks of identical sizes

- Break code blocks into 64 bytes and pad using nop



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!

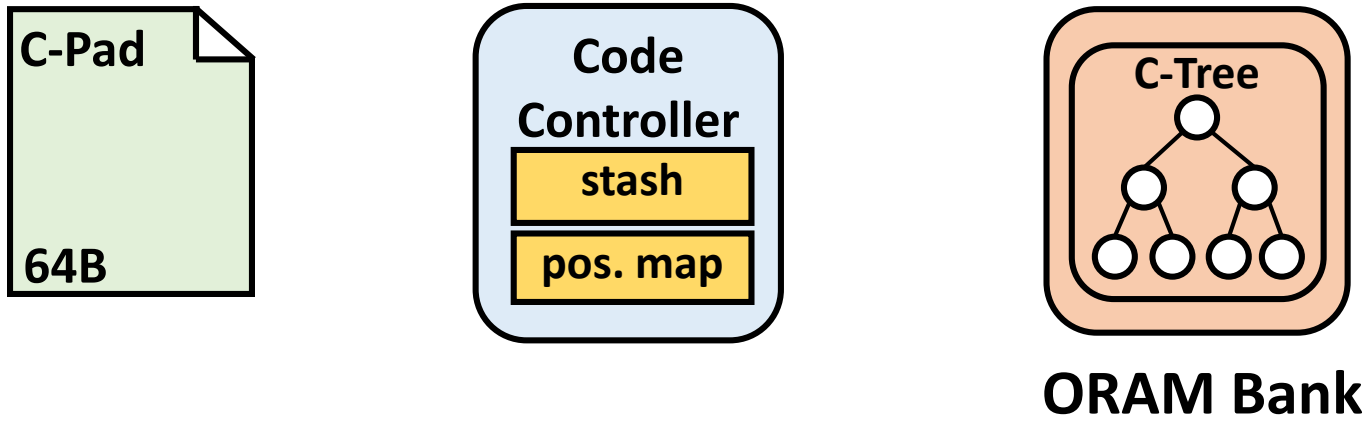
C2. Securely loading C/D-Pad

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.

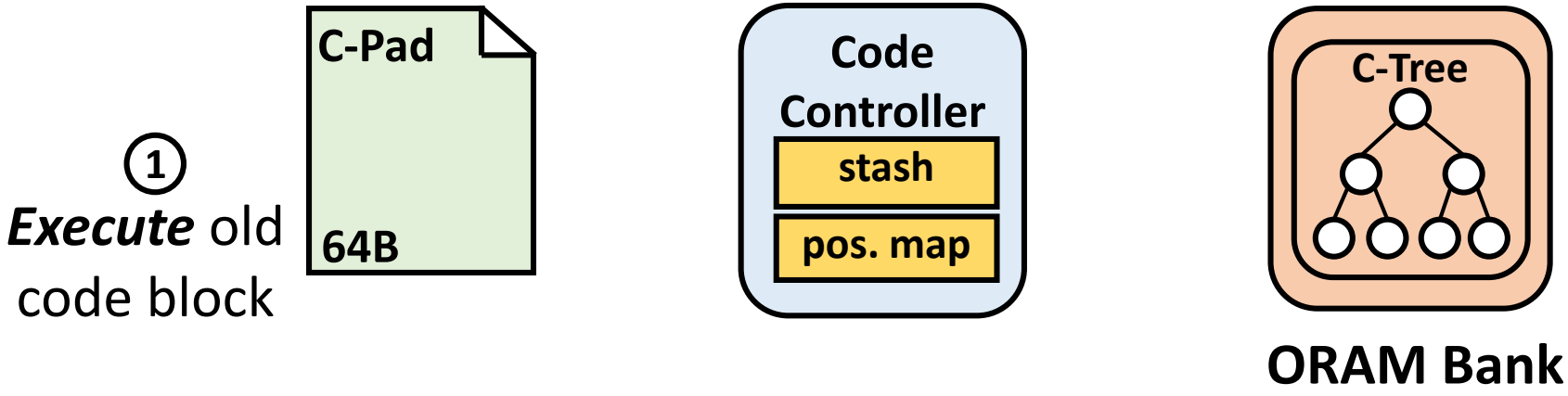
C2. Securely loading C/D-Pad

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



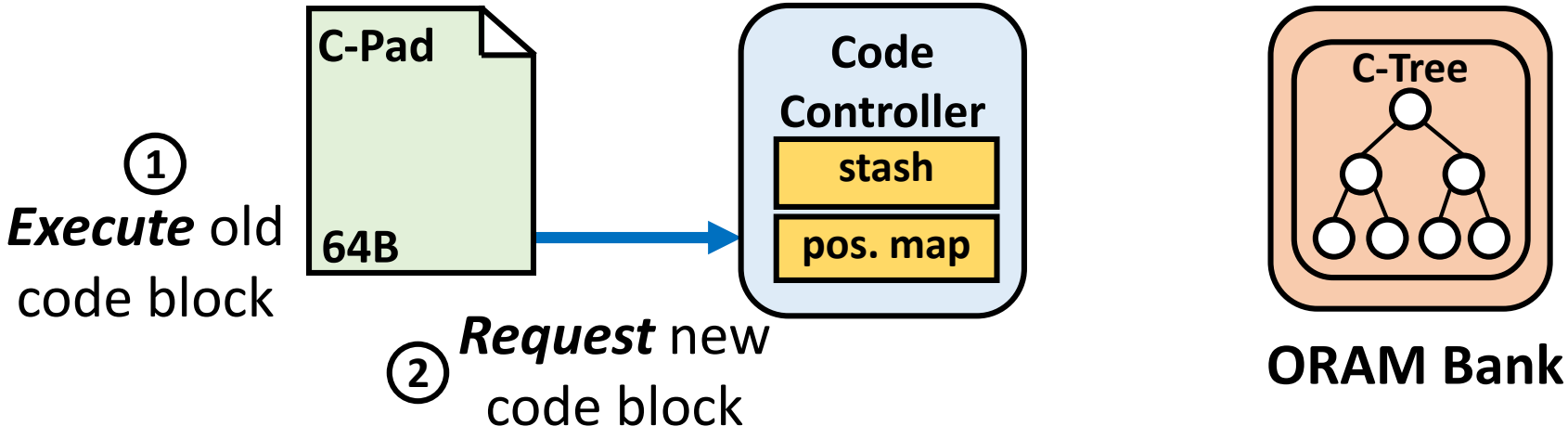
C2. Securely loading C/D-Pad

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



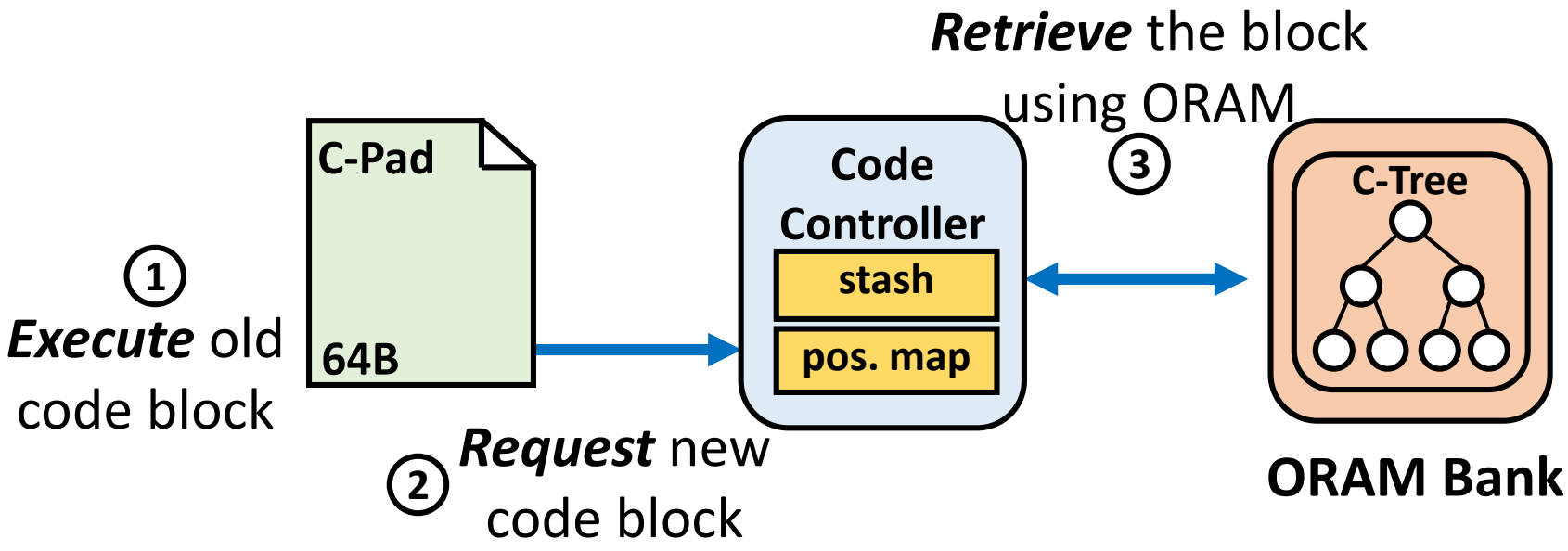
C2. Securely loading C/D-Pad

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



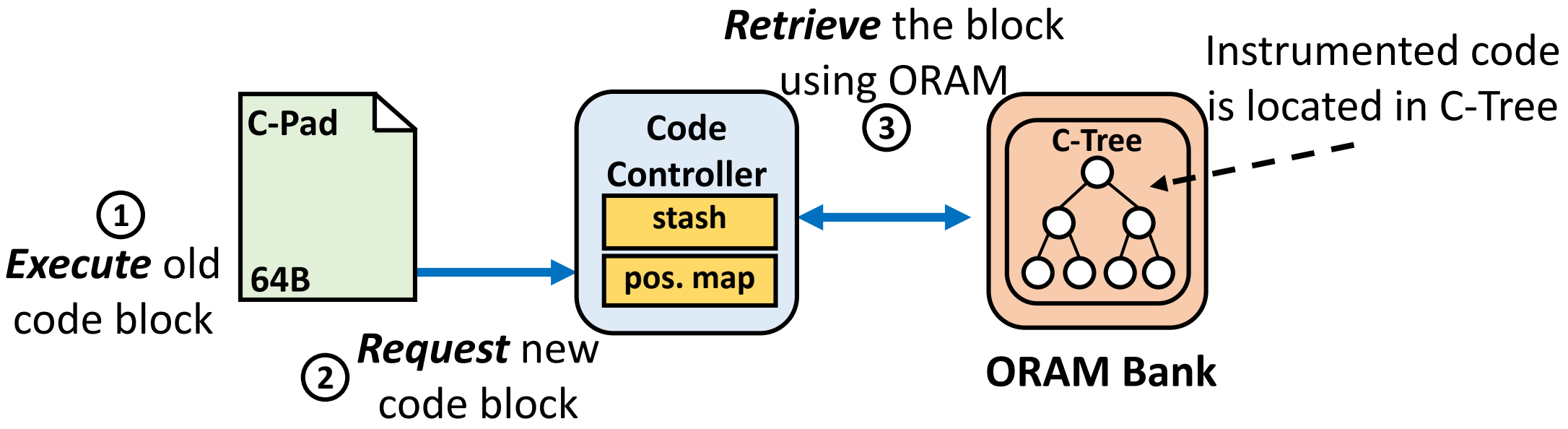
C2. Securely loading C/D-Pad

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



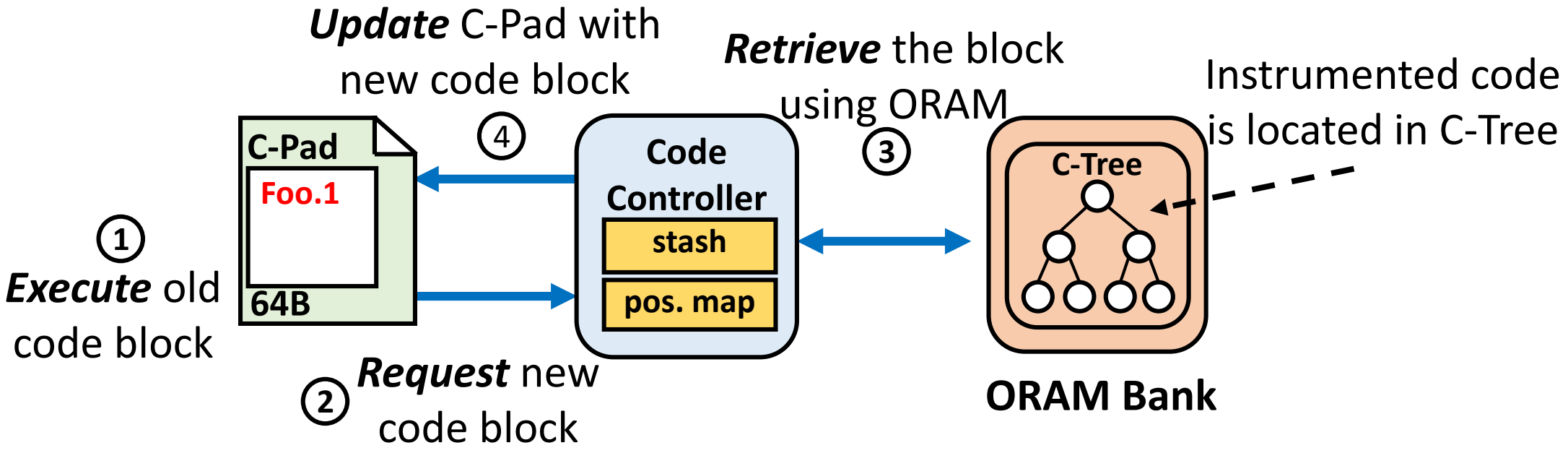
C2. Securely loading C/D-Pad

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



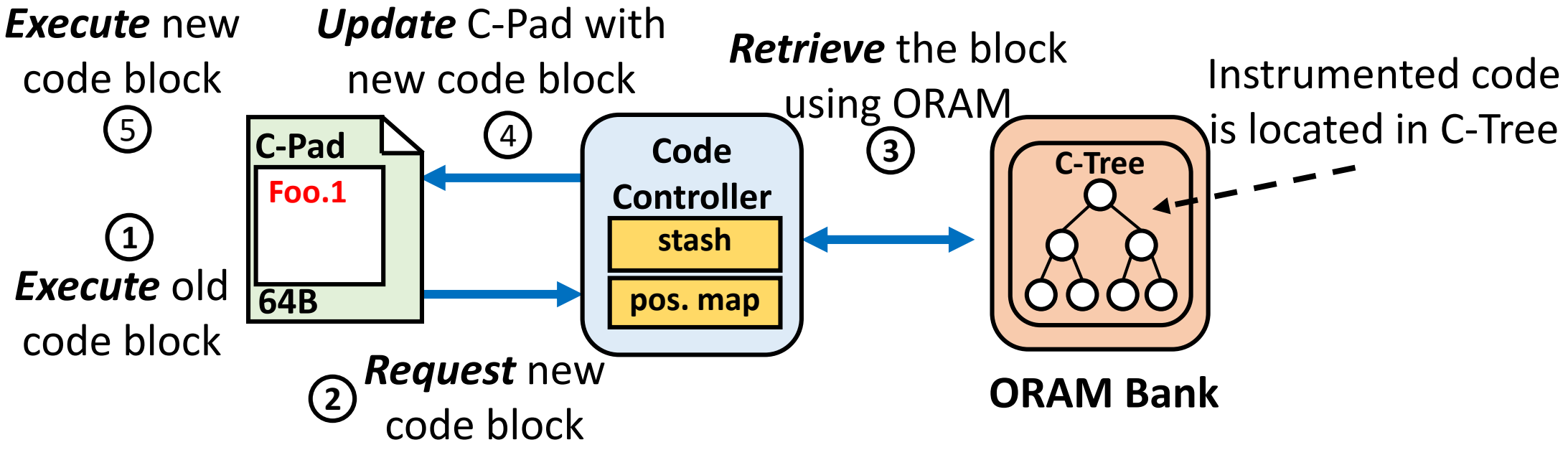
C2. Securely loading C/D-Pad

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



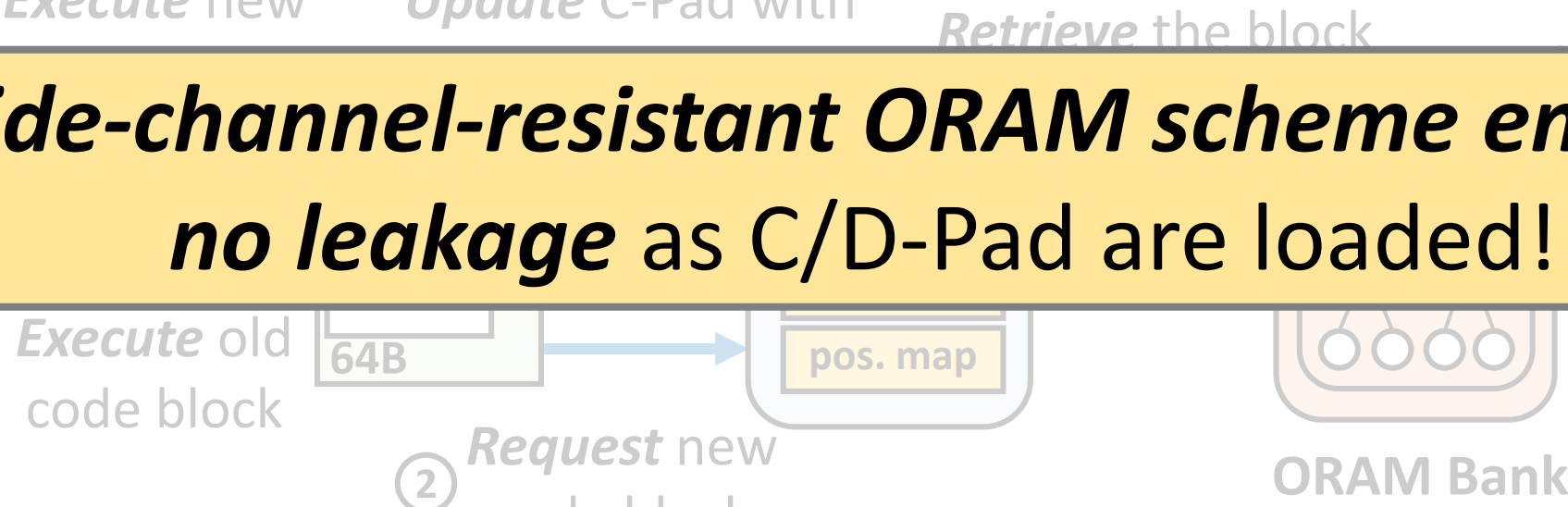
C2. Securely loading C/D-Pad

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



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!

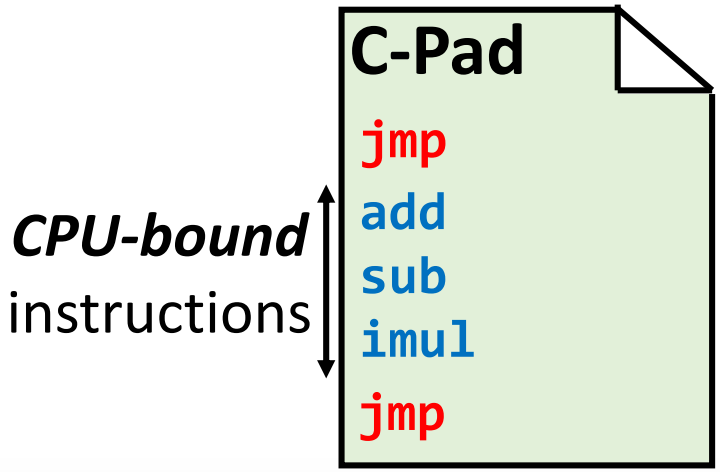
C3. Align branches to/from C-Pad

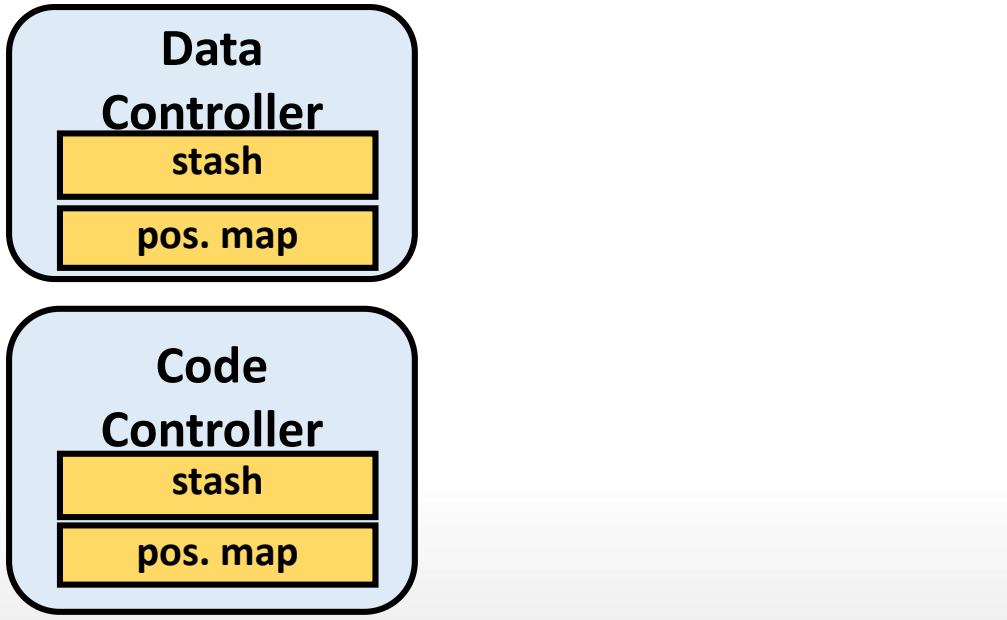
C3. Align branches to/from C-Pad

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

C3. Align branches to/from C-Pad

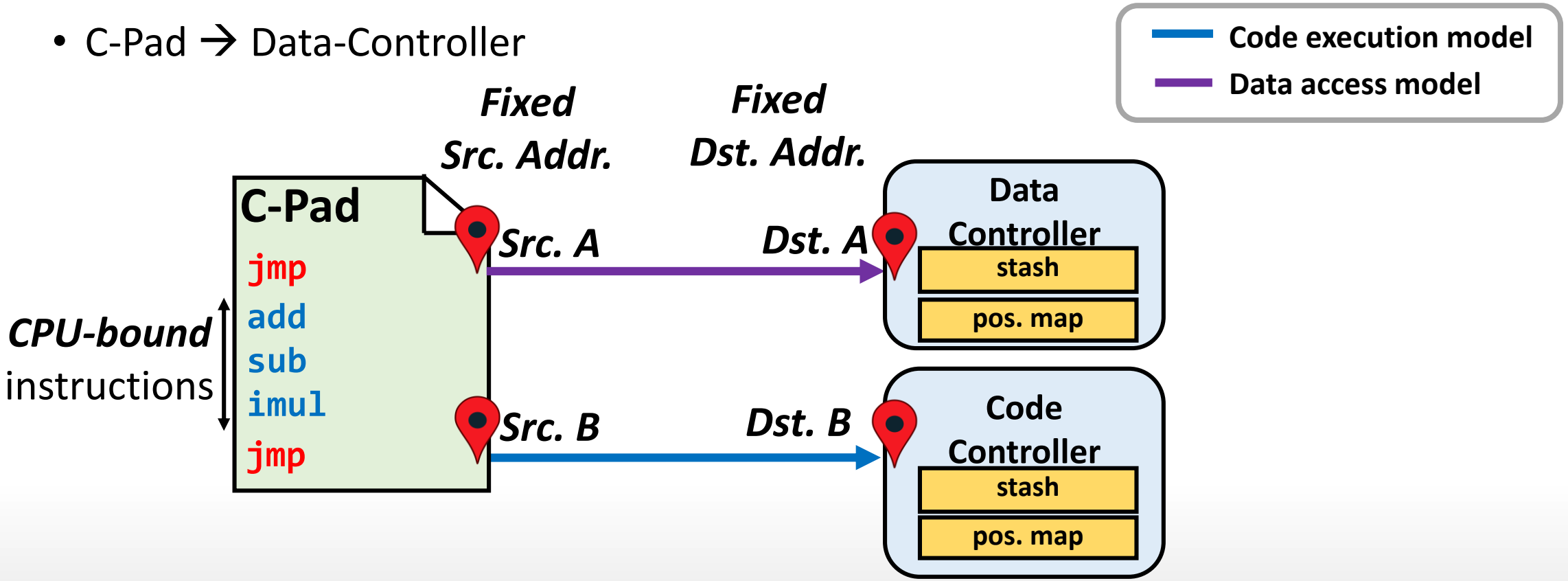
- Each instrumented code block has two branches to fixed locations
  - C-Pad → Code-Controller
  - C-Pad → Data-Controller





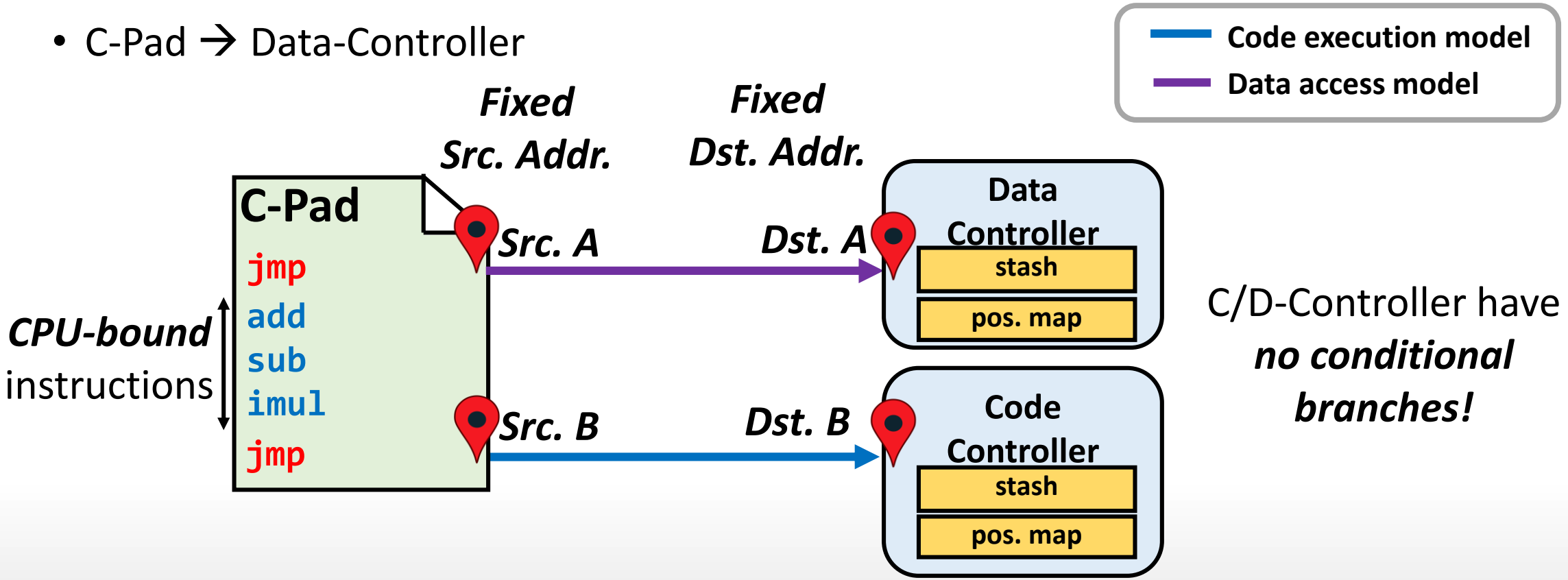
C3. Align branches to/from C-Pad

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



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

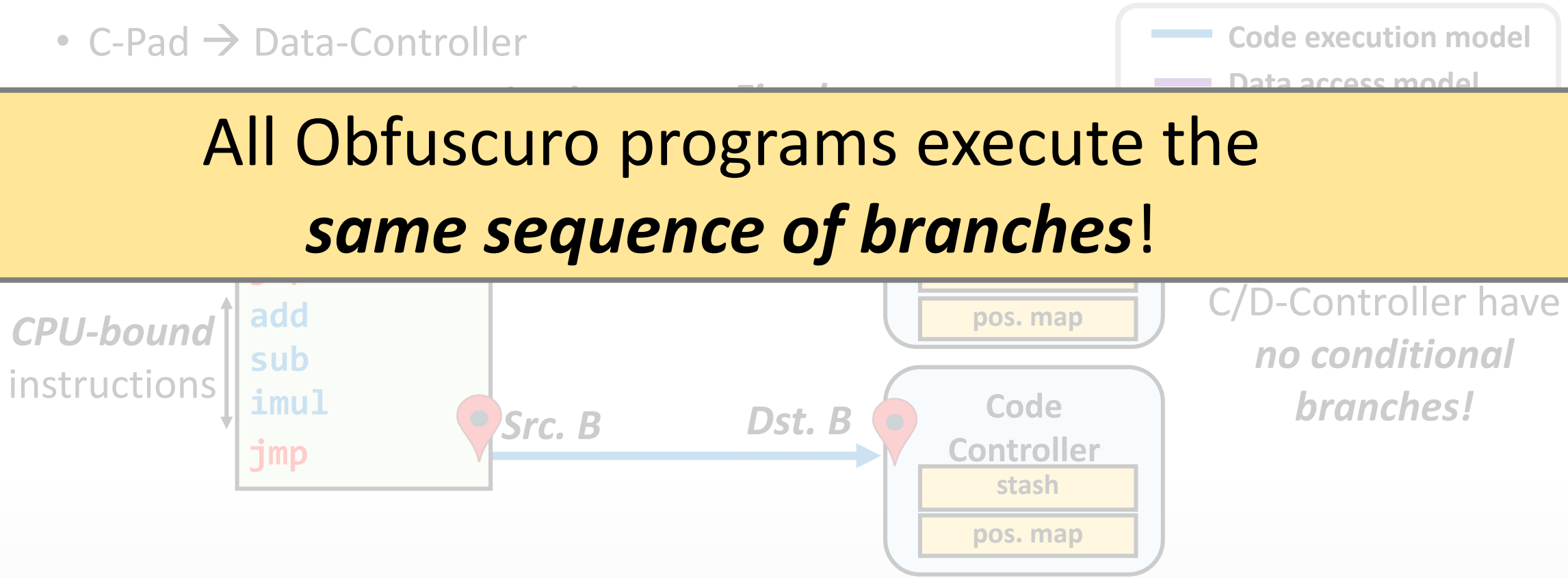


C/D-Controller have ***no conditional branches!***

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***!



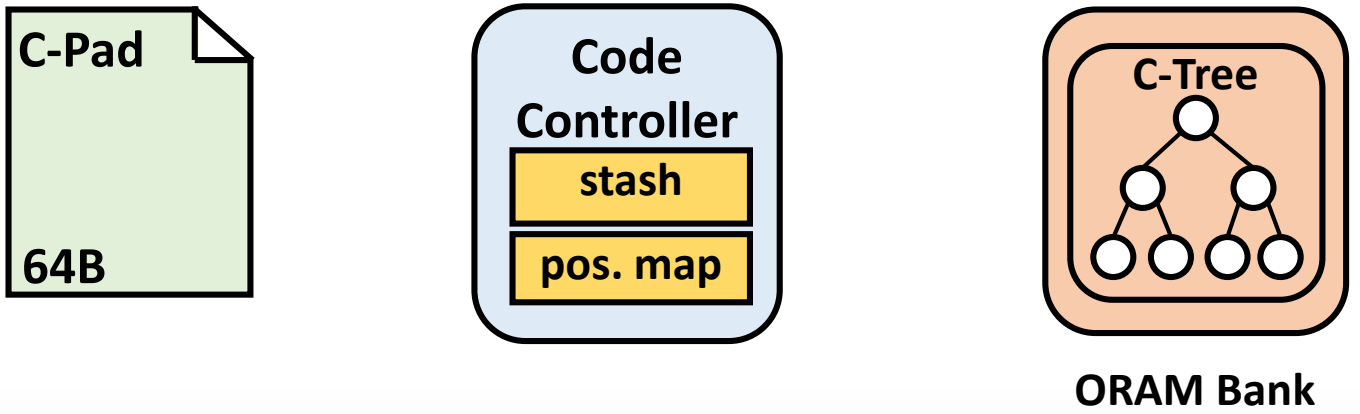
C4. Ensuring execution time consistency

C4. Ensuring execution time consistency

- The program executes fixed number of code blocks

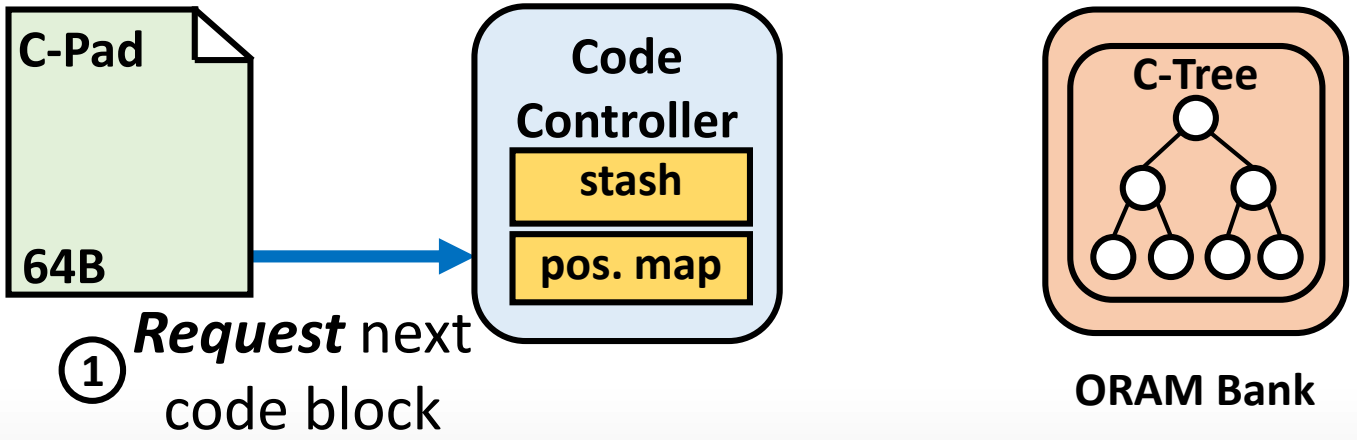
C4. Ensuring execution time consistency

- The program executes fixed number of code blocks



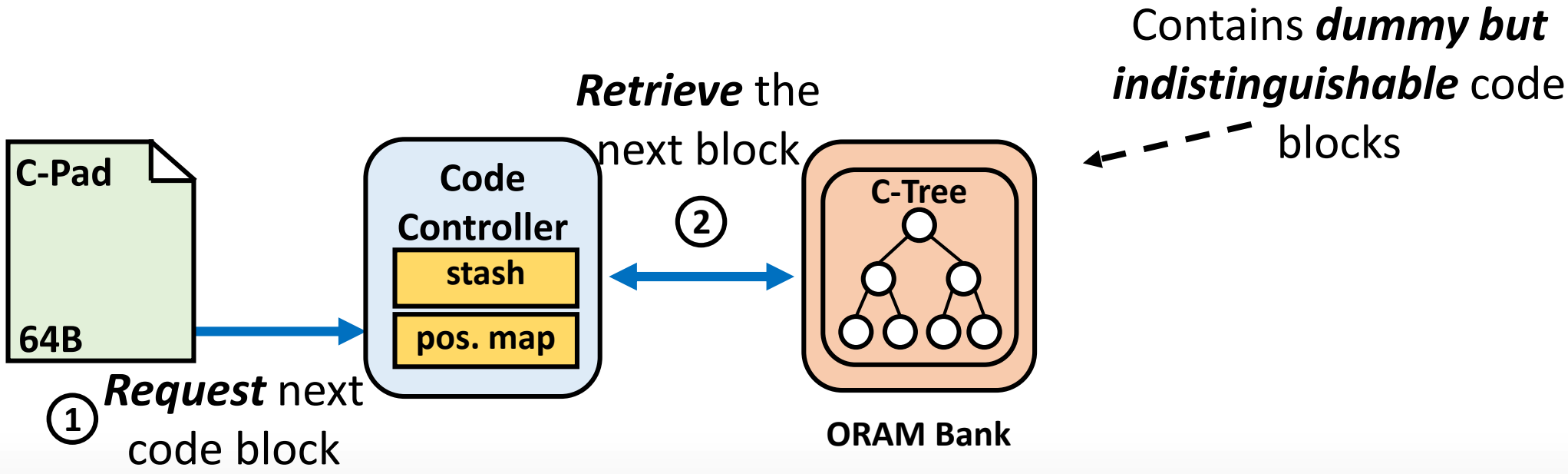
C4. Ensuring execution time consistency

- The program executes fixed number of code blocks



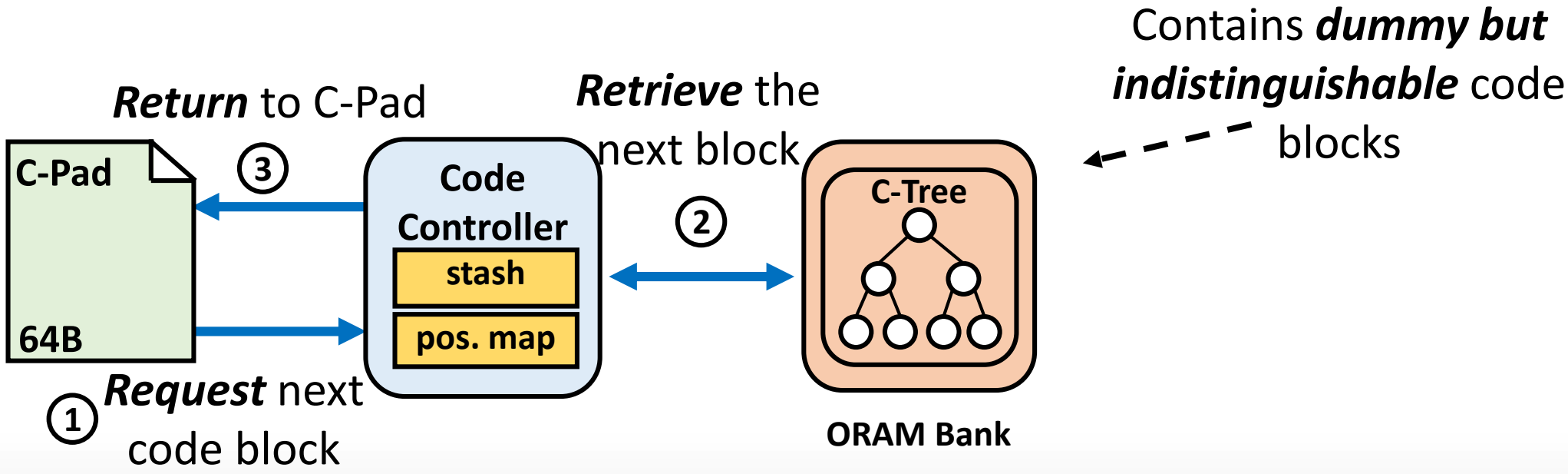
C4. Ensuring execution time consistency

- The program executes fixed number of code blocks



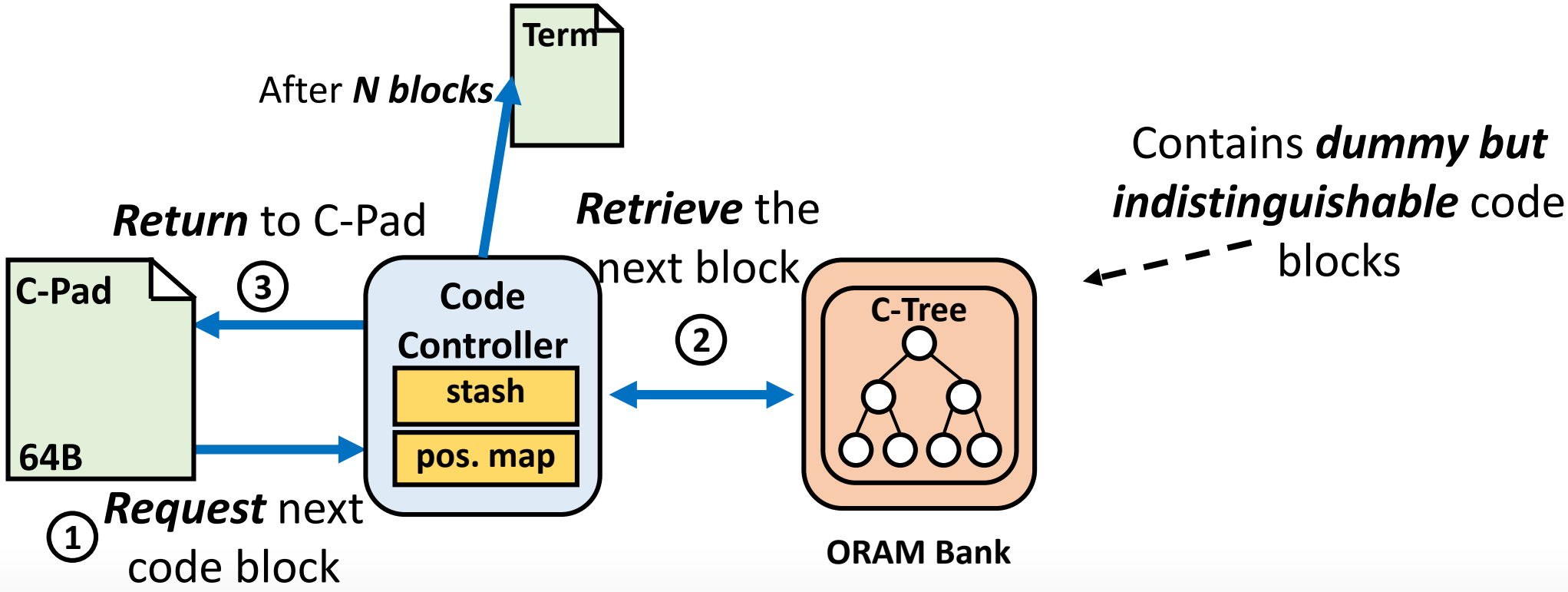
C4. Ensuring execution time consistency

- The program executes fixed number of code blocks



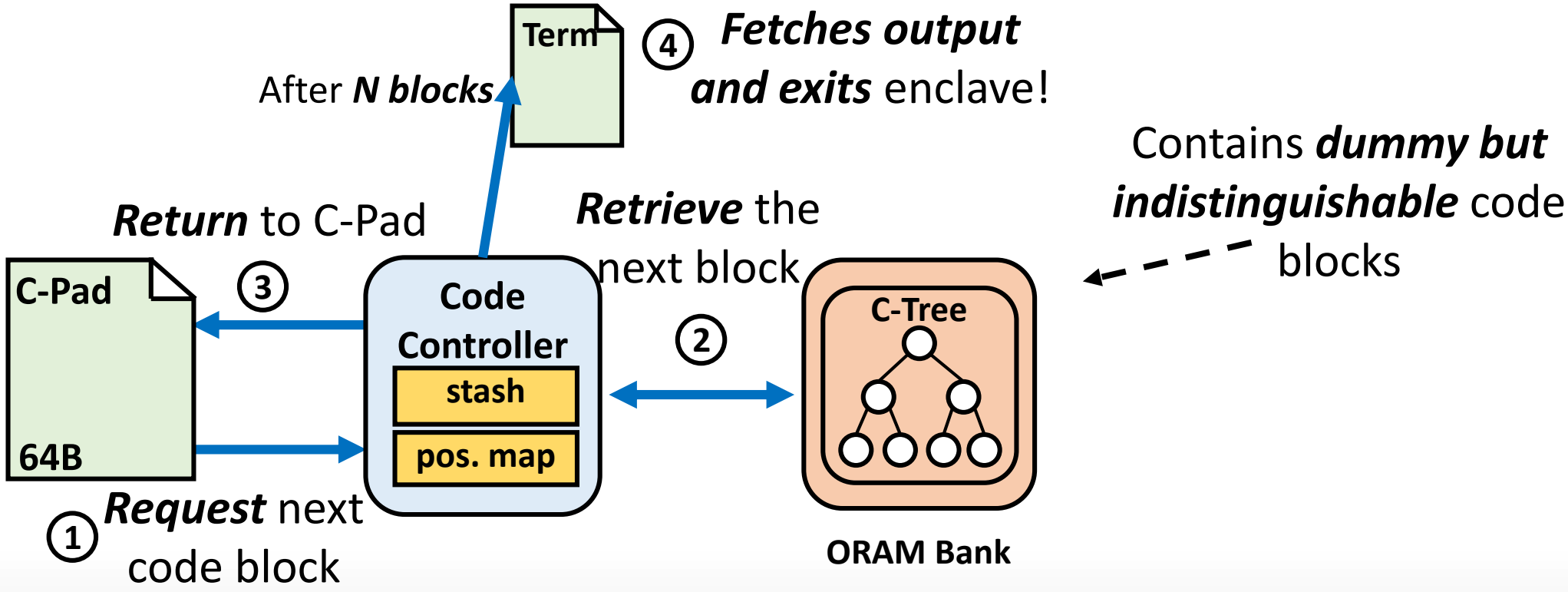
C4. Ensuring execution time consistency

- The program executes fixed number of code blocks



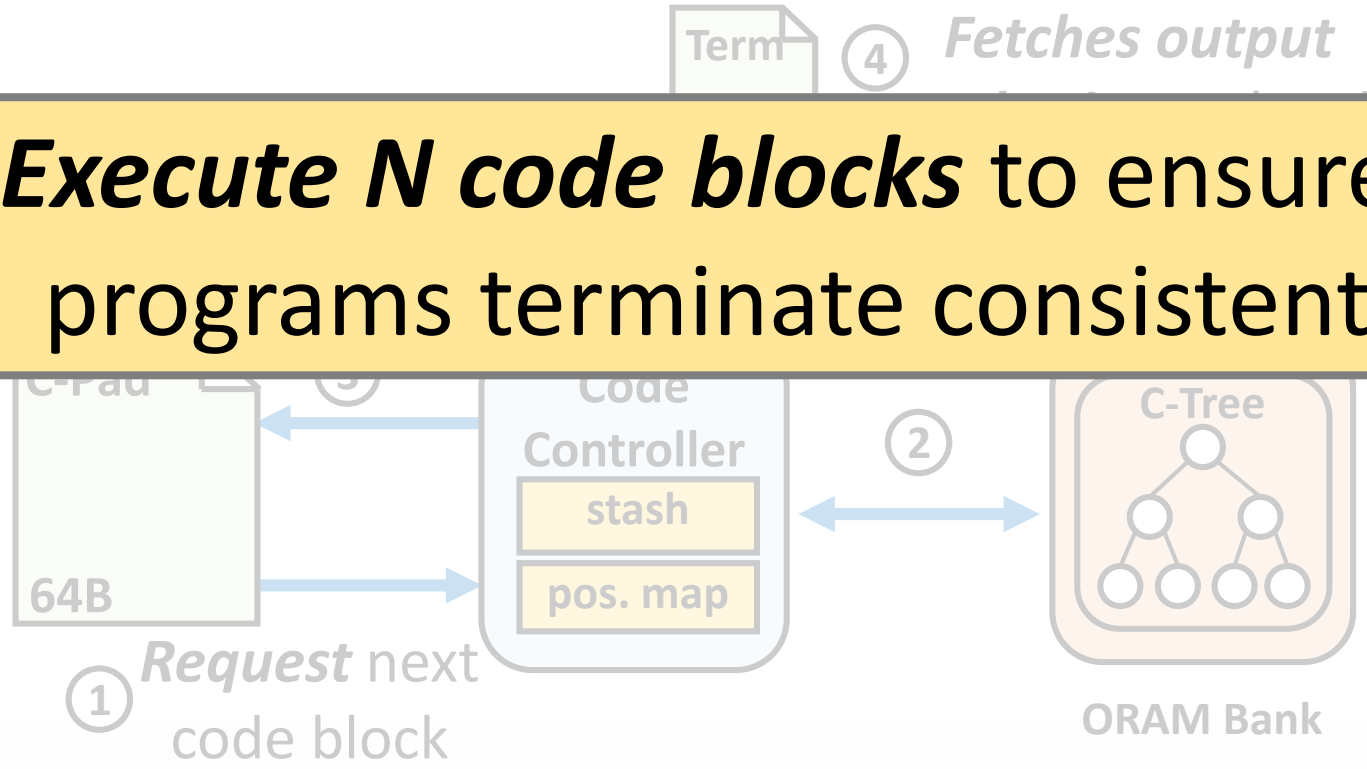
C4. Ensuring execution time consistency

- The program executes fixed number of code blocks



C4. Ensuring execution time consistency

- The program executes fixed number of code blocks



Execute N code blocks to ensure all programs terminate consistently!

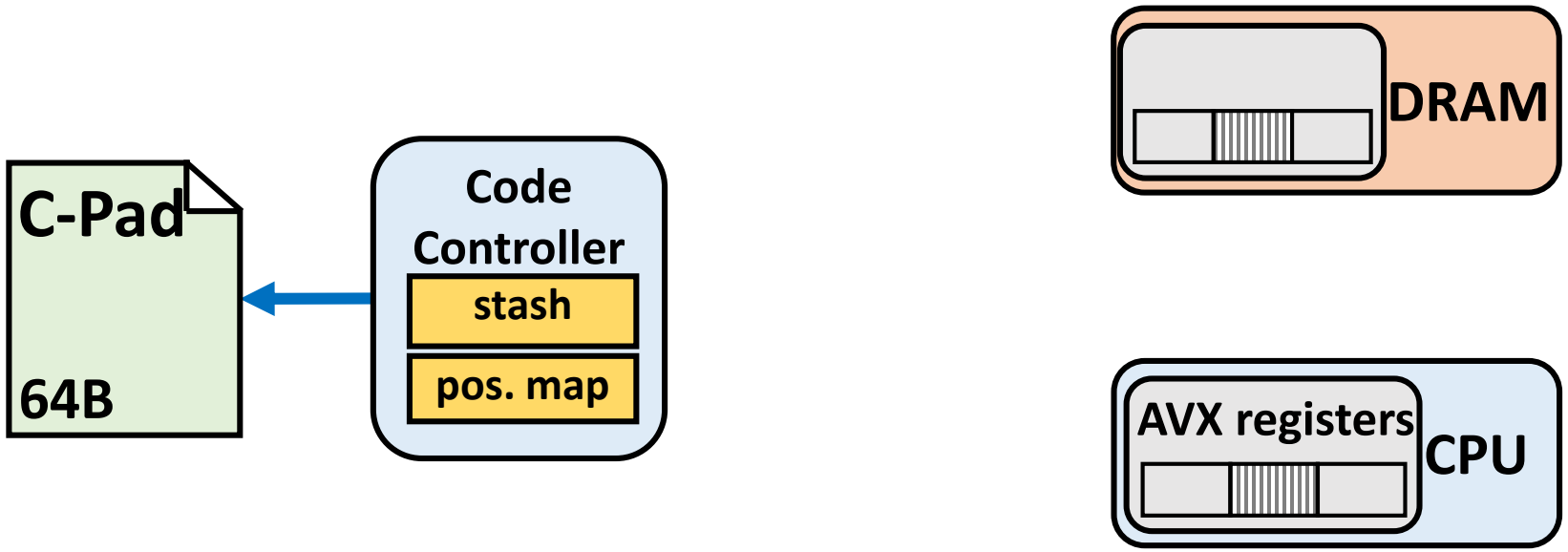
Faster memory store for enclaves

Faster memory store for enclaves

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

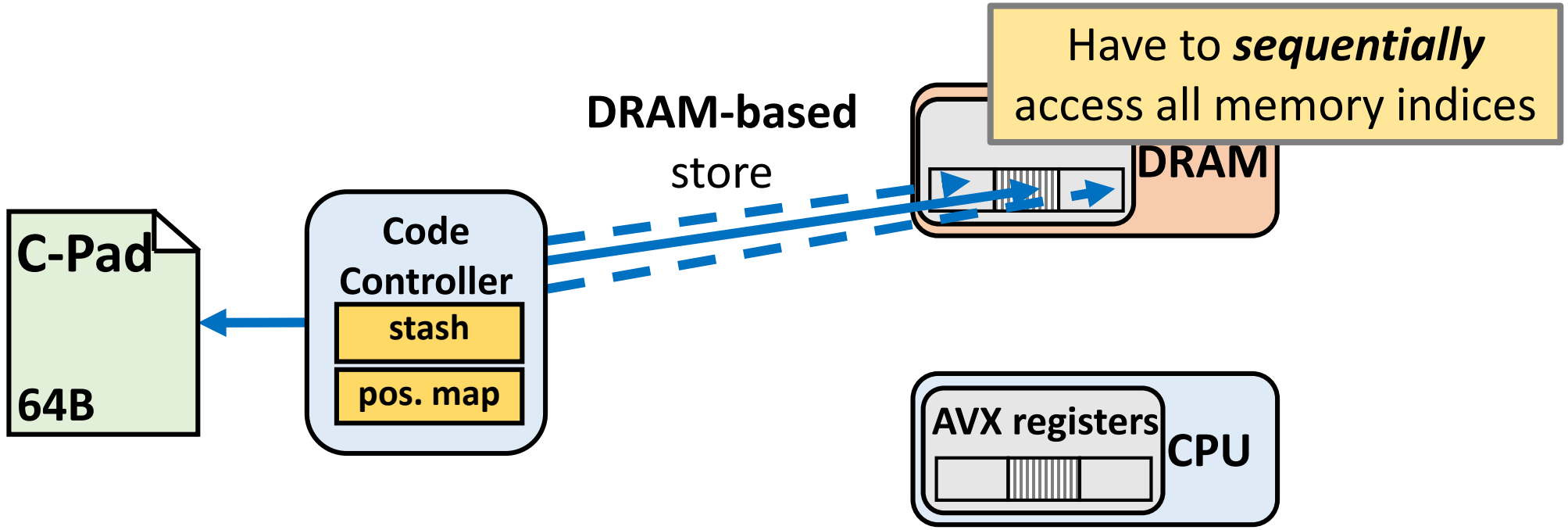
Faster memory store for enclaves

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



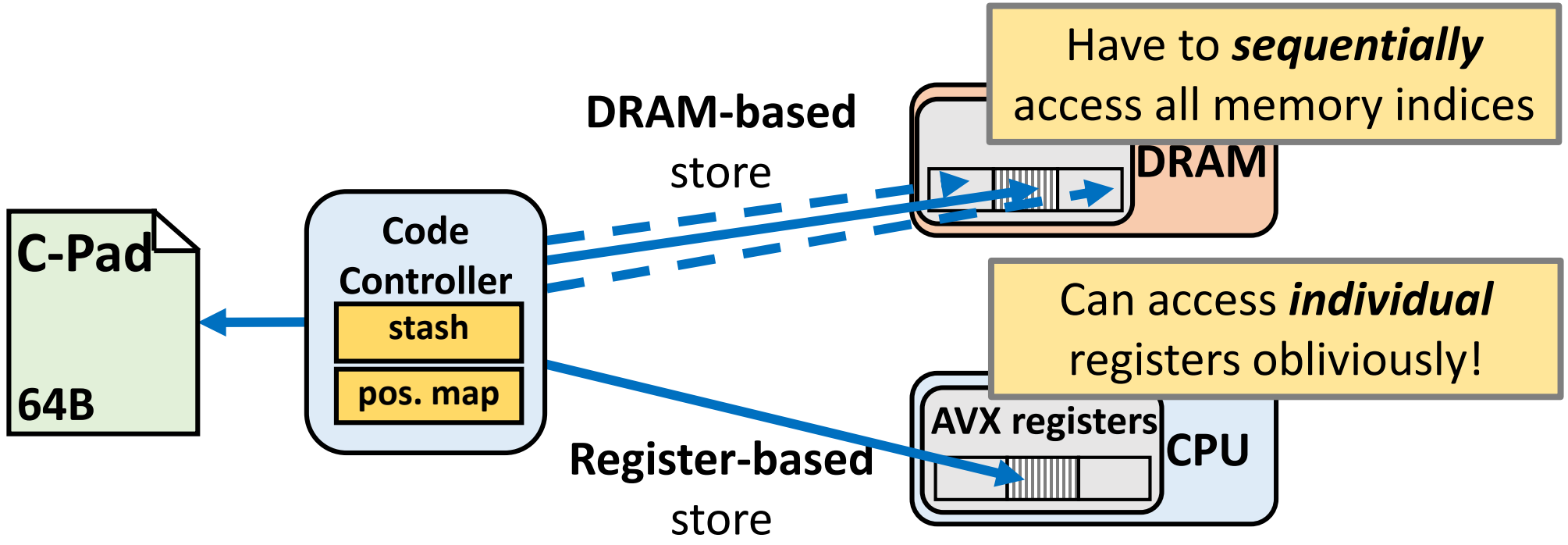
Faster memory store for enclaves

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



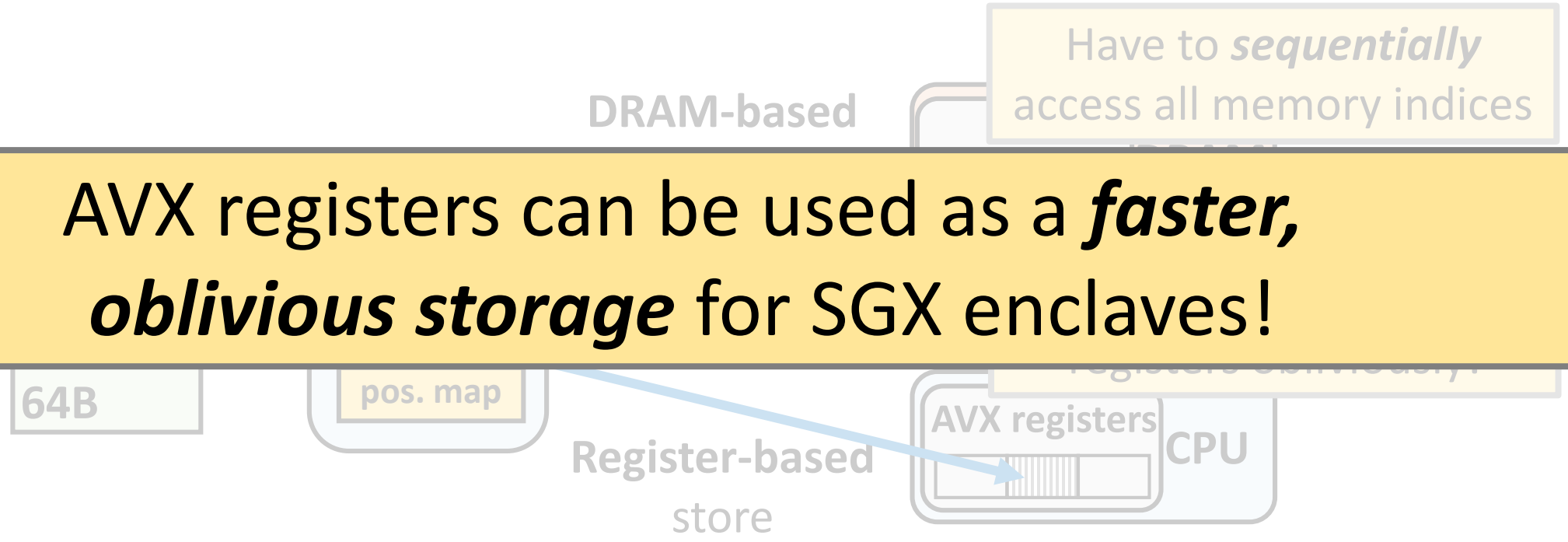
Faster memory store for enclaves

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



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

Implementation

- **LLVM compiler suite (3117 LoC)**
  - Breaks all code into similar blocks **(C1)**
  - Instrument and align all control and data-flow instructions **(C3)**

Implementation

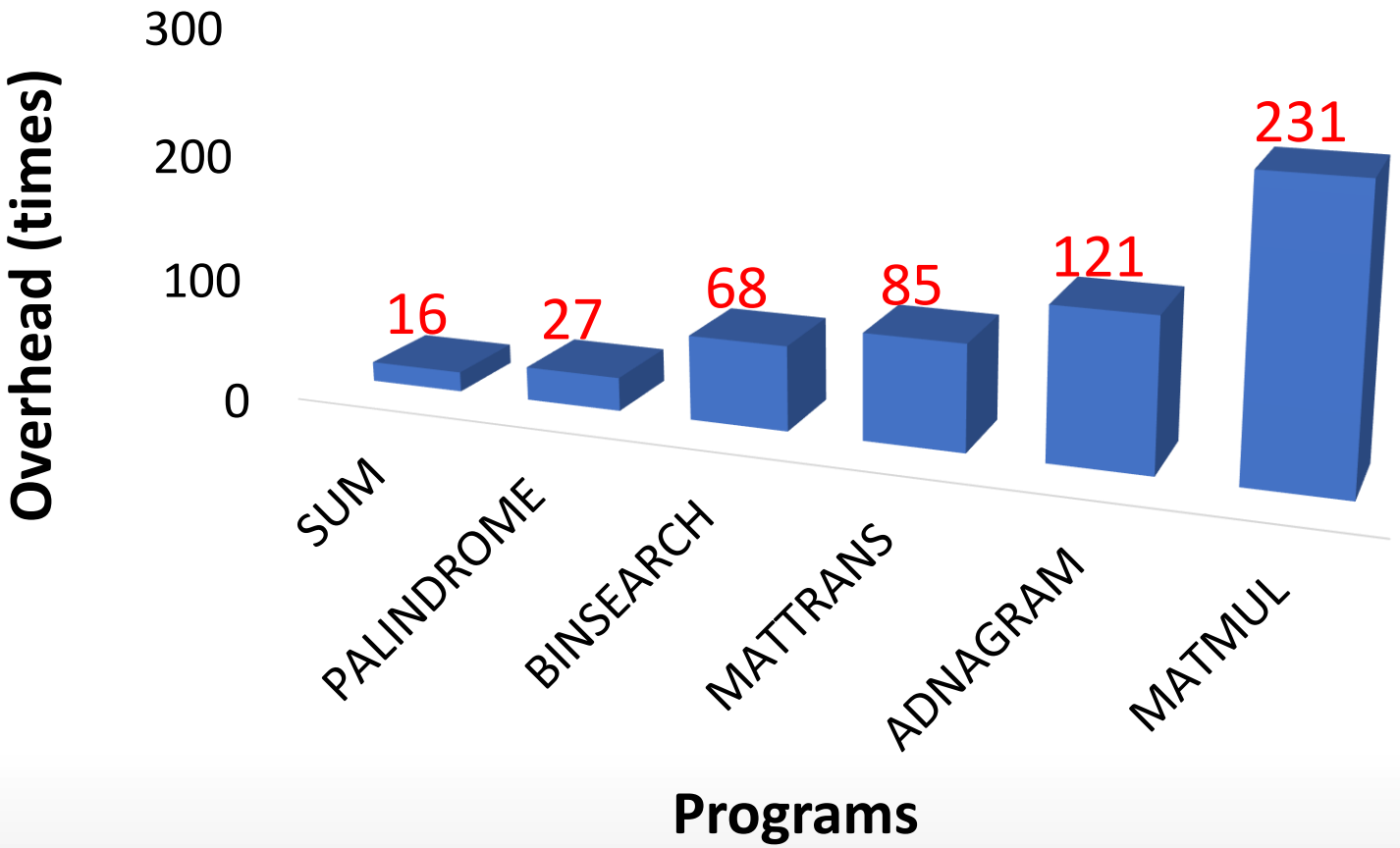
- **LLVM compiler suite (3117 LoC)**
  - Breaks all code into similar blocks **(C1)**
  - Instrument and align all control and data-flow instructions **(C3)**

- **Runtime library (2179 LoC)**
  - Initializes ORAM trees and performs secure ORAM operations **(C2)**
  - Terminate program and fetch output **(C4)**

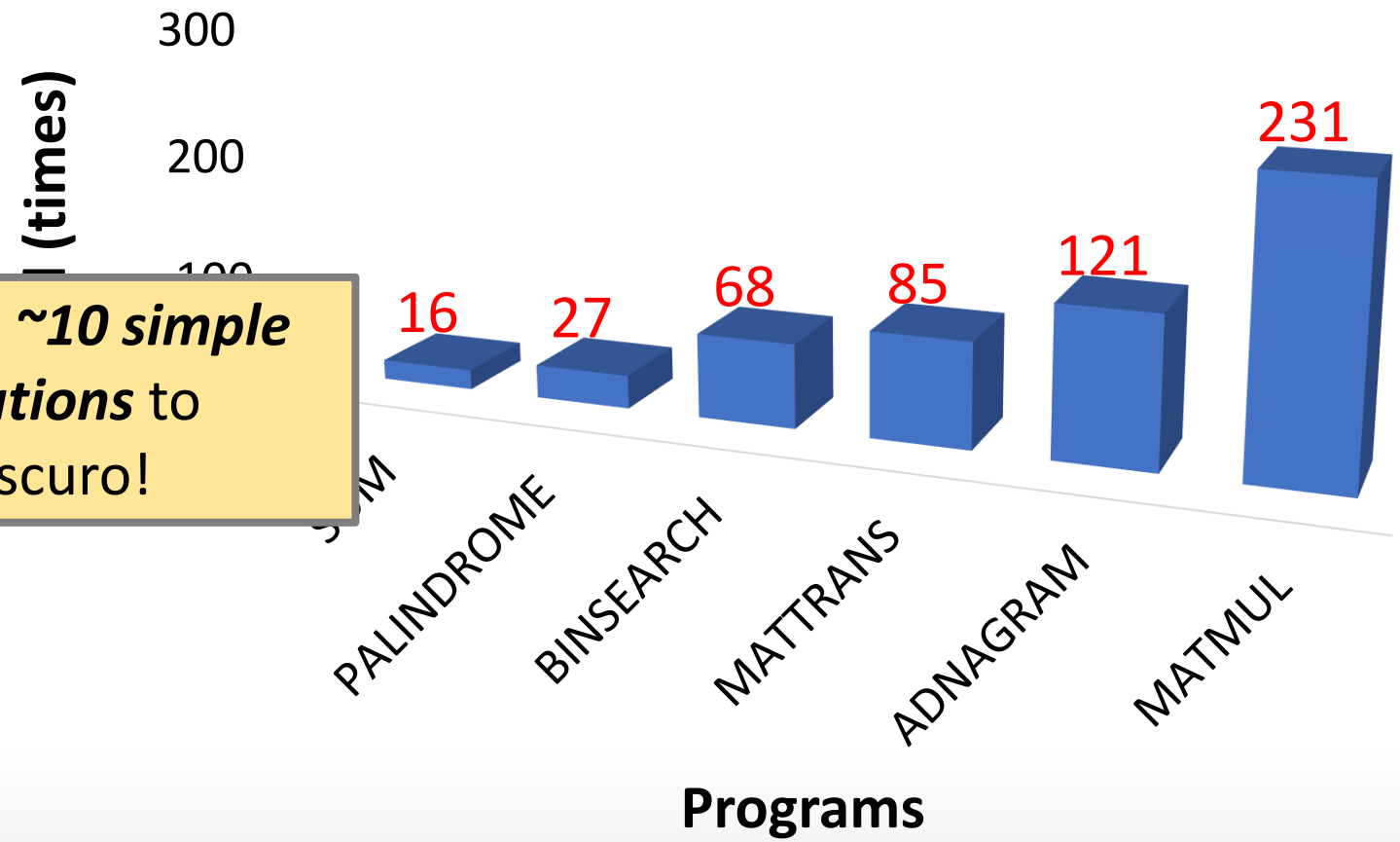
Implementation

- **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



Performance Evaluation

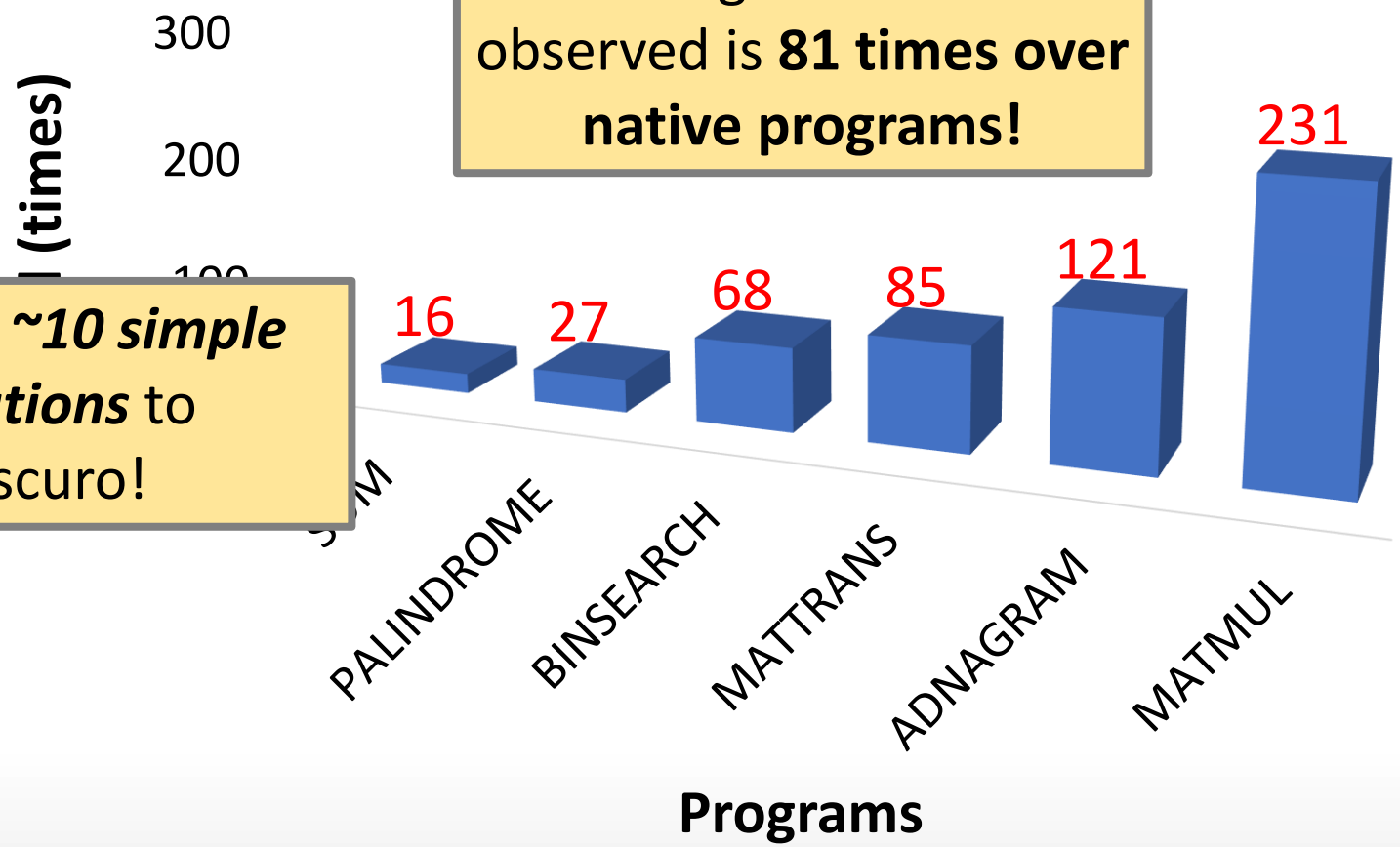


We ported ***~10 simple applications*** to Obfuscuro!

Performance Evaluation

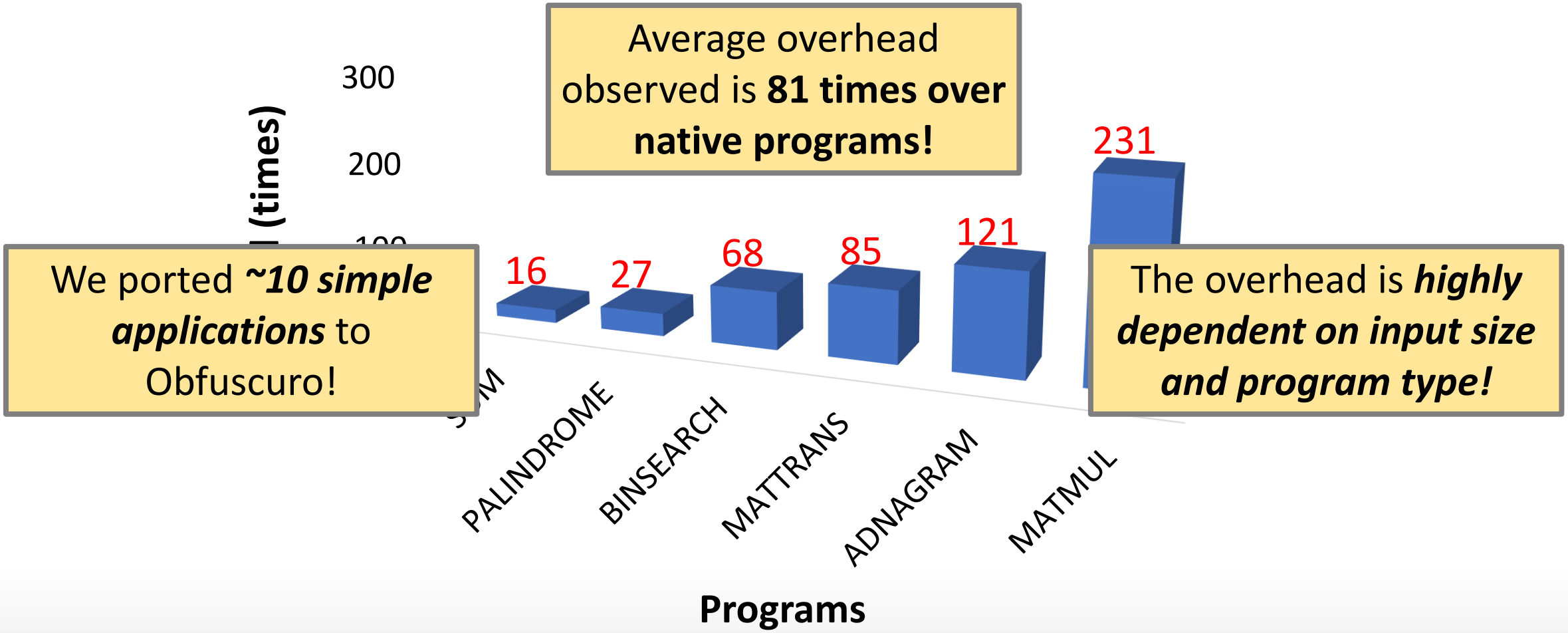
Average overhead observed is **81 times over native programs!**

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



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!

Ending Remarks!

1. Program obfuscation is a ***remarkable dream*** to achieve

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

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

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***

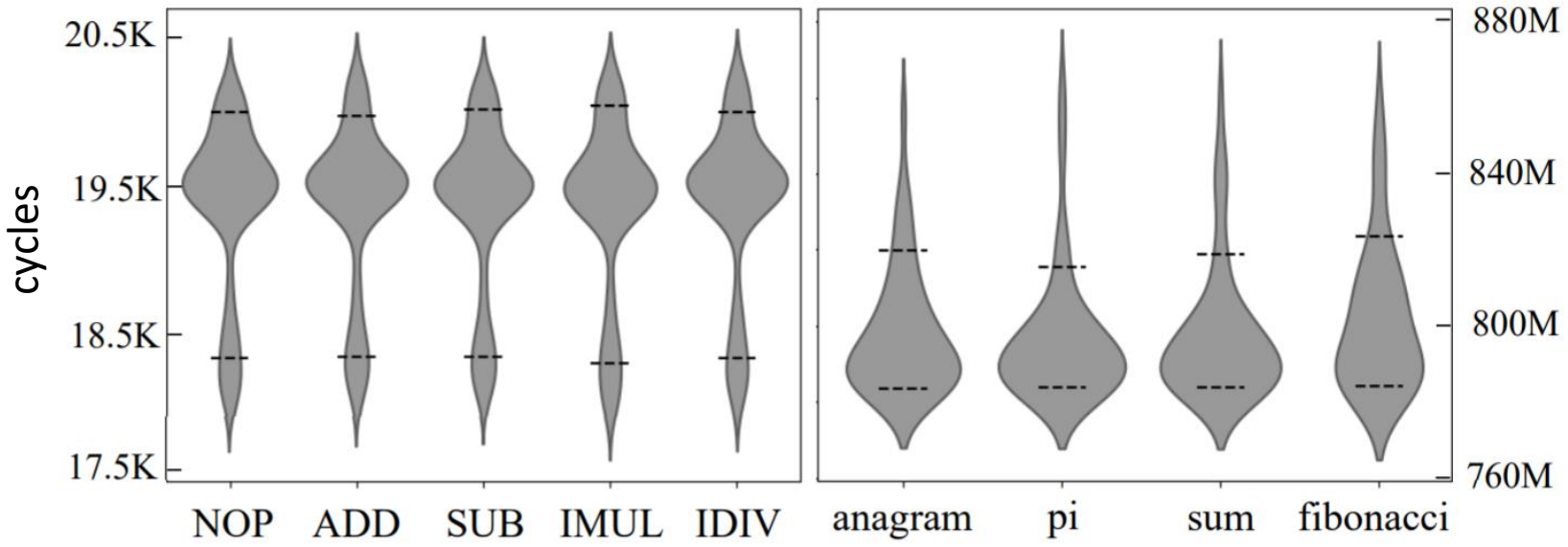
Adil Ahmad

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!