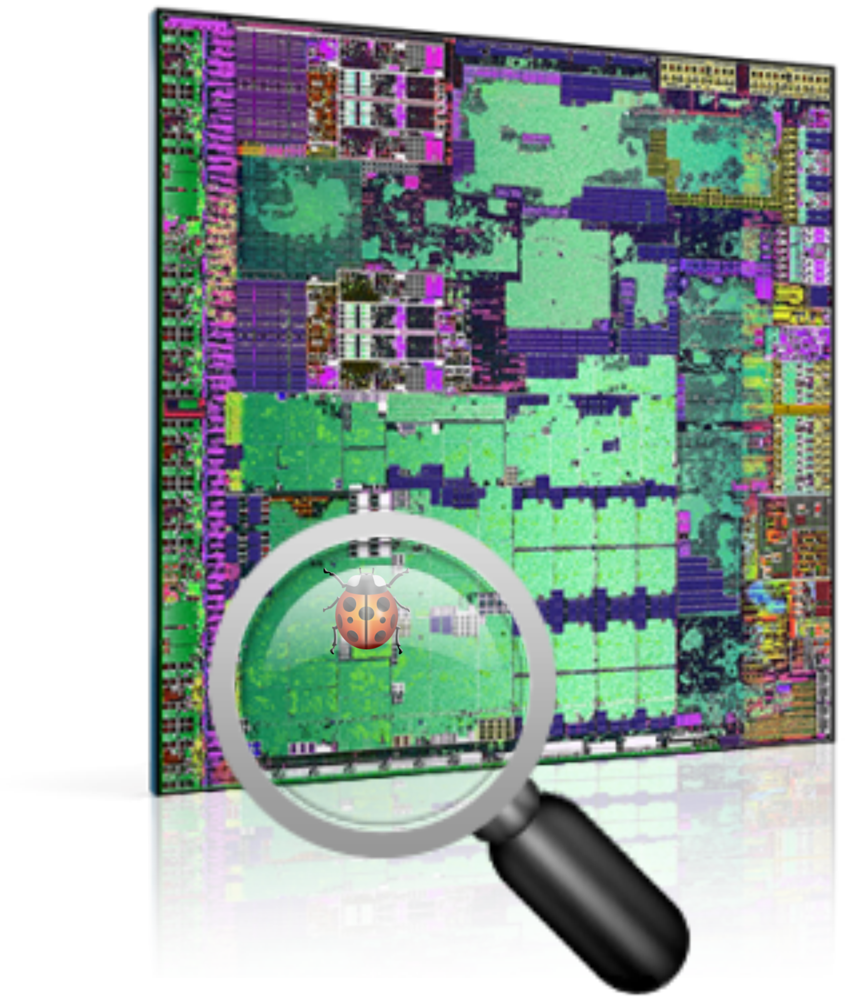
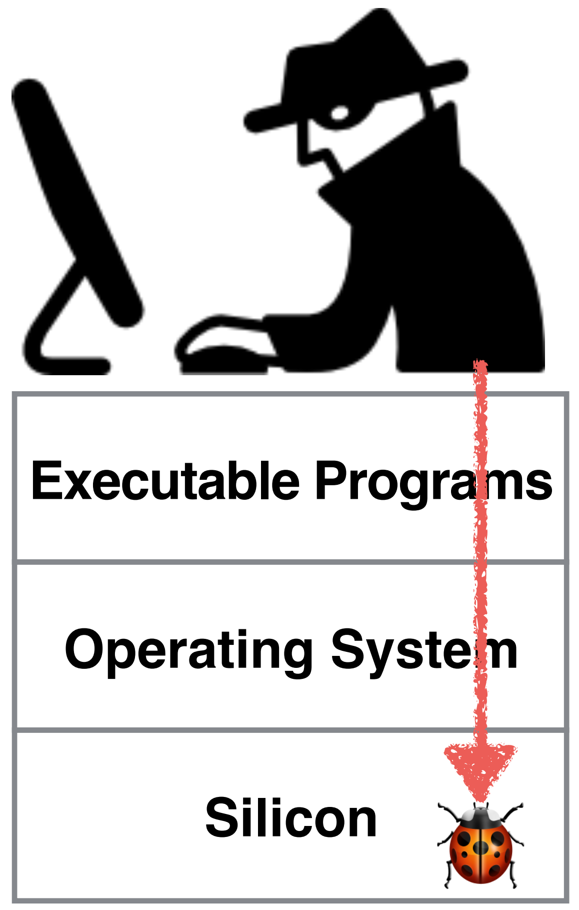
Identifying Security Critical Properties for the Dynamic Verification of a Processor

Rui Zhang, Natalie Stanley, Christopher Griggs, Andrew Chi, Cynthia Sturton

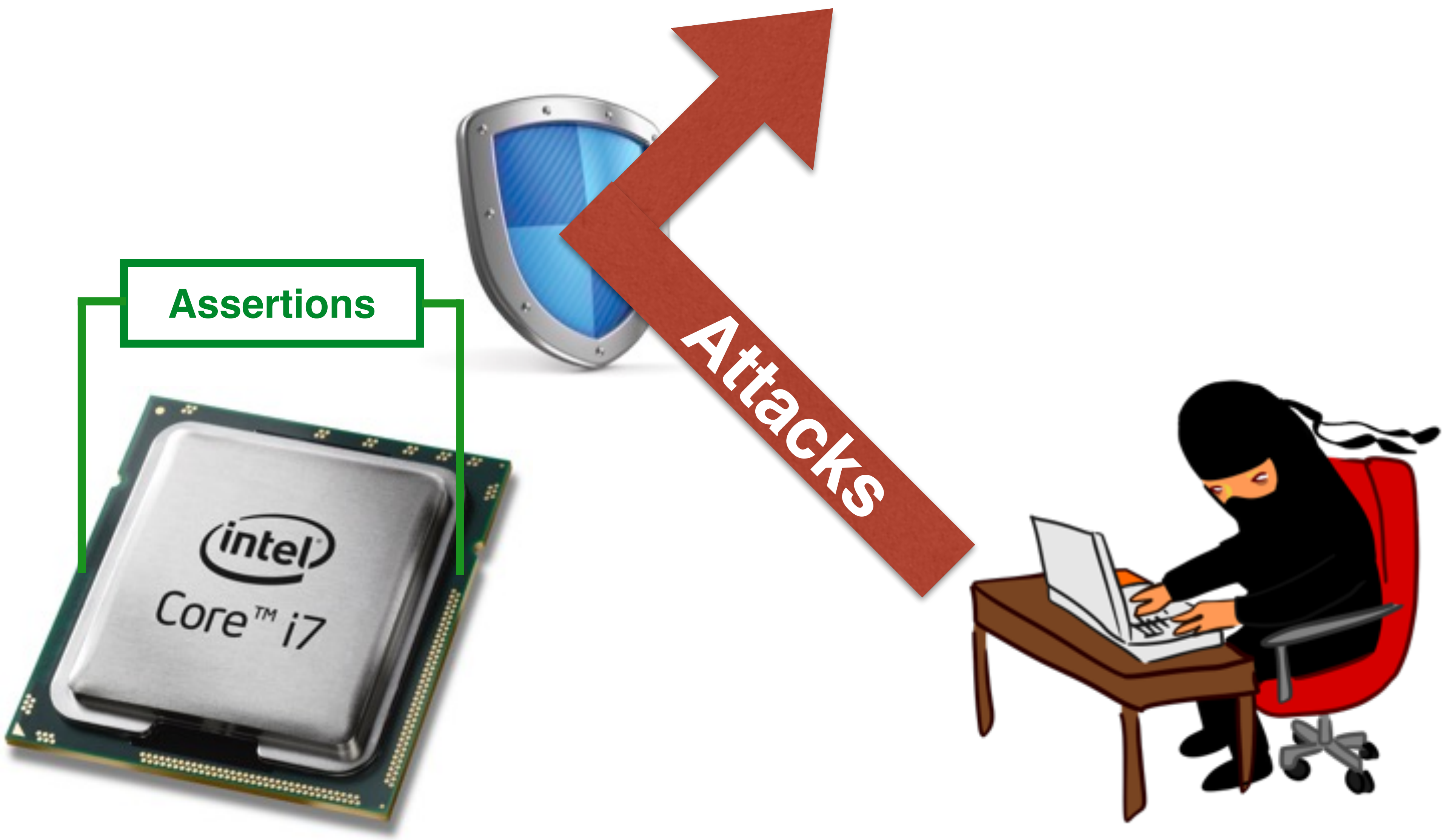
THE UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

Processor Bugs can Create Security Vulnerabilities





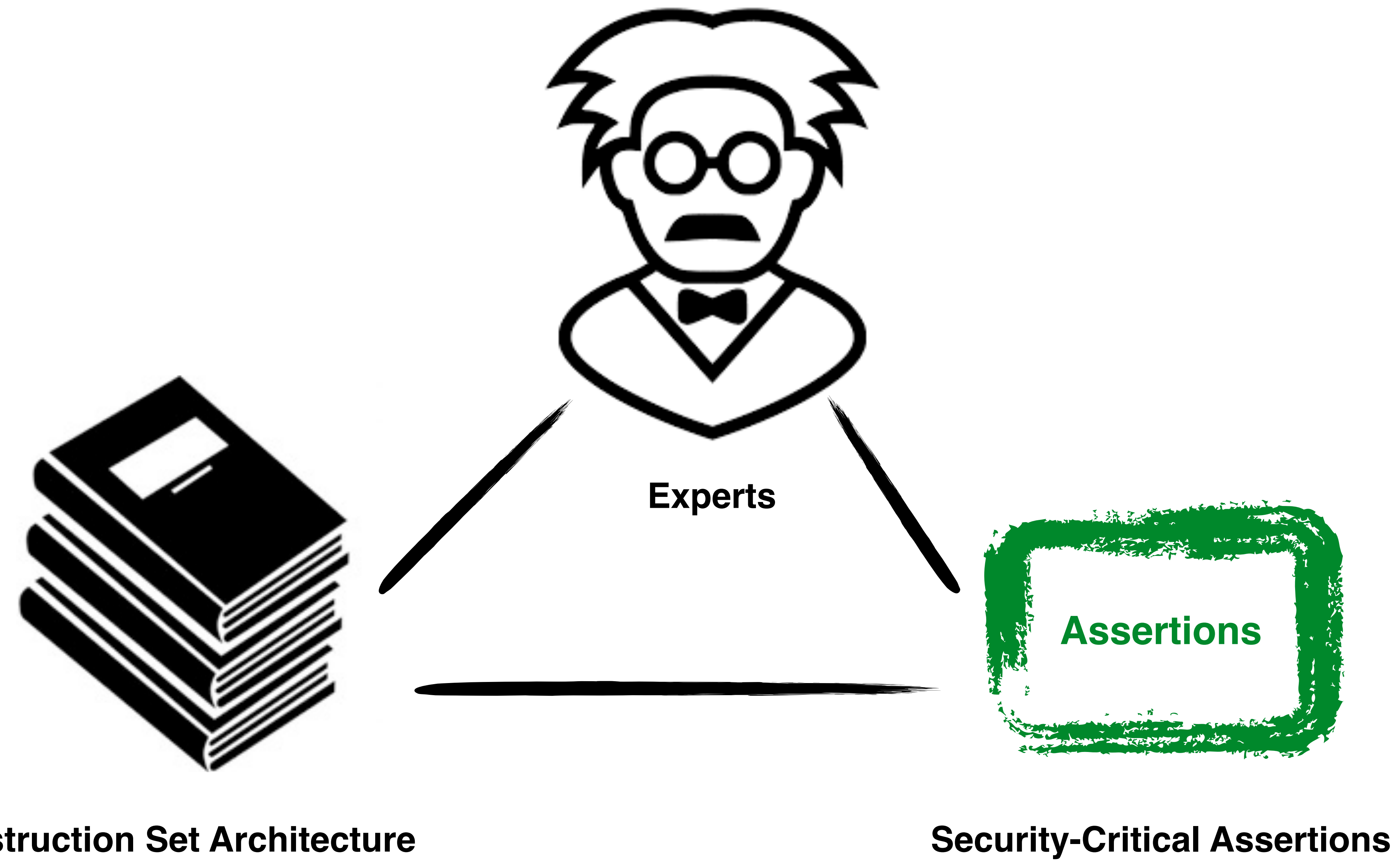
Dynamic Verification for Security



Research Question

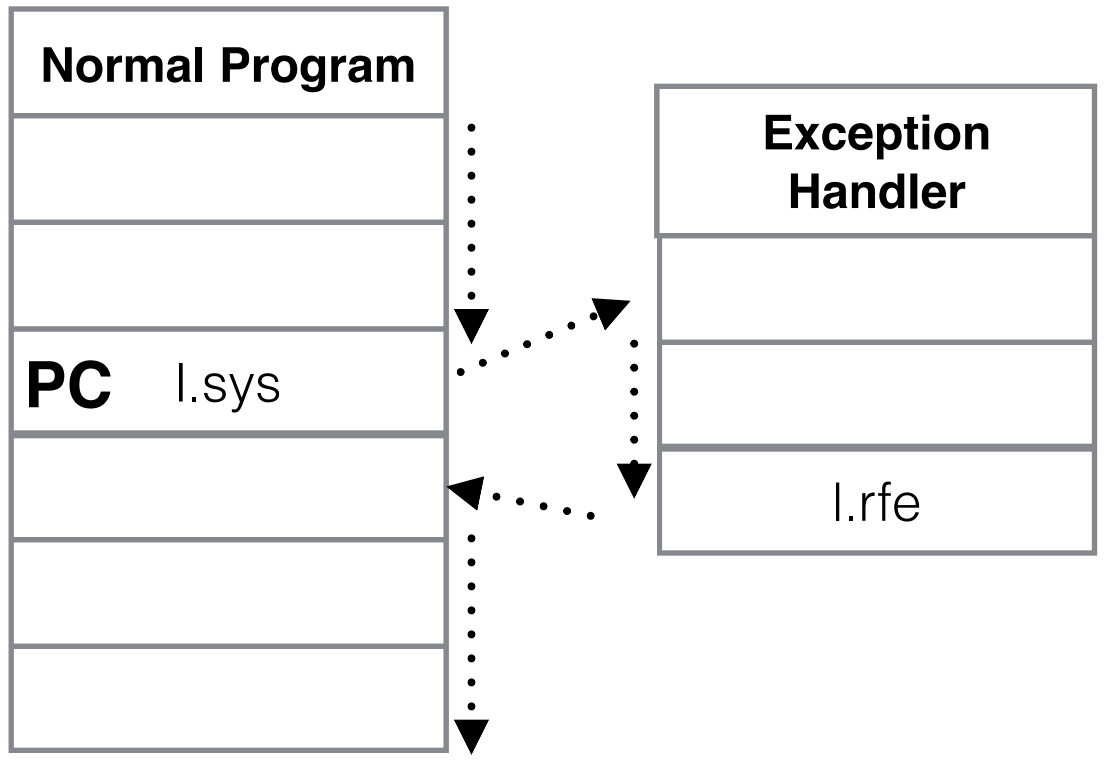
How to find the security properties we should protect for a processor?

The State of the Art: Human Expertise and Judgment



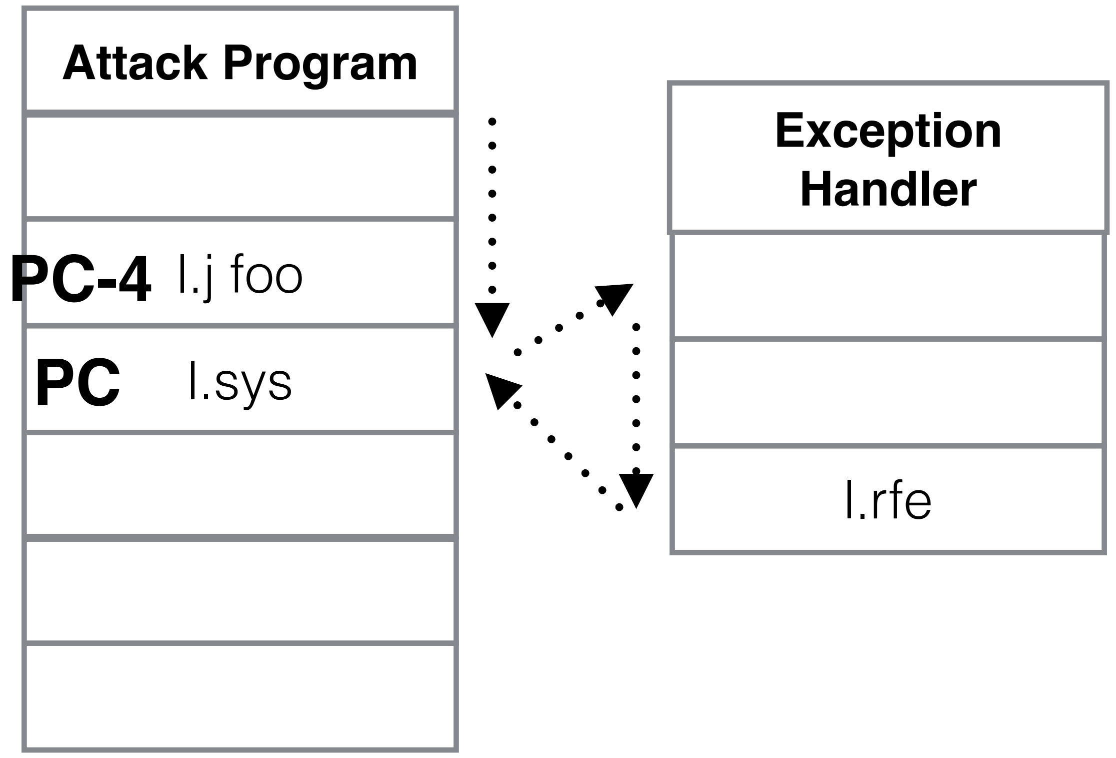
Vulnerability Example: DoS Attack

Normal



EPCR = PC + 4

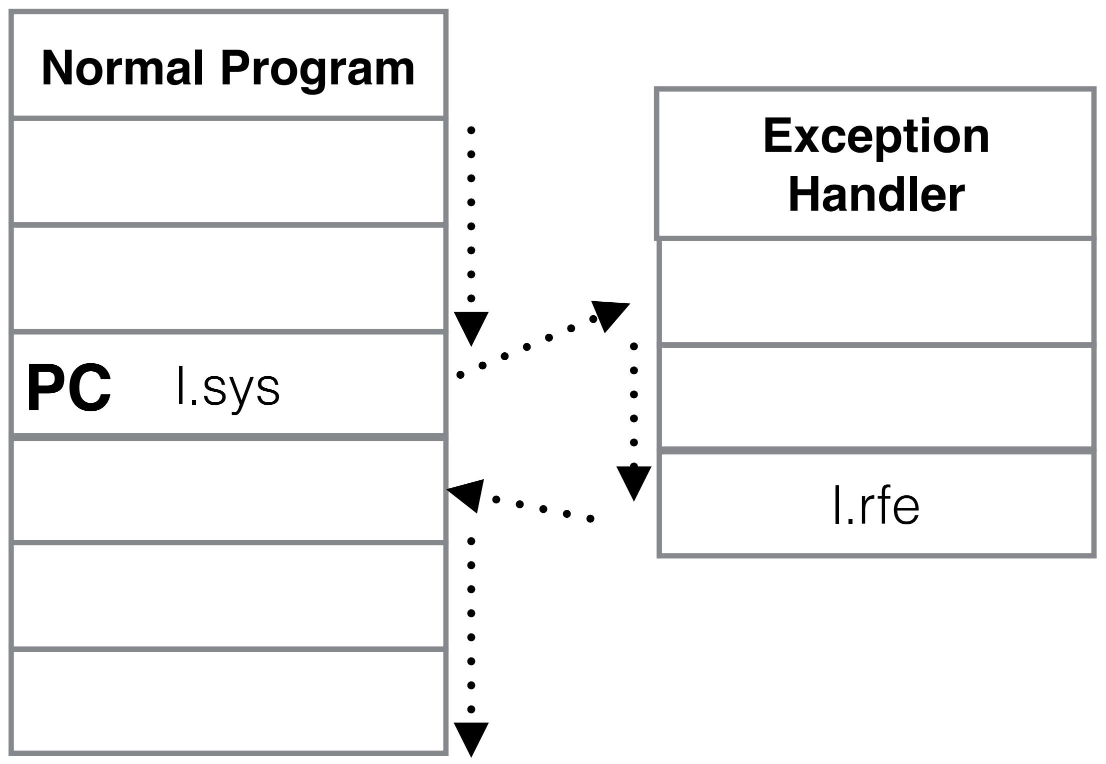
Syscall in Delay Slot



EPCR = PC

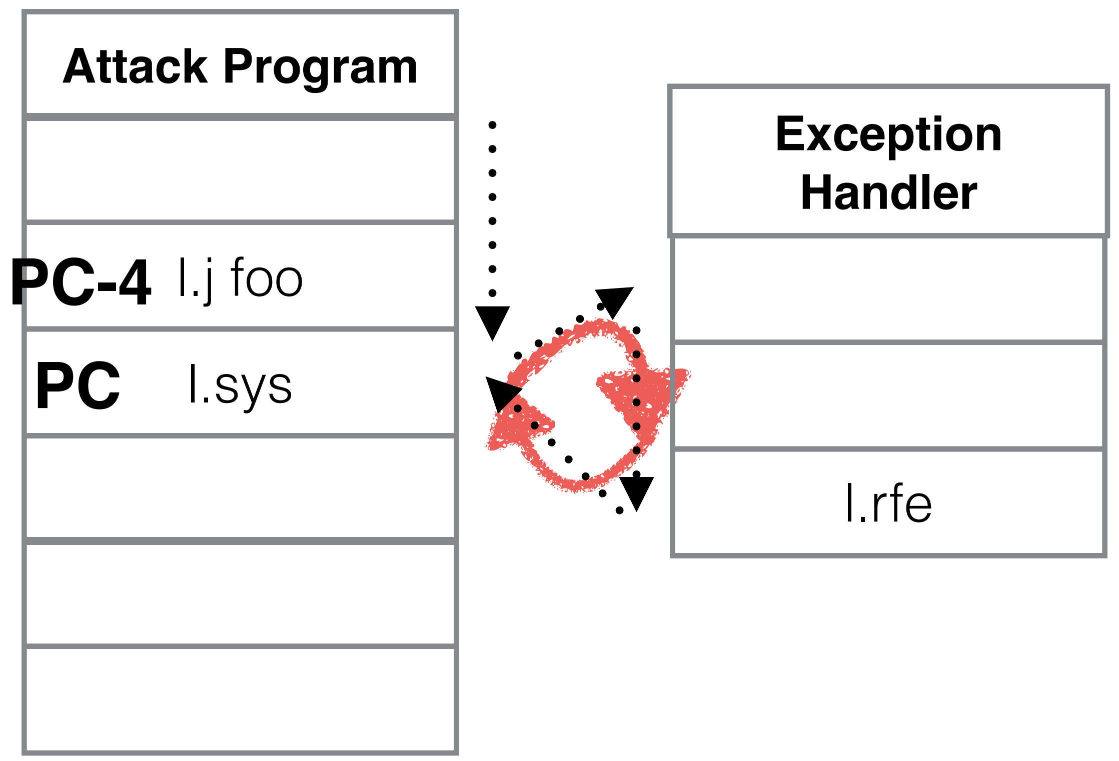
Vulnerability Example: DoS Attack

Normal



EPCR = PC + 4

Syscall in Delay Slot



EPCR = PC

Observation

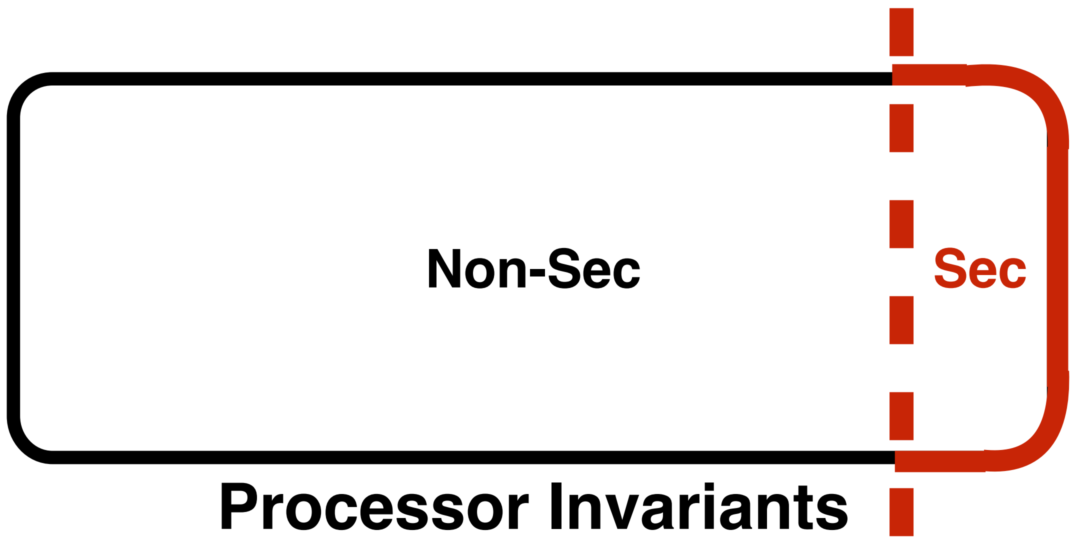
Observation:

- Security-critical bugs are vulnerabilities precisely because they violate some underlying security property

Goal:

- Identify security properties for a processor

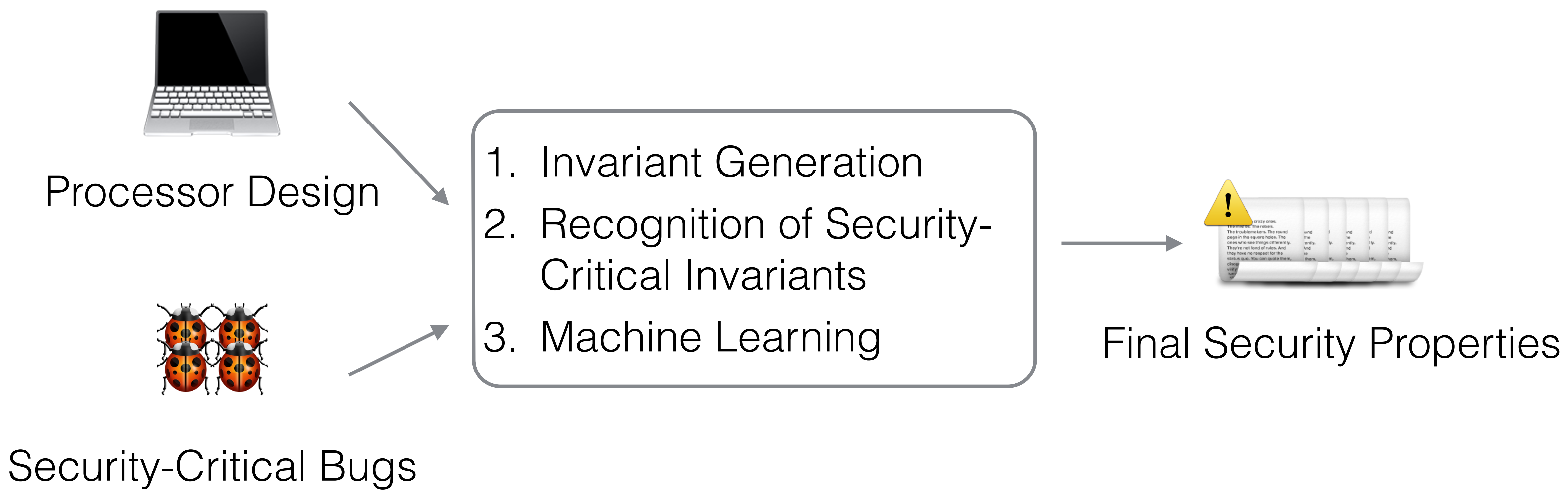
Our Approach



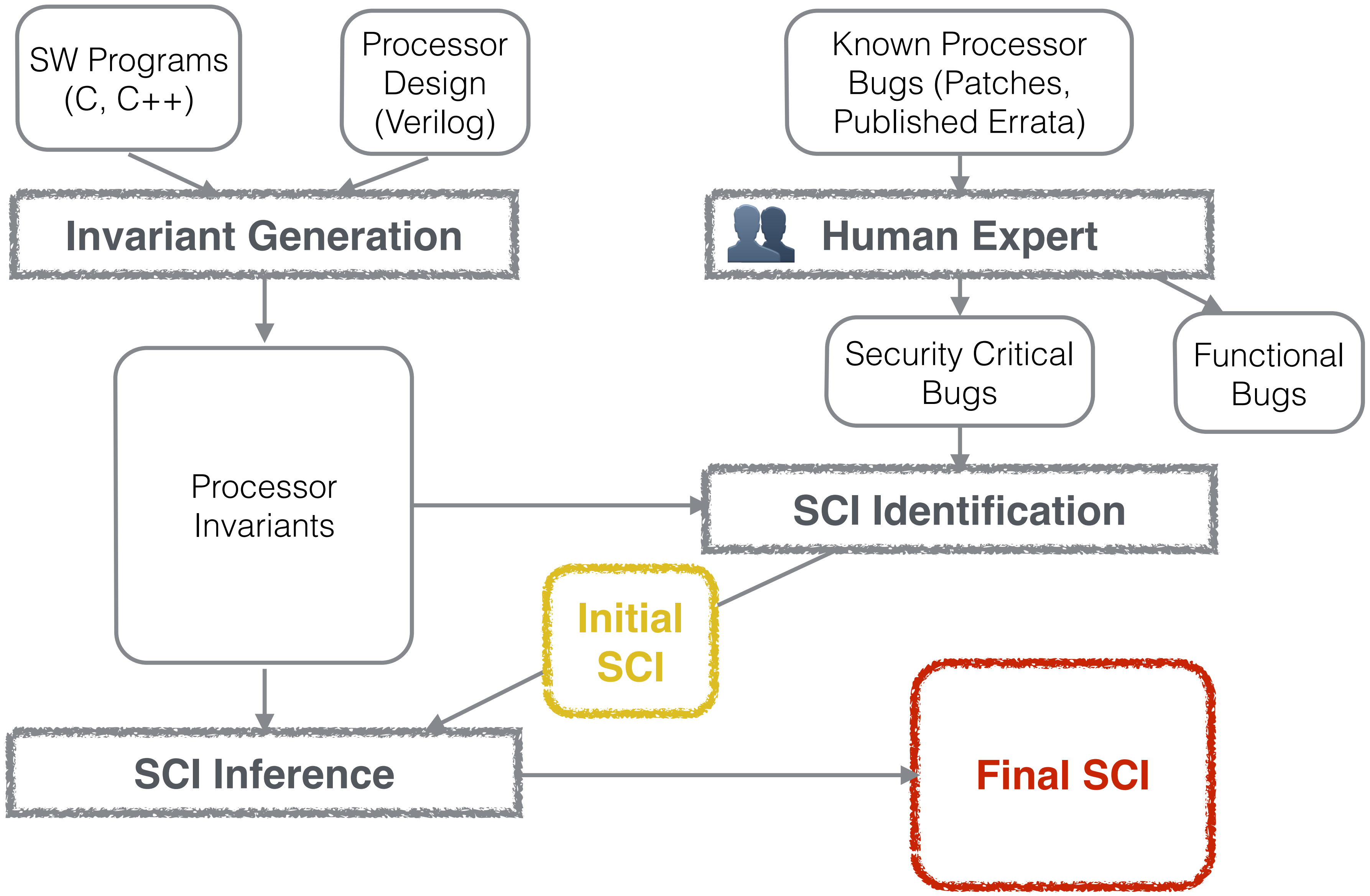
Known Security Bugs $\longrightarrow$ Security Properties

Machine Learning $\longrightarrow$ Additional Security Properties

Our Approach

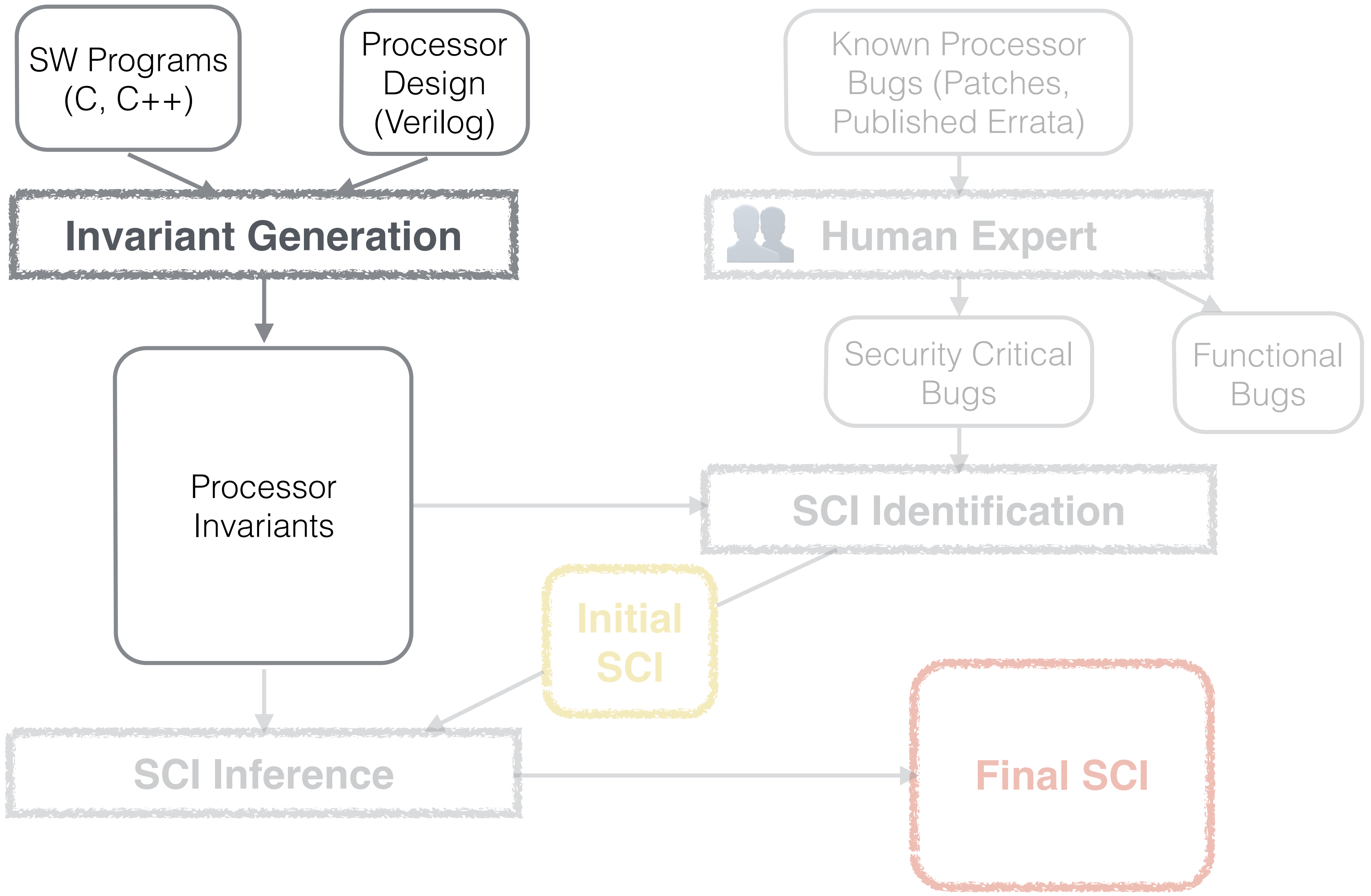


SCIFinder Tool Chain



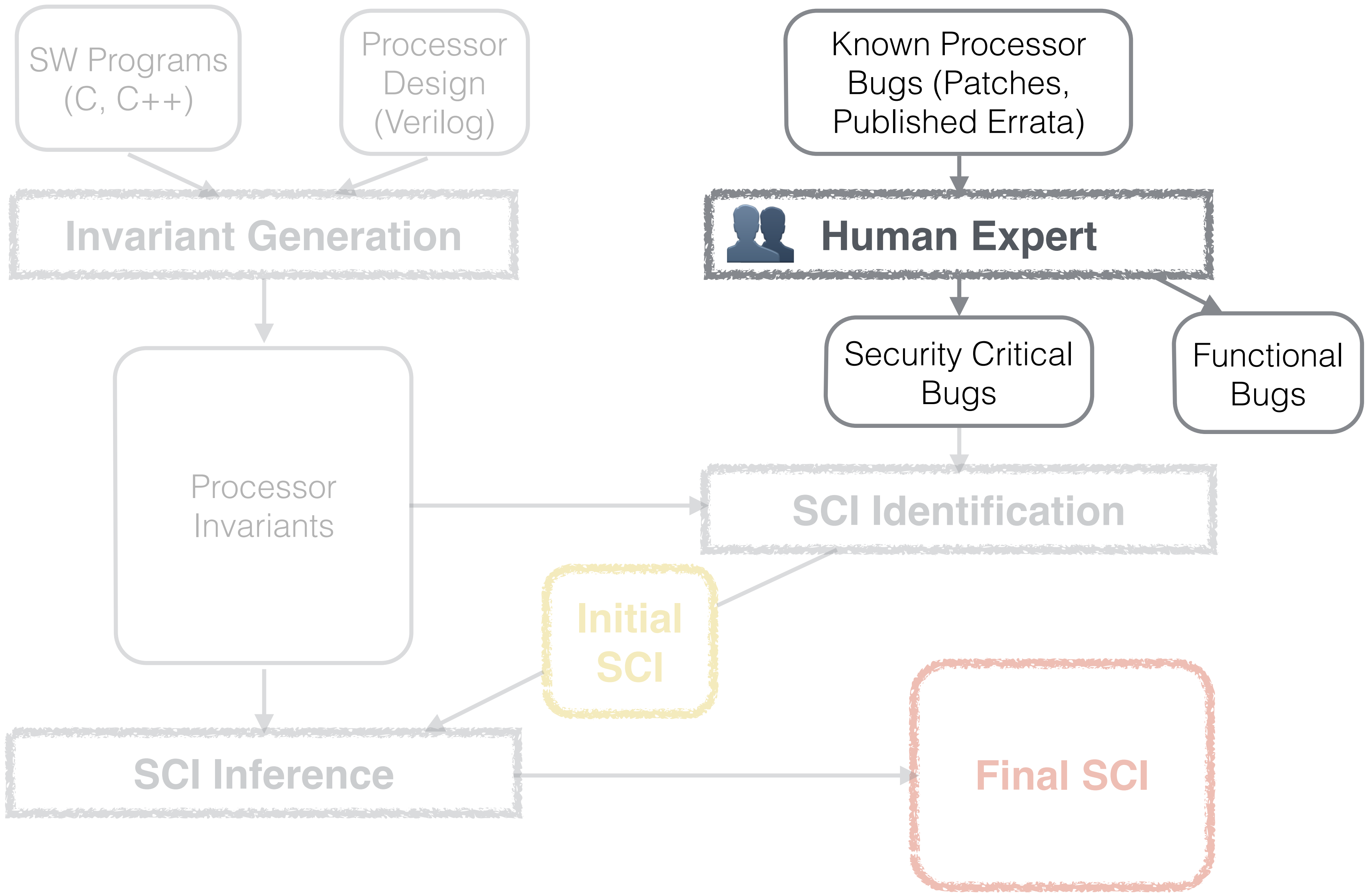
Workflow of SCIFinder

SCIFinder Tool Chain



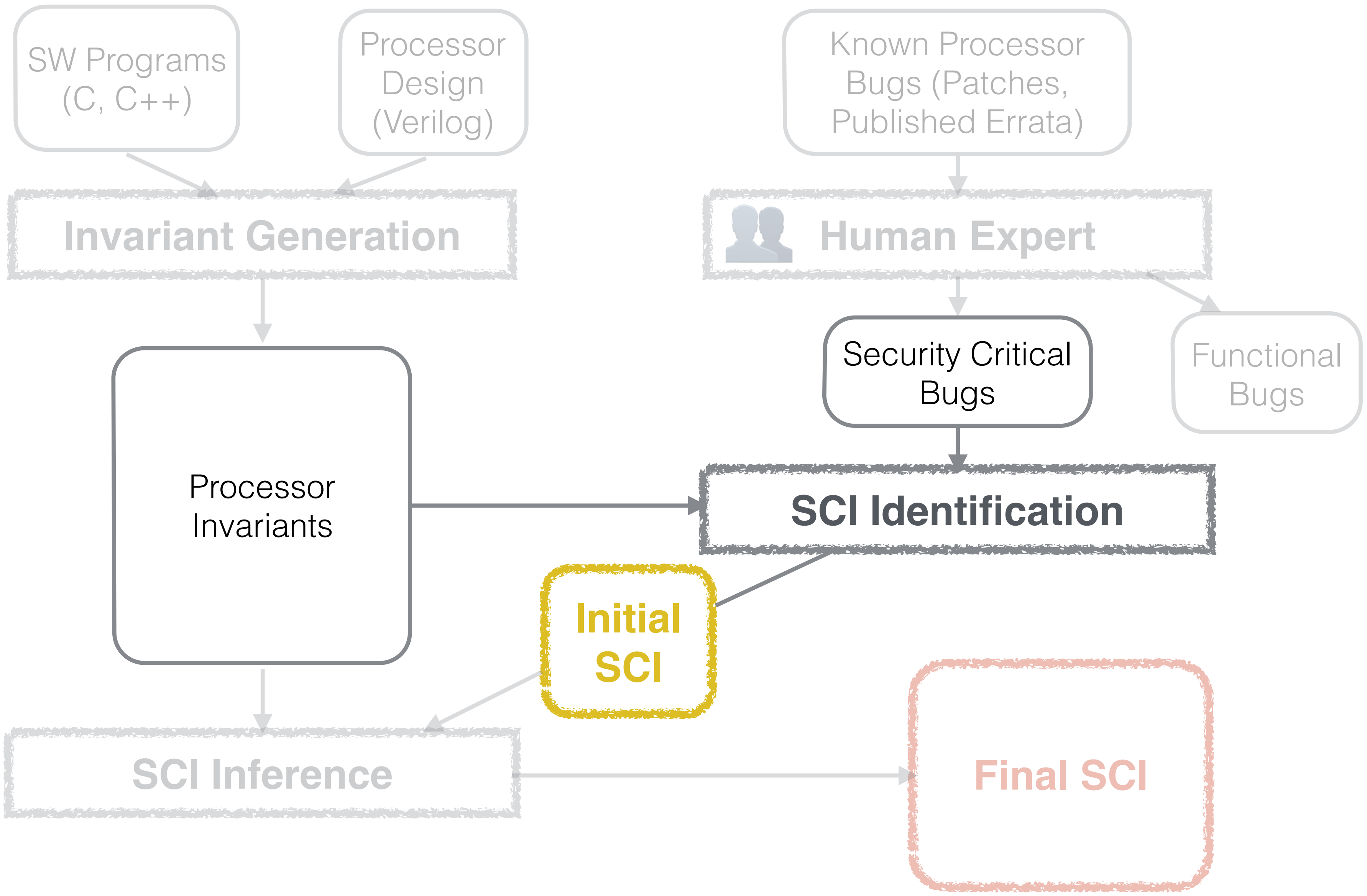
Workflow of SCIFinder

SCIFinder Tool Chain



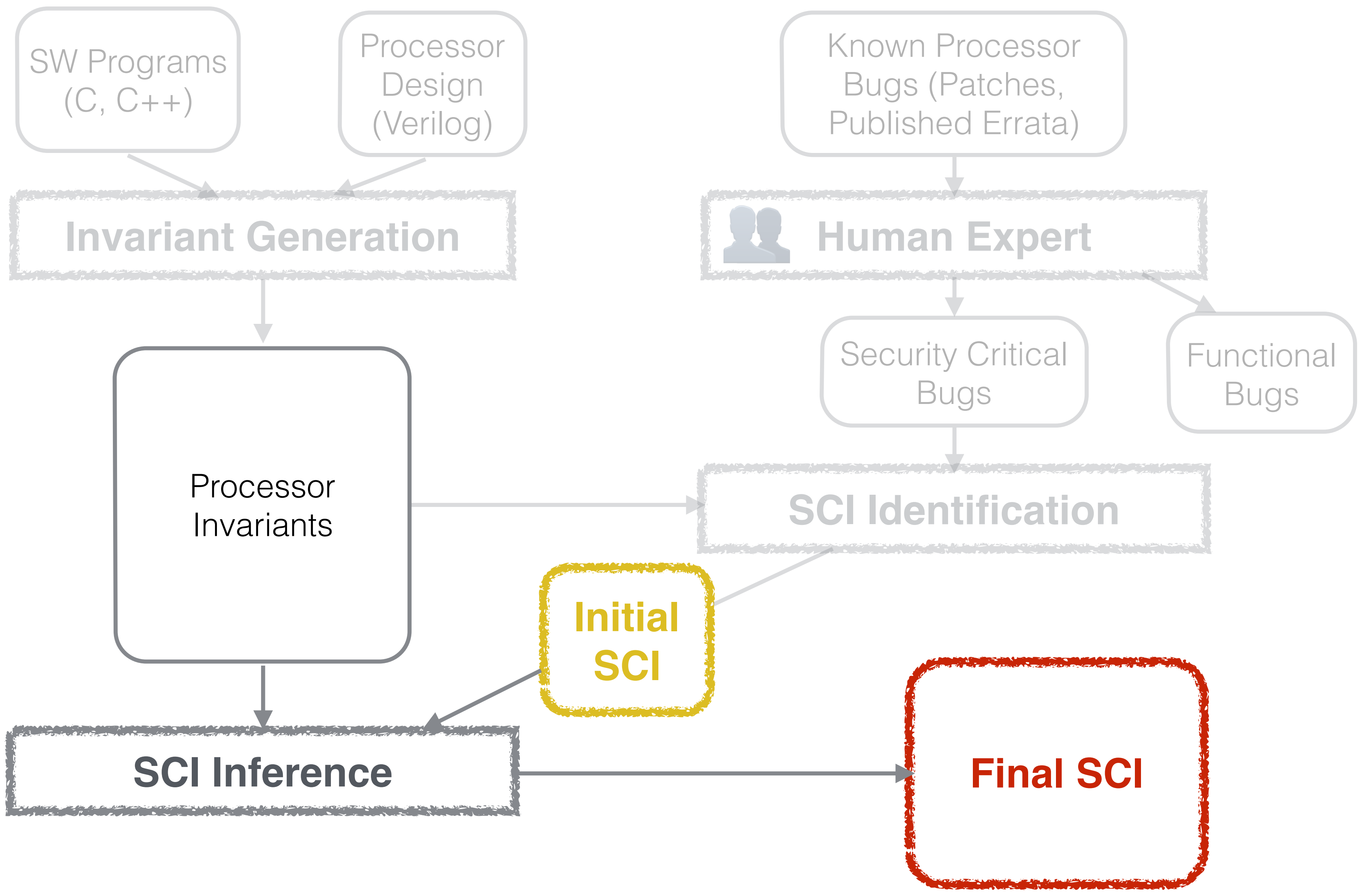
Workflow of SCIFinder

SCIFinder Tool Chain



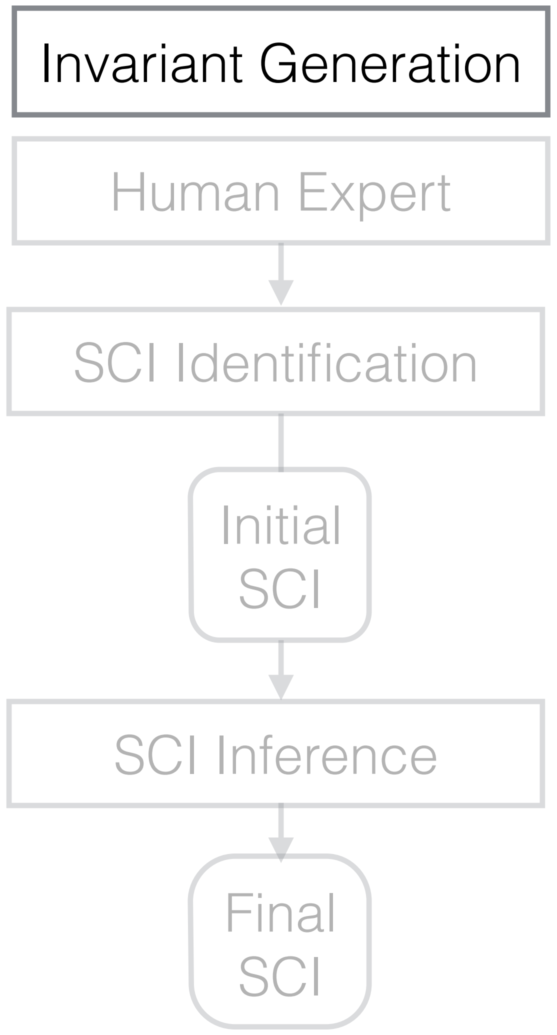
Workflow of SCIFinder

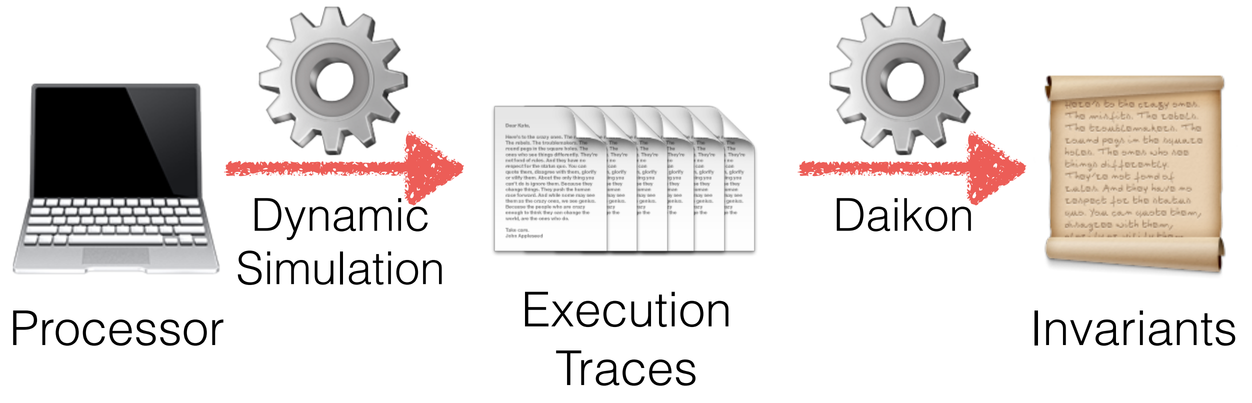
SCIFinder Tool Chain



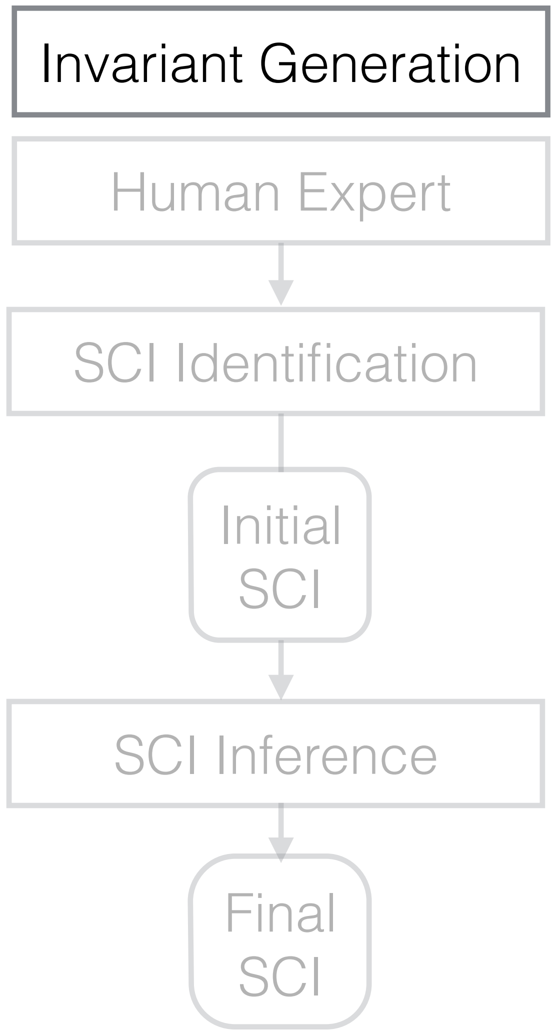
Workflow of SCIFinder

Invariant Generation





Invariant Generation



Simulation Tools:

Icarus Verilog

Execution Traces:

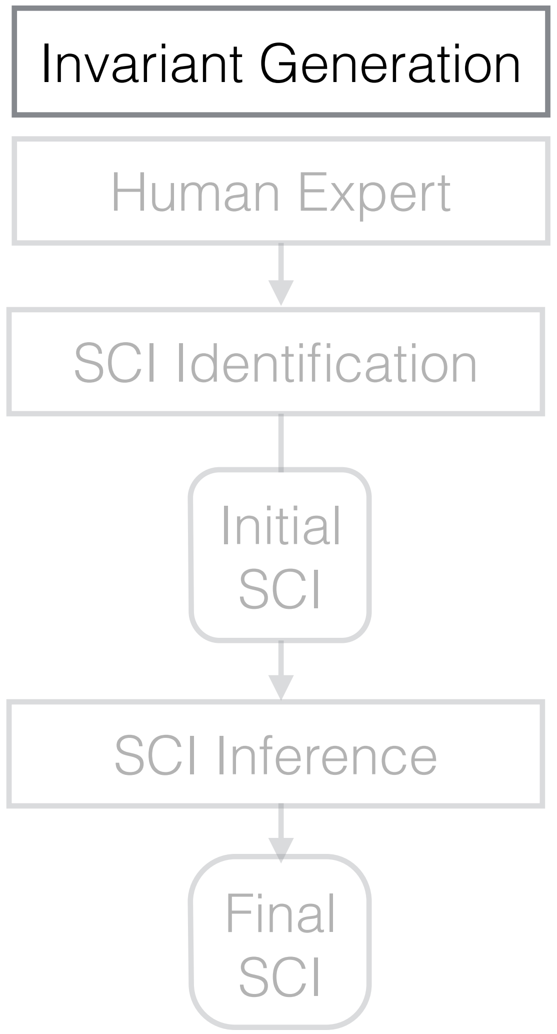
Program Counter

Instruction

```
......
EXECUTED(       1589): 000060b8:  e0c67000
GPR 0: 00000000  GPR 1: 00000001  GPR 2: 00000000  GPR 3: 00002640
GPR 4: 00000040  GPR 5: 00001000  GPR 6: 00000750  GPR 7: 00000008
GPR 8: 00000001  GPR 9: 00002038  GPR10: 00000000  GPR11: 00000000
GPR12: 00000000  GPR13: 00000100  GPR14: 00000010  GPR15: 00000000
GPR16: 00000000  GPR17: 00000000  GPR18: 00000000  GPR19: 00000000
GPR20: 00000000  GPR21: 00000000  GPR22: 00000000  GPR23: 00000000
GPR24: 00000000  GPR25: 00000000  GPR26: 00000000  GPR27: 00000000
GPR28: 00000000  GPR29: 00000000  GPR30: 00000000  GPR31: 00000000
SR   : 00008211  EPCR0: 00000000  EEAR0: 00000000  ESR0 : 00008001
......
```

ISA-level Variables

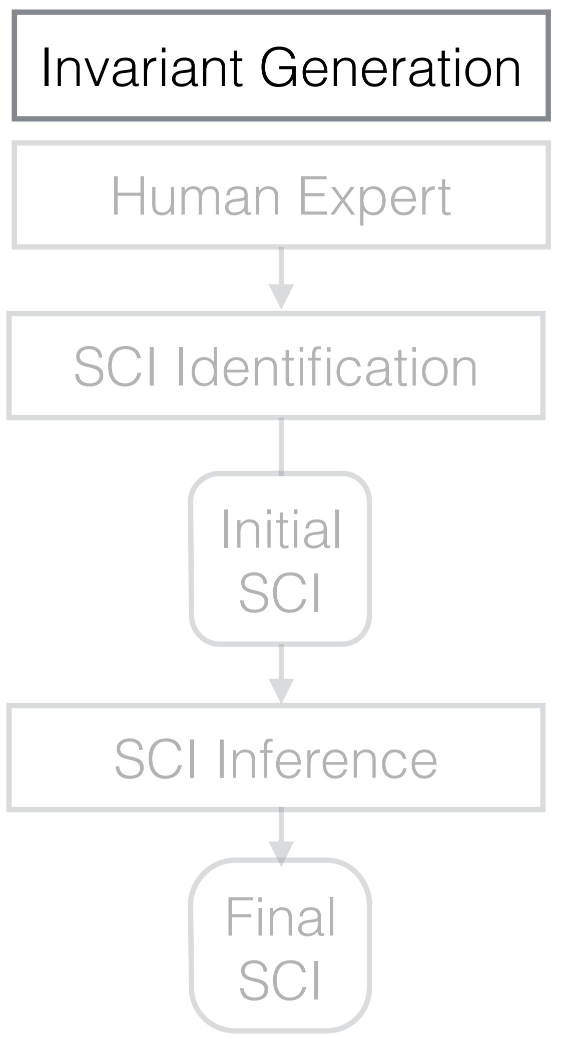
Invariant Generation



Daikon:

- A dynamic invariant detection tool
- An instrumenter: records information about variable values as a program executes
- An inference engine: reads the traces produced by the instrumenter to generate invariants

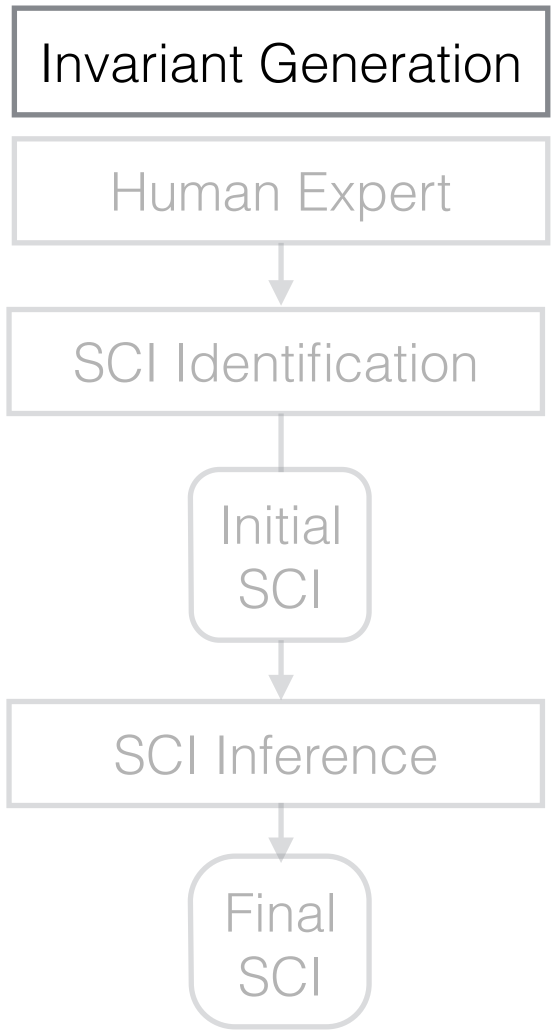
Invariant Generation



Adaptation:

- New Daikon Instrumenter: adapt Daikon to processor execution traces
- ISA-level variables: registers and signals visible to software

Invariant Generation



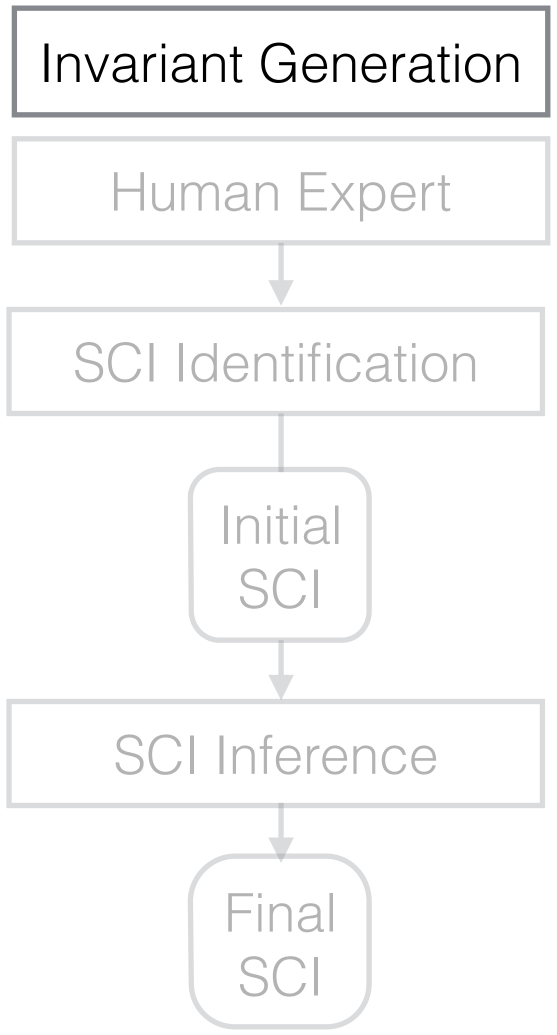
Adaptation:

- New Daikon Instrumenter: adapt Daikon to processor execution traces
- ISA-level variables: registers and signals visible to software
- Configurable: patterns unknown to Daikon, such as bit-packing

Supervision Register: 32-bit special-purpose supervisor-level register

31-28	27-17	16	15	14	13	12
Context ID	Reserved	SPRs User Mode Read Access	Fixed One	Exception Prefix High	Delay Slot Exception	Overflow Flag Exception
11	10	9	8	7	6	5
Overflow Flag	Carry Flag	Flag	CID Enable	Little Endian Enable	Instruction MMU Enable	Data MMU Enable
4	3	2	1	0		
Instructio n Cache Enable	Data Cache Enable	Interrupt Exception Enable	Tick Timer Exception Enable	Supervisor Mode		

Invariant Generation



Adaptation:

- New Daikon Instrumenter: adapt Daikon to processor execution traces
- ISA-level variables: registers and signals visible to software
- Configurable: patterns unknown to Daikon, such as bit-packing
- Carefully handle processor optimizations

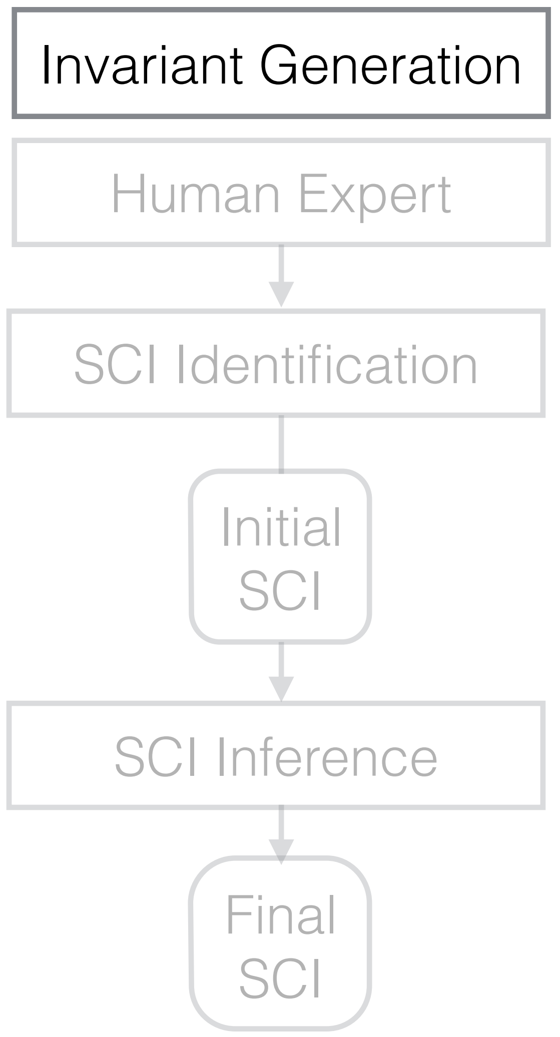
Delay Slot:

```
100 foo:
104     l.nop
108     l.j r9
200 main:
204     l.j foo
208     l.add
```

l.j NPC = PC + 4 ❌

l.j NPC = Target Address ✓

Invariant Generation



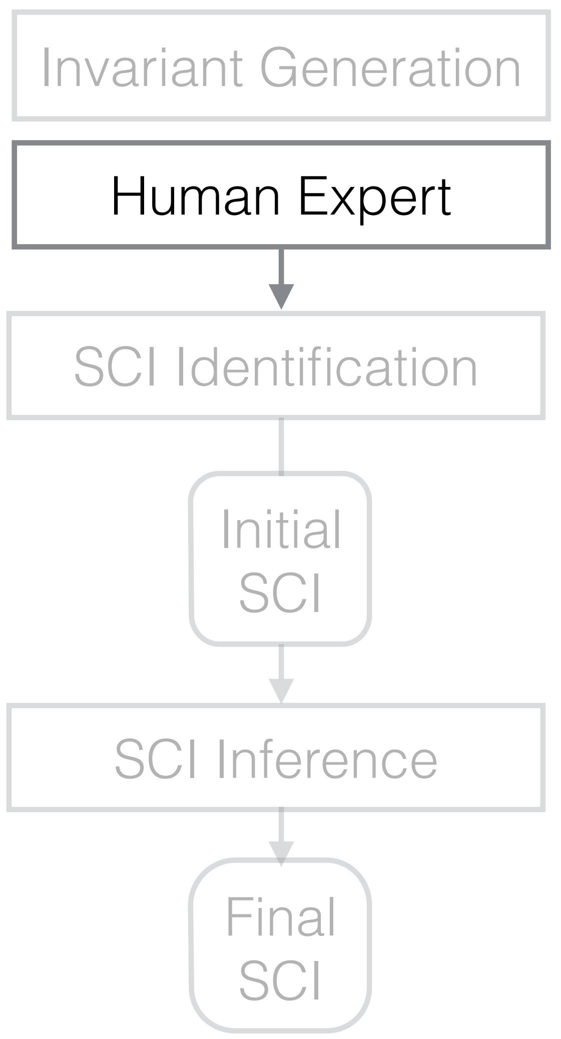
Invariant Format:

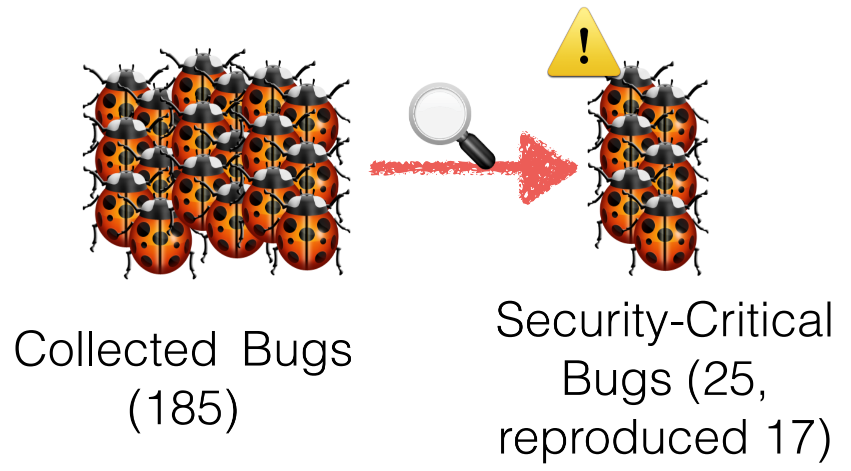
- $I \doteq$ *risingEdge*(*INSN*) → *EXPR*

Invariant Example:

- $I \doteq$ *risingEdge*(I.rfe) → SR = *orig*(ESR0)

Manually Classifying Bugs

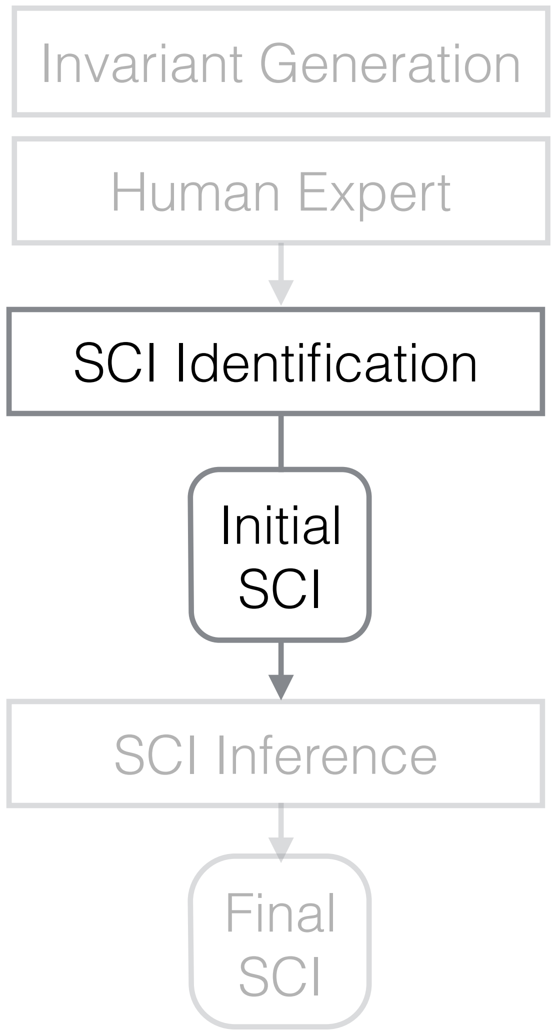




Sources:

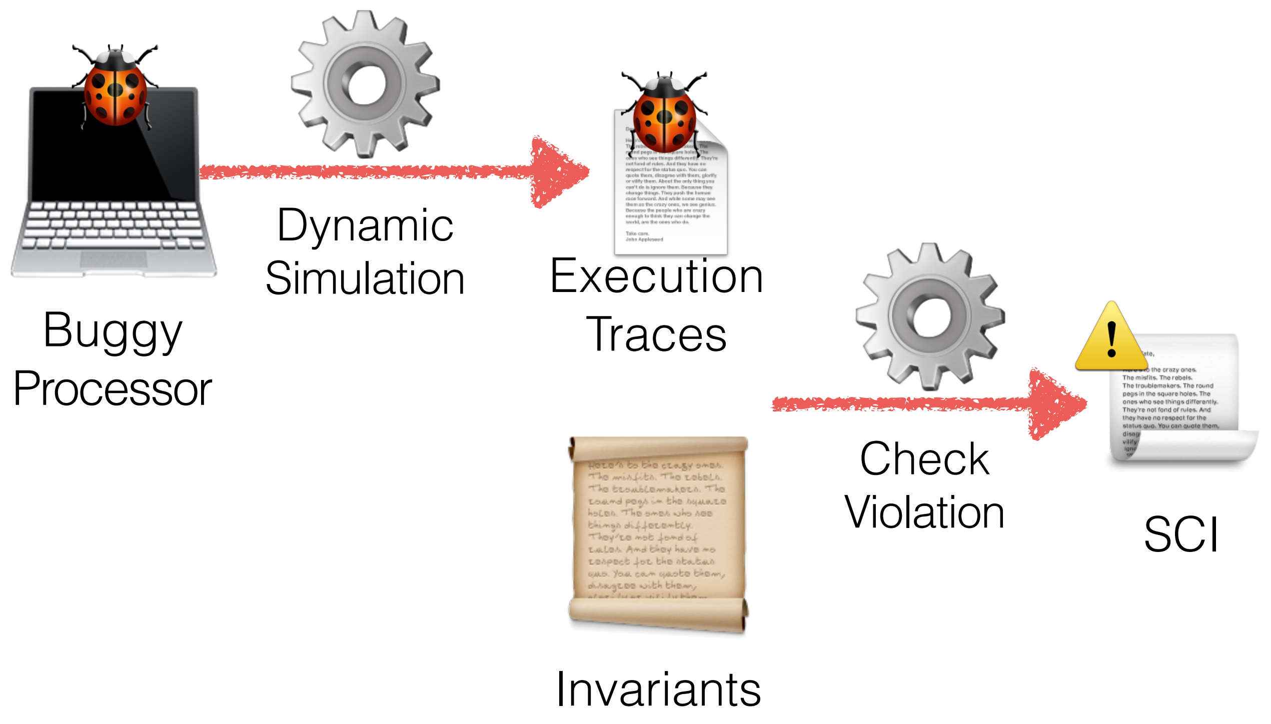
- Processors' bug tracker or Bugzilla sites
- Developers' mail archives
- Commits to the source repository
- Comments in the source code
- Published list of errata

Security-Critical Invariant Identification

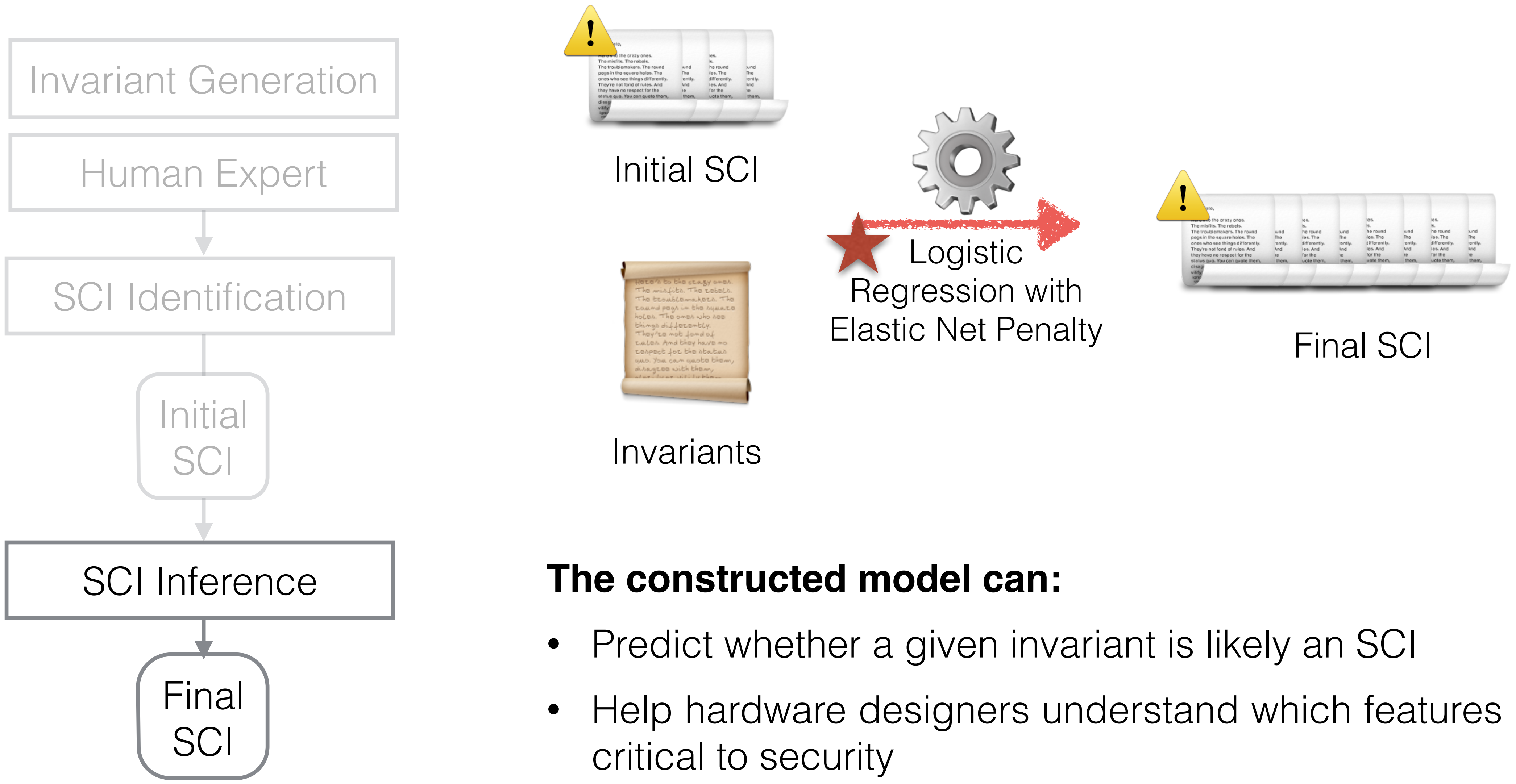


Key Observation:

- Security-critical bugs are vulnerabilities precisely because they violate some underlying security property



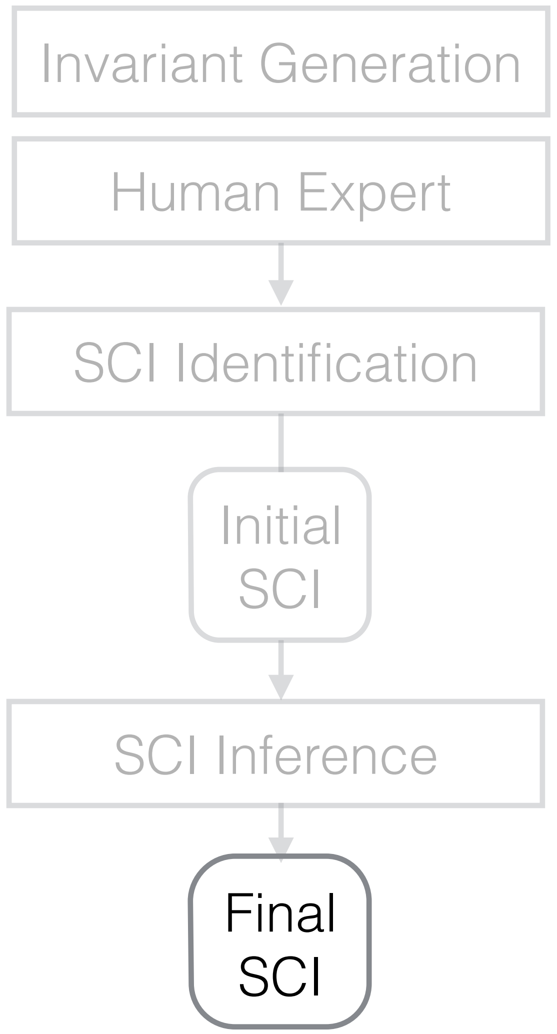
Security-Critical Invariant Inference



The constructed model can:

- Predict whether a given invariant is likely an SCI
- Help hardware designers understand which features are critical to security

False Positives



Sources:

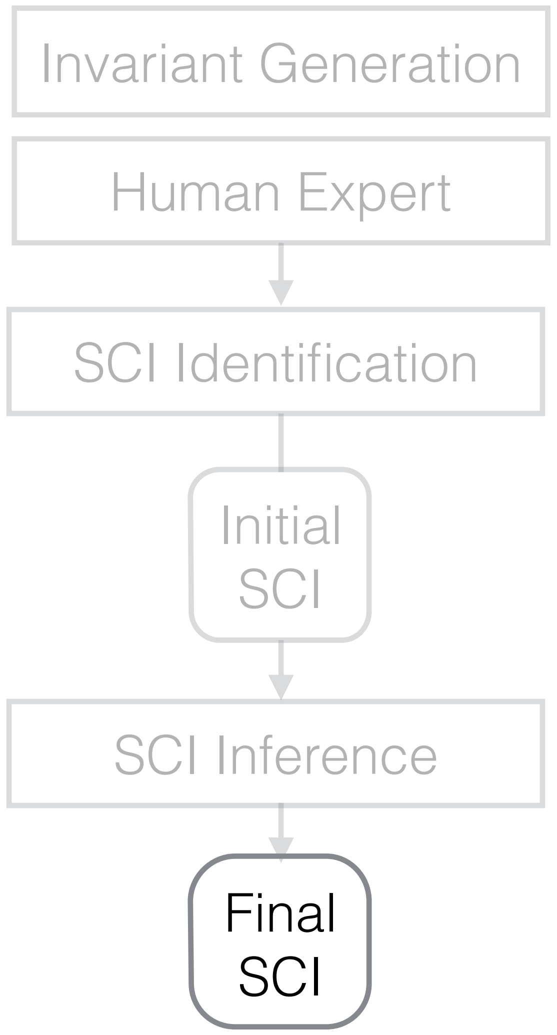
- Generate an invariant that isn't truly an invariant

Inadequate Test Suites

Daikon

Use of a Buggy Processor

False Positives



Sources:

- Generate an invariant that isn't truly an invariant

Solutions:

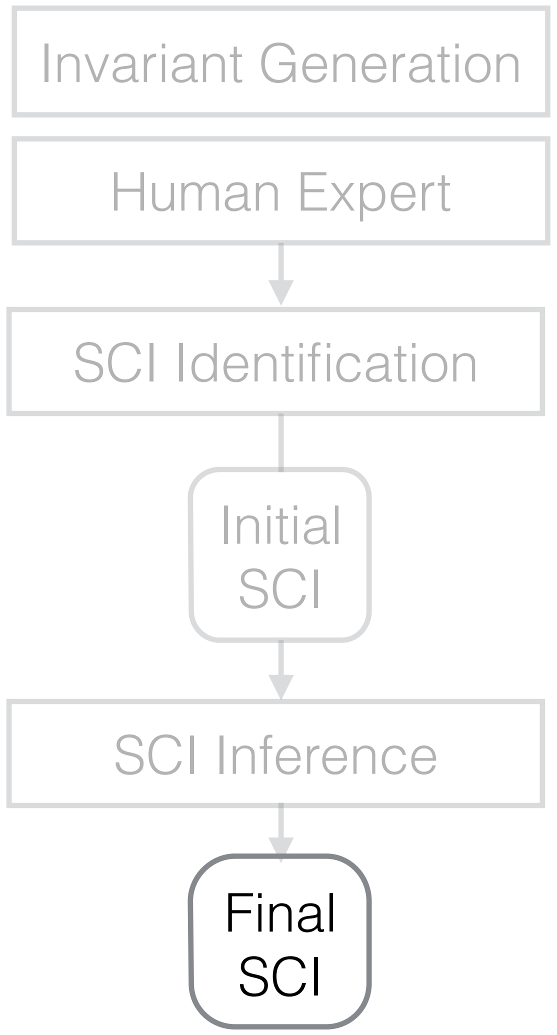
- Rely on human experts to manually remove them

Inadequate Test Suites

Daikon

Use of a Buggy Processor

False Positives



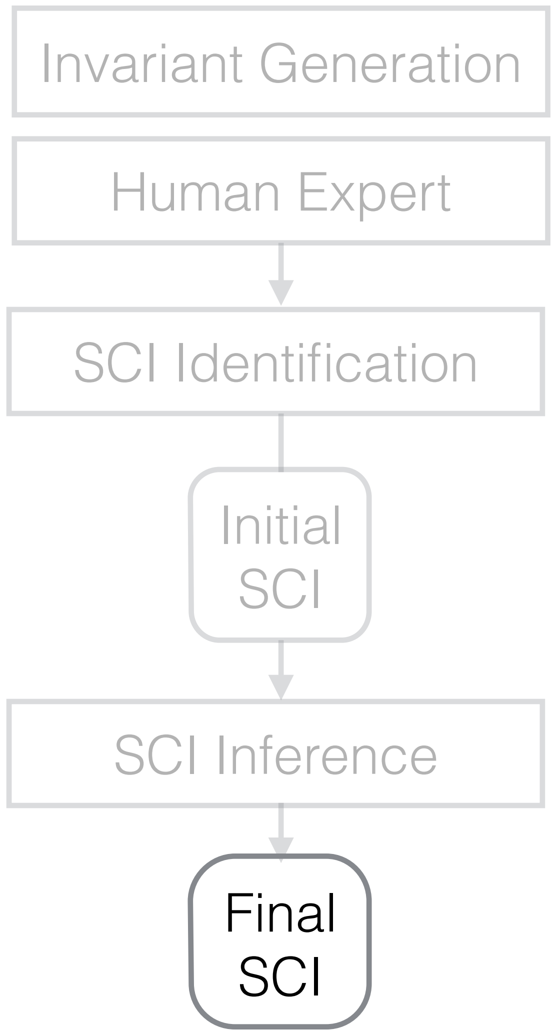
Sources:

- Generate an invariant that isn't truly an invariant
- Classify a non-SCI as security-critical

Solutions:

- Rely on human experts to manually remove them

False Positives



Sources:

- Generate an invariant that isn't truly an invariant
- Classify a non-SCI as security-critical

Solutions:

- Rely on human experts to manually remove them
- Draw a fine line between SCI and non-SCI, add more labeled data, refine machine learning model

Evaluation Methodology

Gather Real-world Security Vulnerabilities:

- Reproduce 17 security-critical bugs from open source processors
- Write attack programs that exploit the vulnerabilities

Generate Security-Critical Properties:

- Run normal programs and attack programs on affected processors
- Record execution traces
- Use SCIFinder to generate SCI

Compare with Prior Work:

- Collect 22 manually written security-critical properties from prior work
- Compare SCI generated by SCIFinder with manually written ones
- Add assertions to detect unknown bugs

Main Results

Security Properties

22

Manually Intensive

SCI Finder

19 + 3 ← **New**

Semi-automatic

Manual Effort: classifying bugs, validating the reported SCI

Results: Comparison to State-of-the-Art

No.	Security Property Description	Found?
p1	Execution privilege matches page privilege	✓
p2	SPR equals GPR in register move instruction	b12
p3	Updates to exception registers make sense	b4 b9 b15
p4	Destination matches the target	✓
p5	Memory value in equals register value out	b14
p6	Register value in equals memory value out	b16 b17
p7	Memory address equals effective address	✓
p8	Privilege escalates correctly	✓
p9	Privilege deescalates correctly	✓
p10	Jumps update the PC correctly	✗
p11	Jumps update the LR correctly	b13
p12	Instruction is in a valid format	b11
p13	Continuous Control Flow	b5
p14	Exception return updates state correctly	b1 b5
p15	Register change implies that it is the instruction target	✓
p16	SR is not written to a GPR in user mode	✗
p17	Interrupt implies handled	b8
p18	Instruction unchanged in pipeline	--
p19	SPR modified only in supervisor mode	✓
p20	Enter supervisor mode is on reset or exception	✓
p21	Exception handling implies exception mechanism activated	b8
p22	Unspecified custom instructions are not allowed	✗
p23	Exception handler accessed only during exception, in supervisor mode, or on reset	b8
p24	Page fault generated if MMU detects an access control violation	--
p25	UART output changes on a write command from CPU	--
p26	Only transmit command or initialization change Ethernet data output	--
p27	Debug Unit's value and control registers only accessible from supervisor mode	--

Properties from SPECS

[H.S.K. ASPLOS 2015]

Properties from Security-Checker

[B.H.I. HOST 2011]

Results: Comparison to State-of-the-Art

No.	Security Property Description	Found?
p1	Execution privilege matches page privilege	✓
p2	SPR equals GPR in register move instruction	b12
p3	Updates to exception registers make sense	b4 b9 b15
p4	Destination matches the target	✓
p5	Memory value in equals register value out	b14
p6	Register value in equals memory value out	b16 b17
p7	Memory address equals effective address	✓
p8	Privilege escalates correctly	✓
p9	Privilege deescalates correctly	✓
p10	Jumps update the PC correctly	✗
p11	Jumps update the LR correctly	b13
p12	Instruction is in a valid format	b11
p13	Continuous Control Flow	b5
p14	Exception return updates state correctly	b1 b5
p15	Register change implies that it is the instruction target	✓
p16	SR is not written to a GPR in user mode	✗
p17	Interrupt implies handled	b8
p18	Instruction unchanged in pipeline	--
p19	SPR modified only in supervisor mode	✓
p20	Enter supervisor mode is on reset or exception	✓
p21	Exception handling implies exception mechanism activated	b8
p22	Unspecified custom instructions are not allowed	✗
p23	Exception handler accessed only during exception, in supervisor mode, or on reset	b8
p24	Page fault generated if MMU detects an access control violation	--
p25	**UART output changes on a write command from CPU**	--
p26	**Only transmit command or initialization change Ethernet data output**	--
p27	**Debug Unit's value and control registers only accessible from supervisor mode**	--

Properties Outside of Processor Core

Results: Comparison to State-of-the-Art

No.	Security Property Description	Found?
p1	Execution privilege matches page privilege	✓
p2	SPR equals GPR in register move instruction	b12
p3	Updates to exception registers make sense	b4 b9 b15
p4	Destination matches the target	✓
p5	Memory value in equals register value out	b14
p6	Register value in equals memory value out	b16 b17
p7	Memory address equals effective address	✓
p8	Privilege escalates correctly	✓
p9	Privilege deescalates correctly	✓
p10	Jumps update the PC correctly	✗
p11	Jumps update the LR correctly	b13
p12	Instruction is in a valid format	b11
p13	Continuous Control Flow	b5
p14	Exception return updates state correctly	b1 b5
p15	Register change implies that it is the instruction target	✓
p16	SR is not written to a GPR in user mode	✗
p17	Interrupt implies handled	b8
p18	**Instruction unchanged in pipeline**	--
p19	SPR modified only in supervisor mode	✓
p20	Enter supervisor mode is on reset or exception	✓
p21	Exception handling implies exception mechanism activated	b8
p22	Unspecified custom instructions are not allowed	✗
p23	Exception handler accessed only during exception, in supervisor mode, or on reset	b8
p24	**Page fault generated if MMU detects an access control violation**	--
p25	UART output changes on a write command from CPU	--
p26	Only transmit command or initialization change Ethernet data output	--
p27	Debug Unit's value and control registers only accessible from supervisor mode	--

Properties Needing Micro-architectural States

Results: Comparison to State-of-the-Art

No.	Security Property Description	Found?
p1	Execution privilege matches page privilege	✓
p2	**SPR equals GPR in register move instruction**	**b12**
p3	**Updates to exception registers make sense**	**b4 b9 b15**
p4	Destination matches the target	✓
p5	**Memory value in equals register value out**	**b14**
p6	**Register value in equals memory value out**	**b16 b17**
p7	Memory address equals effective address	✓
p8	Privilege escalates correctly	✓
p9	Privilege deescalates correctly	✓
p10	Jumps update the PC correctly	✗
p11	**Jumps update the LR correctly**	**b13**
p12	**Instruction is in a valid format**	**b11**
p13	**Continuous Control Flow**	**b5**
p14	**Exception return updates state correctly**	**b1 b5**
p15	Register change implies that it is the instruction target	✓
p16	SR is not written to a GPR in user mode	✗
p17	**Interrupt implies handled**	**b8**
p18	Instruction unchanged in pipeline	--
p19	SPR modified only in supervisor mode	✓
p20	Enter supervisor mode is on reset or exception	✓
p21	**Exception handling implies exception mechanism activated**	**b8**
p22	Unspecified custom instructions are not allowed	✗
p23	**Exception handler accessed only during exception, in supervisor mode, or on**	**b8**
p24	Page fault generated if MMU detects an access control violation	--
p25	UART output changes on a write command from CPU	--
p26	Only transmit command or initialization change Ethernet data output	--
p27	Debug Unit's value and control registers only accessible from supervisor mode	--

Properties Found in the Identification Step

Results: Comparison to State-of-the-Art

No.	Security Property Description	Found?
p1	Execution privilege matches page privilege	✓
p2	**SPR equals GPR in register move instruction**	**b12**
p3	**Updates to exception registers make sense**	**b4 b9 b15** 👉
p4	Destination matches the target	✓
p5	**Memory value in equals register value out**	**b14**
p6	**Register value in equals memory value out**	**b16 b17**
p7	Memory address equals effective address	✓
p8	Privilege escalates correctly	✓
p9	Privilege deescalates correctly	✓
p10	Jumps update the PC correctly	✗
p11	**Jumps update the LR correctly**	**b13**
p12	**Instruction is in a valid format**	**b11**
p13	**Continuous Control Flow**	**b5**
p14	**Exception return updates state correctly**	**b1 b5**
p15	Register change implies that it is the instruction target	✓
p16	SR is not written to a GPR in user mode	✗
p17	**Interrupt implies handled**	**b8**
p18	Instruction unchanged in pipeline	--
p19	SPR modified only in supervisor mode	✓
p20	Enter supervisor mode is on reset or exception	✓
p21	**Exception handling implies exception mechanism activated**	**b8**
p22	Unspecified custom instructions are not allowed	✗
p23	**Exception handler accessed only during exception, in supervisor mode, or on**	**b8**
p24	Page fault generated if MMU detects an access control violation	--
p25	UART output changes on a write command from CPU	--
p26	Only transmit command or initialization change Ethernet data output	--
p27	Debug Unit's value and control registers only accessible from supervisor mode	--

One property can be identified from different bugs

Results: Comparison to State-of-the-Art

No.	Security Property Description	Found?
p1	Execution privilege matches page privilege	✓
p2	**SPR equals GPR in register move instruction**	**b12**
p3	**Updates to exception registers make sense**	**b4 b9 b15**
p4	Destination matches the target	✓
p5	**Memory value in equals register value out**	**b14**
p6	**Register value in equals memory value out**	**b16 b17**
p7	Memory address equals effective address	✓
p8	Privilege escalates correctly	✓
p9	Privilege deescalates correctly	✓
p10	Jumps update the PC correctly	✗
p11	**Jumps update the LR correctly**	**b13**
p12	**Instruction is in a valid format**	**b11**
p13	**Continuous Control Flow**	**b5** 👉
p14	**Exception return updates state correctly**	**b1 b5** 👉
p15	Register change implies that it is the instruction target	✓
p16	SR is not written to a GPR in user mode	✗
p17	**Interrupt implies handled**	**b8**
p18	Instruction unchanged in pipeline	--
p19	SPR modified only in supervisor mode	✓
p20	Enter supervisor mode is on reset or exception	✓
p21	**Exception handling implies exception mechanism activated**	**b8**
p22	Unspecified custom instructions are not allowed	✗
p23	**Exception handler accessed only during exception, in supervisor mode, or on**	**b8**
p24	Page fault generated if MMU detects an access control violation	--
p25	UART output changes on a write command from CPU	--
p26	Only transmit command or initialization change Ethernet data output	--
p27	Debug Unit's value and control registers only accessible from supervisor mode	--

One property can be identified from different bugs

Different properties can be identified from the same bug

Results: Comparison to State-of-the-Art

No.	Security Property Description	Found?
p1	Execution privilege matches page privilege	✓
p2	**SPR equals GPR in register move instruction**	**b12**
p3	**Updates to exception registers make sense**	**b4 b9 b15**
p4	Destination matches the target	✓
p5	**Memory value in equals register value out**	**b14**
p6	**Register value in equals memory value out**	**b16 b17**
p7	Memory address equals effective address	✓
p8	Privilege escalates correctly	✓
p9	Privilege deescalates correctly	✓
p10	Jumps update the PC correctly	✗
p11	**Jumps update the LR correctly**	**b13**
p12	**Instruction is in a valid format**	**b11**
p13	**Continuous Control Flow**	**b5**
p14	**Exception return updates state correctly**	**b1 b5**
p15	Register change implies that it is the instruction target	✓
p16	SR is not written to a GPR in user mode	✗
p17	**Interrupt implies handled**	**b8** 👉
p18	Instruction unchanged in pipeline	--
p19	SPR modified only in supervisor mode	✓
p20	Enter supervisor mode is on reset or exception	✓
p21	**Exception handling implies exception mechanism activated**	**b8** 👉
p22	Unspecified custom instructions are not allowed	✗
p23	**Exception handler accessed only during exception, in supervisor mode, or on**	**b8** 👉
p24	Page fault generated if MMU detects an access control violation	--
p25	UART output changes on a write command from CPU	--
p26	Only transmit command or initialization change Ethernet data output	--
p27	Debug Unit's value and control registers only accessible from supervisor mode	--

One property can be identified from different bugs

Different properties can be identified from the same bug

A single SCI can concisely represent multiple manually written properties

Results: Comparison to State-of-the-Art

No.	Security Property Description	Found?
p1	**Execution privilege matches page privilege**	✓
p2	SPR equals GPR in register move instruction	b12
p3	Updates to exception registers make sense	b4 b9 b15
p4	**Destination matches the target**	✓
p5	Memory value in equals register value out	b14
p6	Register value in equals memory value out	b16 b17
p7	**Memory address equals effective address**	✓
p8	**Privilege escalates correctly**	✓
p9	**Privilege deescalates correctly**	✓
p10	Jumps update the PC correctly	✗
p11	Jumps update the LR correctly	b13
p12	Instruction is in a valid format	b11
p13	Continuous Control Flow	b5
p14	Exception return updates state correctly	b1 b5
p15	**Register change implies that it is the instruction target**	✓
p16	SR is not written to a GPR in user mode	✗
p17	Interrupt implies handled	b8
p18	Instruction unchanged in pipeline	--
p19	**SPR modified only in supervisor mode**	✓
p20	**Enter supervisor mode is on reset or exception**	✓
p21	Exception handling implies exception mechanism activated	b8
p22	Unspecified custom instructions are not allowed	✗
p23	Exception handler accessed only during exception, in supervisor mode, or on reset	b8
p24	Page fault generated if MMU detects an access control violation	--
p25	UART output changes on a write command from CPU	--
p26	Only transmit command or initialization change Ethernet data output	--
p27	Debug Unit's value and control registers only accessible from supervisor mode	--

Properties Found in the Inference Step

Results: Comparison to State-of-the-Art

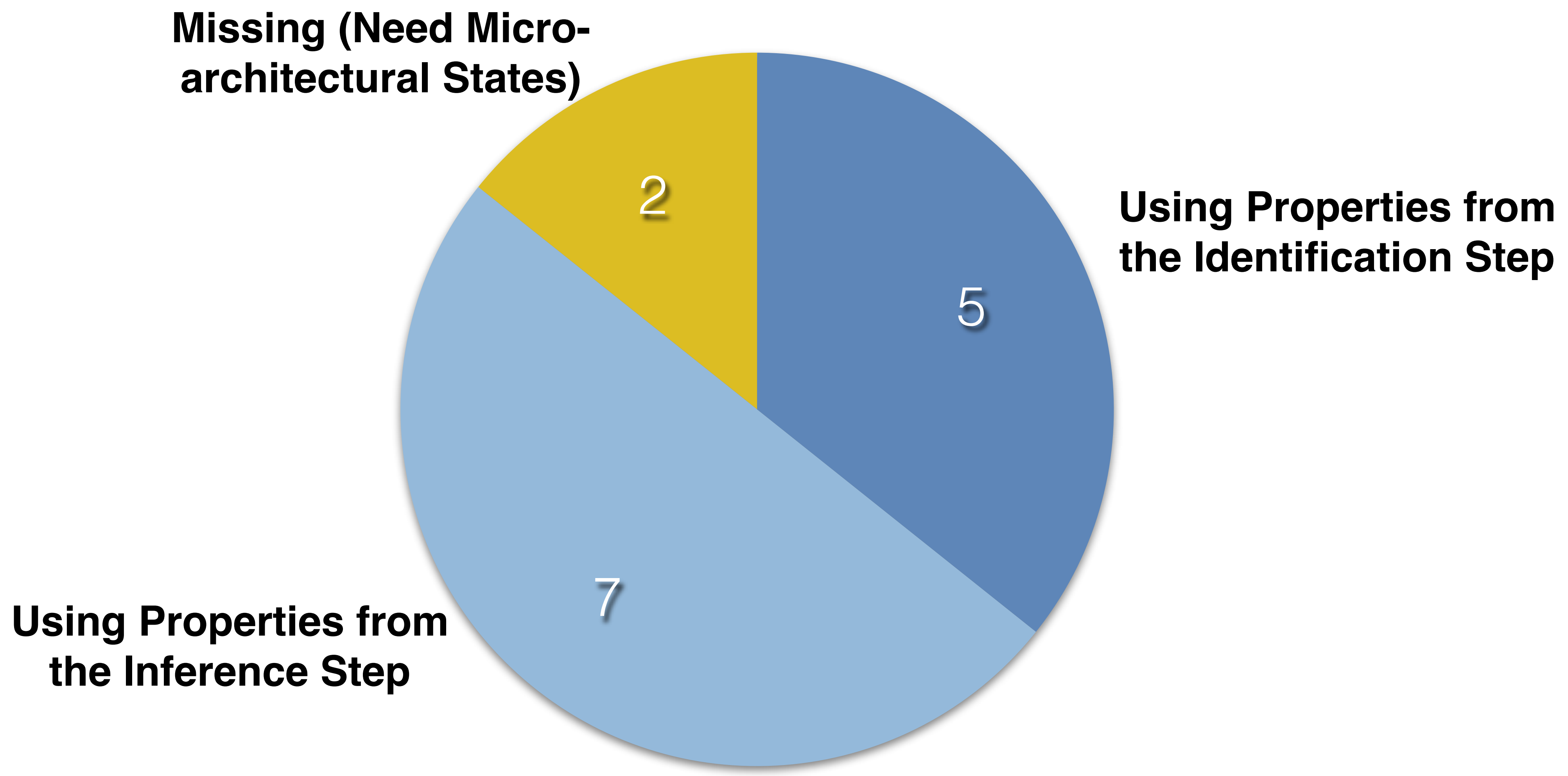
No.	Security Property Description	Found?
p1	Execution privilege matches page privilege	✓
p2	SPR equals GPR in register move instruction	b12
p3	Updates to exception registers make sense	b4 b9 b15
p4	Destination matches the target	✓
p5	Memory value in equals register value out	b14
p6	Register value in equals memory value out	b16 b17
p7	Memory address equals effective address	✓
p8	Privilege escalates correctly	✓
p9	Privilege deescalates correctly	✓
p10	**Jumps update the PC correctly**	✗
p11	Jumps update the LR correctly	b13
p12	Instruction is in a valid format	b11
p13	Continuous Control Flow	b5
p14	Exception return updates state correctly	b1 b5
p15	Register change implies that it is the instruction target	✓
p16	**SR is not written to a GPR in user mode**	✗
p17	Interrupt implies handled	b8
p18	Instruction unchanged in pipeline	--
p19	SPR modified only in supervisor mode	✓
p20	Enter supervisor mode is on reset or exception	✓
p21	Exception handling implies exception mechanism activated	b8
p22	**Unspecified custom instructions are not allowed**	✗
p23	Exception handler accessed only during exception, in supervisor mode, or on reset	b8
p24	Page fault generated if MMU detects an access control violation	--
p25	UART output changes on a write command from CPU	--
p26	Only transmit command or initialization change Ethernet data output	--
p27	Debug Unit's value and control registers only accessible from supervisor mode	--

Properties Not Found

Results: New Security Properties Found

No.	Security Property Description	Found?
p28	Flags that influence control flow should be set correctly	b6 b7
p29	Calculation of memory address or memory data is correct	b3 b10
p30	Link address is not modified during function call execution	✓

Results: Stopping New Bugs



Result of detecting 14 AMD errata from SPECS project
(bugs not used in the development of the assertions).

Summary

SCIFinder:

- Generates security-critical invariants semi-automatically
- Requires a list of known security-critical bugs and a processor design

Main Results:

- The final SCI set covers 86.4% of the manually crafted security properties
- We identify 3 new properties not seen in prior work

Website:

- https://cs.unc.edu/~csturton/SCIFinder/