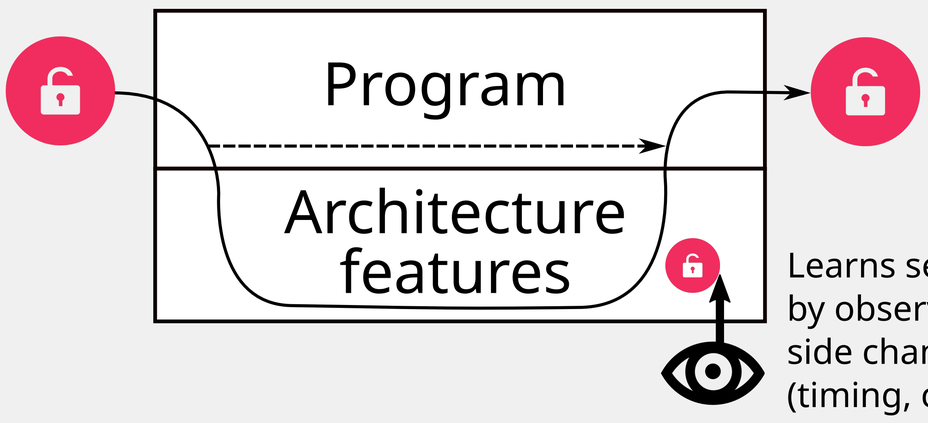
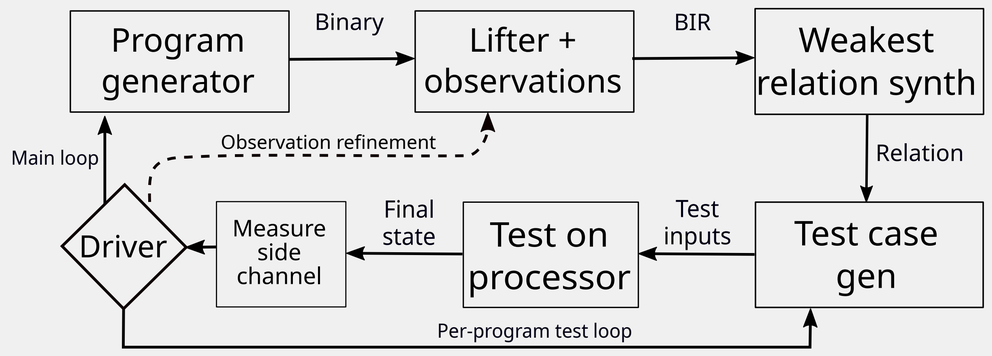
Validation of Abstract Side-channel Models for Computer Architectures

Andreas Lindner (KTH), Hamed Nemati (CISPA), Matthias Stockmayer (CISPA),
Pablo Buiras (KTH), Roberto Guanciale (KTH) and Swen Jacobs (CISPA)

Programs may leak secrets via side channels



SCAM-V pipeline



State of the Art

- **Hardware complexity** makes it tricky to analyze **side-channel security**
- Abstract **models** of side-channels approximate them with system state **observations**
- They assume **soundness**:

  Observational equivalence ⇒ Indistinguishability to attacker in real hw

- But models **not always sound**

  e.g. Spectre attacks

Lifting and BIR (intermediate language)

```
    B.eq l2                [l0: CJMP Z l2 l1]
    mul x1 x2 x3     →     [l1: X1= X2*X3; JMP l2]
l2: ldr x2 {x1} +8         [l2: OBS([tag(X1)]);
                               X2= LOAD(MEM, X1);
                               X1= X1+8;
                               HALT]
```

Example: observing **cache tag bits** of every load Observation statement added to IR before LOAD

Relation synthesis and Test generation

Symbolic execution:

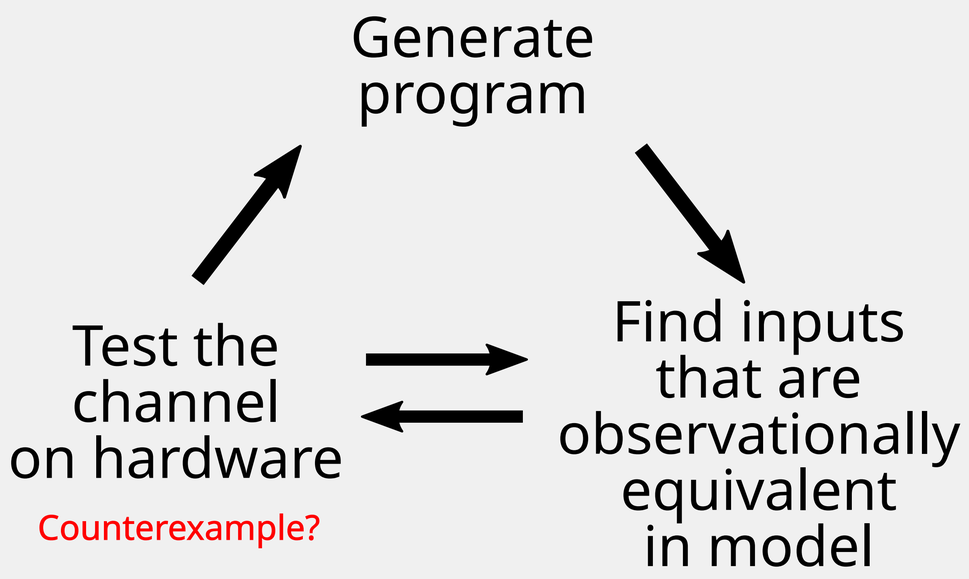
$$(Z \wedge Z' \Rightarrow tag(X1) = tag(X1')) \wedge (Z \wedge \neg Z' \Rightarrow tag(X1) = tag(X2'*X3')) \wedge (\neg Z \wedge Z' \Rightarrow tag(X2*X3) = tag(X1')) \wedge (\neg Z \wedge \neg Z' \Rightarrow tag(X2*X3) = tag(X2'*X3')) \wedge \textit{address constraints (...)}$$

SMT solver (Z3) finds satisfying assignment s1 and s2

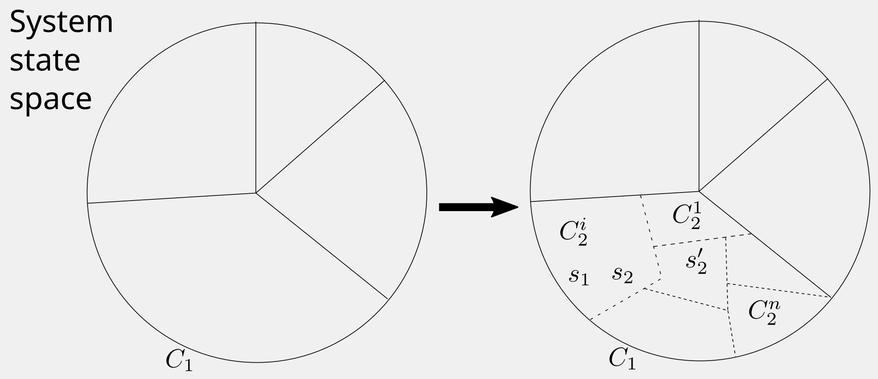
Our proposal

Side
Channel
Abstract
Model
Validator

Use testing to validate the model against real hardware



Observation refinement (Search steering)



Partitioned by obs. equivalence in original model

(each one too big)

Partitioned by obs. equivalence in more precise model

s1, s2 chosen to be in different (smaller) partitions