Evaluating Security Specification Mining for a CISC Architecture

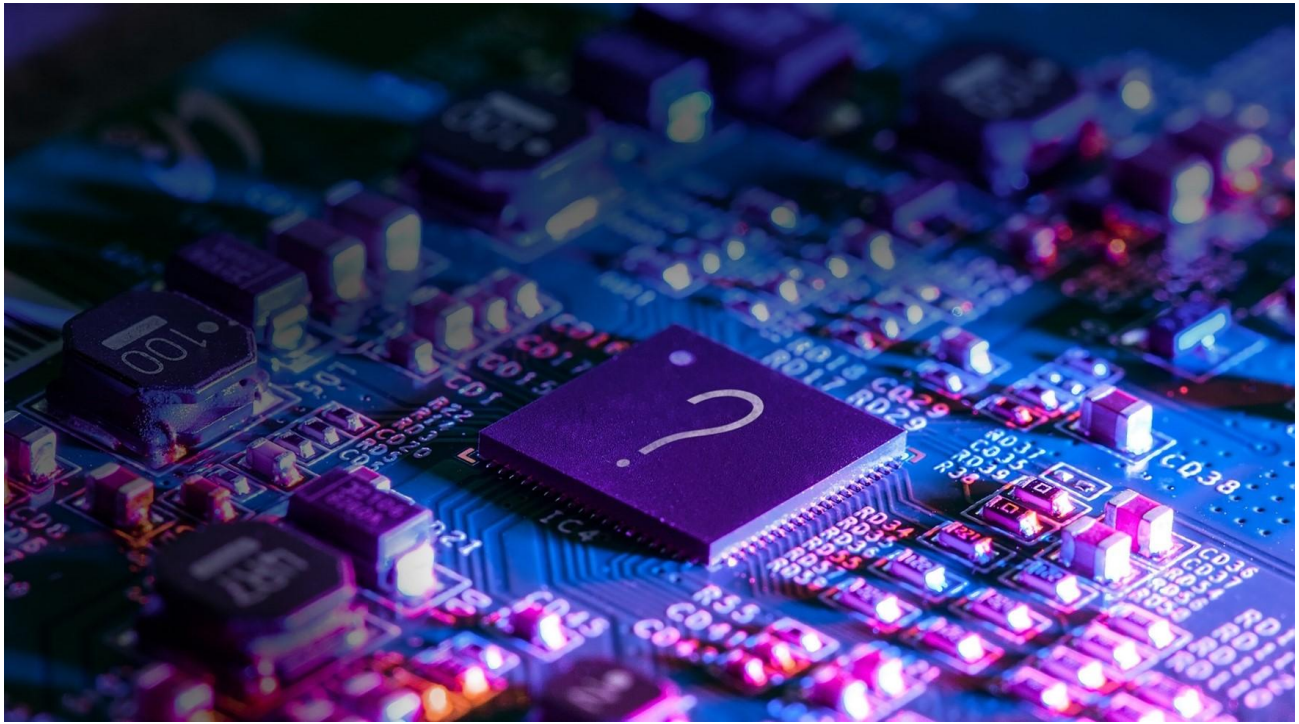
Hardware Security @ UNC

Calvin Deutschbein

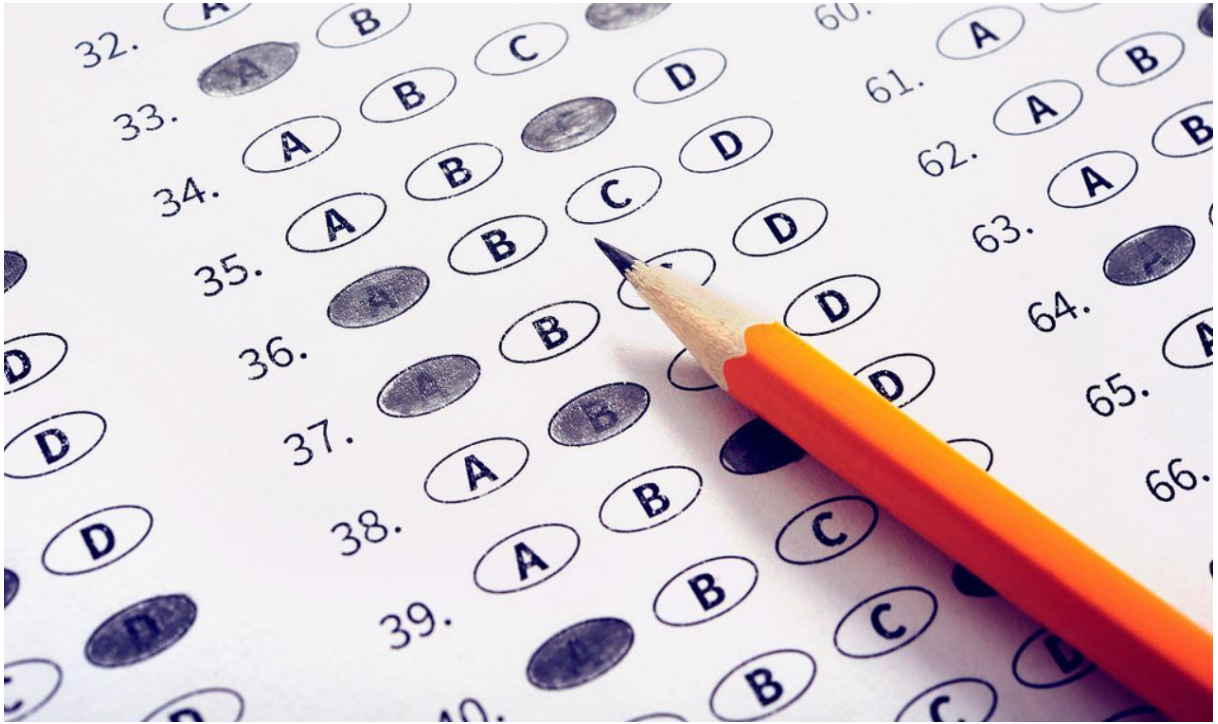
Cynthia Sturton

THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL

Hardware May Contain Security Bugs



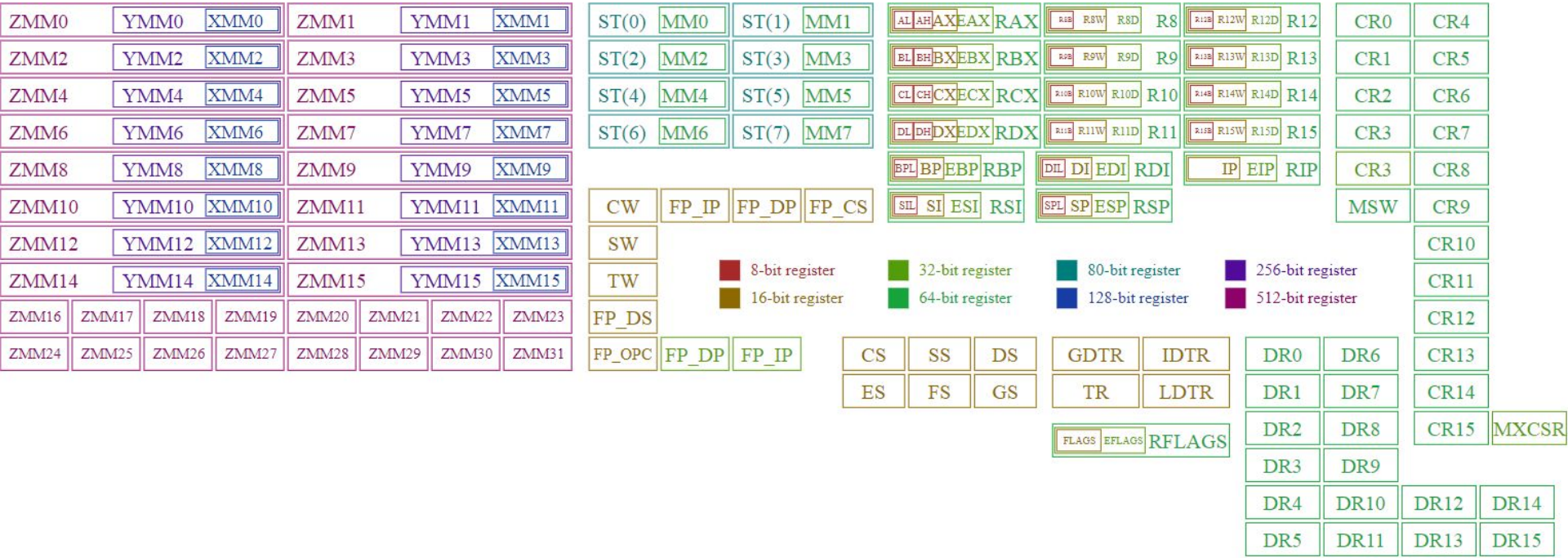
Testing Works – But What To Test For?



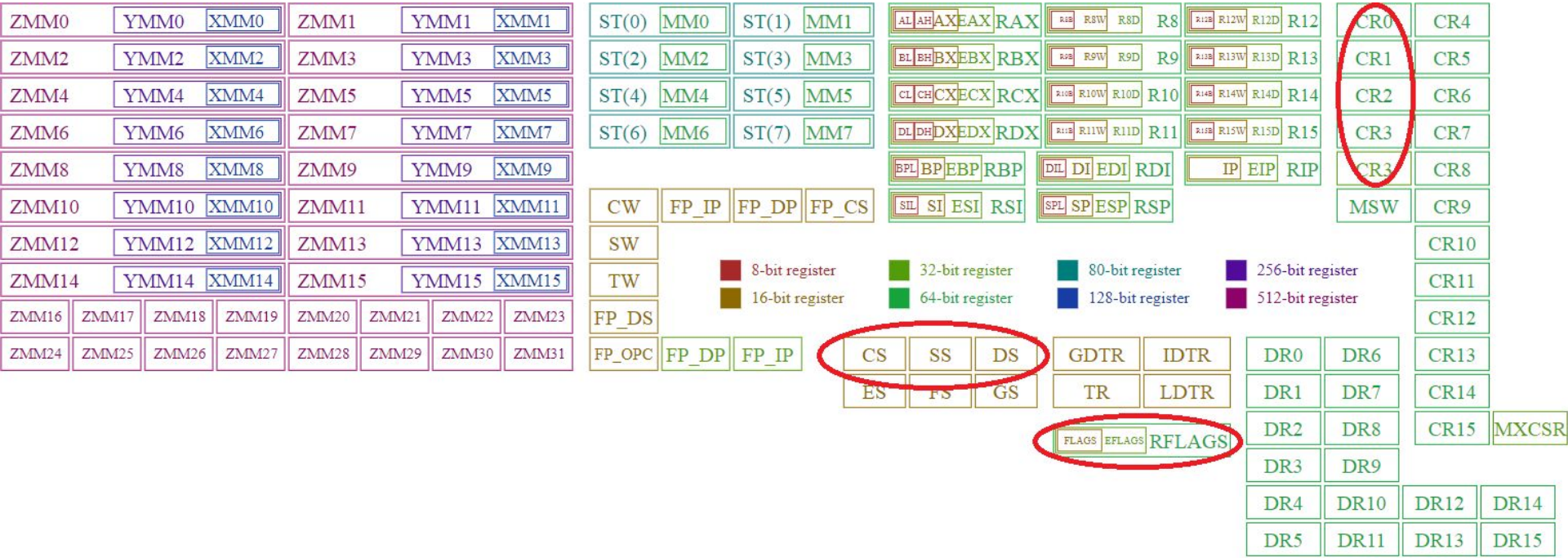
Research Question

How can properties that model secure behavior of closed source CISC designs be discovered using specification mining?

Today: 100s of registers, 10s of modules



Few registers are control signals!



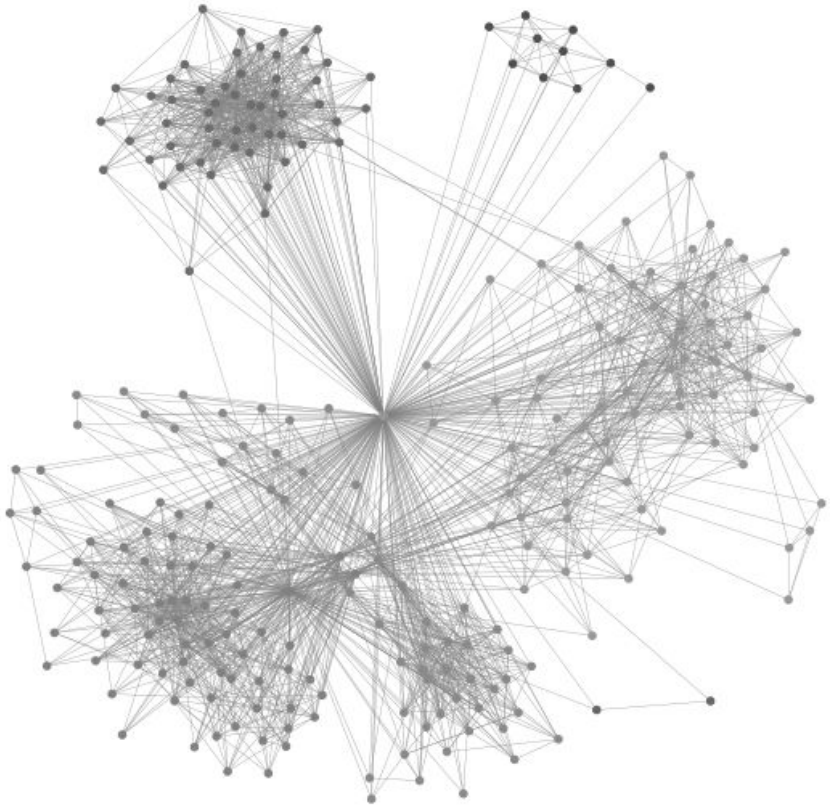
Method: Mining Closed Source CISC

How can properties that model secure behavior of closed source CISC designs be discovered using specification mining?

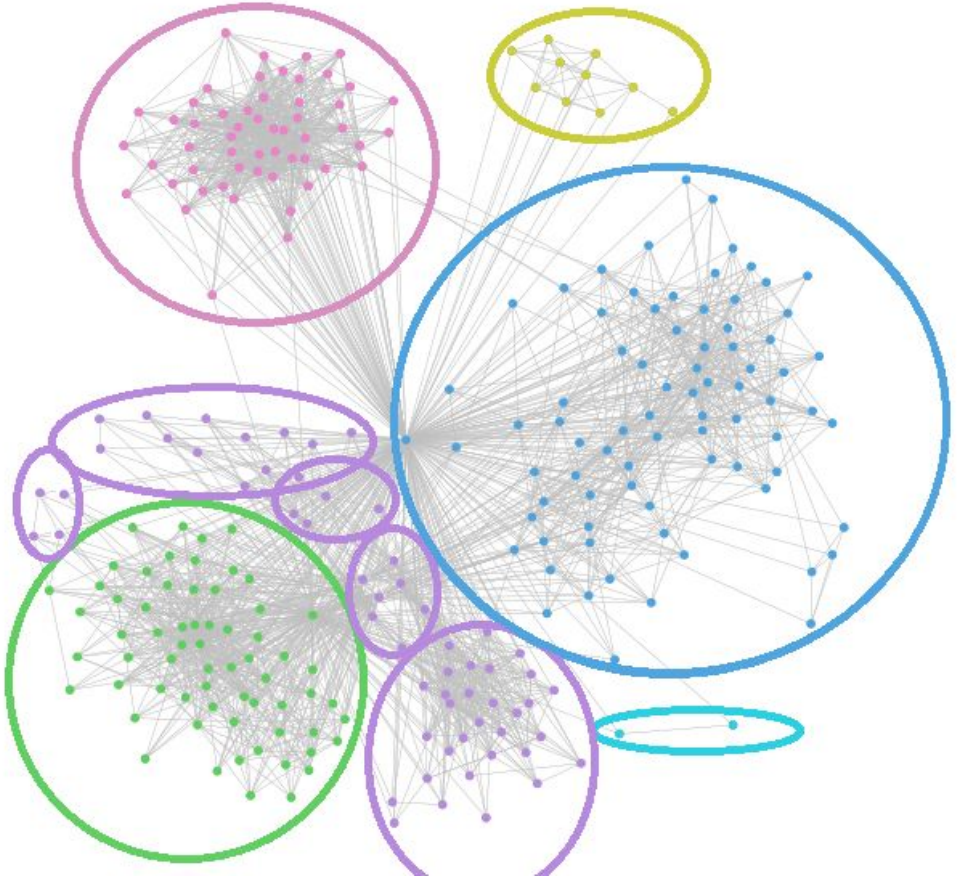
A two step approach:

- Mining for control signals in the design
- Mining preconditioned on those control signals

Naive Mining Produces Trillions of Properties



Control Signals Partition the Space



Control Signals Partition the Space

Transitions between partition - by changing a control signal - are well defined.

Finding properties within a partition is less expensive.

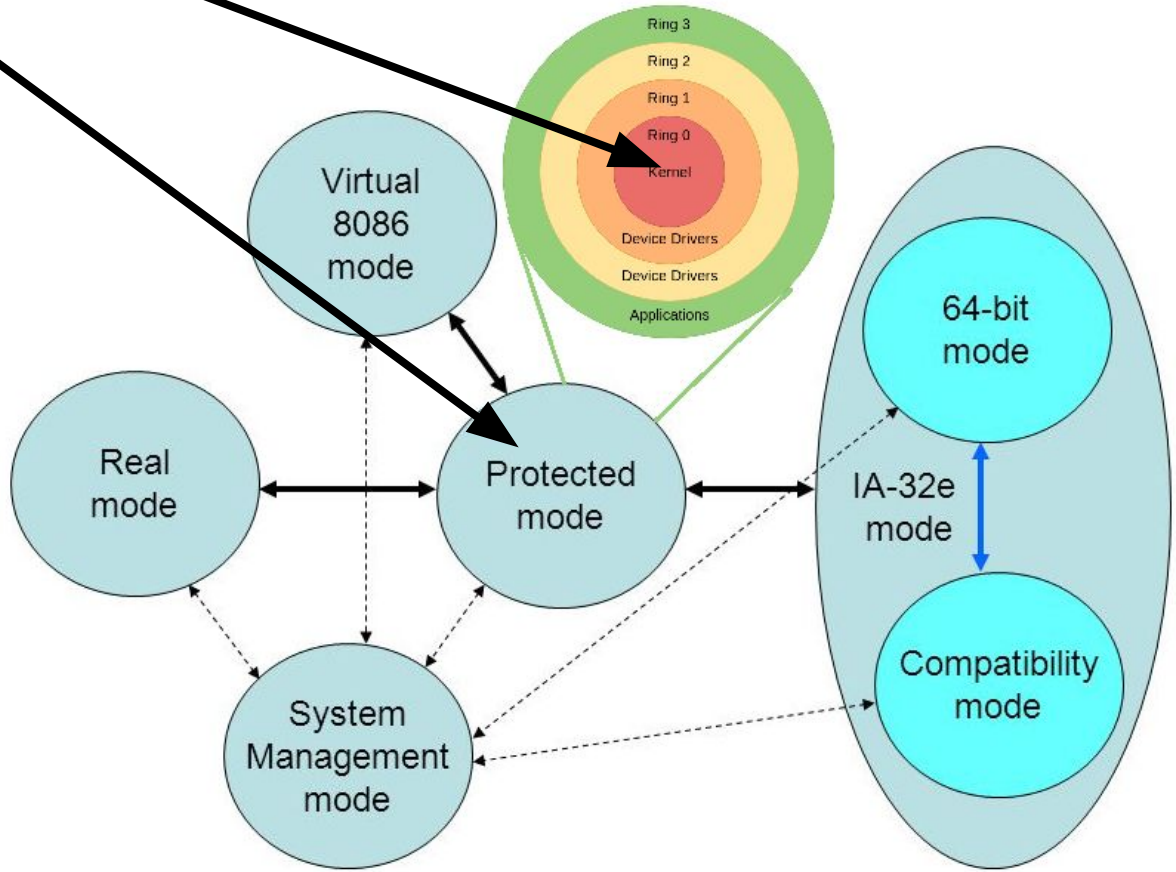
Partition Example

For example, the IOPL (I/O privilege level) signal can only be changed at “Ring 0” in protected mode, that is

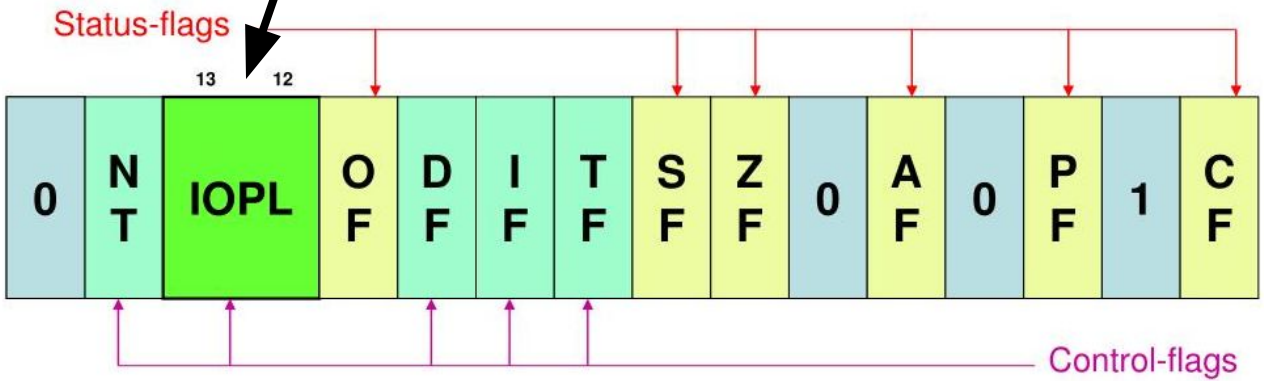
PME=1 & !CPL=0 ==> IOPL=orig(IOPL)

“Protected mode at privilege level other than zero means IOPL can’t change”

PME=1 & !CPL=0 ==> IOPL=orig(IOPL)



!CPL==0 & PME==1 ==> **IOPL==orig(IOPL)**



x86 EFLAGS register

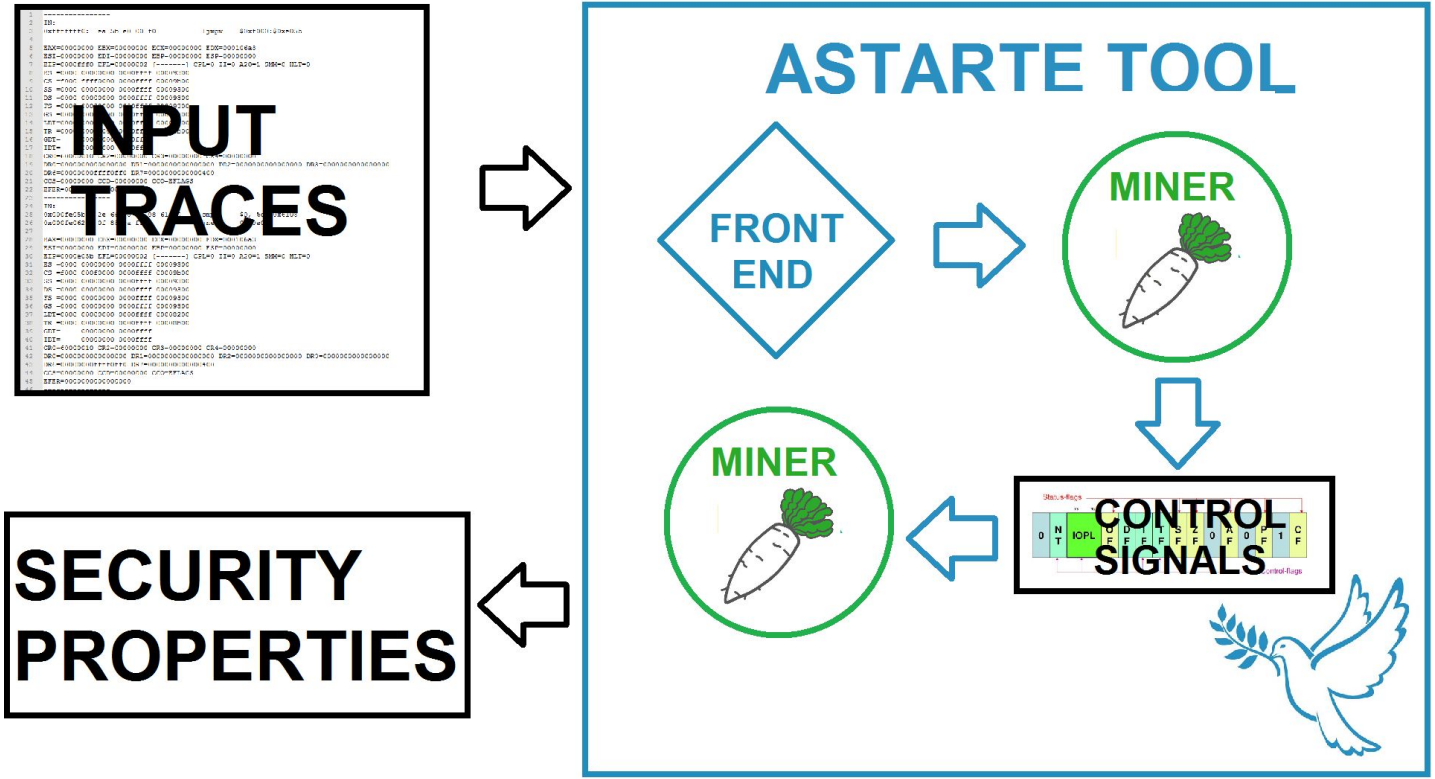
Why look at control signals?

The x86 architecture is large and complex, but can be mined effectively.

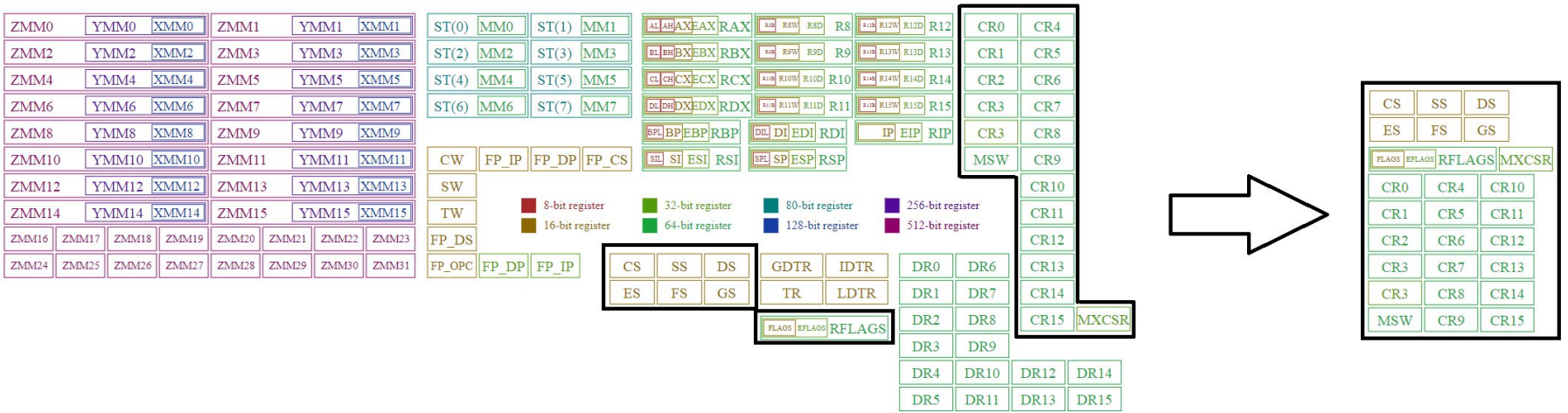
- Control signals implement **secure** computing.
- Control signals can behave differently **across architectures**.
- Control signals **refine** the search for security properties.

These control signals can be discovered automatically.

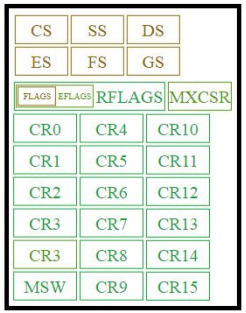
So we created a tool to find properties using signals.

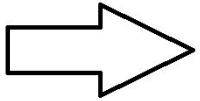


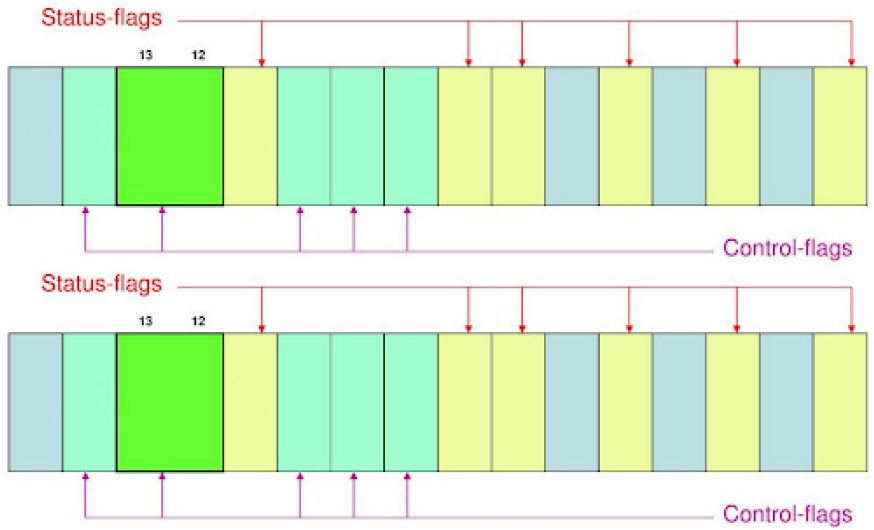
Front End: Registers Placed in Groups



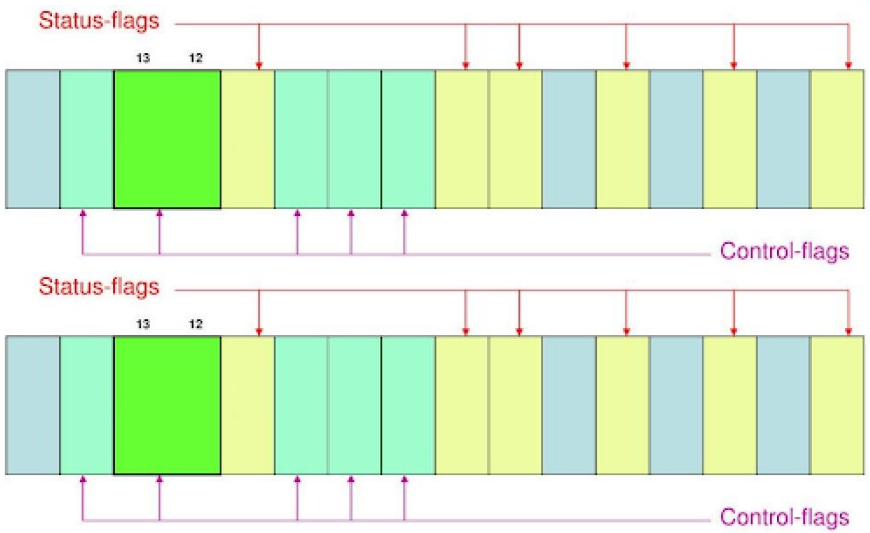
Front End: Control Signals Split into Bits

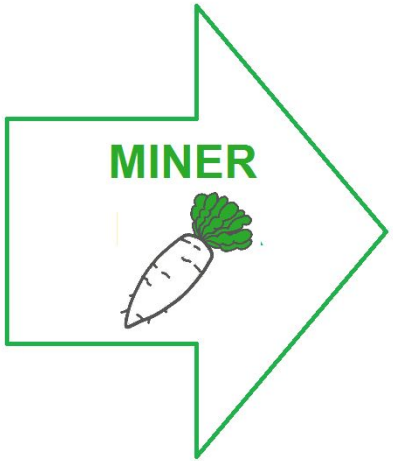




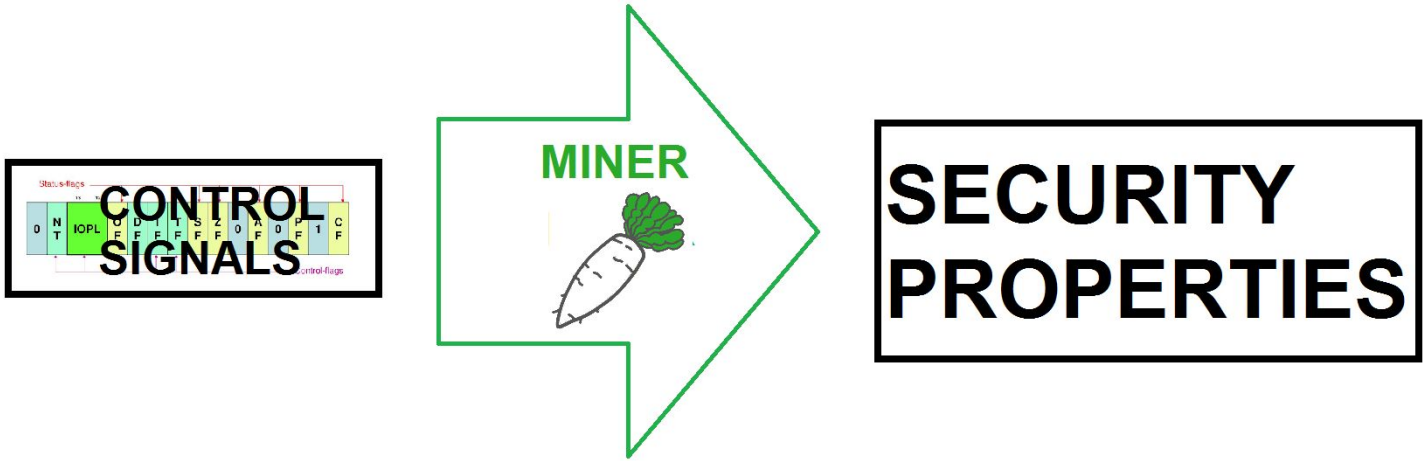


First Mining: Find Control Signals



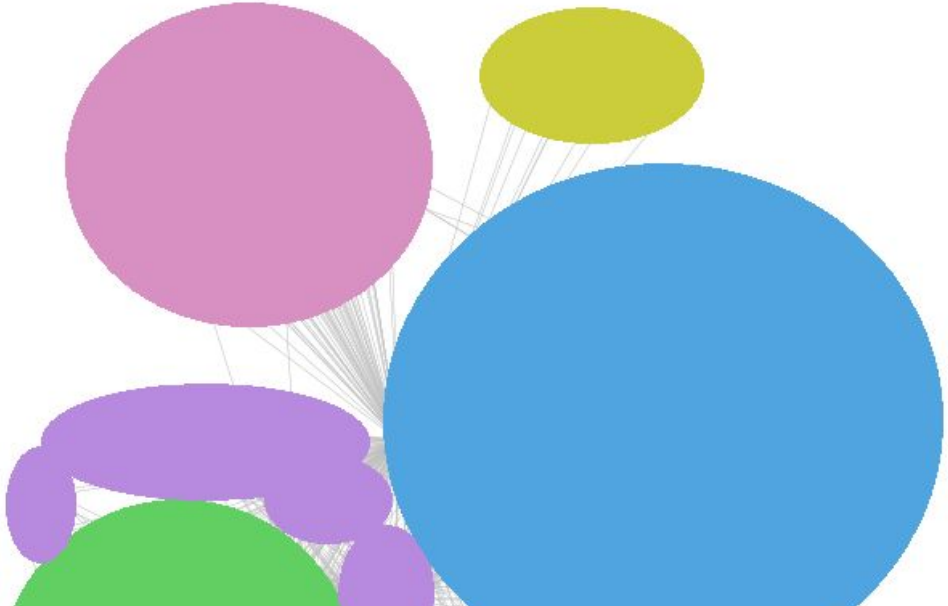


Second Mining: Precondition on Control Signals

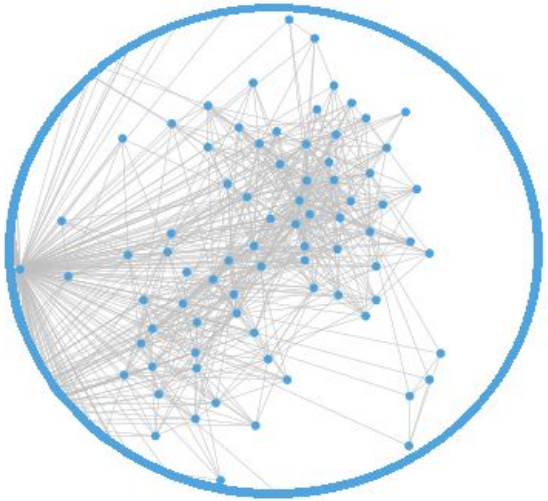


Control Signals Partition the Space

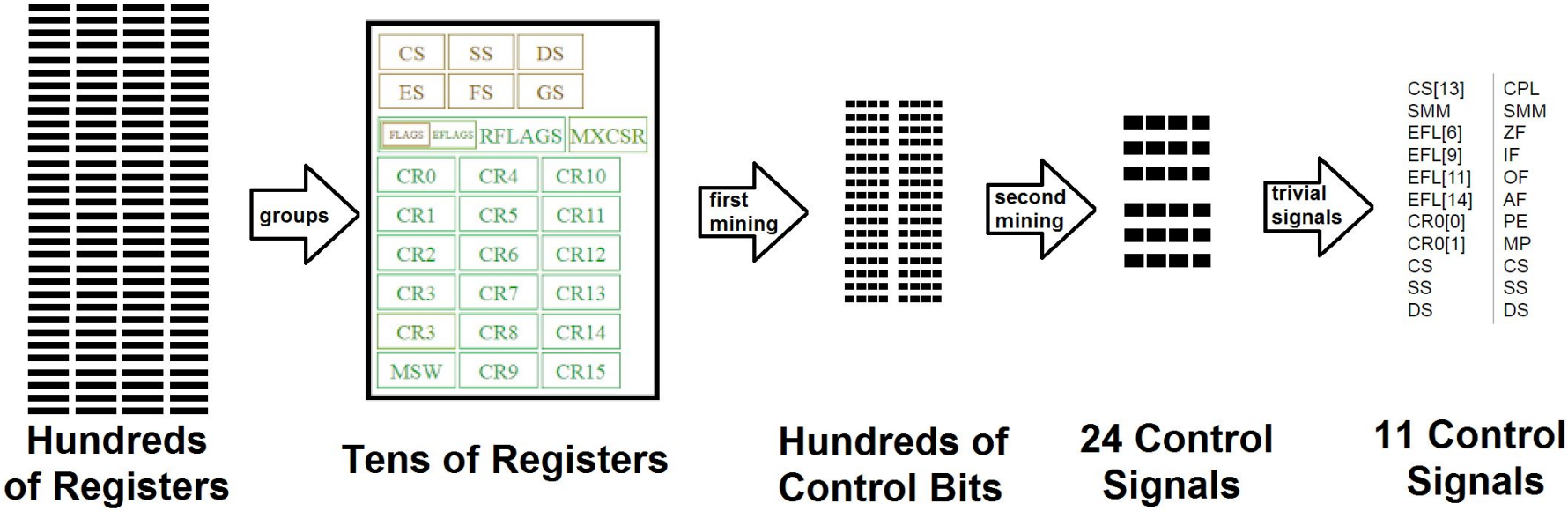
Preconditions capturing changes to signals capture transitions between different modes of the processor.



Preconditions holding signals constant capture the behavior defined by a control bit taking on a certain value.



Property Refinement



Security Control Signals

CS[13]	CPL	Current Privilege Level	Gives Ring in Protected Mode
SMM	SMM	System Management Mode	Gives “Ring -2” or System Management Mode
EFL[6]	ZF	Zero Flag	Indicates Zero result for Arithmetic
EFL[9]	IF	Interrupt enable Flag	Allows or Disallows Interrupts
EFL[11]	OF	Overflow Flag	Indicates Zero result for Arithmetic
EFL[14]	AF	Adjust Flag	Indicates Carry result for Arithmetic
CR0[0]	PE	Protected mode Enable	Gives whether Protected Mode is active
CR0[1]	MP	Monitor co-processor	Controls (F)WAIT instructions
CS	CS	Code Segment	Holds current code segment pointer
SS	SS	Stack Segment	Holds current stack segment pointer
DS	DS	Data Segment	Holds current data segment pointer

Evaluation

Developed and ran Astarte over IvyBridge x86 with bare metal and OS traces.

Considered output properties versus manual and historical bugs.

Considered output properties as implemented by various operating systems.

Results

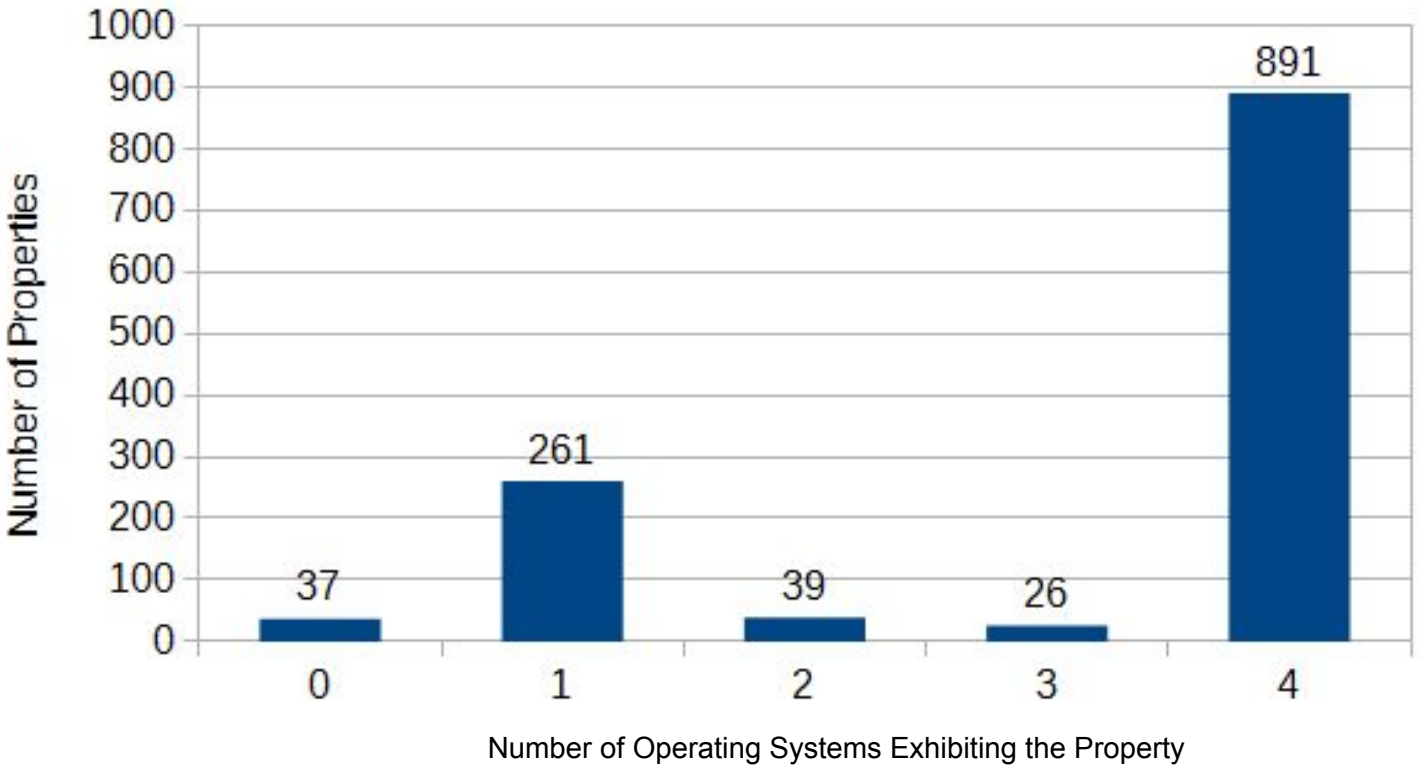
Input traces cover:

- Ubuntu
- seL4
- Solaris
- FreeDOS

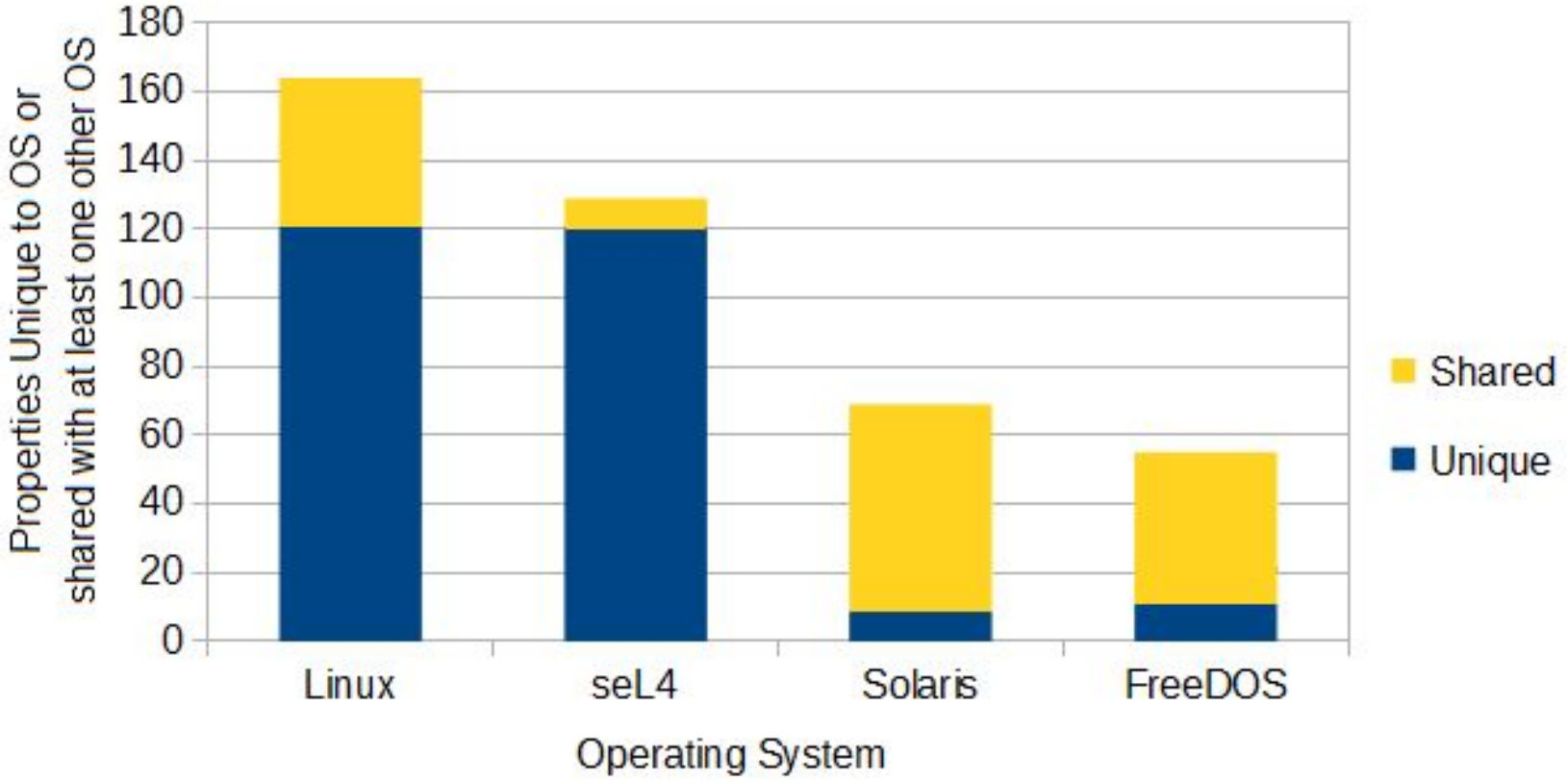
Output properties capture:

- 23/29 known properties
- Prevent 2/2 known bugs
- 1400 total properties
- 892 in ALL OSes

Most properties occured in one or all OSes



Most shared properties were not in seL4



Astarte: Mining Closed Source CISC

Specification mining can discover security properties preconditioned on control signals in closed source CISC designs.

Thank you!