CoppeliaPayload: Automatically Generating Hardware Attacks

Martin Meng, Cynthia Sturton, UNC Chapel Hill

DESIGN AUTOMATION CONFERENCE 57 — FROM CHIPS TO SYSTEMS - LEARN TODAY, CREATE TOMORROW — DAC YF

THE UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

Introduction

Hardware Attacks

- Attack examples will help designers better understand security threats.
- Developing hardware attacks manually is challenging.

Goal

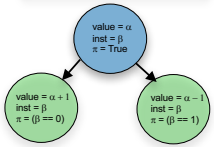
Given identified triggers, automatically generate hardware attacks.

Background

Symbolic Execution

- Explore all possible paths of a program with symbolic inputs.
- Associated with a symbolic exploration tree.
- e.g.

```
value <= inst ? value - 1:
                value + 1;
```

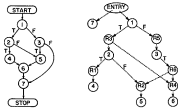


Recursive, Backward Strategy

- Usage: find assertion violations in hardware.
- Apply symbolic execution to find the assertion violation in one cycle.
- Recursively find the previous states until the initial state is found.

Dependency Analysis

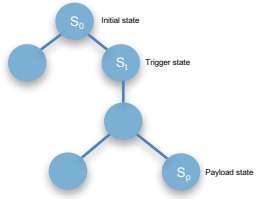
Program dependence graph: represents data and control dependences of a program.



Problem Formulation

Given a processor design and a trigger state S_t,

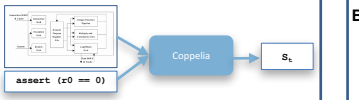
- define a payload state S_p, (e.g. npc == &foo)
- find symbolic exploration trees T_0, …, T_k such that
  1. the root node of T_0 is S_t
  2. the leaf node n_k of T_k is S_p, and
  3. the leaf node n_j of T_j is the root node of n_{j+1} of T_{j+1}.



Design

Inputs: processor design + property.

Preprocessing



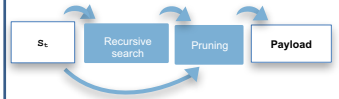
Choosing Payload Assertion

Payload assertion depends on the class of the property in the trigger state.

CF	Control Flow
XR	Exception Related
MA	Memory Access
IE	Correct Instructions
CR	Correctly Update Results

Building the Payload

- Run the recursive, backward search to find a sequence of instructions from S_t to S_p.
- Use dependency analysis to prune the search towards utilizing the signals in the trigger.



Example

Bug:

```
assign a_lt_b = comp_op[3] ? ((a[width-1] & !b[width-1]) |
  (!a[width-1] & !b[width-1] & result_sum[width-1]) |
  (a[width-1] & b[width-1] & result_sum[width-1])) :
  (a < b);                    // Bug Free Version
  result_sum[width-1];  // Buggy Version
```

Property:

```
~(
      ((or1200_ctrl.ex_insn & 'hFFE00000) >> 21  == 1826)
      &&
      (operand_a > operand_b)
)
|| (or1200_sprs.to_sr[9] == 1)
|| (rst == 1)
```

Trigger:

```
l.movhi     r16 0x8000
l.sfgtu     r16 r0
```

Payload:

```
l.bnf       attack_function
```

Undesired payload:

```
l.j         attack_function
```

Contact me at: martinmq@live.unc.edu

THE UNIVERSITY of NORTH CAROLINA at CHAPEL HILL