6.888 Secure Hardware Design

Mengjia Yan

Fall 2020

Today's Agenda

- Introduce yourself
- Logistics
- Course Overview

Introduce Yourself

Course Logistics

Basic Administrivia

- Instructor:
  - Mengjia Yan <mengjia@csail.mit.edu>
- TA:
  - Miles Dai <milesdai@mit.edu>
- Mailing List:
  - 6888-fa20-staff@csail.mit.edu

- Website: http://csg.csail.mit.edu/6.888Yan/
  - Paper readings
  - Syllabus
  - Assignments
- Piazza:
  - Announcements
  - Discussions
- HotCRP: Submit paper reviews
- Canvas: Submit project proposals & reports

Course Website

Date	Topic	Speaker	Reading	Notes
9/16 (Wed)	Micro-architectural Side Channel	Mengjia	Kiriansky et al. DAWG: A defense against cache timing attacks in speculative execution processors. MICRO. 2018. **Optional:** Qian et al. A survey of microarchitectural timing attacks and countermeasures on contemporary hardware. Journal of Cryptographic Engineering (2018).	
9/21 (Mon)	Traditional Side Channels	Mengjia	Percival, Colin. Cache missing for fun and profit. (2005). **Optional:** Yarom et al. FLUSH+RELOAD: a high resolution, low noise, L3 cache side-channel attack. USENIX Security. 2014. Liu et al. Last-level cache side-channel attacks are practical. S&P, 2015.	Lab Assigned
9/23 (Wed)	Transient Execution Attacks	Mengjia	Kocher et al. Spectre attacks: Exploiting speculative execution. S&P. 2019. **Optional:** Canella et al. A systematic evaluation of transient execution attacks and defenses. USENIX Security. 2019.	
9/28 (Mon)	Transient Execution Defenses		Yu et al. Speculative Taint Tracking (STT) A Comprehensive Protection for Speculatively Accessed Data. MICRO. 2019. **Optional:** Guarnieri et al. Hardware-Software Contracts for Secure Speculation. arXiv preprint. 2020.	
9/30 (Wed)	Hardware to Enforce Non-interference		Tiwari et al. Complete information flow tracking from the gates up. ASPLOS. 2009. **Optional:** Ferraiuolo et al. HyperFlow: A processor architecture for nonmalleable, timing-safe information flow security. CCS. 2018.	

Pre-requisites and Recommendation

- Pre-requisite:
  - Basic computation structure course (6.004)

- Recommended but not required
  - System security and software security courses (6.858, 6.857)
  - Advanced computer architecture course (6.823)
  - Basic applied cryptography (6.875)

Assignments and Grading

- Paper reviews (2 papers/week) - 25%
  - 500 word summary + 1-2 discussion questions
- Seminars - 15%
  - Discussion lead for 1-2 papers - 10%
  - Participation - 5%
- Lab assignments - 15%
- Research project - 50%
  - Proposal – 10%
  - Weekly report + Checkpoint – 10%
  - Final report – 15%
  - Final presentation – 15%

Seminar Format

- Every student will write a review for each paper
  - 500 word summary, comments on pros and cons, and key takeaways
  - 1-2 discussion questions
  - Due @midnight before each class
  - Submit via HotCRP (visible after the due time)

- Each paper will have one student as the lead presenter
  - ~45 min presentation: A good opportunity to practice presentation skills
  - Send slides to me 24 hours before the lecture
  - Design a poll question
  - *I may invite the authors of the paper to attend the presentation (opportunities to ask questions that only the authors can answer)*

Presentation Format

- Background and Motivation
- Threat Model
- Key technical ideas (*insights*), main contributions
- Strengths/Weaknesses
- Directions for future work
- Several questions for discussion

Lab Assignments (3.5 weeks)

- Team of 2 persons
  1) Dead drop: Build a communication channel via hardware resource contention
  2) Capture the flag: Steal a secret via hardware resource contention

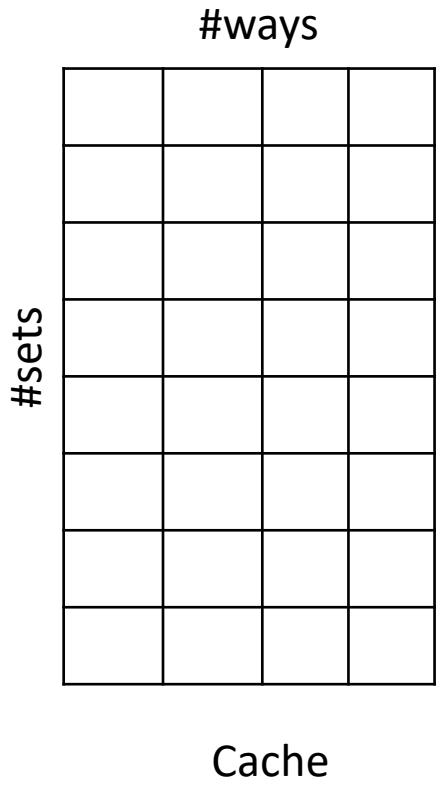
- Opportunities to turn into final projects

Dead Drop

- Communicate via hardware resource contention

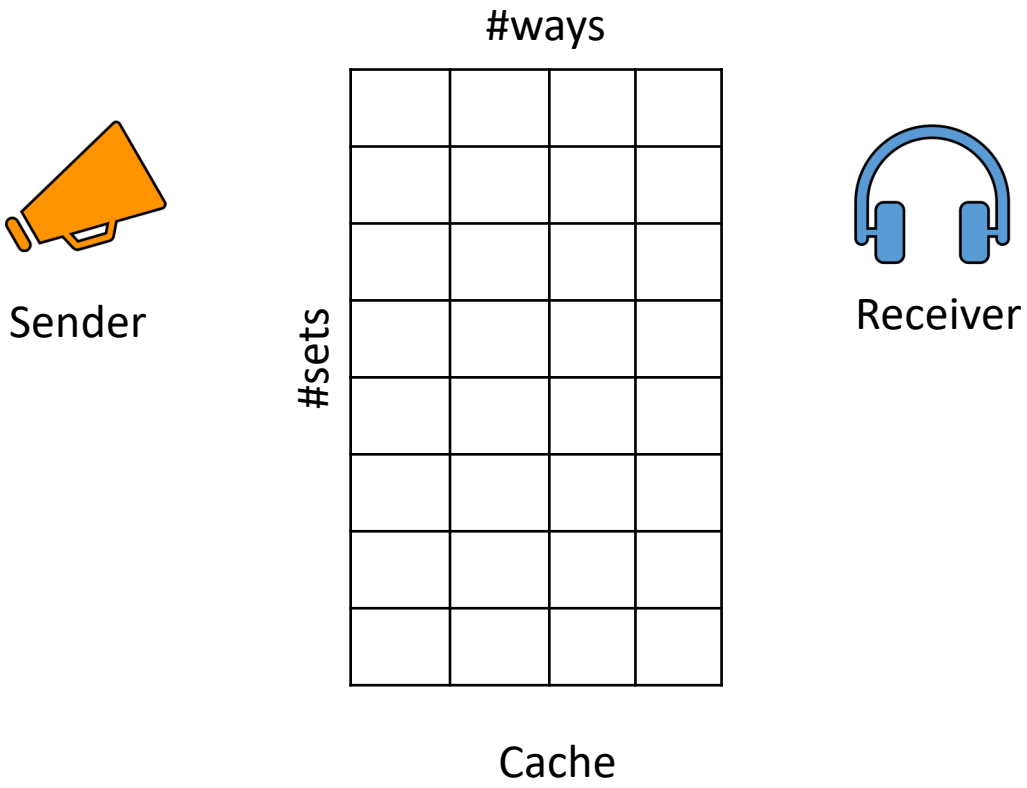
Dead Drop

- Communicate via hardware resource contention



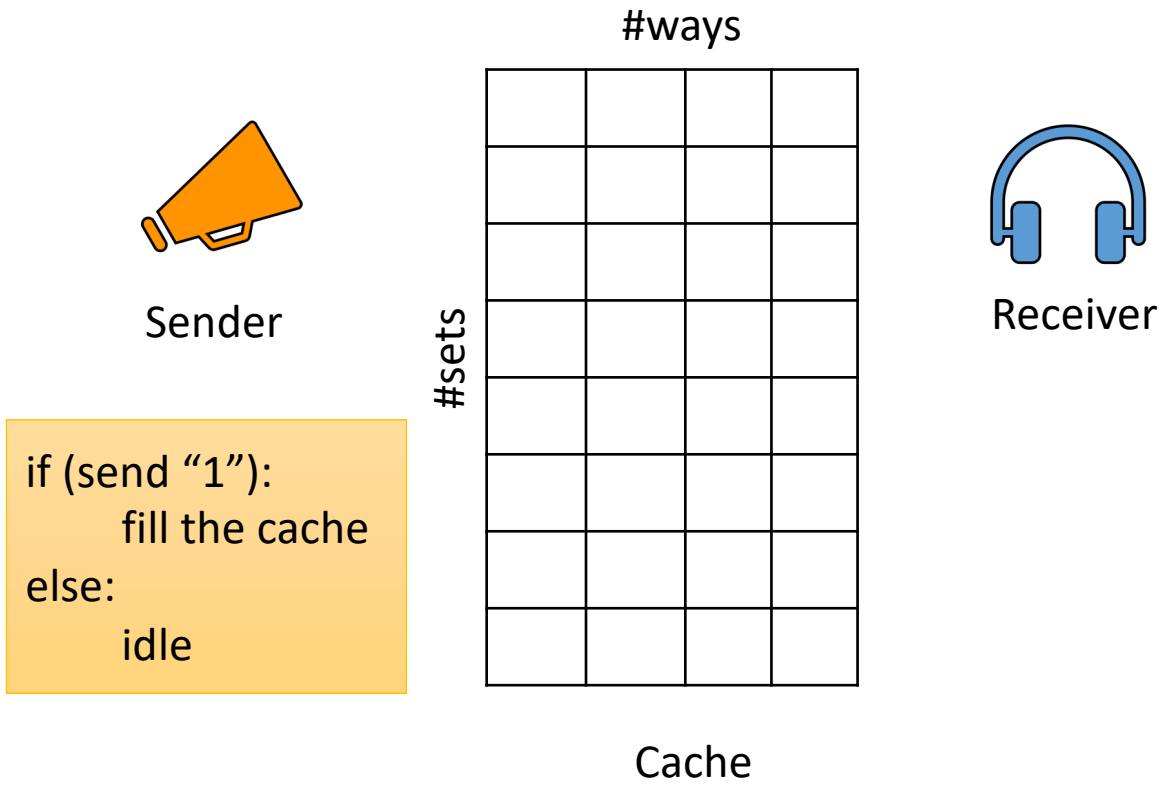
Dead Drop

- Communicate via hardware resource contention



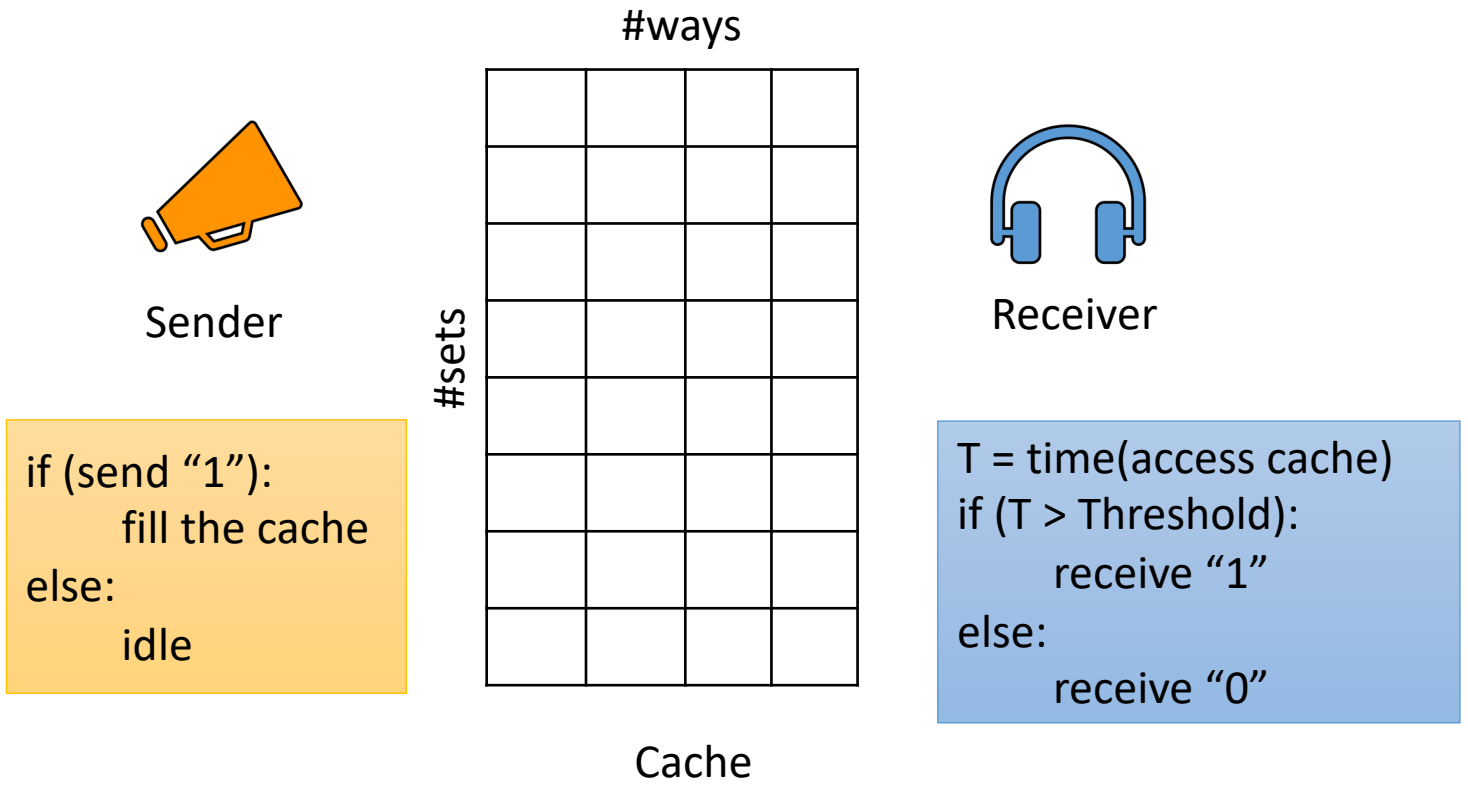
Dead Drop

- Communicate via hardware resource contention



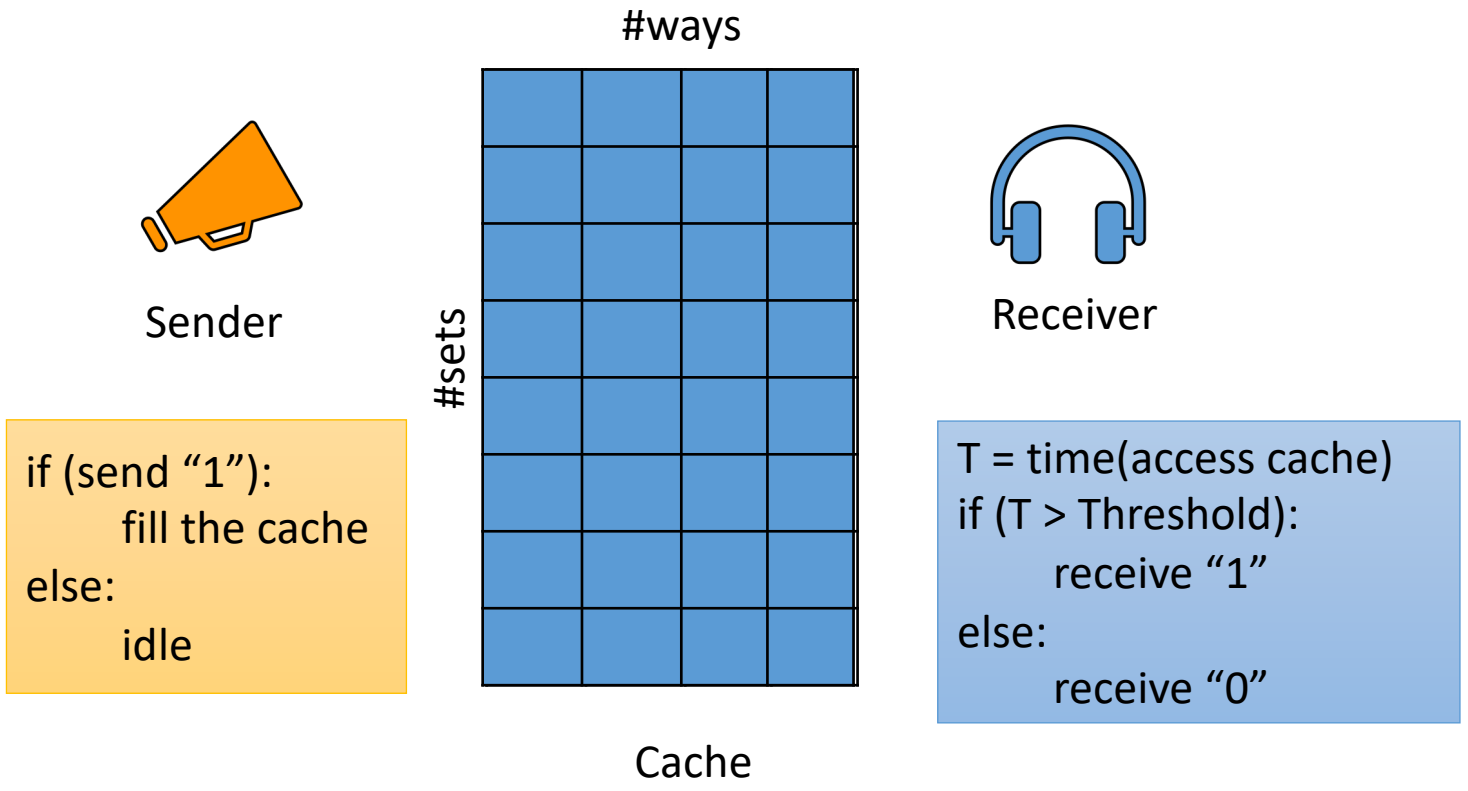
Dead Drop

- Communicate via hardware resource contention



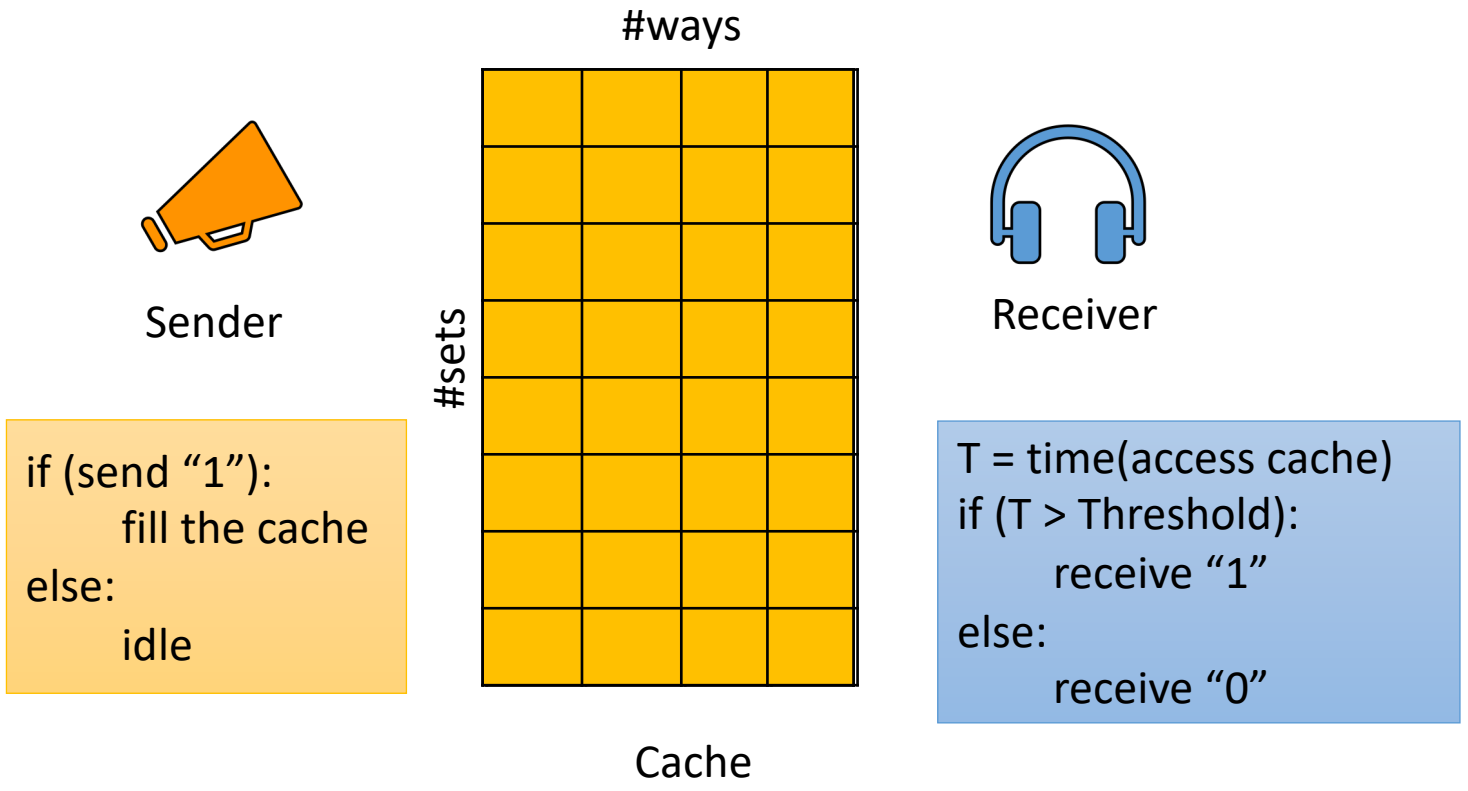
Dead Drop

- Communicate via hardware resource contention



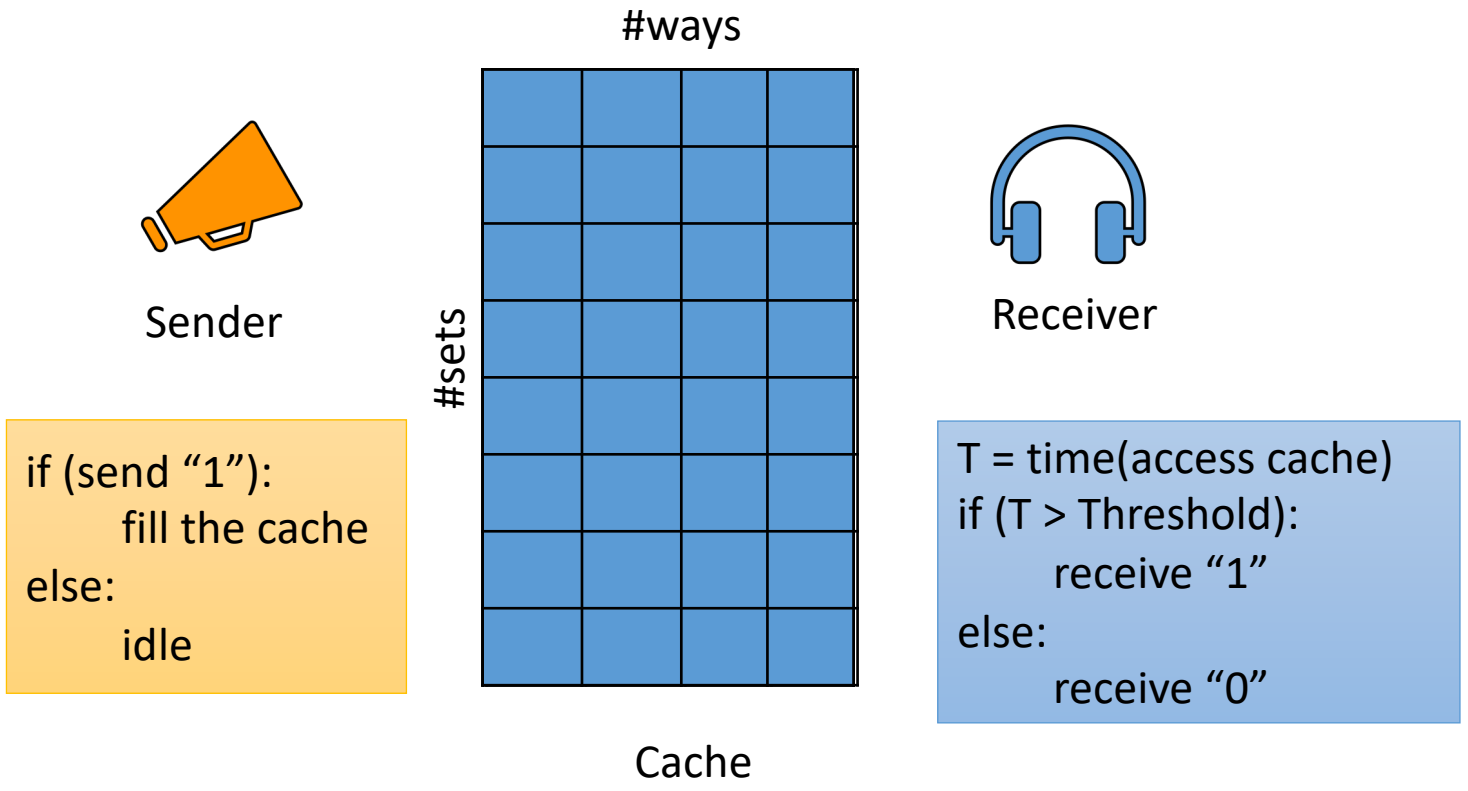
Dead Drop

- Communicate via hardware resource contention



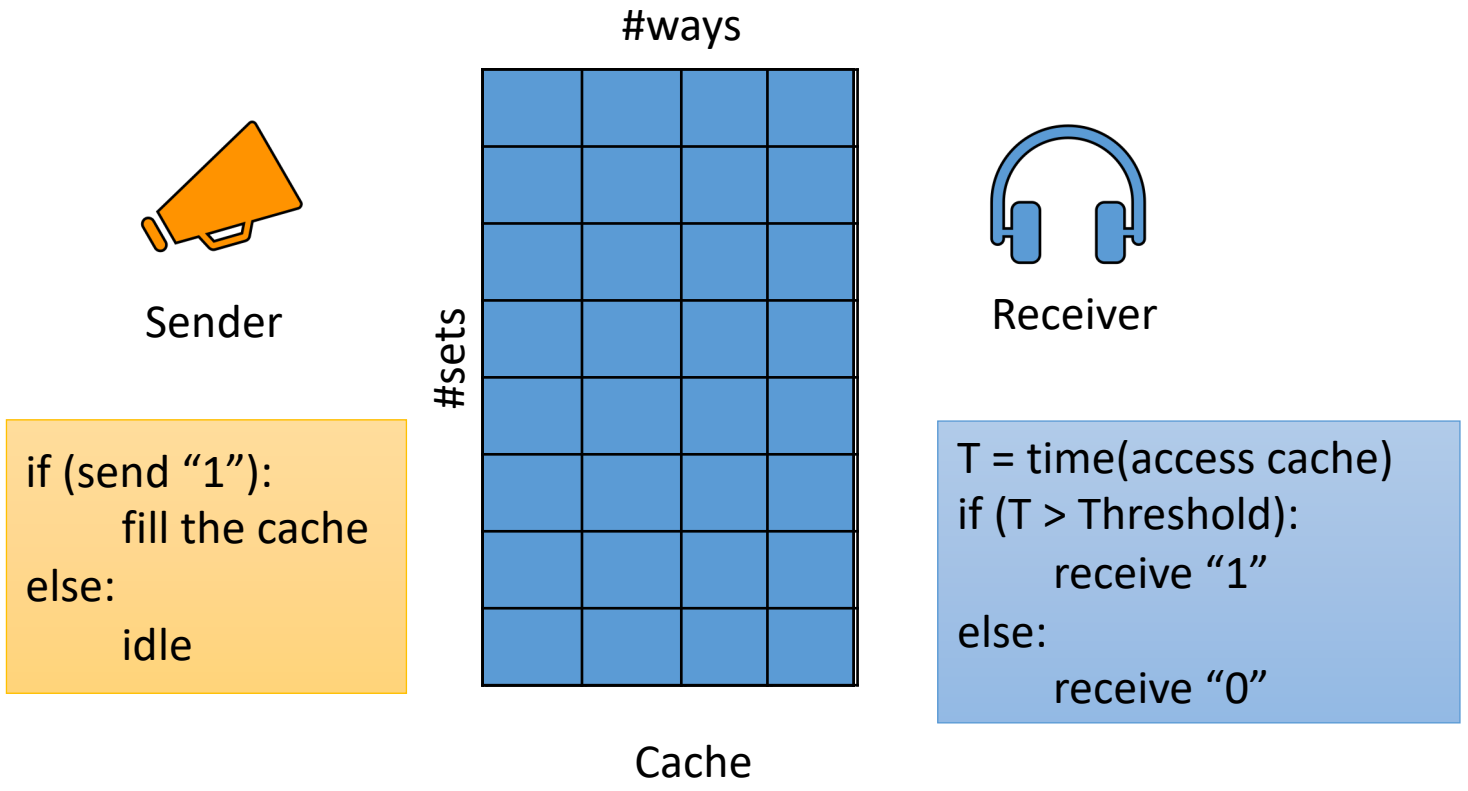
Dead Drop

- Communicate via hardware resource contention



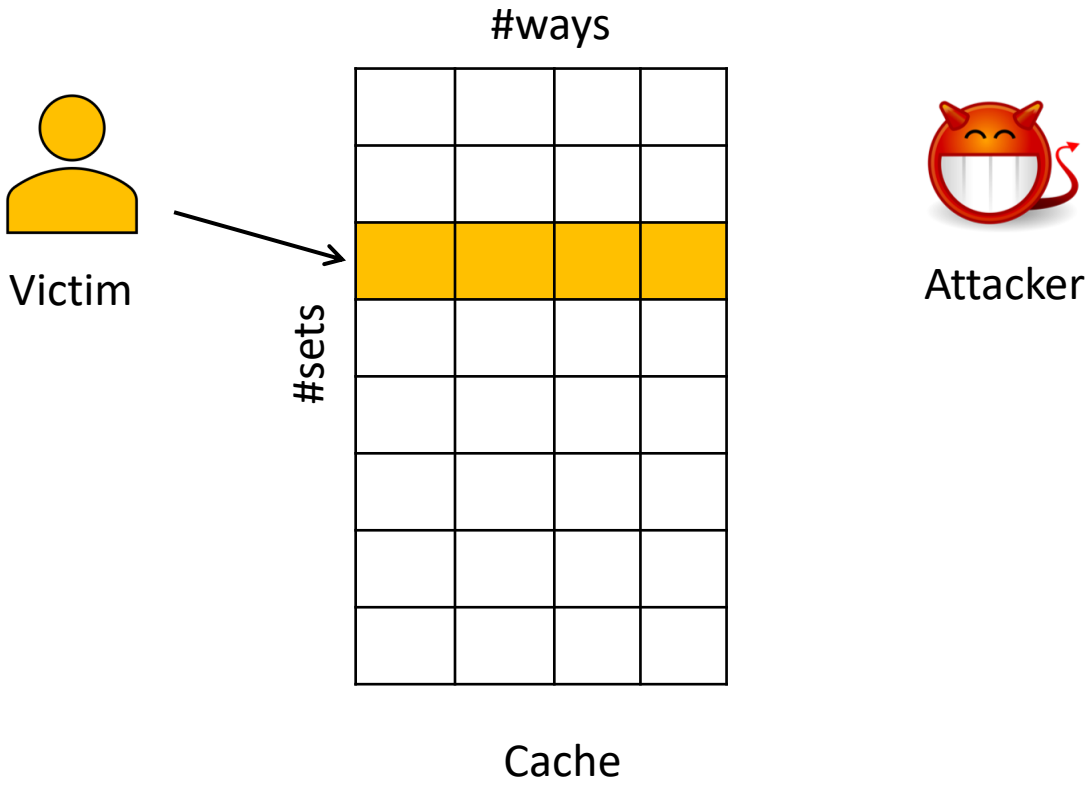
Dead Drop

- Communicate via hardware resource contention



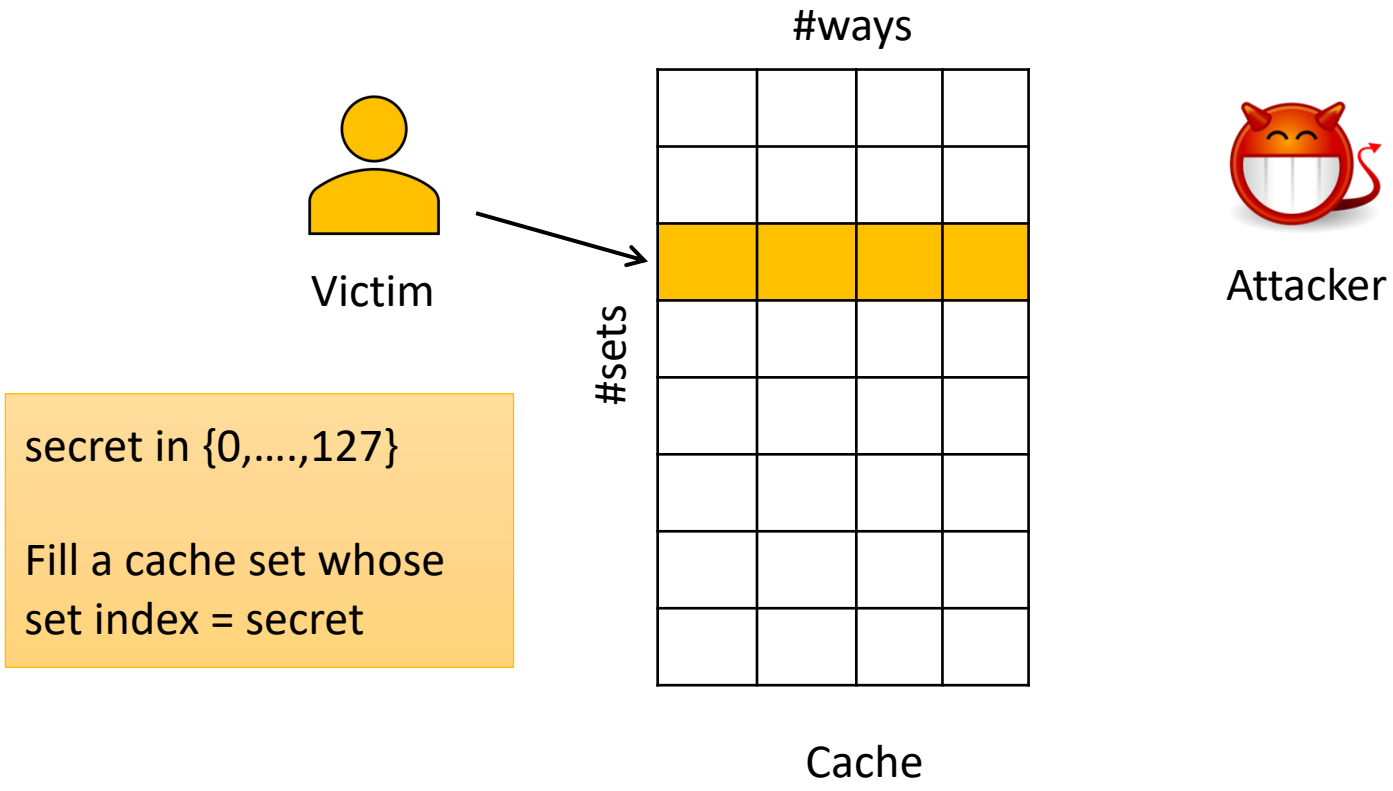
Capture the Flag

- Steal secrets via hardware resource contention



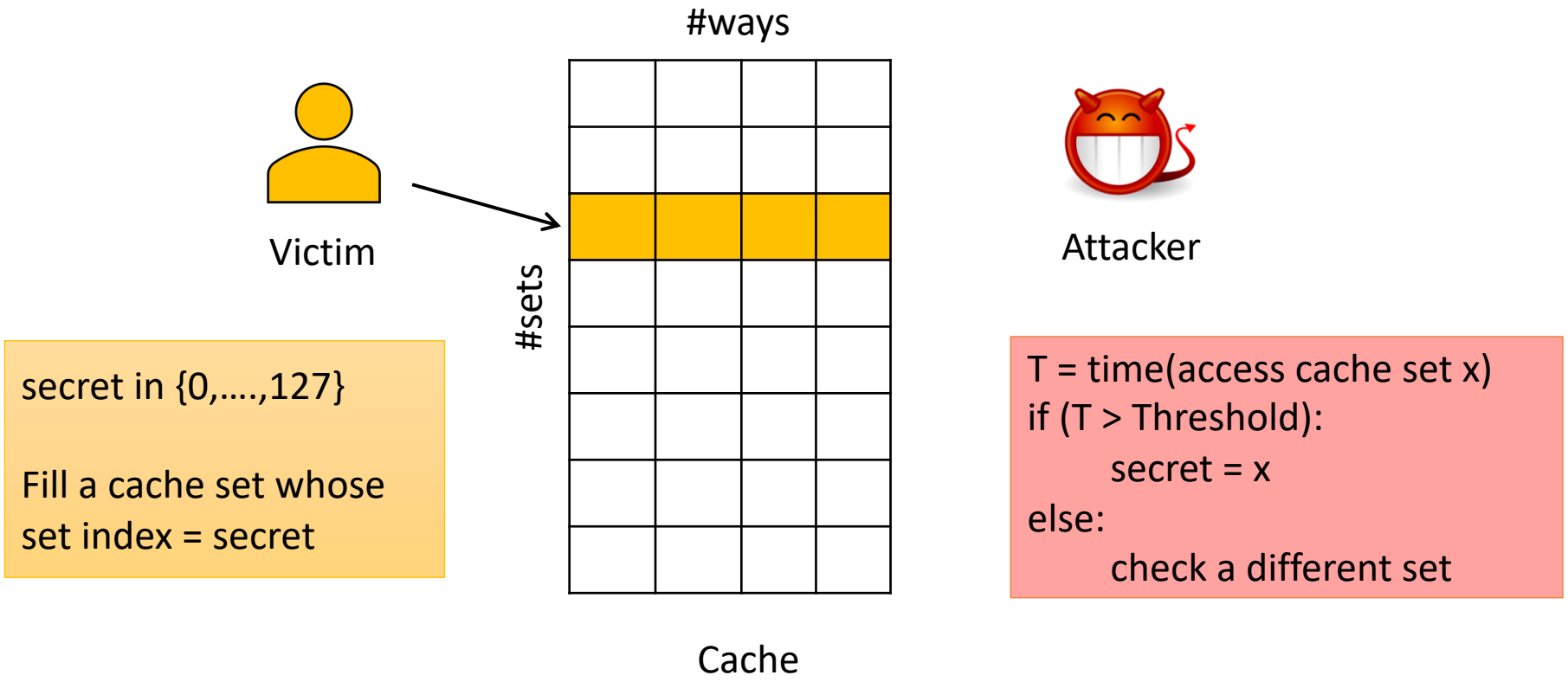
Capture the Flag

- Steal secrets via hardware resource contention



Capture the Flag

- Steal secrets via hardware resource contention

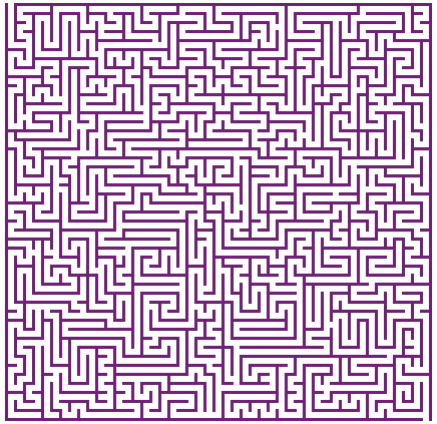


Final Project (8 weeks)

- Original research project
- Solo or 2 person groups
- Deliverables
  - Proposal (schedule pre-proposal meetings with me)
  - Weekly report (short and informal) + Checkpoint (5 min presentation)
  - Final report + Final presentation
- Open-ended topics
  - Must have some hardware security angle

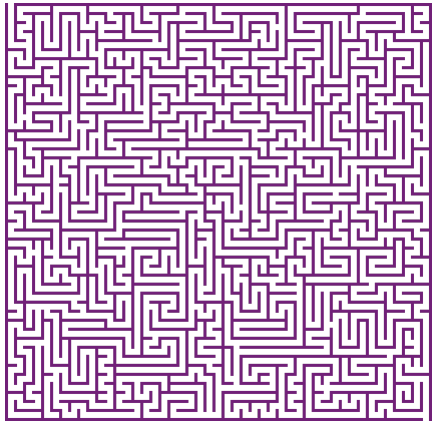
Hardware Security: The Evil and The Good

- Attack modern processors
  - To thoroughly understand HW vulnerabilities

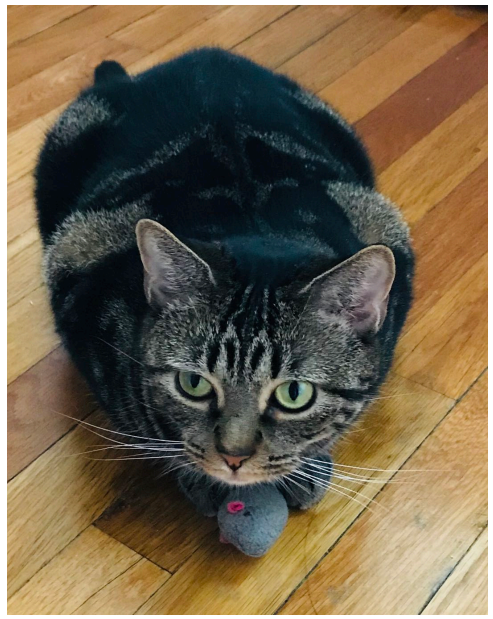


Hardware Security: The Evil and The Good

- Attack modern processors
  - To thoroughly understand HW vulnerabilities



- Secure computation on HW
  - e.g., data oblivious abstraction, enclave abstraction



Course Project Examples

{Attacks, Defenses} x {Theory, Practice}

- Attack + Practice
  - Discover an exploit in existing processors or existing applications
- Attack + Theory
  - What architectural principles fundamentally leak what degree of privacy
- Defense + Practice
  - Mitigate an existing threat using SW/HW
- Defense + Theory
  - Mitigate broad classes of present+future threats

Collaboration Policy and Warning

- Discussions are always encouraged.
- You should carefully acknowledge all contributions of ideas by others, whether from classmates or from sources you have read.
- MIT academic integrity guidelines

Warning

- Please don’t attack other people’s computers or information without their prior permission.
- MIT network rules

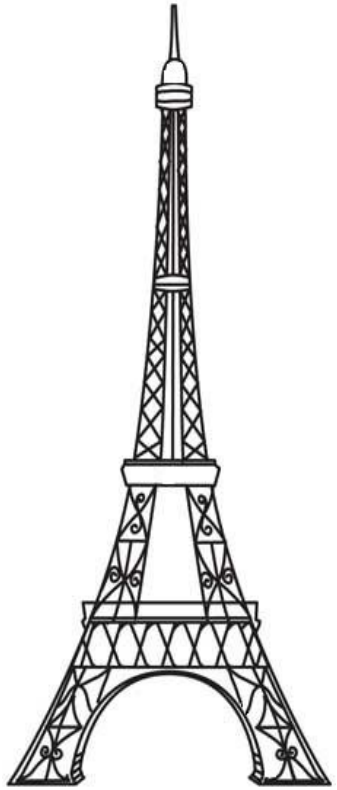
TODO Today

- Check the paper list on http://csg.csail.mit.edu/6.888Yan/schedule.html

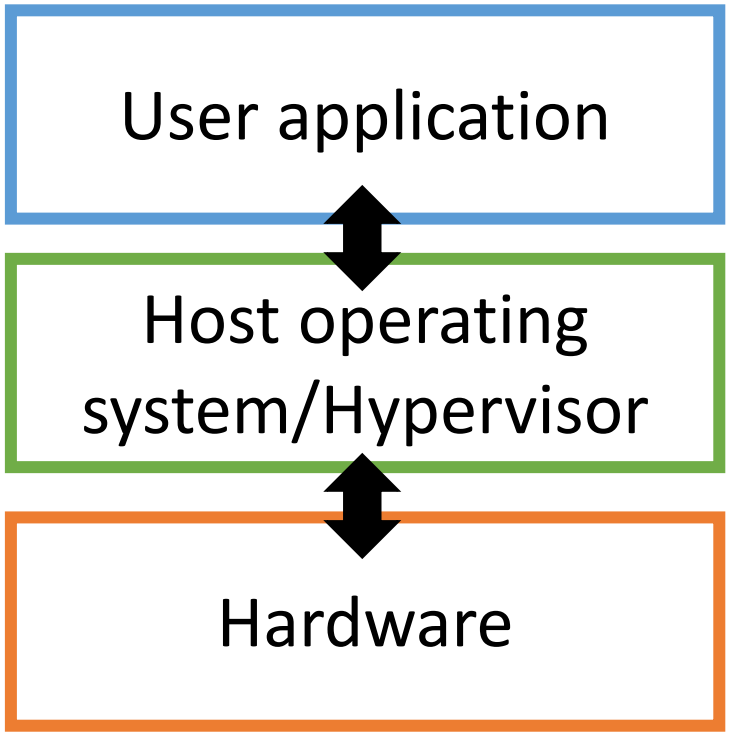
- Fill the google form https://forms.gle/G6gh6sEYJ4UY24ePA
  - your background/interests (e.g., microarchitecture, theoretical crypto, system security)
  - Top 5 papers that you would like to present

Course Overview

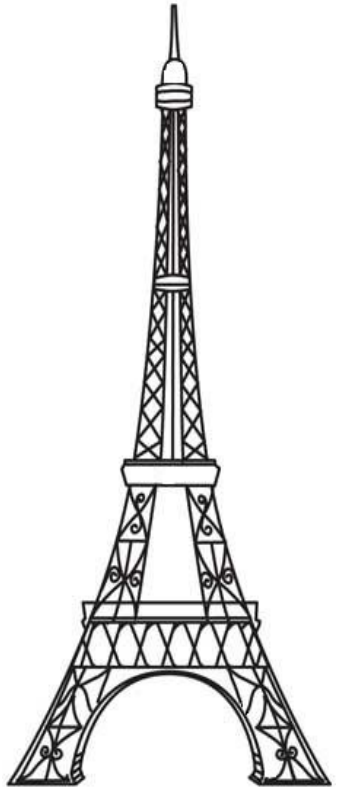
Why Hardware Security?



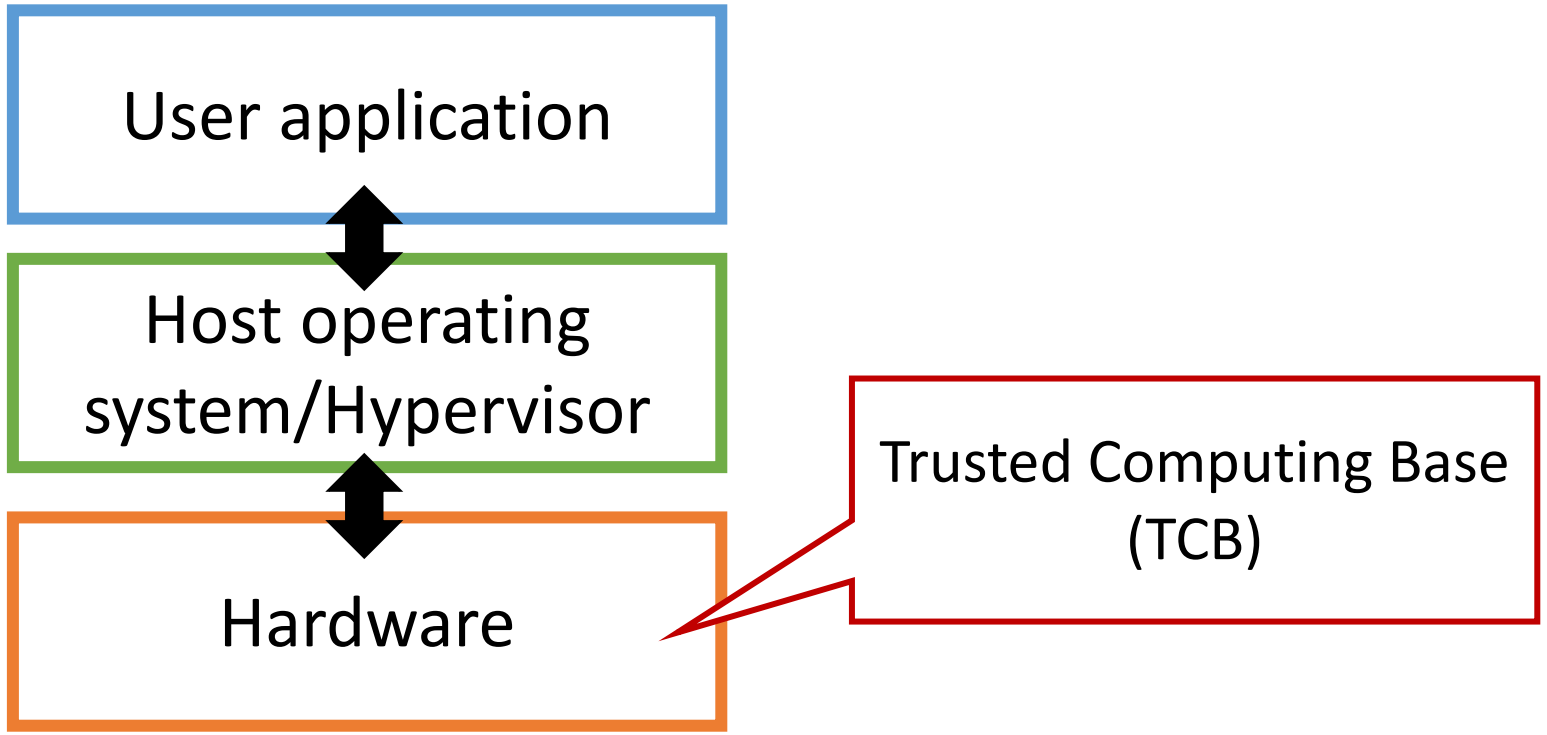
Computing Systems



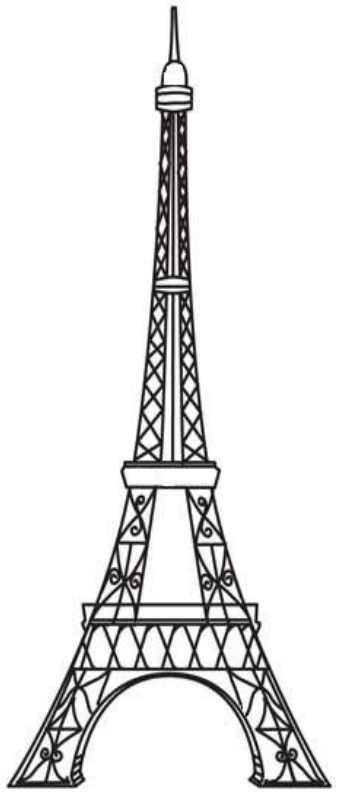
Why Hardware Security?



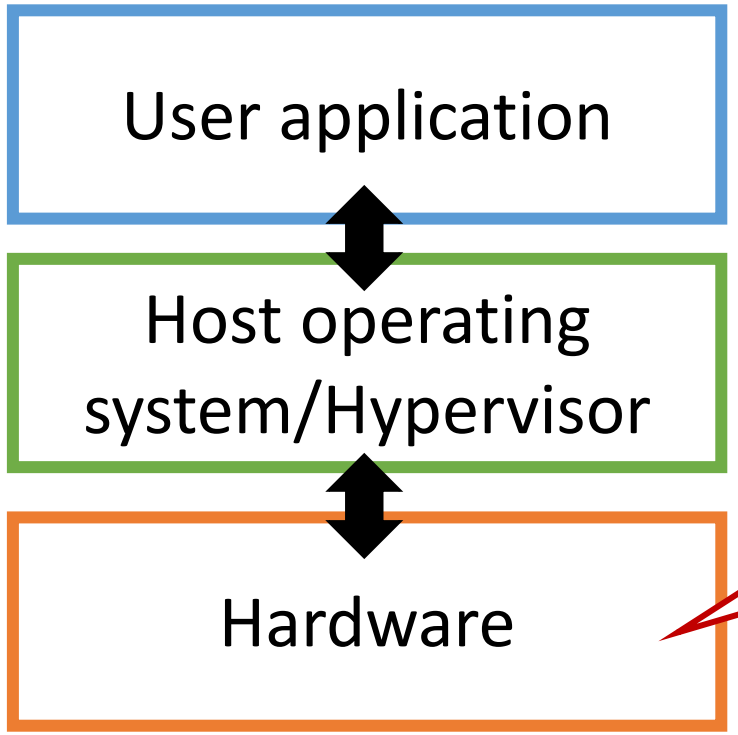
Computing Systems



Why Hardware Security?



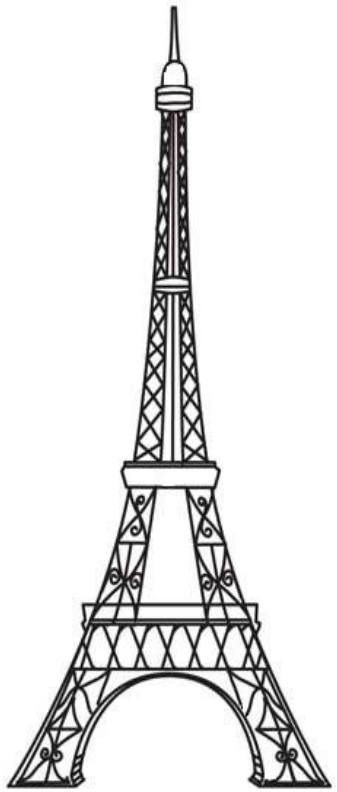
Computing Systems



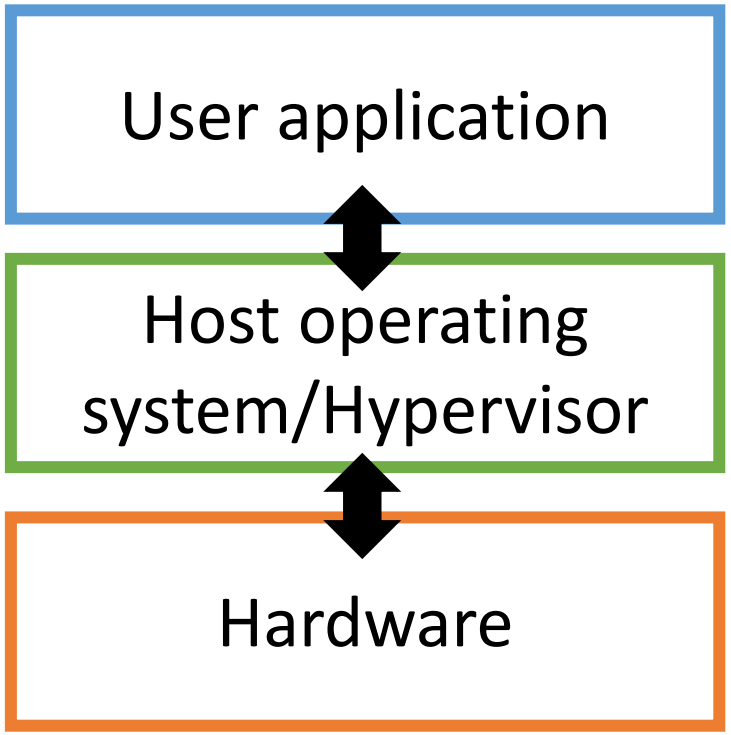
- What is the interface between SW and HW?

Trusted Computing Base (TCB)

Why Hardware Security TODAY?

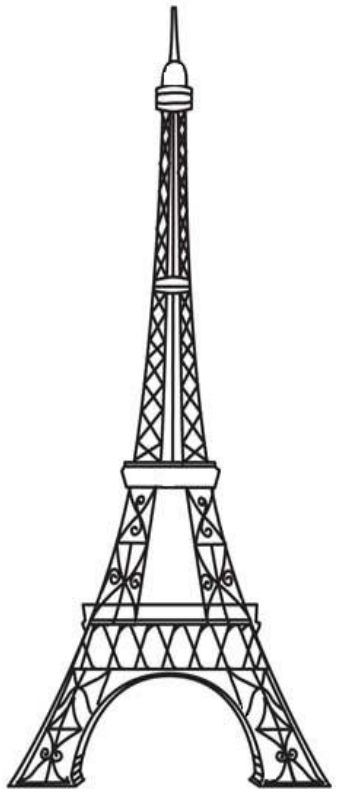


Computing Systems

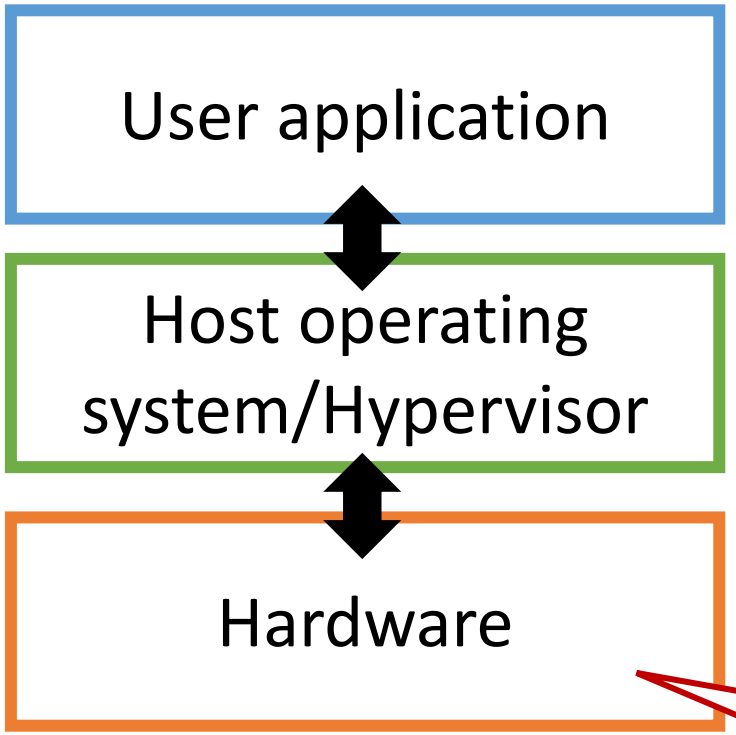


E.g, after Spectre and Meltdown

Why Hardware Security TODAY?



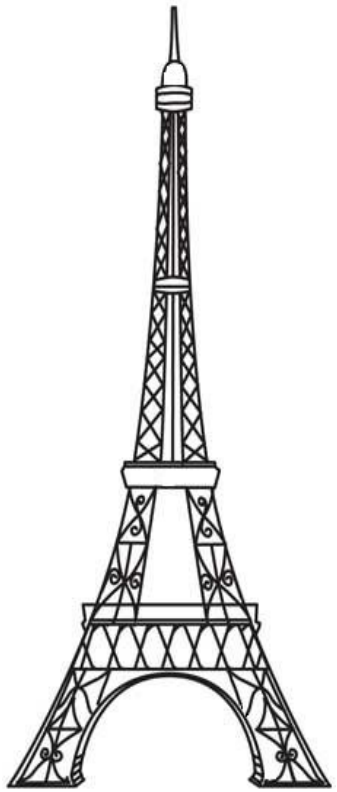
Computing Systems



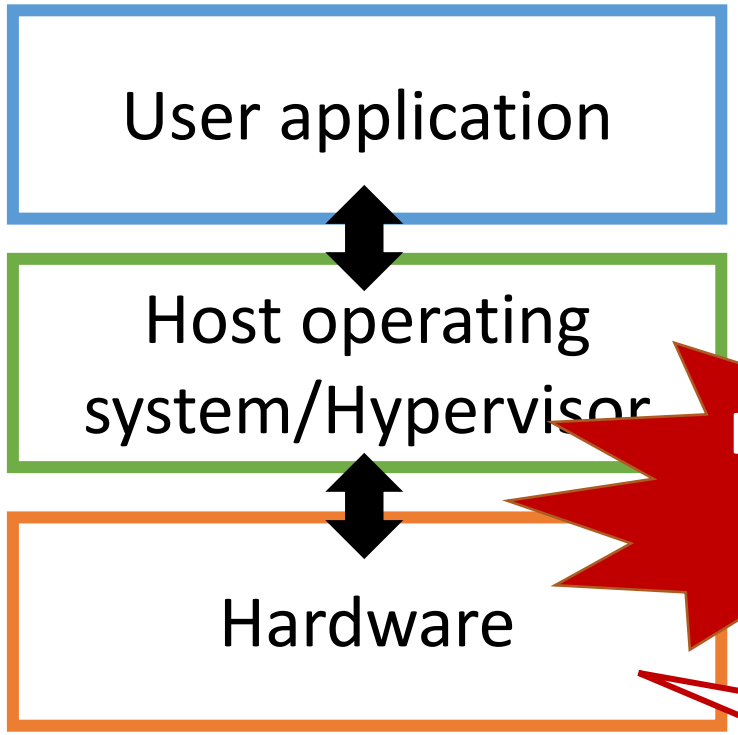
E.g, after Spectre and Meltdown

Open the Pandora's box

Why Hardware Security TODAY?



Computing Systems



E.g, after Spectre and Meltdown

Insufficient ISA

Open the Pandora's box

Preview of Modules/Topics

- Introduction

1) Micro-architecture Side Channel
2) Enclaves
3) Opensource Hardware and Verification
4) Physical Side Channels
5) Memory Safety

Introduction

- Commercial processor architectures that include security features:
  - LPAR in IBM mainframes (1970s)
  - IBM 4758 (2000s)
  - ARM TrustZone (2000s)
  - Intel TXT & TPM module (2000s)
  - Intel SGX (mid 2010s)
  - AMD SEV (late 2010s)

Micro-architecture Side Channels

Victim

A Channel
(a micro-architecture structure)

Attacker

[*] *Kiriansky et al. DAWG: a defense against cache timing attacks in speculative execution processors. MICRO'18*

Micro-architecture Side Channels

Access cache set [secret]

Victim

A Channel
(a micro-architecture structure)

Attacker

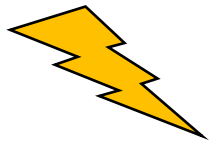
[*] *Kiriansky et al. DAWG: a defense against cache timing attacks in speculative execution processors. MICRO'18*

Micro-architecture Side Channels

Access cache set [secret]

secret-dependent execution

Victim



A Channel
(a micro-architecture structure)

Attacker

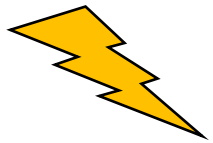
[*] *Kiriansky et al. DAWG: a defense against cache timing attacks in speculative execution processors. MICRO'18*

Micro-architecture Side Channels

Access cache set [secret]

secret-dependent execution

Victim

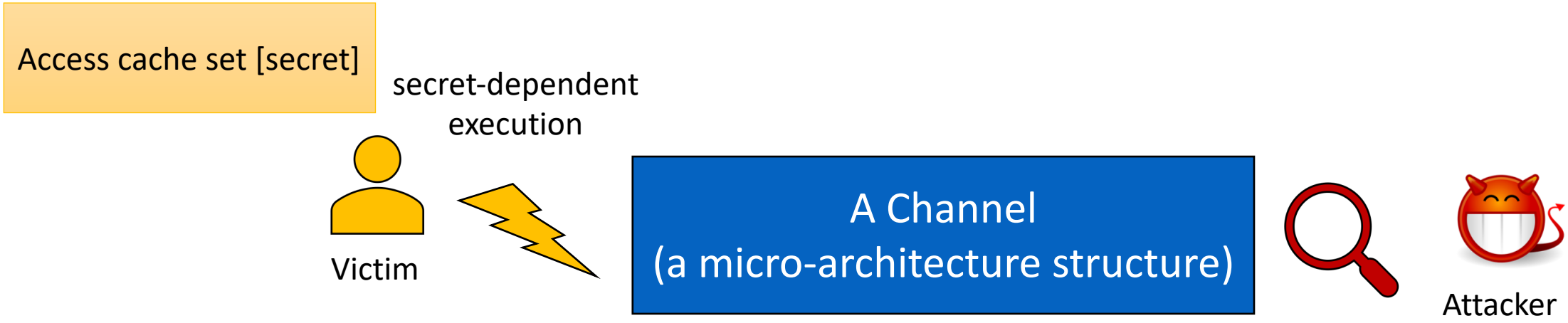


A Channel
(a micro-architecture structure)

Attacker

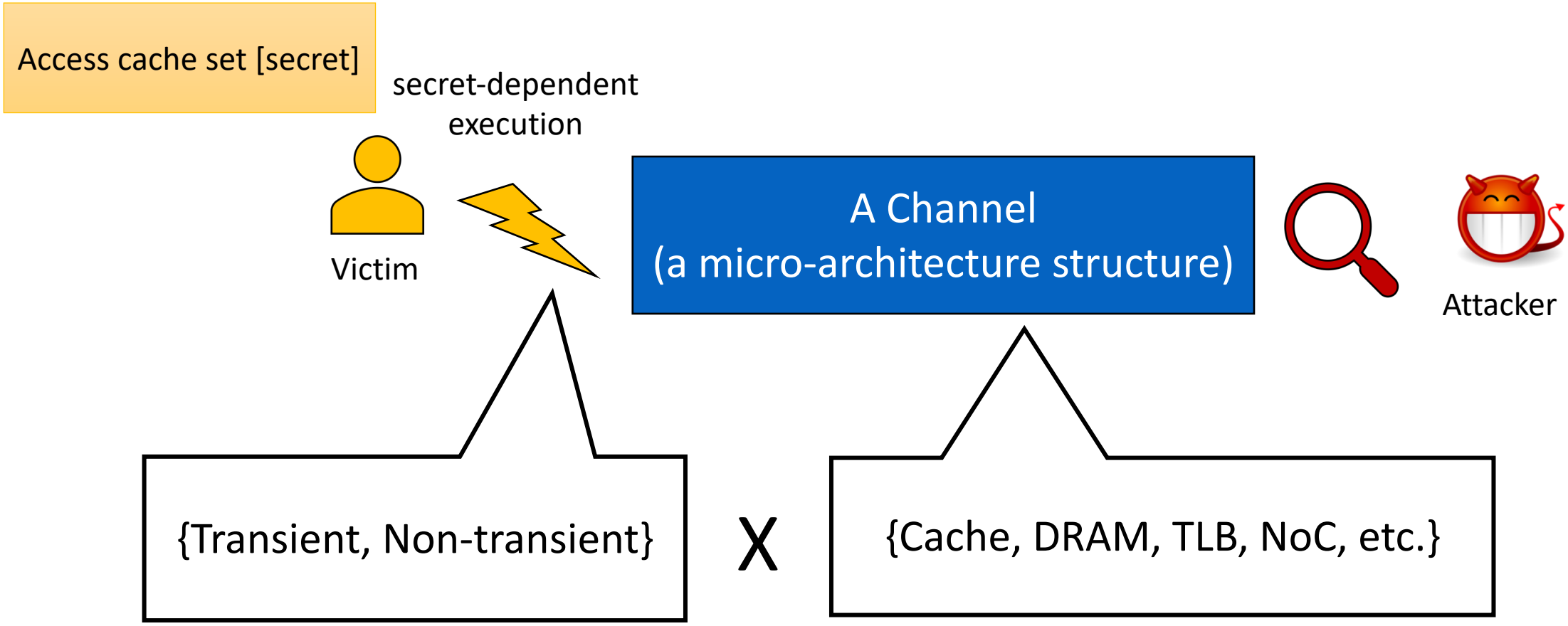
[*] *Kiriansky et al. DAWG: a defense against cache timing attacks in speculative execution processors. MICRO'18*

Micro-architecture Side Channels



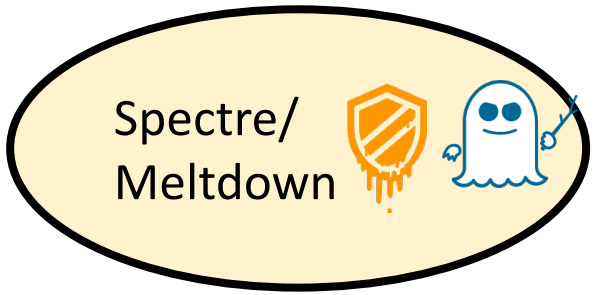
[*] *Kiriansky et al. DAWG: a defense against cache timing attacks in speculative execution processors. MICRO'18*

Micro-architecture Side Channels

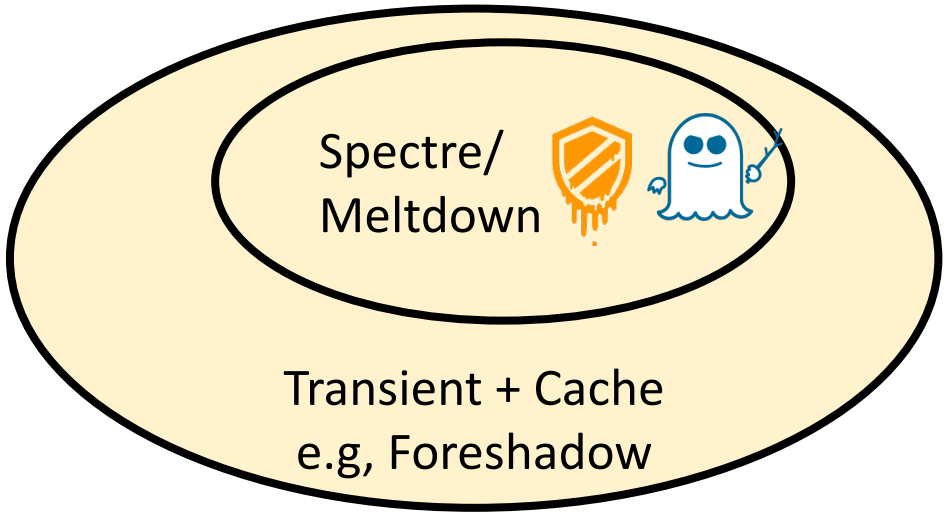


[*] *Kiriansky et al. DAWG: a defense against cache timing attacks in speculative execution processors. MICRO'18*

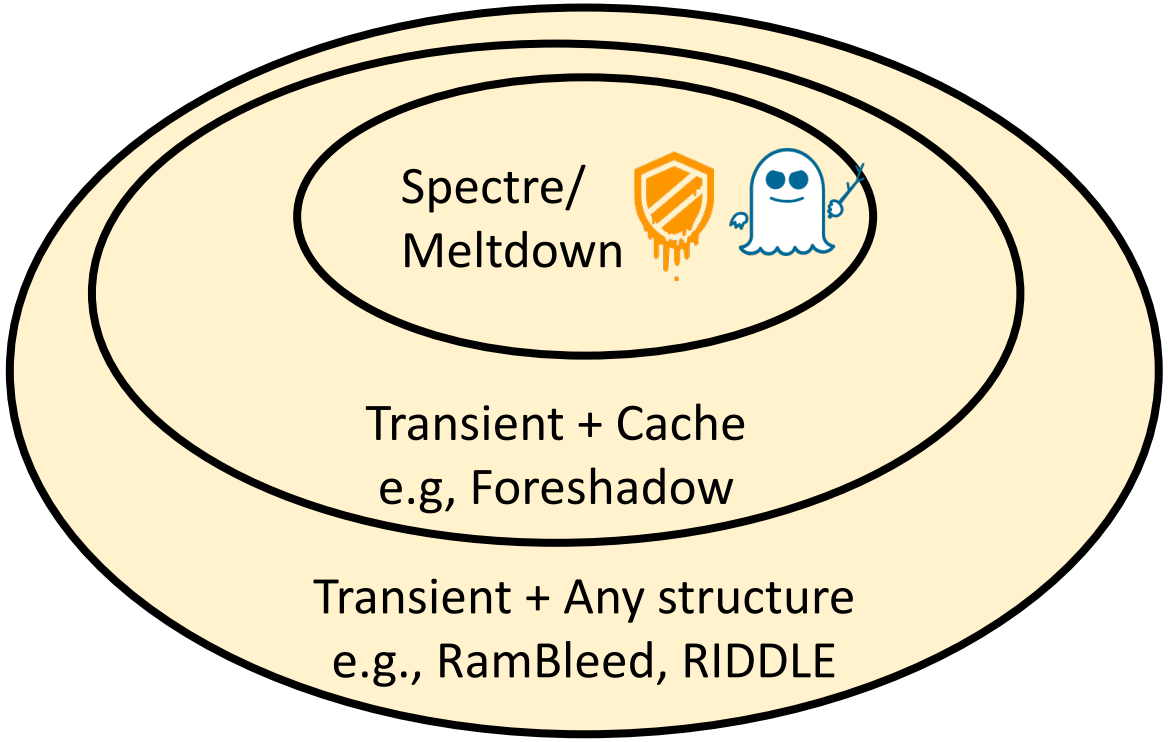
Micro-architecture Side Channel



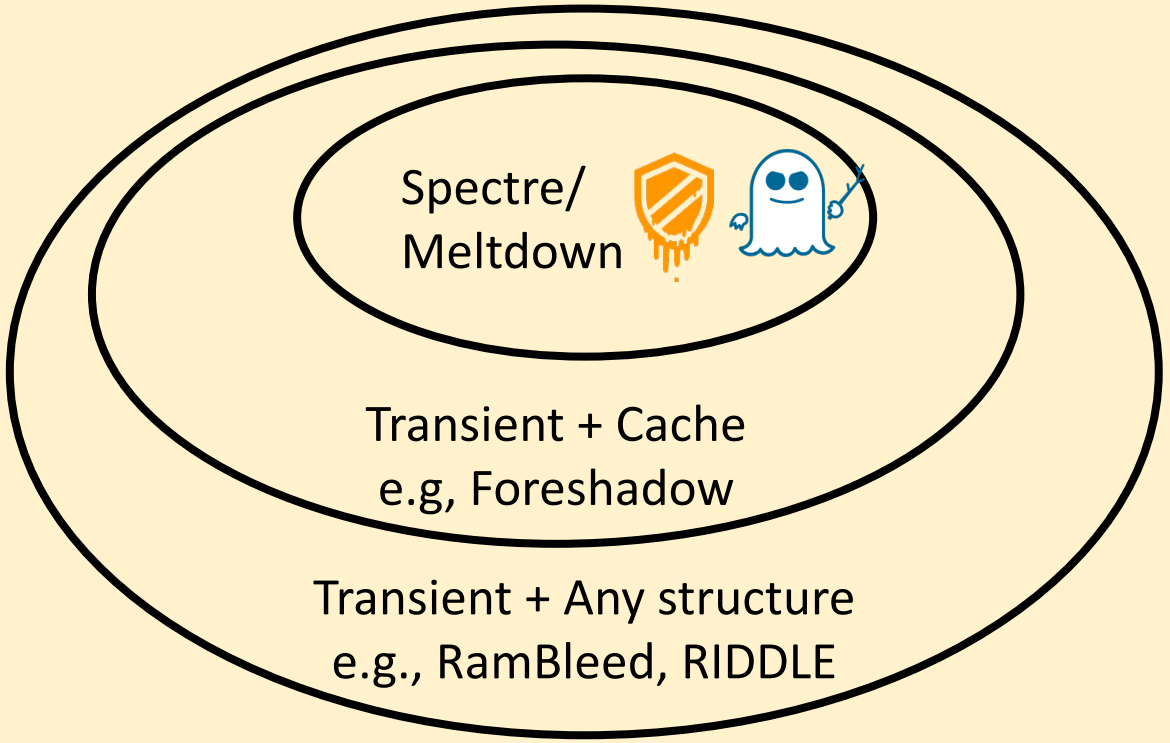
Micro-architecture Side Channel



Micro-architecture Side Channel



Micro-architecture Side Channel



Non-transient + Any structure

Micro-architecture Side Channels

Micro-architecture Side Channels

Defenses:

[*] *Kiriansky et al. DAWG: a defense against cache timing attacks in speculative execution processors. MICRO'18*

Micro-architecture Side Channels

Defenses:

Block creation of signals:
Oblivious execution,
speculative execution defenses, etc.

[*] *Kiriansky et al. DAWG: a defense against cache timing attacks in speculative execution processors. MICRO'18*

Oblivious Programming

Victim

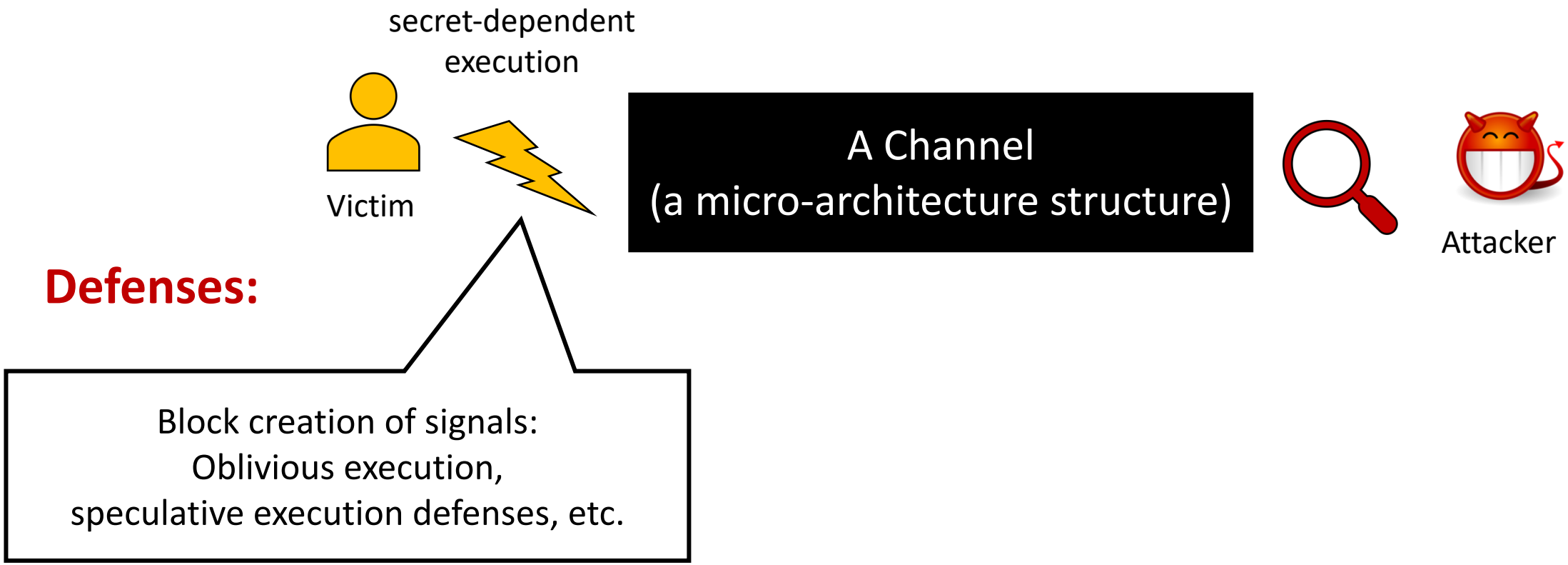
secret in {0,….,127}

Access cache set [secret]

secret in {0,….,127}

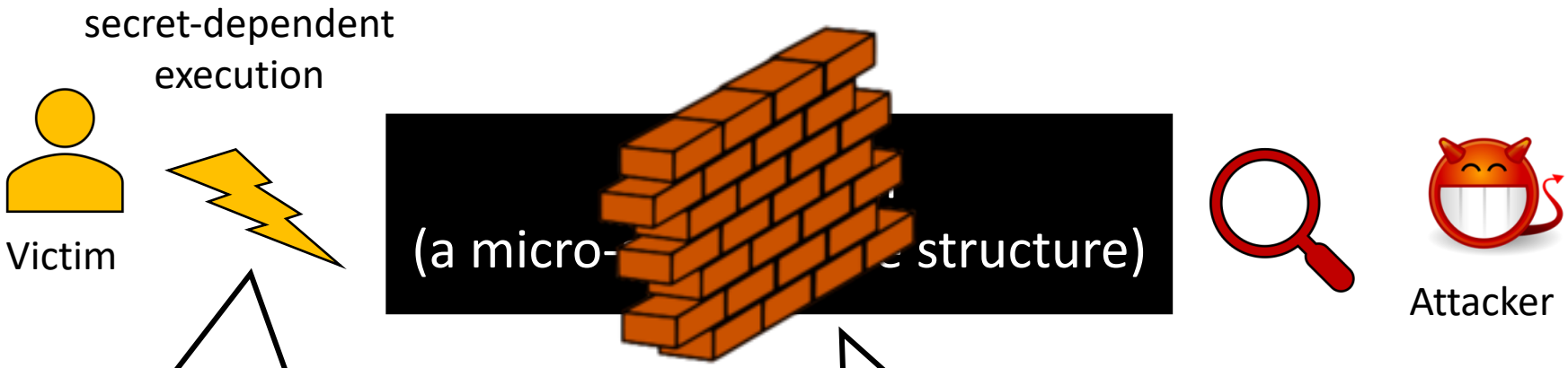
For I from 0 to 127:
    access cache set [i]

Micro-architecture Side Channels



[*] *Kiriansky et al. DAWG: a defense against cache timing attacks in speculative execution processors. MICRO'18*

Micro-architecture Side Channels



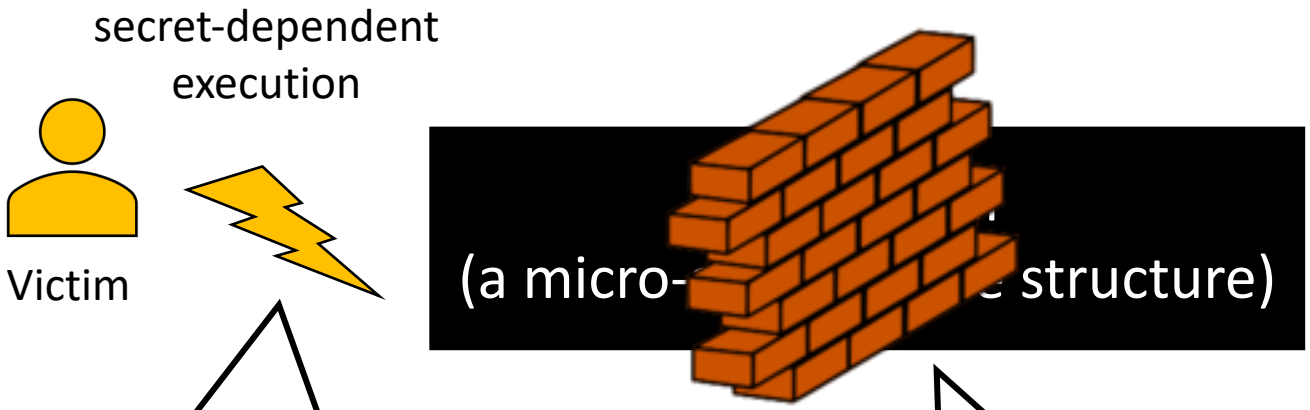
Defenses:

Block creation of signals:
Oblivious execution,
speculative execution defenses, etc.

Close the channel:
Isolation, etc.

[*] *Kiriansky et al. DAWG: a defense against cache timing attacks in speculative execution processors. MICRO'18*

Micro-architecture Side Channels



Defenses:

Block creation of signals:
Oblivious execution,
speculative execution defenses, etc.

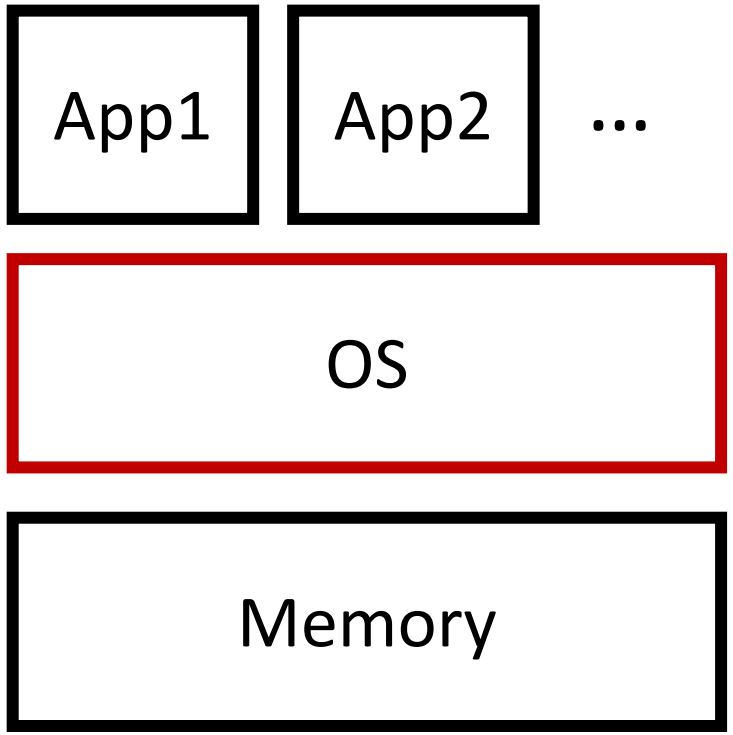
Close the channel:
Isolation, etc.

Block detection of signals:
Randomization, etc.

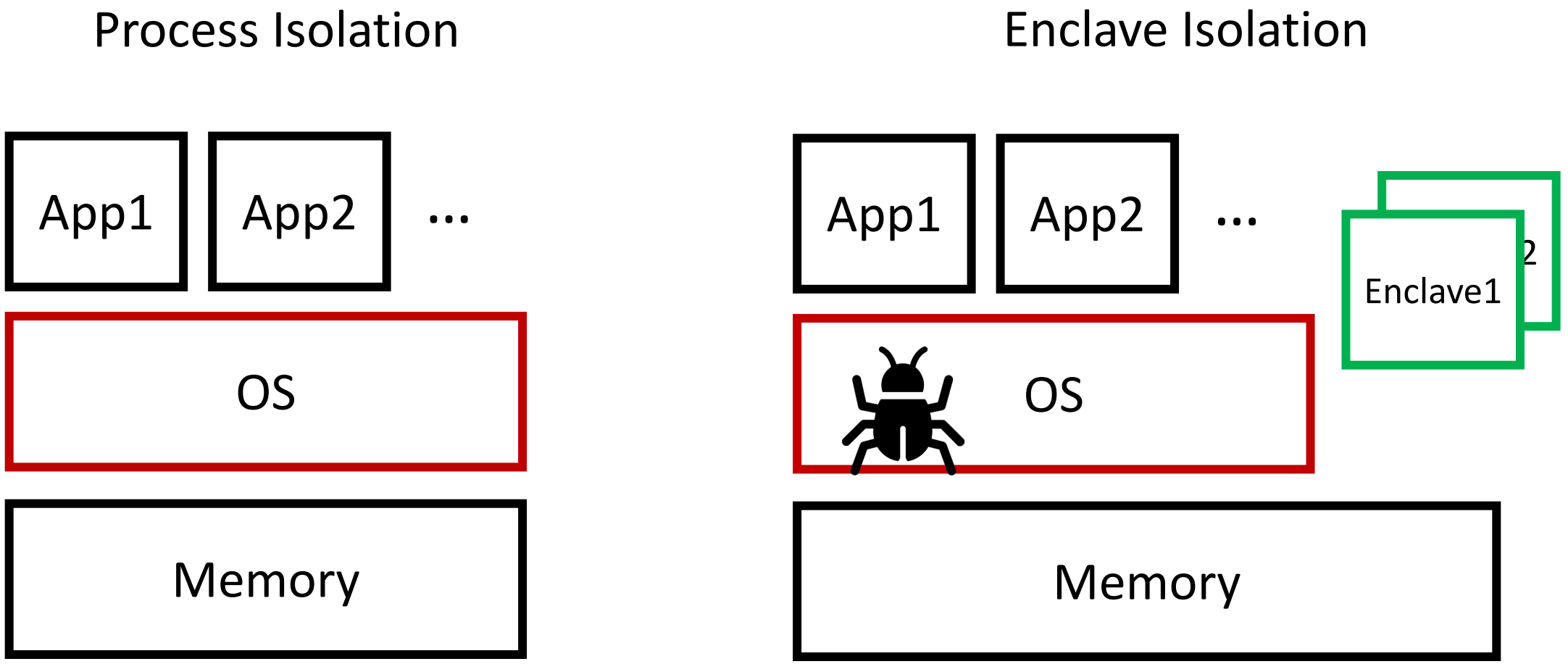
[*] *Kiriansky et al. DAWG: a defense against cache timing attacks in speculative execution processors. MICRO'18*

Enclaves

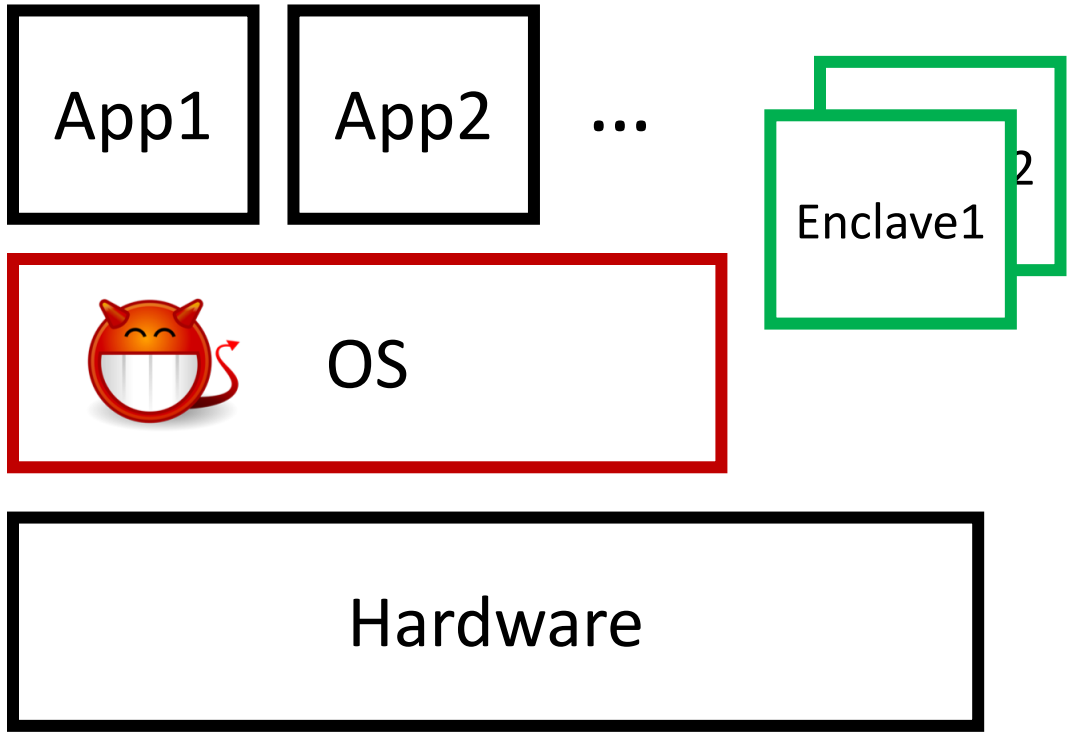
Process Isolation



Enclaves

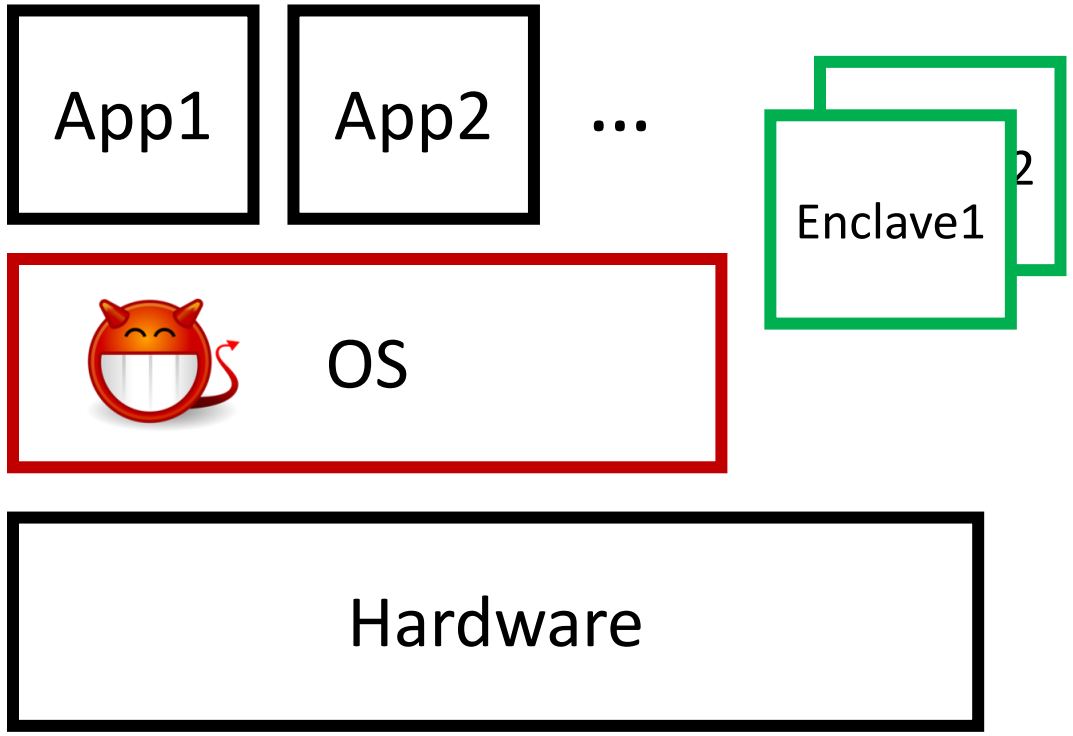

Process Isolation
App1
App2
...
OS
Memory
Enclave Isolation
App1
App2
...
Enclave1
OS
Memory

Enclaves



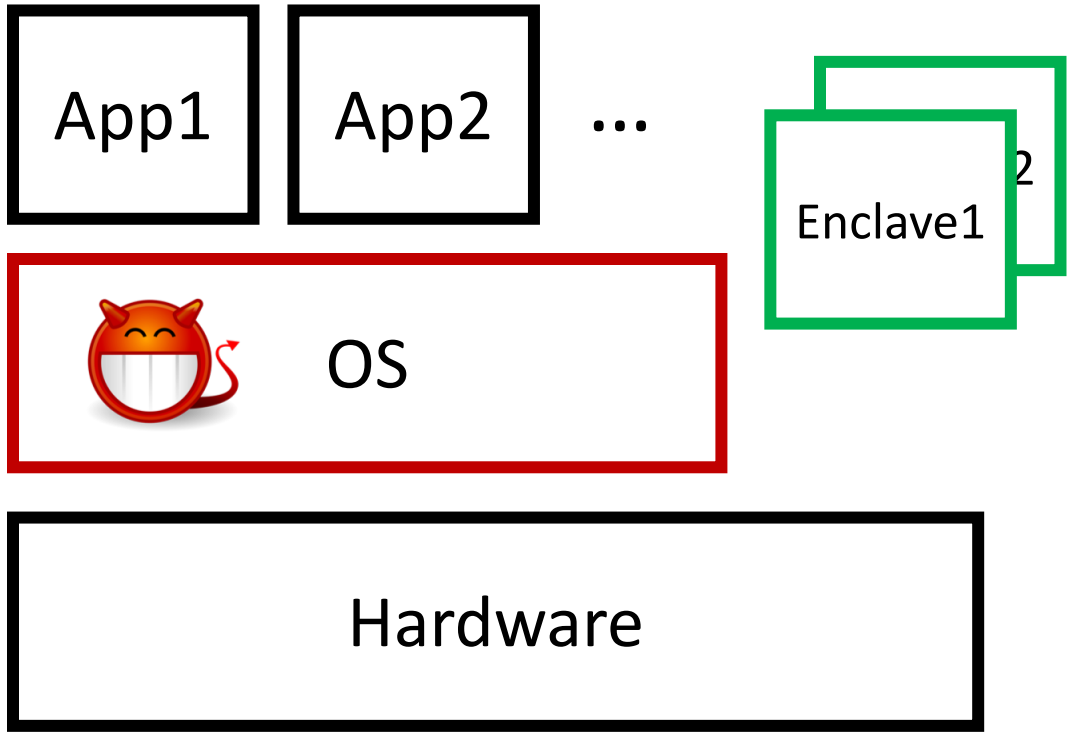
- Side-channel vulnerabilities in an enclave setup?

Enclaves



- Side-channel vulnerabilities in an enclave setup?
- Defend against privileged attackers?

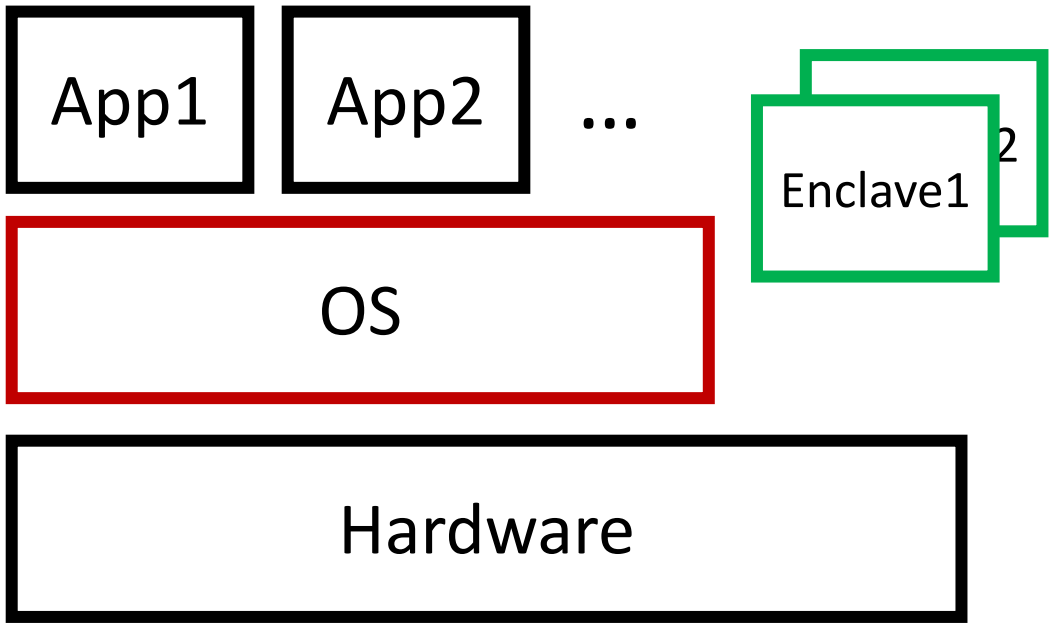
Enclaves



- Side-channel vulnerabilities in an enclave setup?
- Defend against privileged attackers?
- How to write efficient enclave applications?

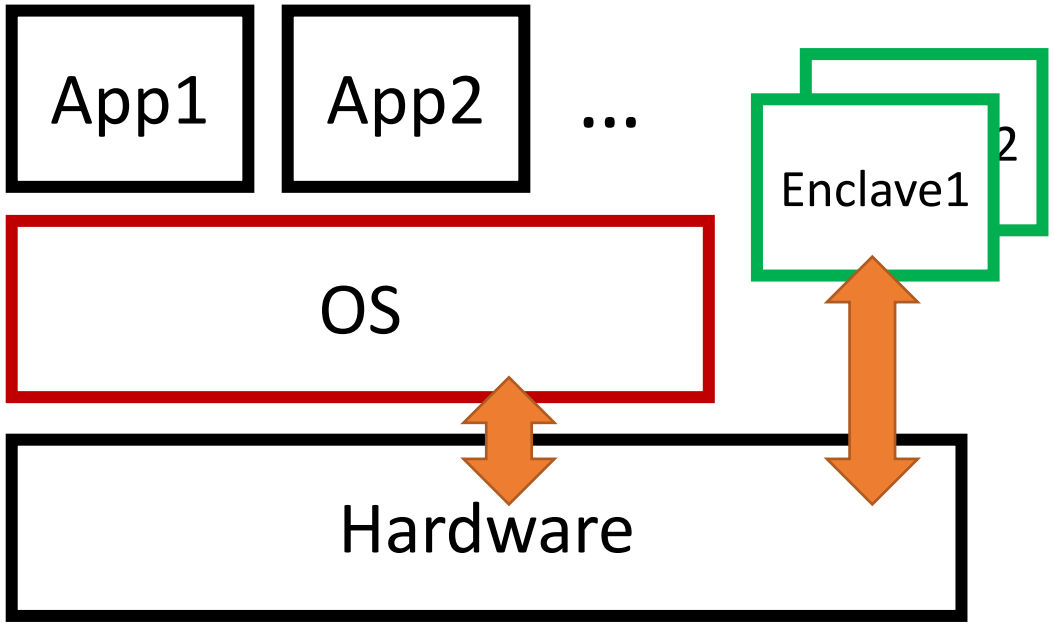
Opensource Hardware and Verification

- Modern hardware is a mess for security. What if you can design everything from scratch?



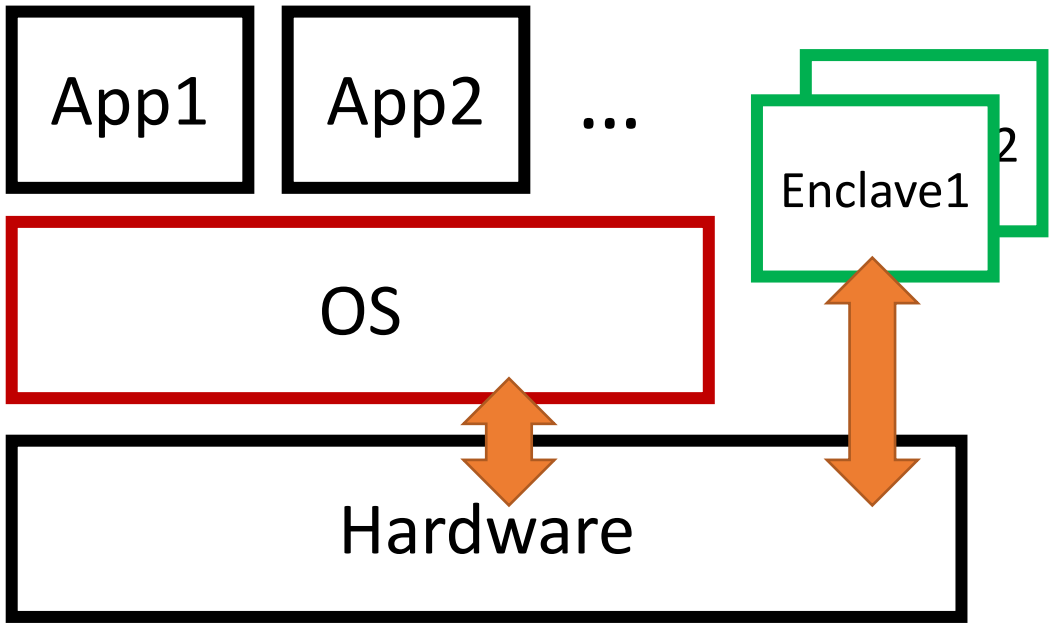
Opensource Hardware and Verification

- Modern hardware is a mess for security. What if you can design everything from scratch?



Opensource Hardware and Verification

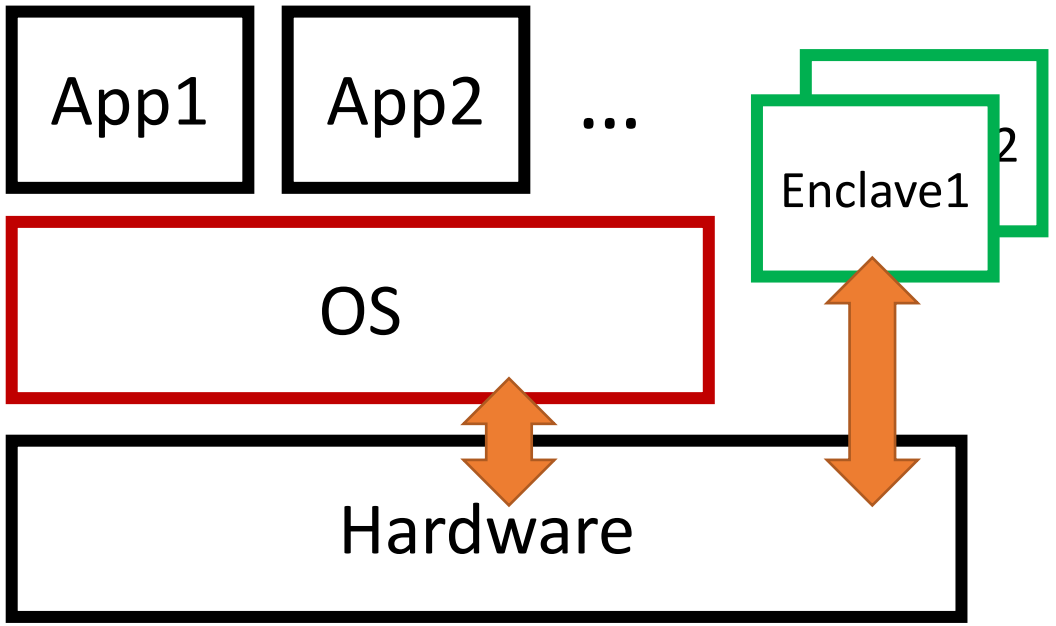
- Modern hardware is a mess for security. What if you can design everything from scratch?



- Many design choices:
  - HW v.s. SW implementations?
  - New abstraction of HW?

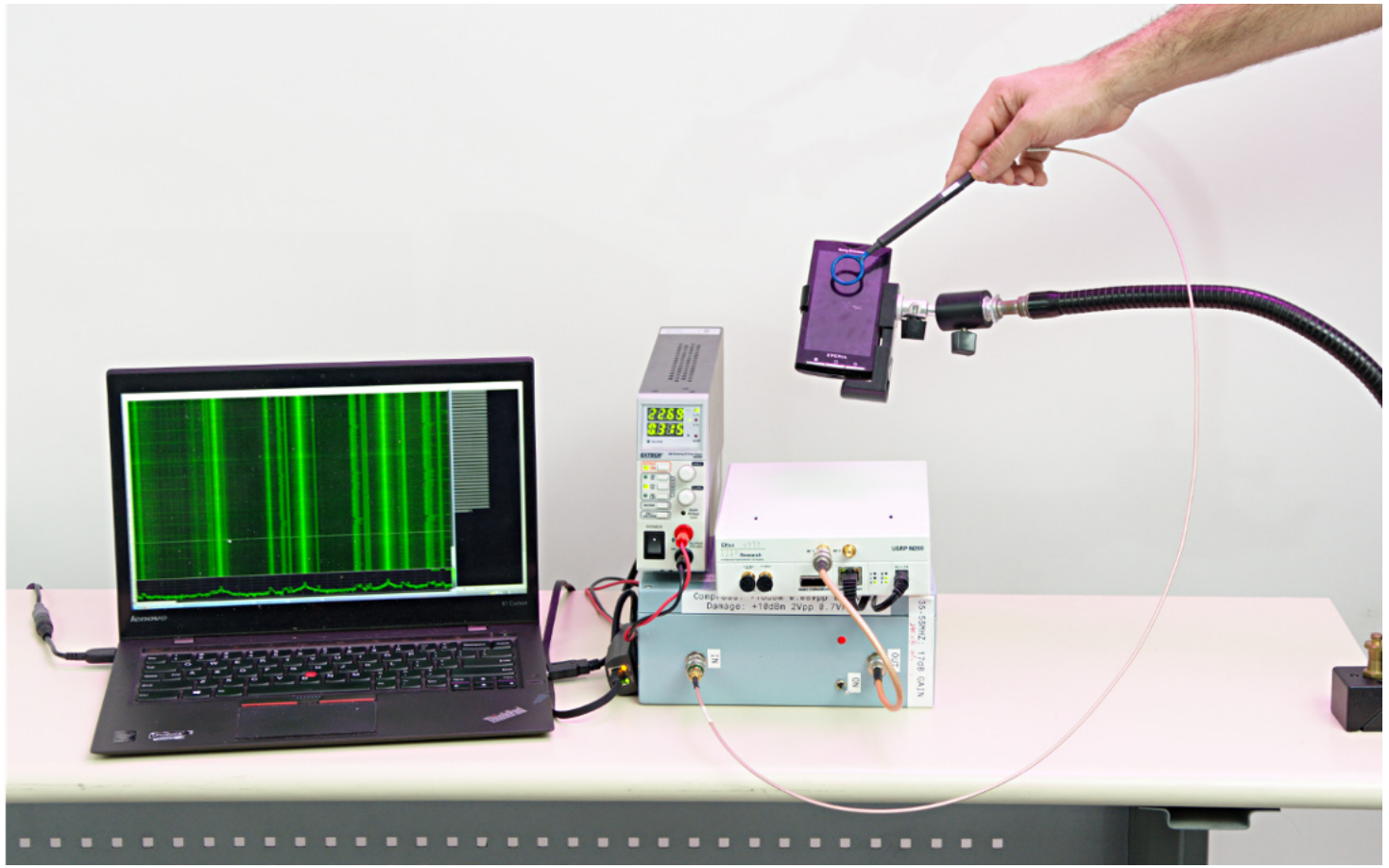
Opensource Hardware and Verification

- Modern hardware is a mess for security. What if you can design everything from scratch?



- Many design choices:
  - HW v.s. SW implementations?
  - New abstraction of HW?
- How to verify the security of HW or multiple layers of a system

Physical Attacks



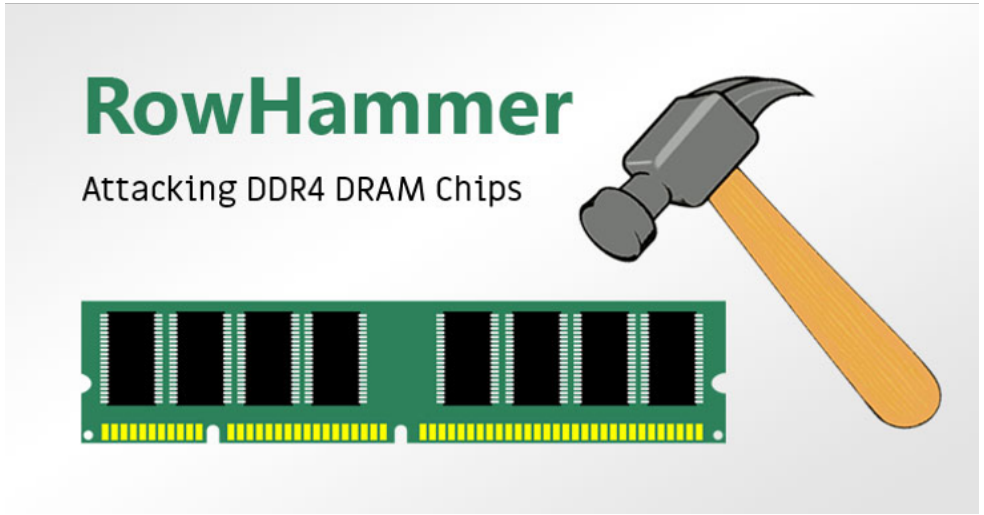
EM side channels to steal bitcoin signing keys

Physical Attacks

- Modern physical side channels can be done remotely

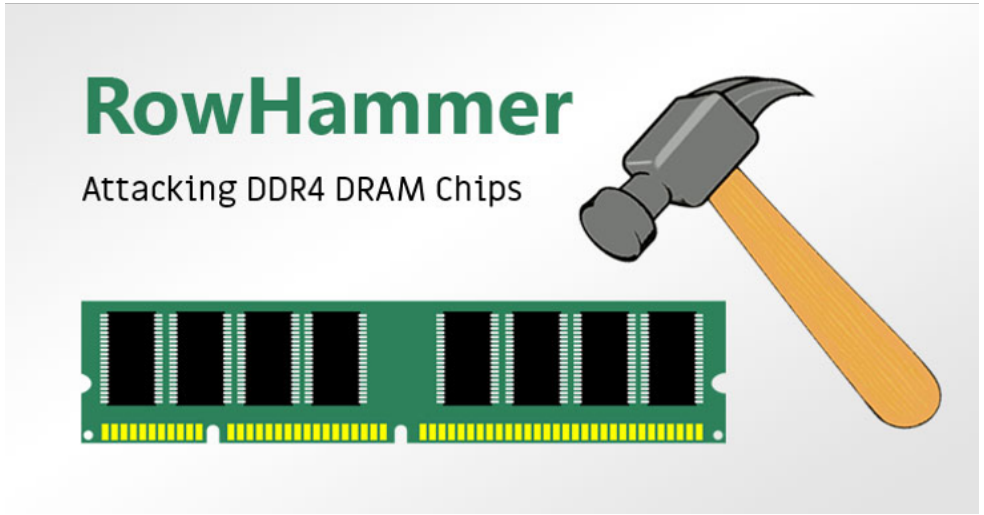
Physical Attacks

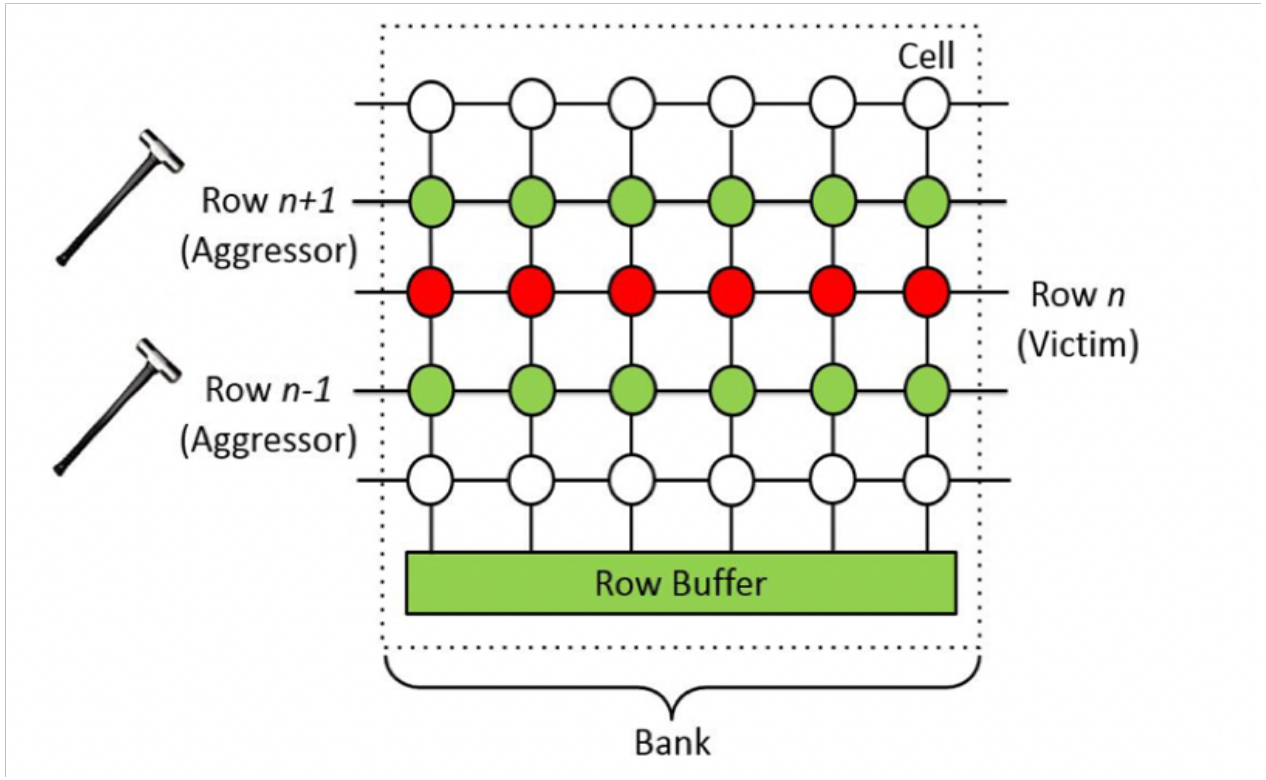
- Modern physical side channels can be done remotely



Physical Attacks

- Modern physical side channels can be done remotely





Memory Safety

- Classical memory safety issues
  - E.g., buffer overflow

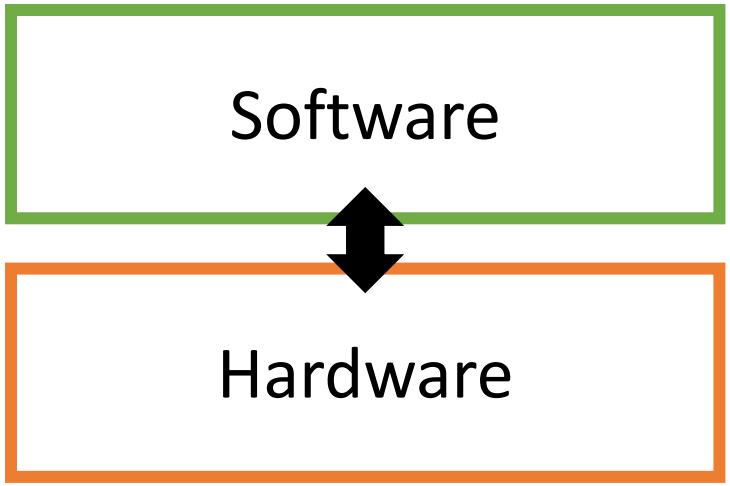
Memory Safety

- Classical memory safety issues
  - E.g., buffer overflow

- HW: accelerators for security checks

Memory Safety

- Classical memory safety issues
  - E.g., buffer overflow
- HW: accelerators for security checks
- A more interesting question: what is a good abstraction?



Next:
Secure Processors in Industry