Validation of Abstract Side-channel Models for Computer Architectures

Andreas Lindner[1], Hamed Nemati[2], Matthias Stockmayer[2], **Pablo Buiras**[1], Roberto Guanciale[1] and Swen Jacobs[2]

(work in progress)
ENTROPY '19,
Stockholm

[1] KTH Royal Institute of Technology

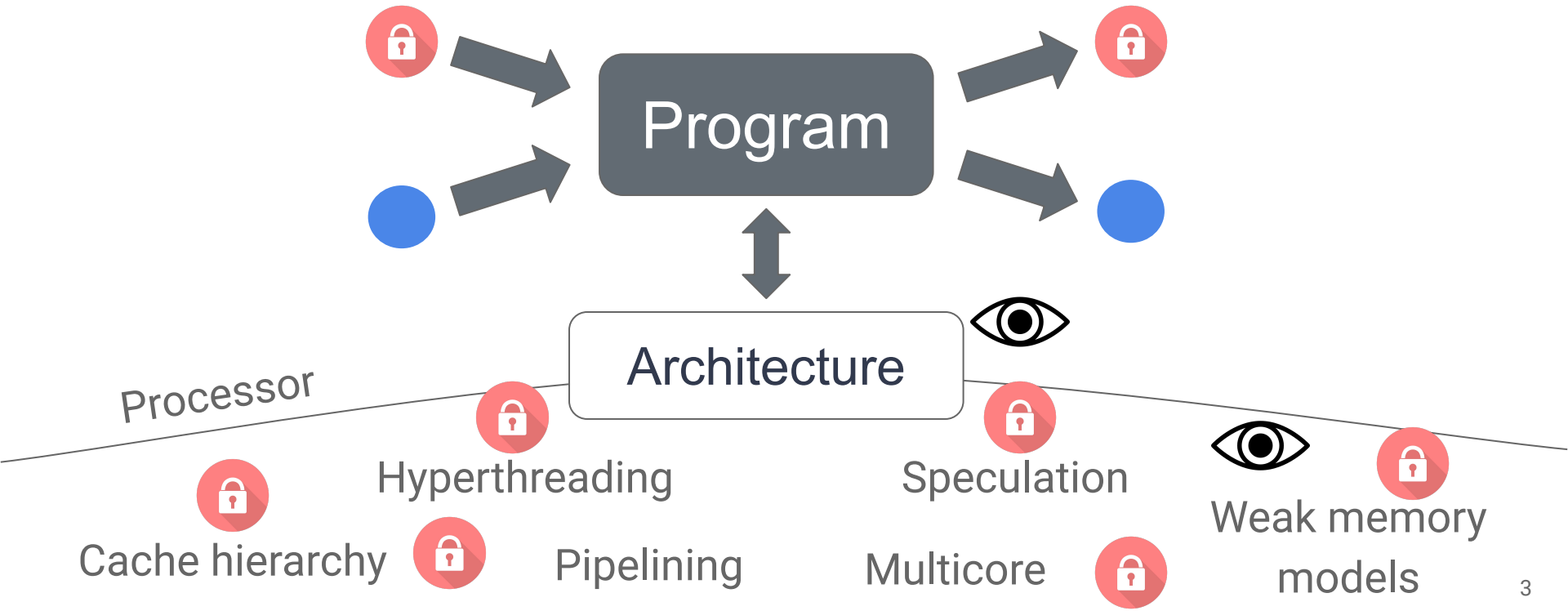
[2] CISPA Helmholtz Center for Information Security

Side channels and Security

Noninterference

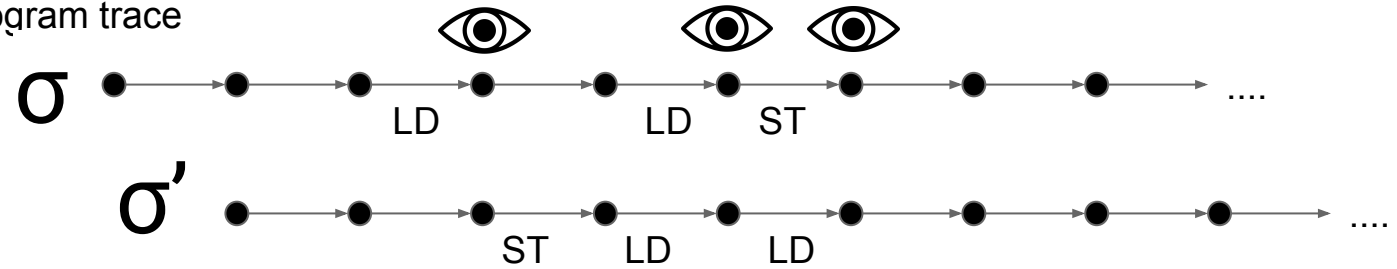
$$P(a) \sim_{\text{👁}} P(b) \text{ if } a \sim_{\text{👁}} b, \ \forall \ a\ b$$

Side channels and Security



Abstract observation models

Program trace



Program text

MOV …
ADD …
CMP ...
LD ...
B label
...

$\sigma \sim_P \sigma'$

Observational Equivalence

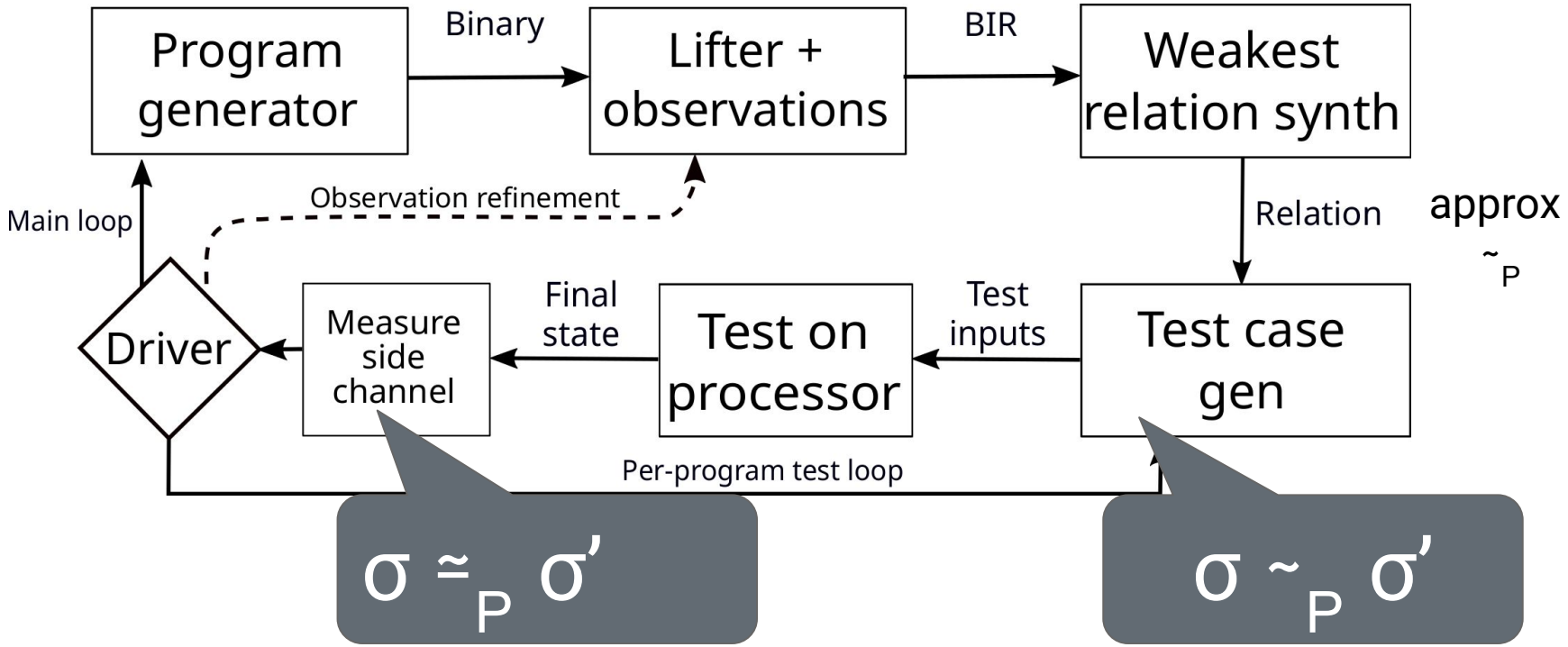
Abstract model

Real hardware

$$\sigma \sim_P \sigma' \quad \overset{?}{\Longrightarrow} \quad \sigma \simeq_P \sigma'$$

SCAM-V: Side Channel Abstract Model Validator

SCAM-V Pipeline Overview



BIR

Abstract Assembly Language

- Infinite number of register variables
- Assignments, jumps, cond. jumps
- BIR expressions: arithmetic, bitwise, etc.
- Memory is an array
- **(Attacker) Observations**

BIR

Observation statements

`OBS(c, exp)`

outputs the evaluation of `exp` in the current state,
if `c` evaluates to true

OBS(true, exp) = OBS(exp)

Lifting and BIR

Binary

```
    B.eq l2
    mul x1 x2 x3
l2: ldr x2 {x1} +8
```

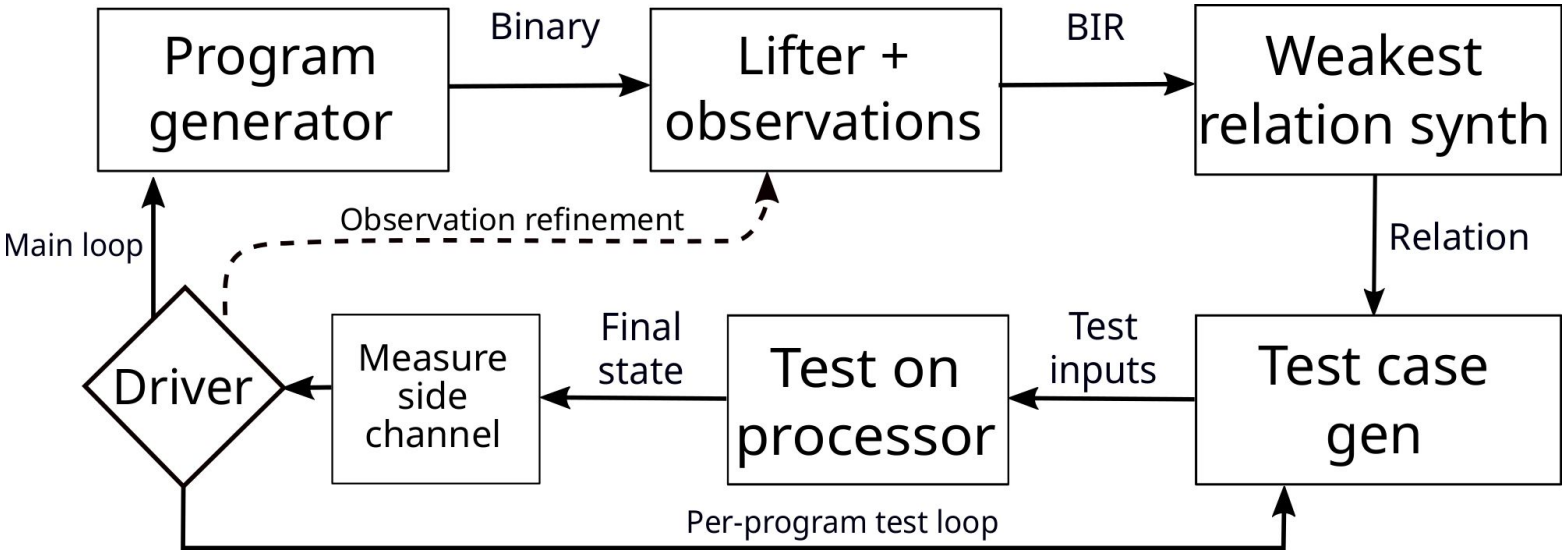
BIR

```
[l0: CJMP Z l2 l1]
[l1: X1= X2*X3; JMP l2]
[l2: OBS([tag(X1)]);
     X2= LOAD MEM, X1); X1= X1+8;
     HALT]
```

Obs model: Cache tag

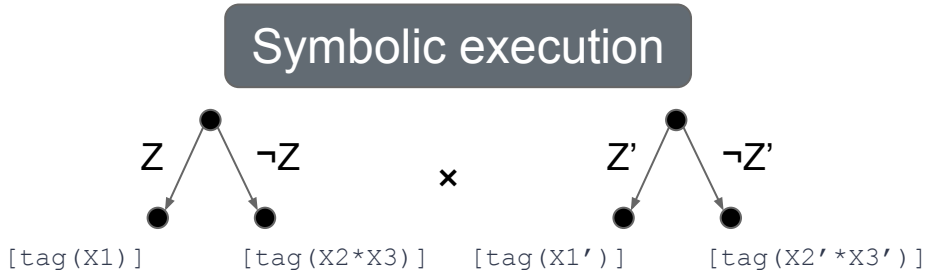
Observation added according to selected model

SCAM-V Pipeline Overview



SCAM-V Weakest Relation Synthesis

```
[l0: CJMP Z l2 l1]
[l1: X1= X2*X3; JMP l2]
[l2: OBS([tag(X1)]);
     X2= LOAD(MEM, X1); X1= X1+8;
     HALT]
```

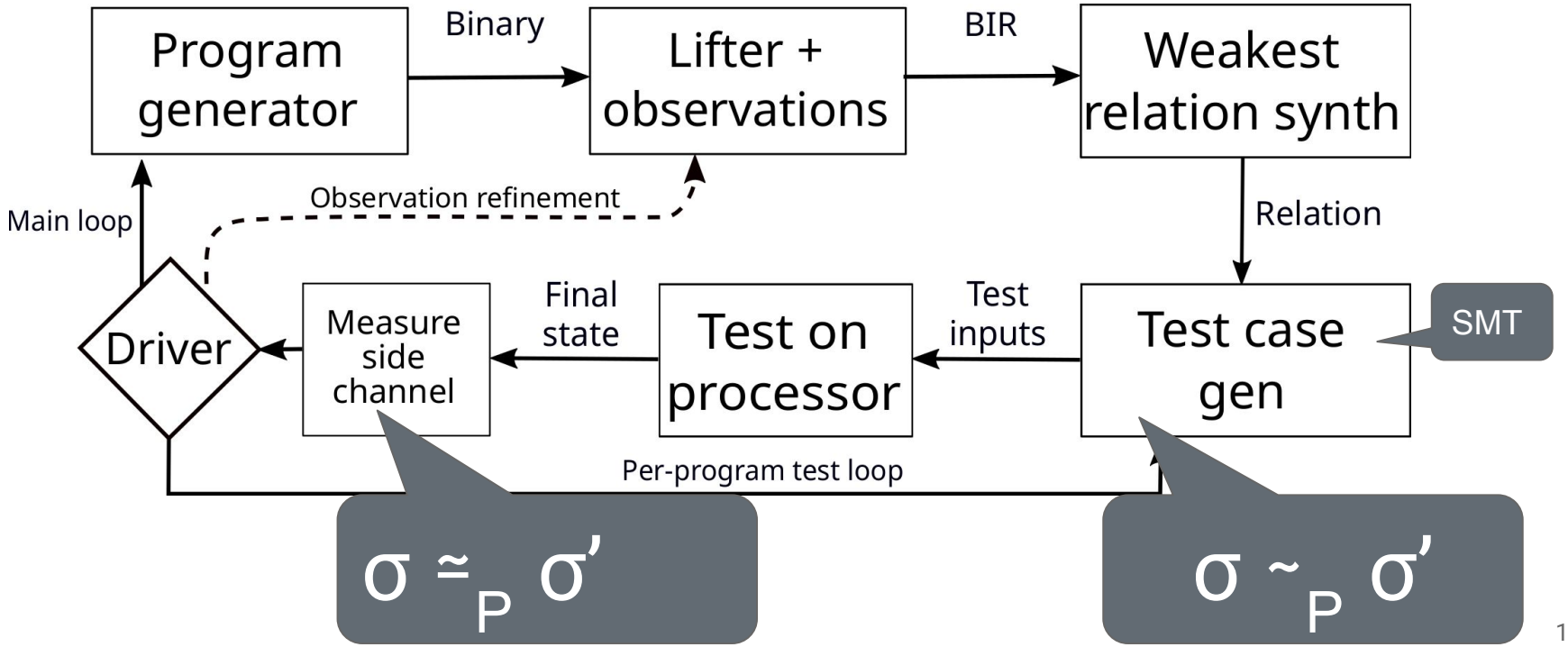


Relation

$\sigma \sim_P \sigma'$

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

SCAM-V Pipeline Overview



Testing / Measuring the channel

- ARMv8 (Raspberry Pi 3)
- **Observation**: tag and operation of every memory access
- **Measuring**: TrustZone instructions for cache inspection

- Cortex-M0 (MicroBit)
- **Observation**: program counter
- **Measuring**: internal system clock

SCAM-V: Side Channel Abstract Model Validator

Summary

- Modern architectures are too complex to directly analyze side-channels
- Abstract models based on system-state observations
    - PC security model, Cache-line + tag of memory access
- **Assumption:**
  States with ***equivalent*** observations in the model are
  indistinguishable to the attacker on real hardware
- Not always true! e.g. Spectre
- SCAM-V **validates** this assumption