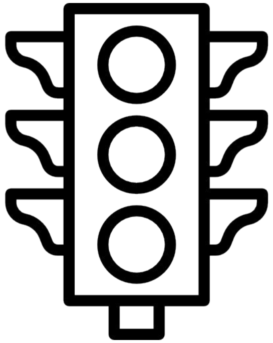
Discovering Blind-Trust Vulnerabilities in PLC Binaries via State Machine Recovery

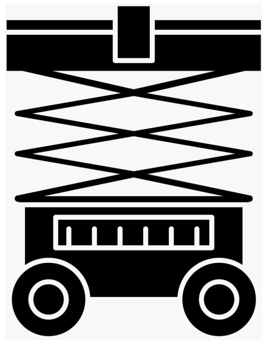
Fangzhou "Bonnie" Dong, Arvind S Raj, Efrén López-Morales, Siyu Liu,
Yan Shoshitaishvili, Tiffany Bao, Adam Doupé, Muslum Ozgur Ozmen, Ruoyu Wang

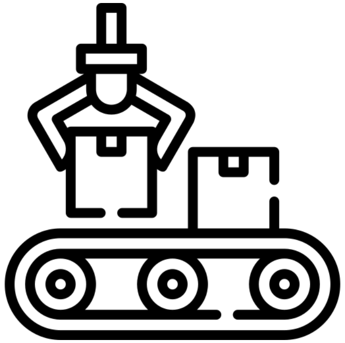
Arizona State University

NM STATE BE BOLD. Shape the Future.® New Mexico State University

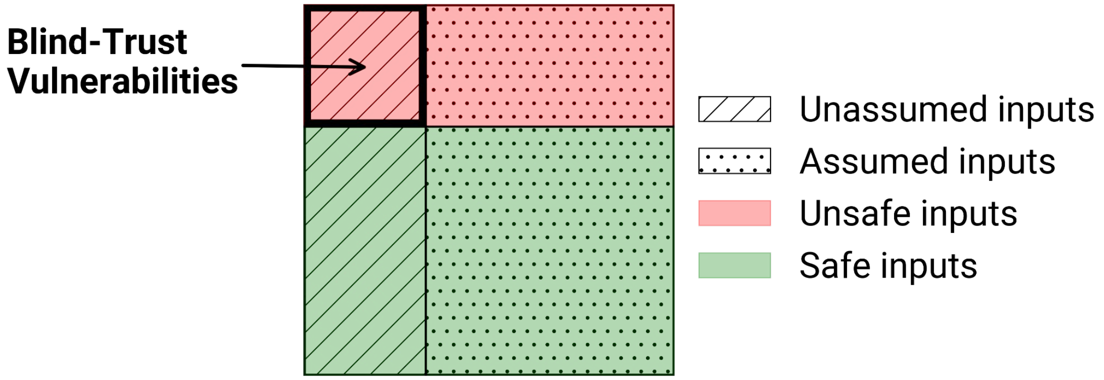
Programmable Logic Controllers


Traffic Light


Warehouse Lifter

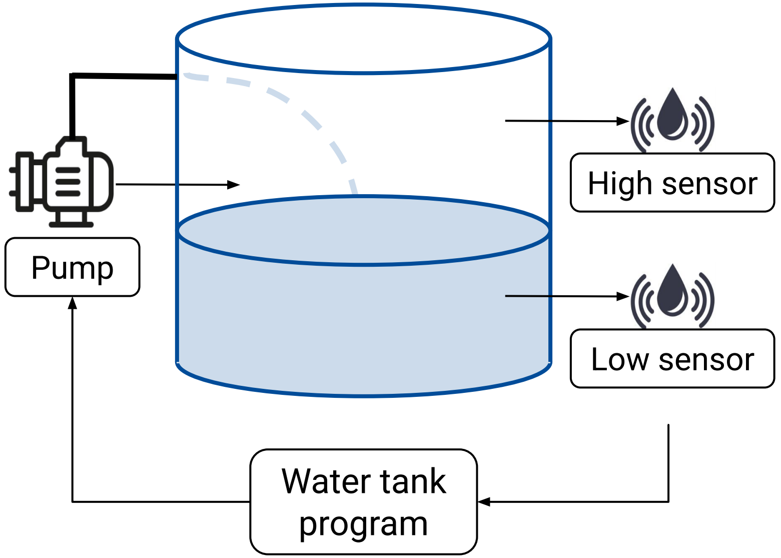

Conveyor System

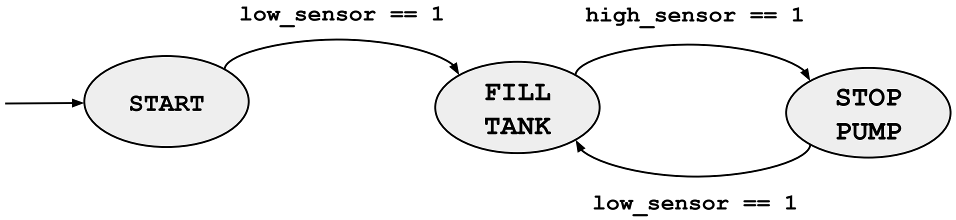
Blind-Trust Vulnerability



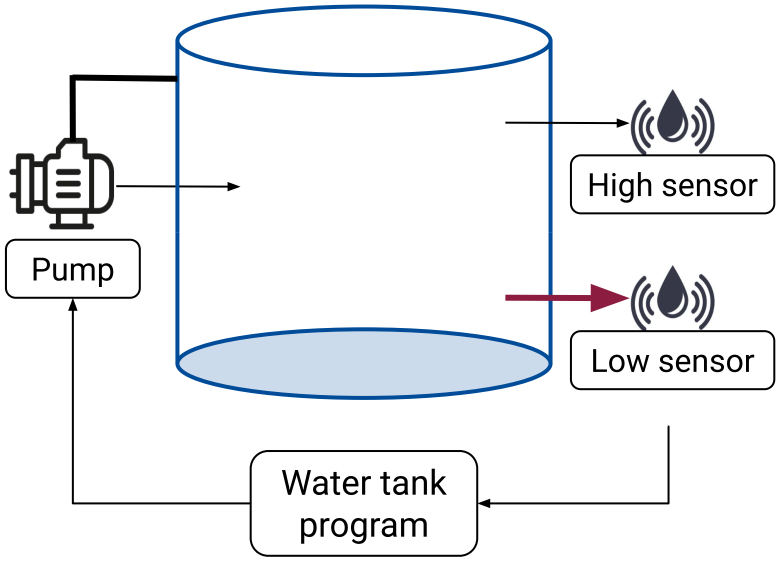
security engineering for future computing

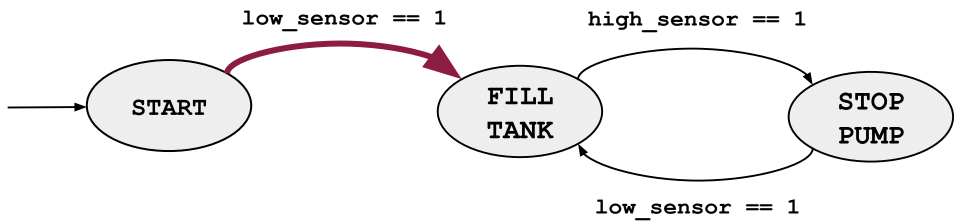
Blind-Trust Vulnerability: Scenario



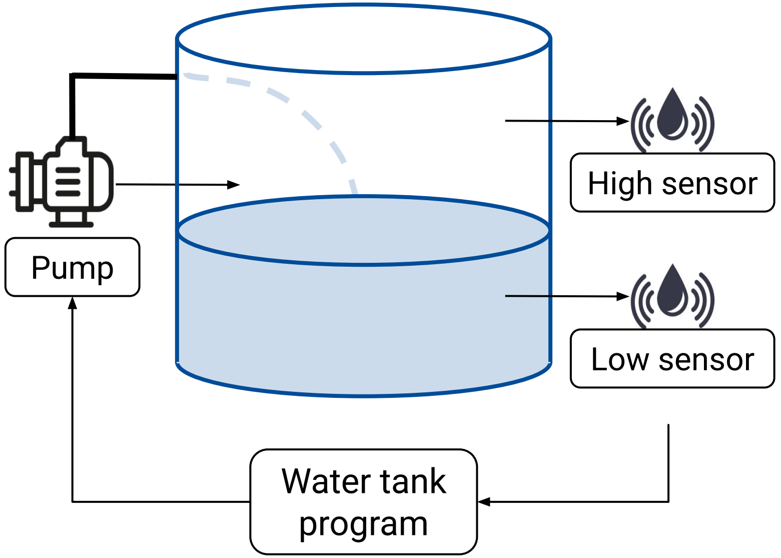

$$\begin{tikzcd}
{} \arrow[r] & \text{START} \arrow[r, bend left, "low\_sensor == 1"] & \text{FILL TANK} \arrow[r, bend left, "high\_sensor == 1"] & \text{STOP PUMP} \arrow[l, bend left, "low\_sensor == 1"]
\end{tikzcd}$$

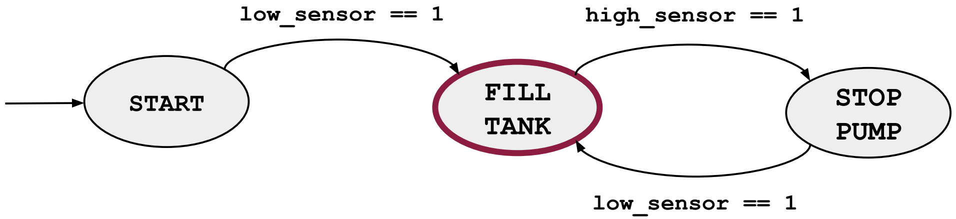
Blind-Trust Vulnerability: Scenario



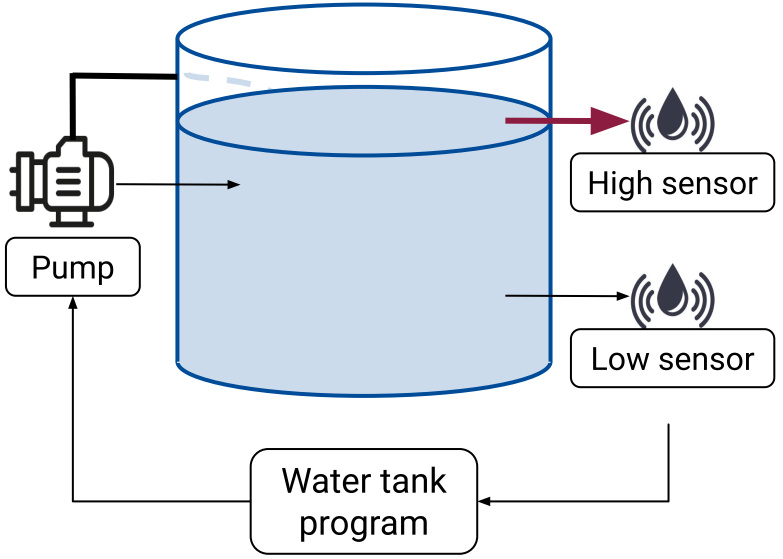


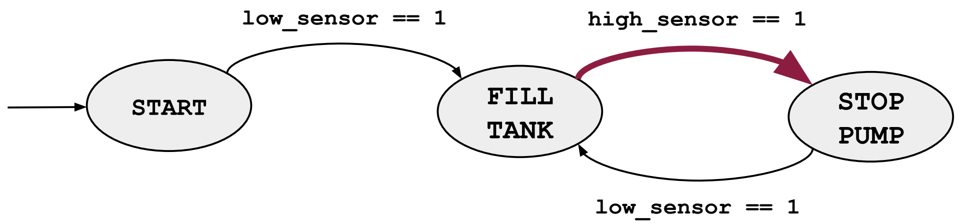
Blind-Trust Vulnerability: Scenario



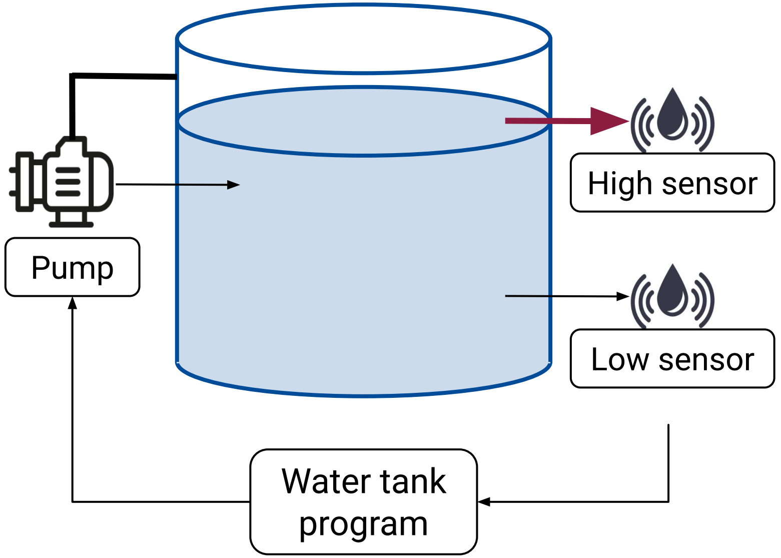


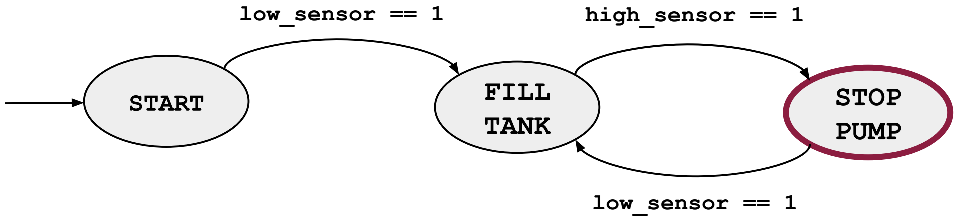
Blind-Trust Vulnerability: Scenario



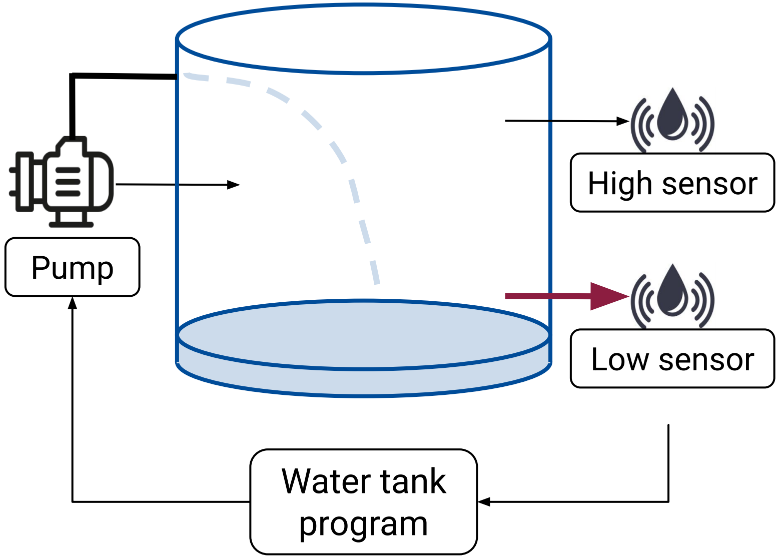


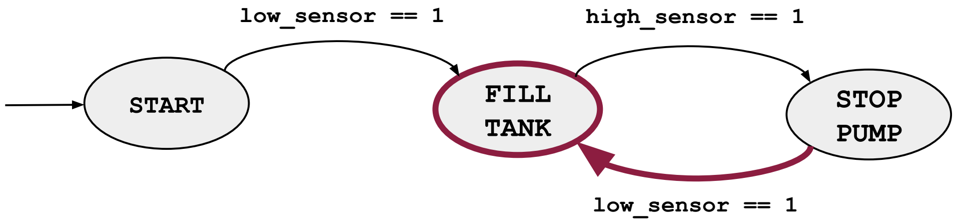
Blind-Trust Vulnerability: Scenario



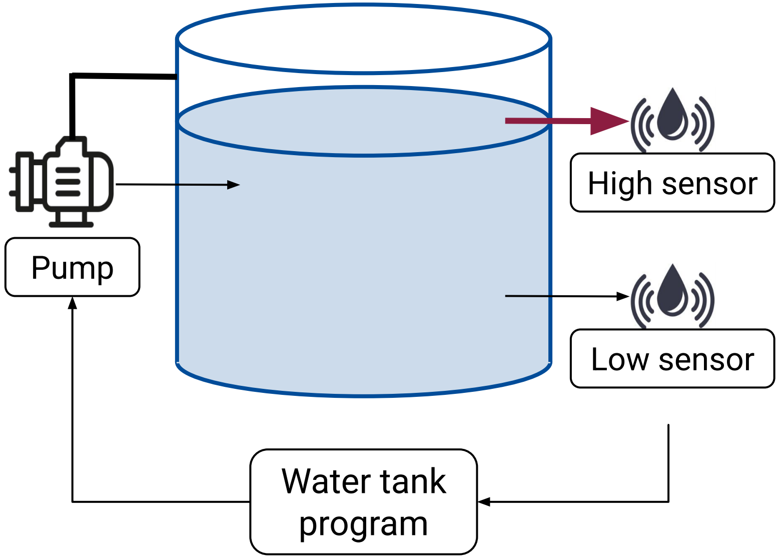


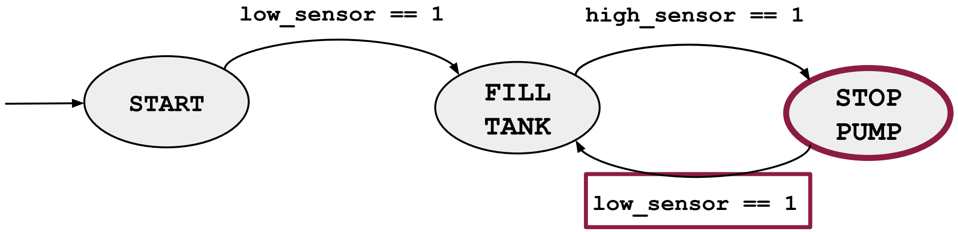
Blind-Trust Vulnerability: Scenario



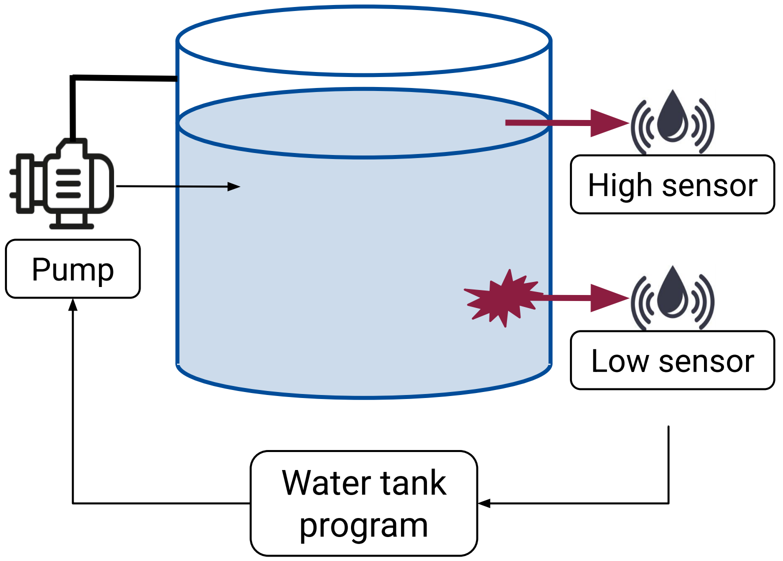


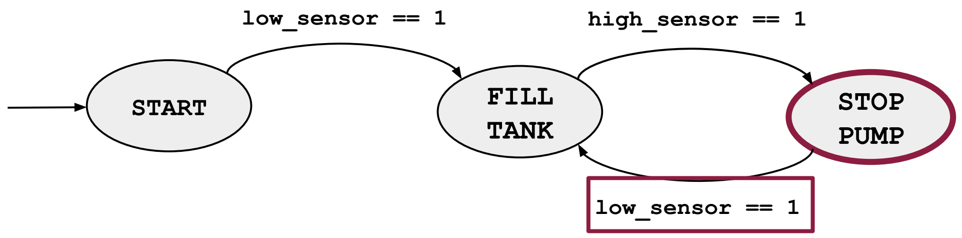
Blind-Trust Vulnerability



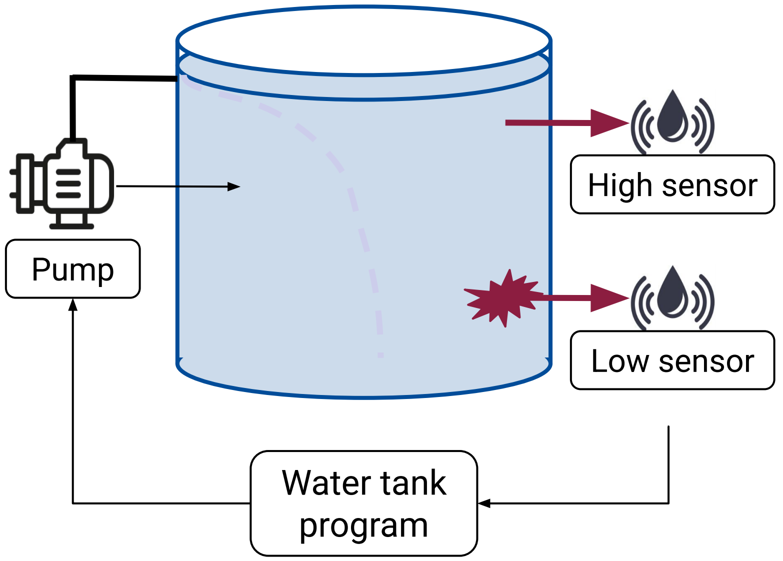


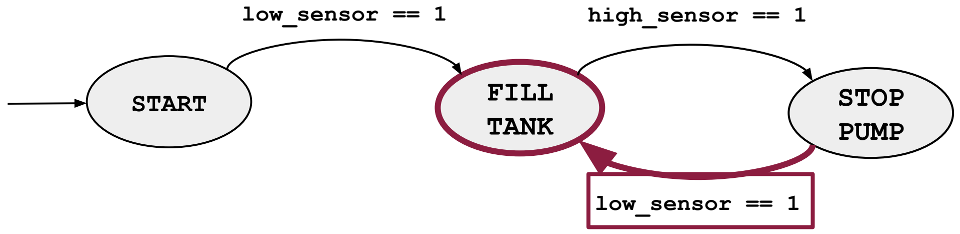
Blind-Trust Vulnerability



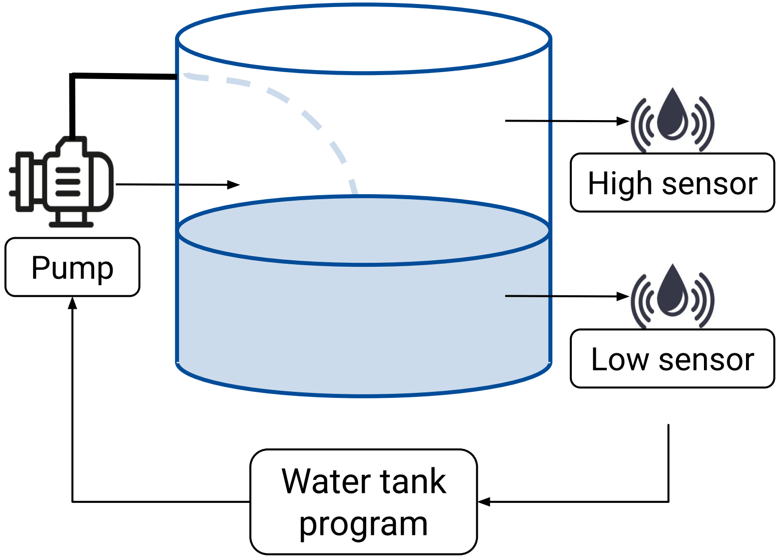


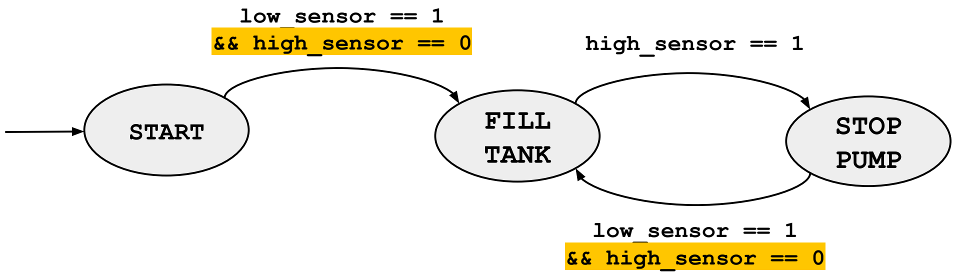
Blind-Trust Vulnerability





Blind-Trust Vulnerability: The Fix



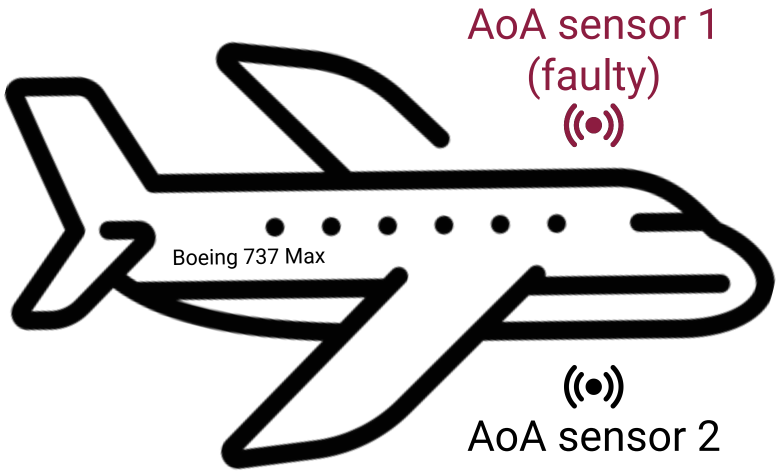


Safety Incidents

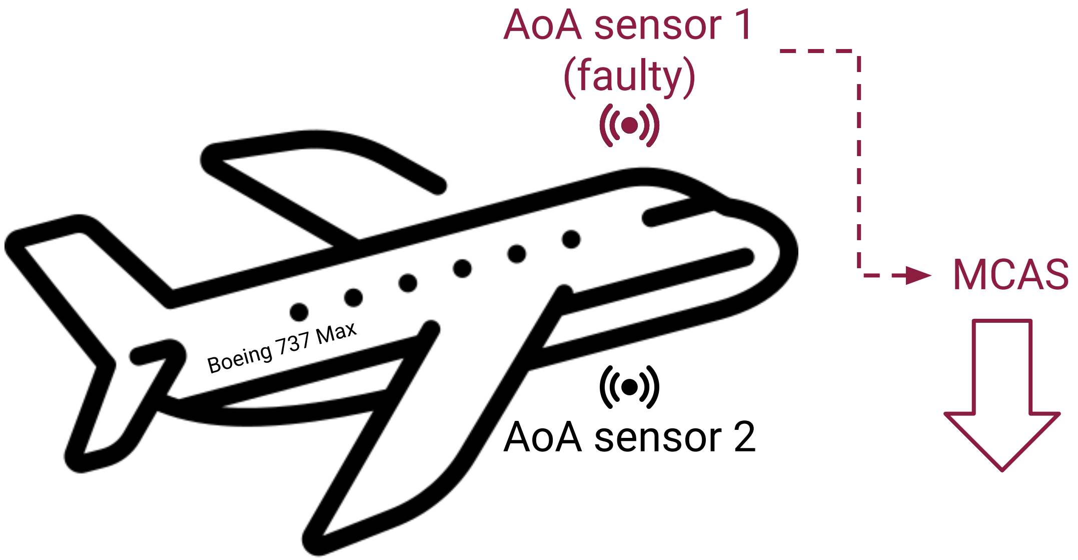
- Boeing 737 MAX
- 2018-2019
- 2 planes crashed
- 346 people died

ETHIOPIA-AIR ACCIDENT

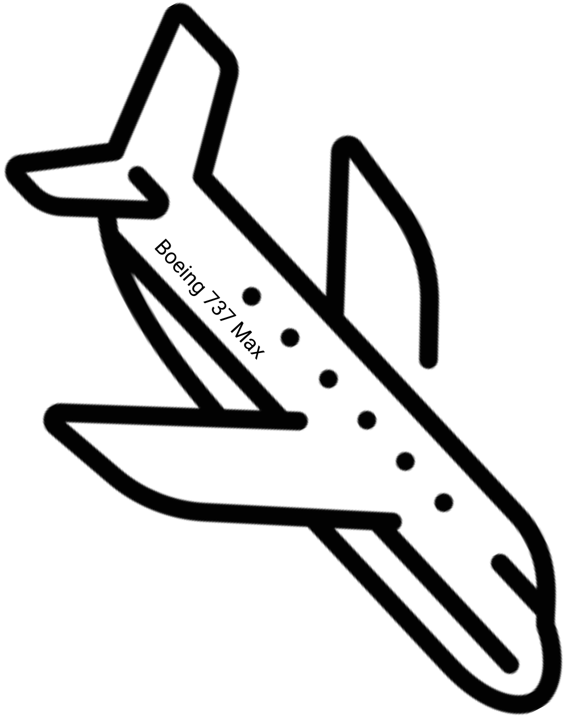
Safety Incidents

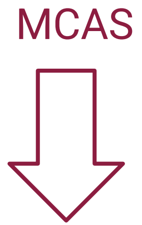


Safety Incidents

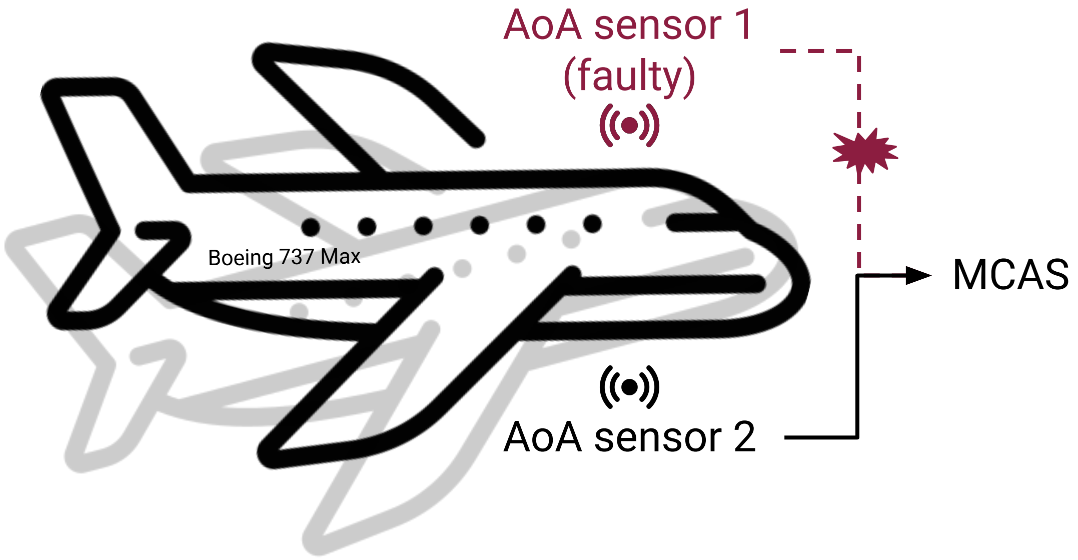


Safety Incidents





Safety Incidents: The Fix



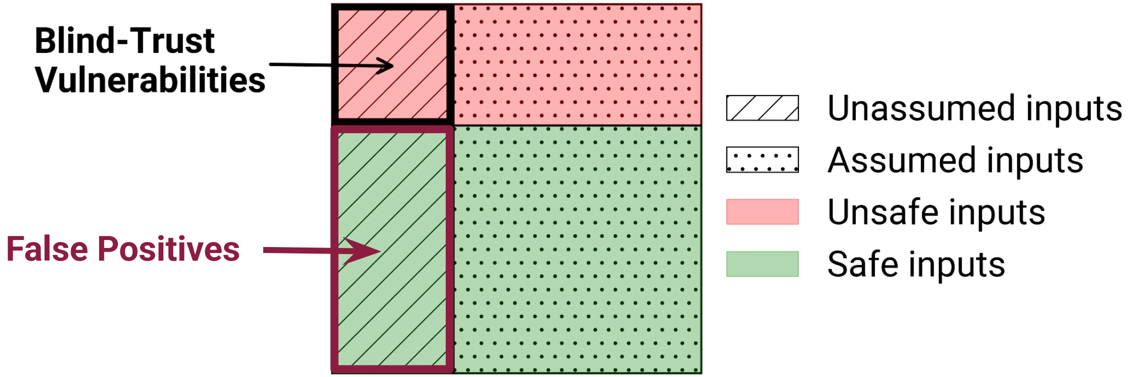
Challenges

Finding BTVs is challenging!

Challenges

Finding BTVs is challenging!

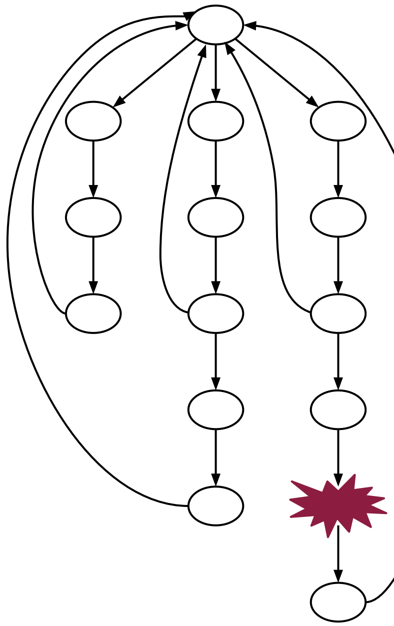
- Not all unassumed inputs are BTVs.



Challenges

Finding BTVs is challenging!

- Not all unassumed inputs are BTVs.
- BTVs may exist in deep states.



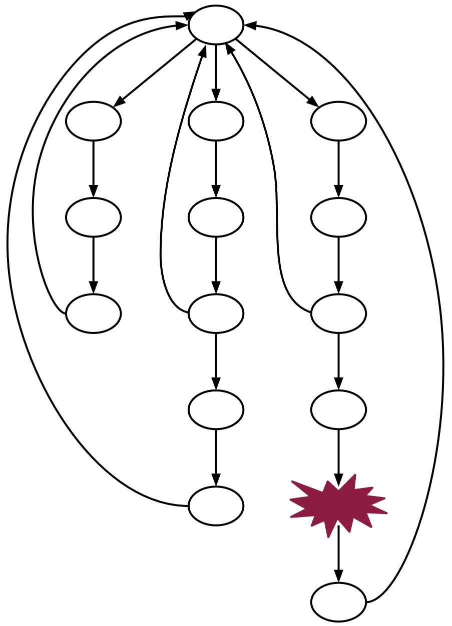
Challenges

Finding BTVs is challenging!

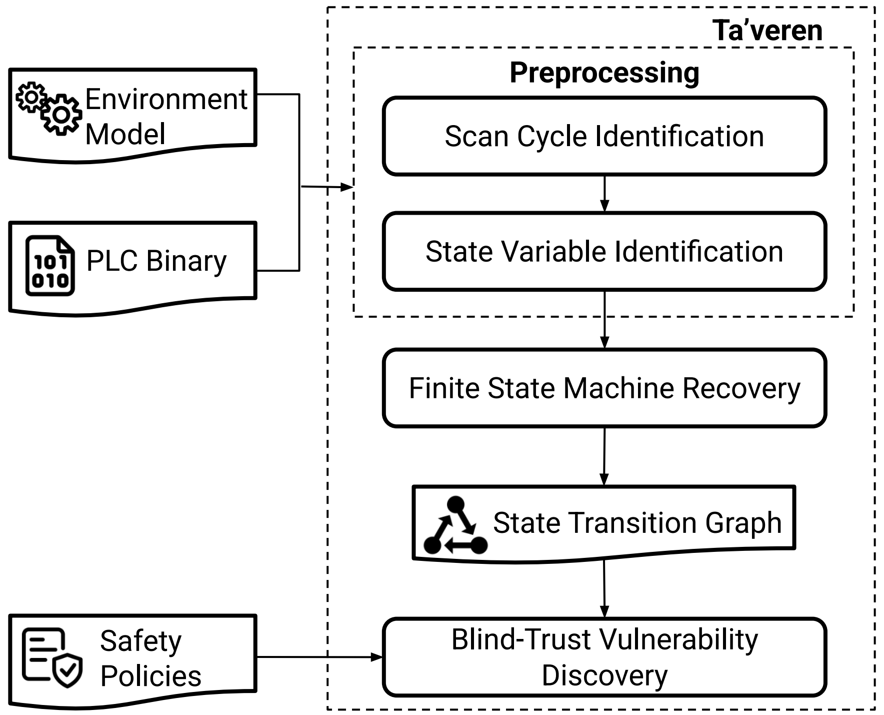
- Not all unassumed inputs are BTVs.
- BTVs may exist in deep states.

Solution: Ta’veren

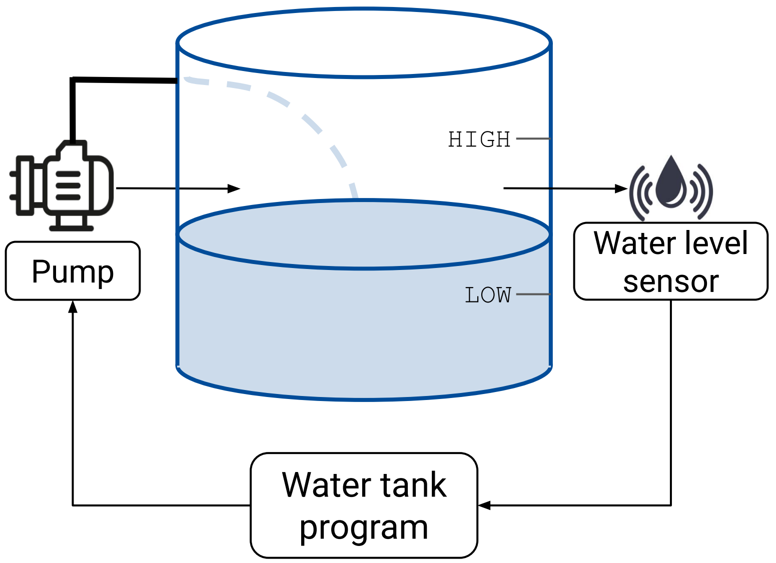
- Finite state machine recovery
- Model checking



Ta’veren

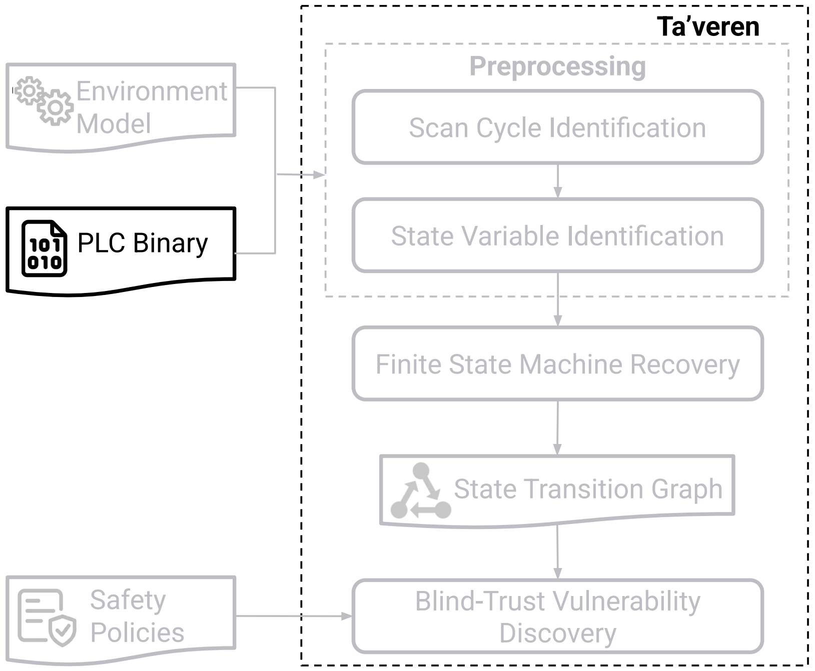

Ta’veren
Preprocessing
Environment Model
PLC Binary
Scan Cycle Identification
State Variable Identification
Finite State Machine Recovery
State Transition Graph
Safety Policies
Blind-Trust Vulnerability Discovery

Scenario: Water Tank (One Sensor)



Ta'veren: Inputs

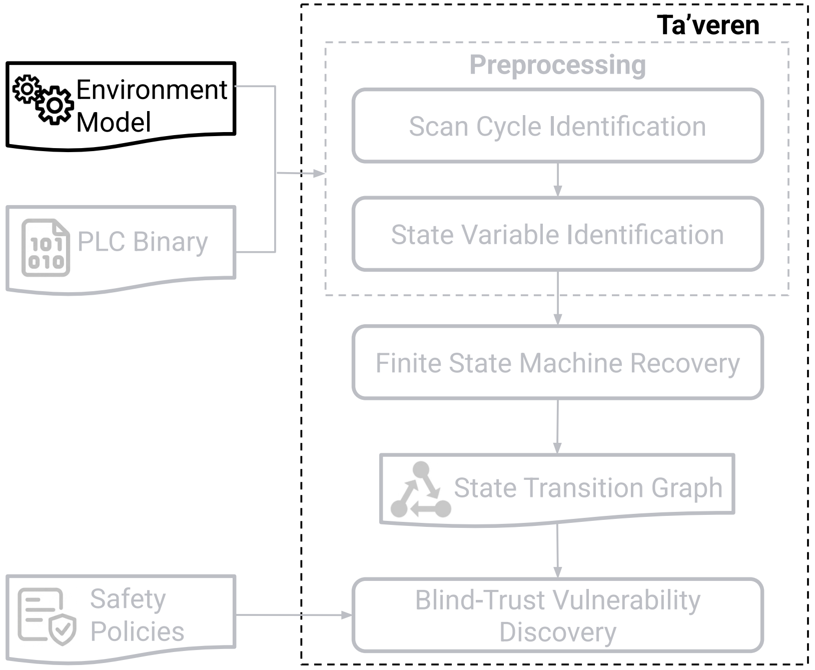
- The binary
  - No access to high-level semantics: names, enums



Ta'veren
Preprocessing
Environment Model
Scan Cycle Identification
PLC Binary
State Variable Identification
Finite State Machine Recovery
State Transition Graph
Safety Policies
Blind-Trust Vulnerability Discovery

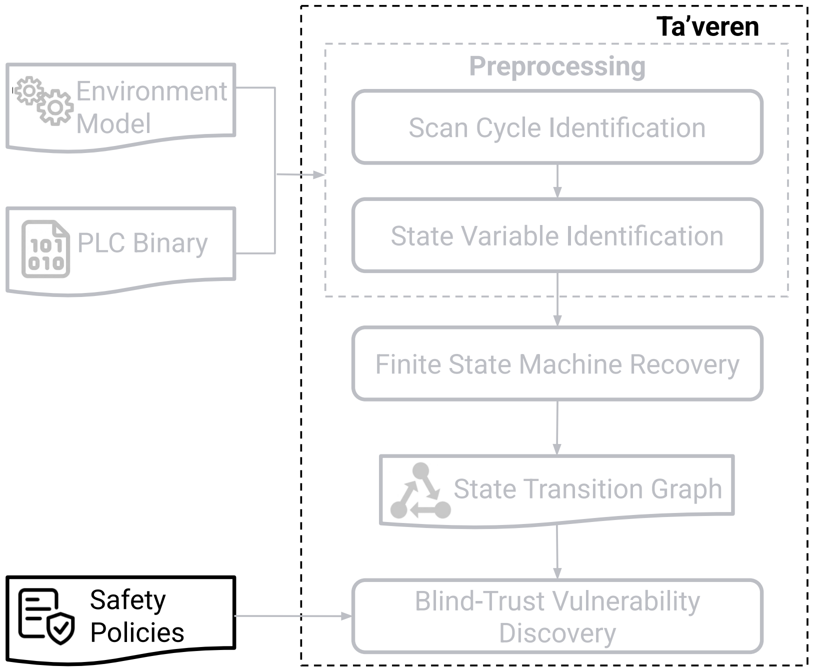
Ta’veren: Inputs

- The binary
  - No access to high-level semantics: names, enums
- Environment model
  - Input variable: **`level`** (`int`)
  - Output variable: **`pump`** (`bool`)


Ta’veren
Preprocessing
Environment Model
PLC Binary
Scan Cycle Identification
State Variable Identification
Finite State Machine Recovery
State Transition Graph
Safety Policies
Blind-Trust Vulnerability Discovery

Ta'veren: Inputs

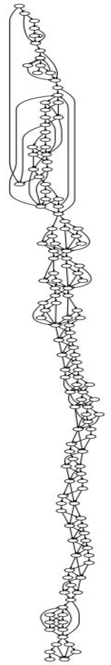
- The binary
  - No access to high-level semantics: names, enums
- Environment model
  - Input variable: **`level`** (`int`)
  - Output variable: **`pump`** (`bool`)
- Safety policies
  - When water level is high, pump must be off

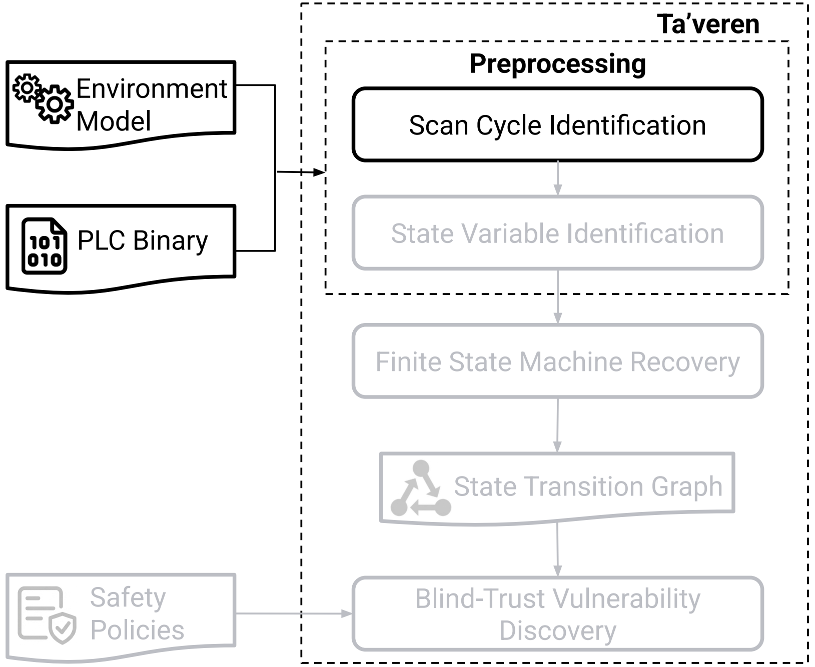


Ta'veren
Preprocessing
Environment Model
PLC Binary
Scan Cycle Identification
State Variable Identification
Finite State Machine Recovery
State Transition Graph
Safety Policies
Blind-Trust Vulnerability Discovery

Ta’veren: Preprocessing

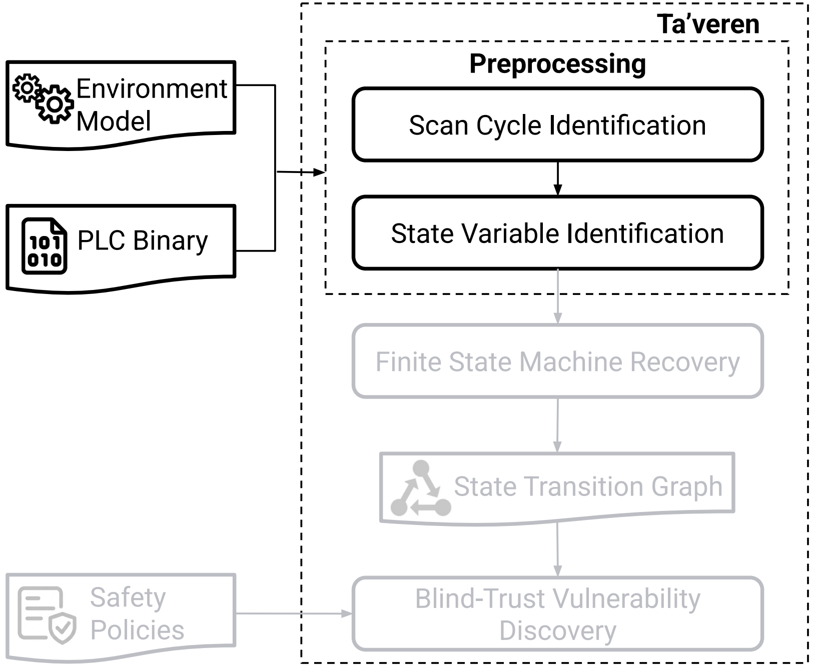
- Scan cycle identification
    - Functions implementing FSM



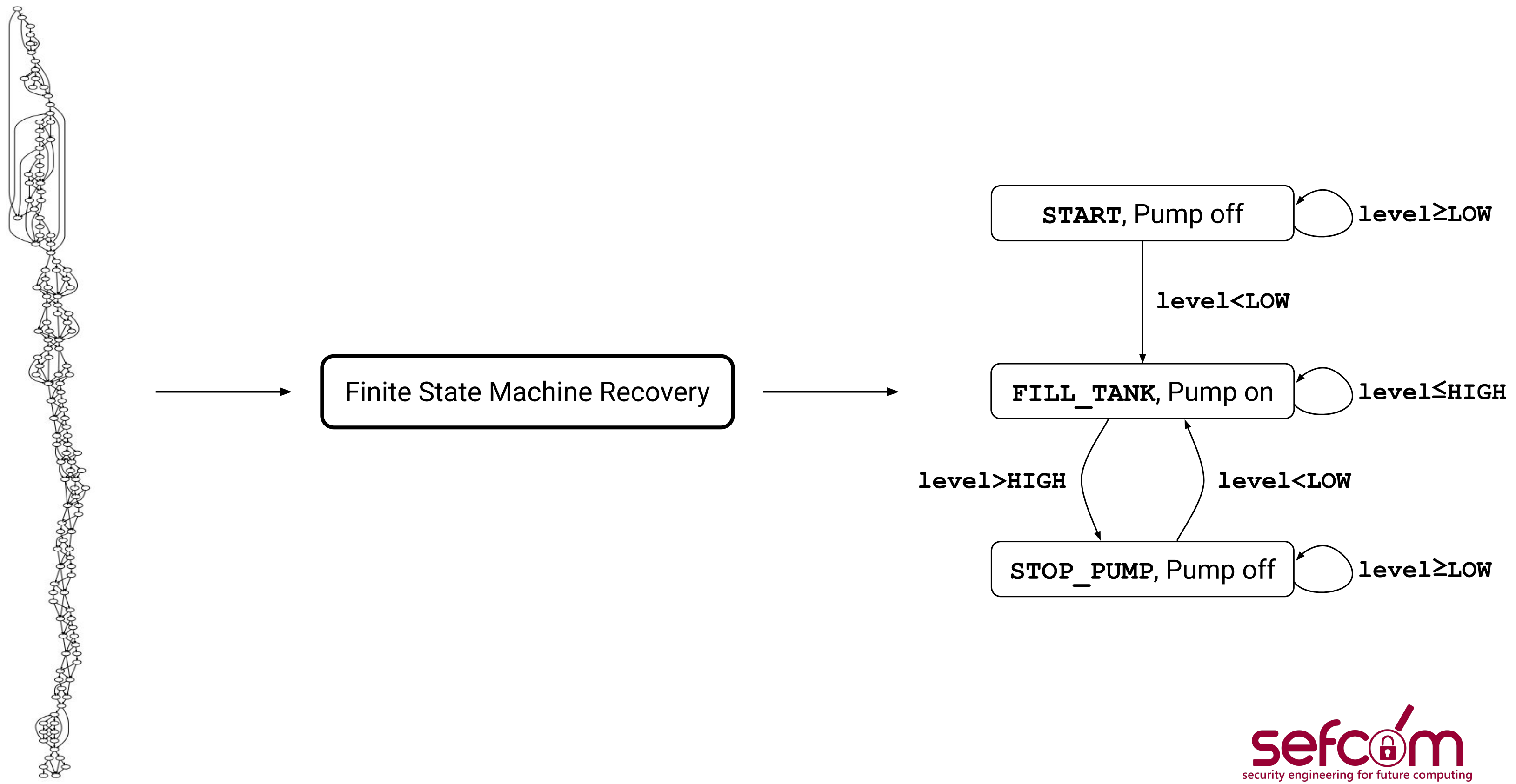

Ta’veren
Preprocessing
Environment Model
PLC Binary
Scan Cycle Identification
State Variable Identification
Finite State Machine Recovery
State Transition Graph
Safety Policies
Blind-Trust Vulnerability Discovery

Ta’veren: Preprocessing

- Scan cycle identification
  - Functions implementing FSM
- State variable identification
  - `(START, FILL_TANK, STOP_PUMP)`

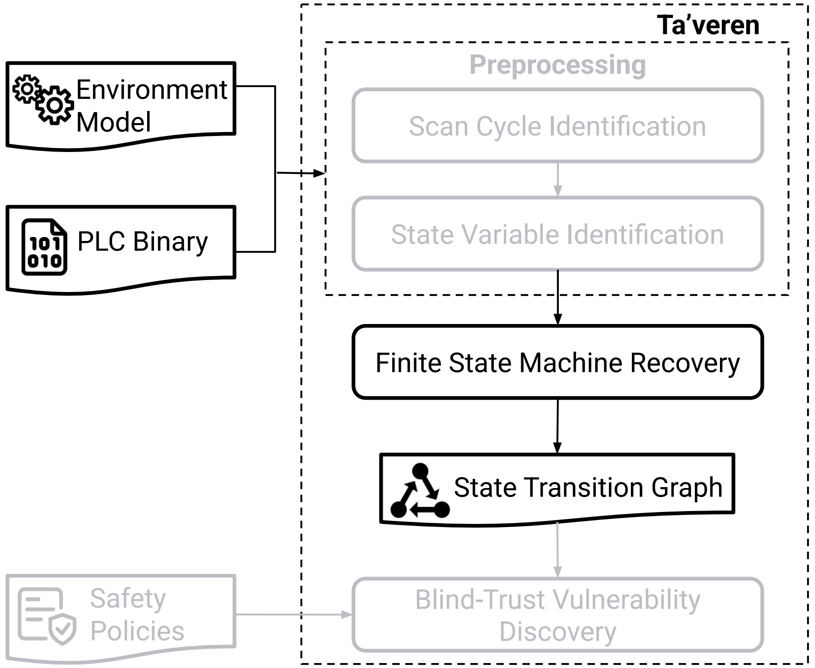

Ta’veren
Preprocessing
Environment Model
PLC Binary
Scan Cycle Identification
State Variable Identification
Finite State Machine Recovery
State Transition Graph
Safety Policies
Blind-Trust Vulnerability Discovery

Ta’veren: FSM Recovery



Ta’veren: FSM Recovery

- State deduplication using abstract states
- Discover meaningful values for input variables, instead of checking every possible input

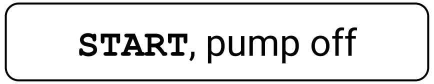

Ta’veren
Preprocessing
Environment Model
PLC Binary
Scan Cycle Identification
State Variable Identification
Finite State Machine Recovery
State Transition Graph
Safety Policies
Blind-Trust Vulnerability Discovery

Ta’veren: FSM Recovery

Worklist:

Concrete state	Input constraints	Prev abs state
S_0	level == 0	∅

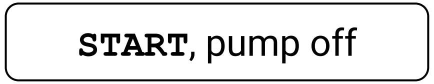
Ta’veren: FSM Recovery



Worklist:

Concrete state	Input constraints	Prev abs state
S_0	level == 0	$\varnothing$

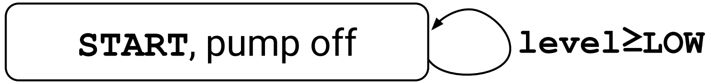
Ta’veren: FSM Recovery



Worklist:

Concrete state	Input constraints	Prev abs state
S_1	level ≥ LOW	**START**, pump off
S_1	level < LOW	**START**, pump off

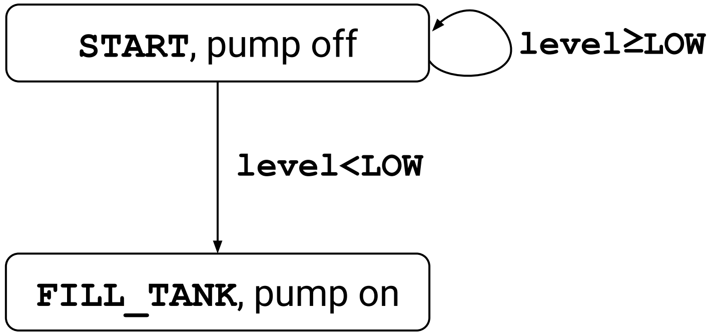
Ta'veren: FSM Recovery



Worklist:

Concrete state	Input constraints	Prev abs state
S_1	level ≥ LOW	**START**, pump off
S_1	level < LOW	**START**, pump off

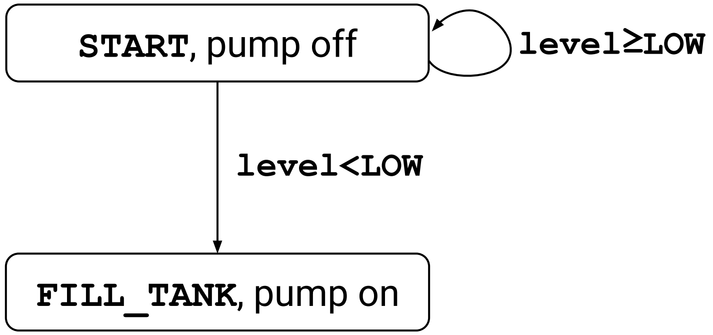
Ta’veren: FSM Recovery



Worklist:

Concrete state	Input constraints	Prev abs state
S_1	level < LOW	START, pump off
S_2	level ≥ LOW	START, pump off
S_2	level < LOW	START, pump off

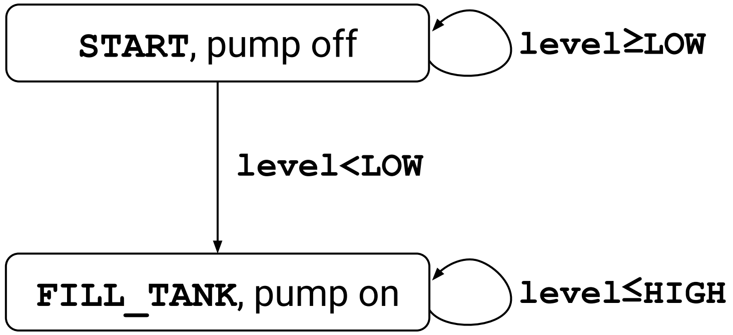
Ta’veren: FSM Recovery



Worklist:

Concrete state	Input constraints	Prev abs state
S_3	level ≤ HIGH	**FILL_TANK**, pump on
S_3	level > HIGH	**FILL_TANK**, pump on

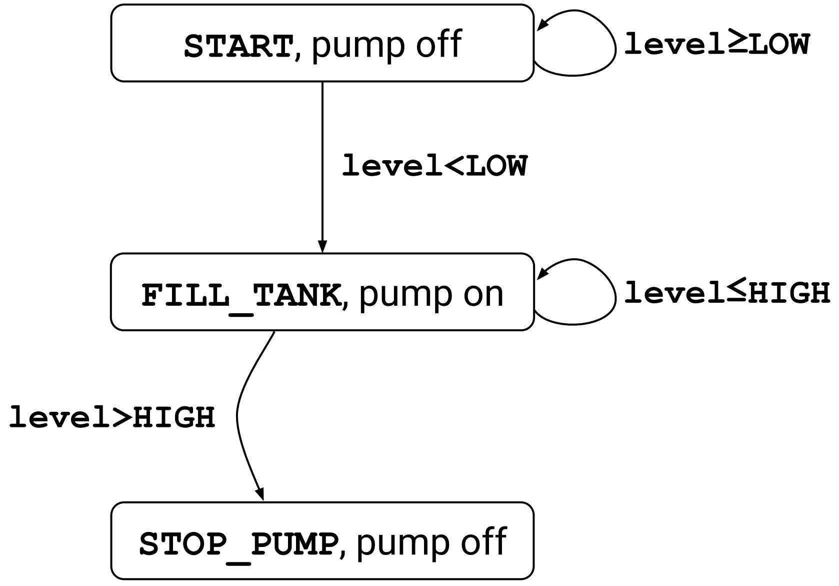
Ta’veren: FSM Recovery



Worklist:

Concrete state	Input constraints	Prev abs state
S_3	level ≤ HIGH	**FILL_TANK**, pump on
S_3	level > HIGH	**FILL_TANK**, pump on

Ta’veren: FSM Recovery

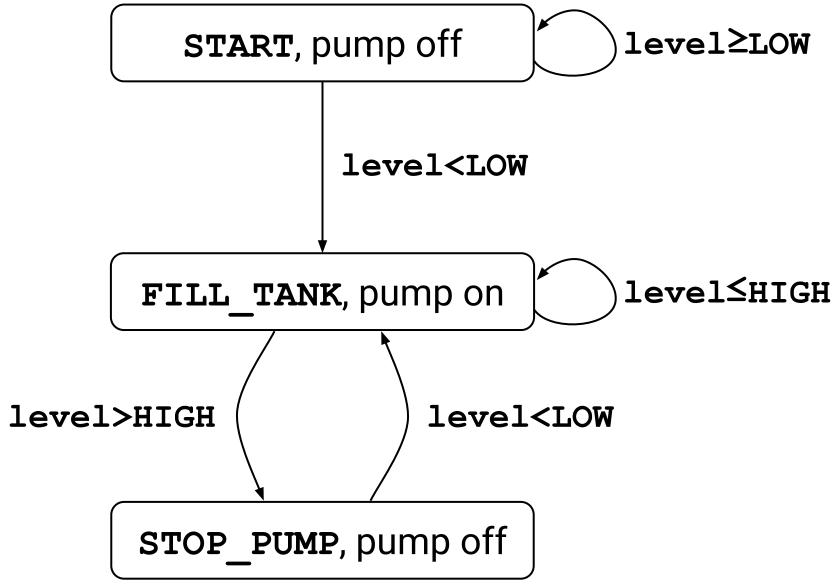


Worklist:

Concrete state	Input constraints	Prev abs state
S_3	level > HIGH	**FILL_TANK**, pump on
S_4	level ≤ HIGH	**FILL_TANK**, pump on
S_4	level > HIGH	**FILL_TANK**, pump on

security engineering for future computing

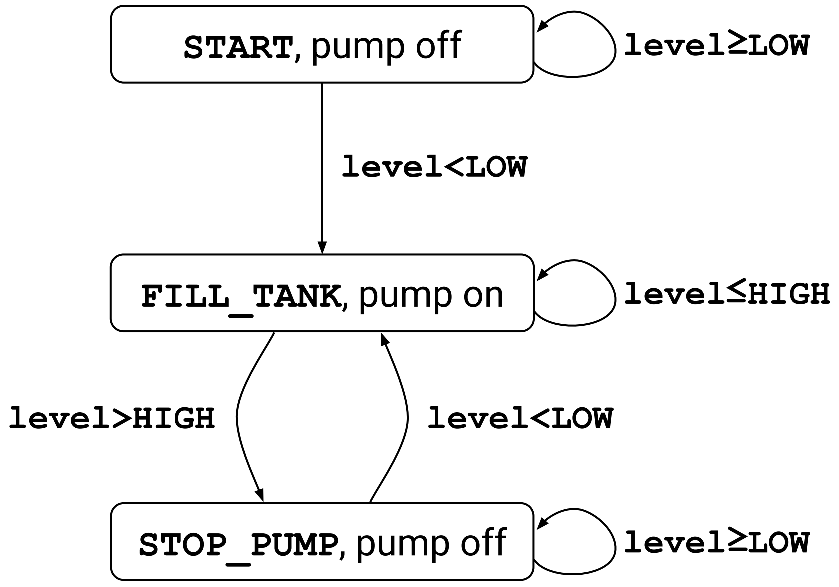
Ta’veren: FSM Recovery



Worklist:

Concrete state	Input constraints	Prev abs state
S_5	level < LOW	**STOP_PUMP**, pump off
S_5	level ≥ LOW	**STOP_PUMP**, pump off

Ta'veren: FSM Recovery

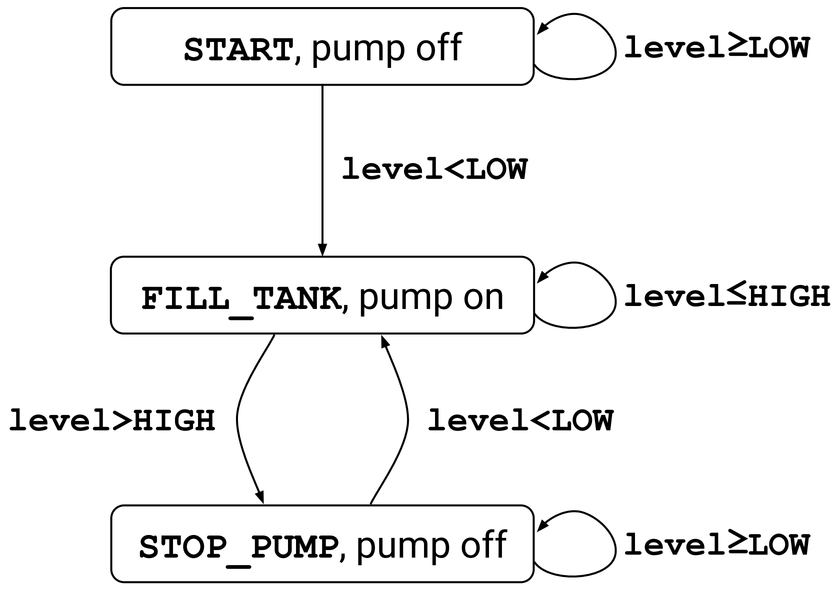


Worklist:

Concrete state	Input constraints	Prev abs state
S_5	level ≥ LOW	**STOP_PUMP**, pump off
S_6	level ≤ HIGH	**FILL_TANK**, pump on
S_6	level > HIGH	**FILL_TANK**, pump on

sefcom
security engineering for future computing

Ta’veren: FSM Recovery



Worklist:

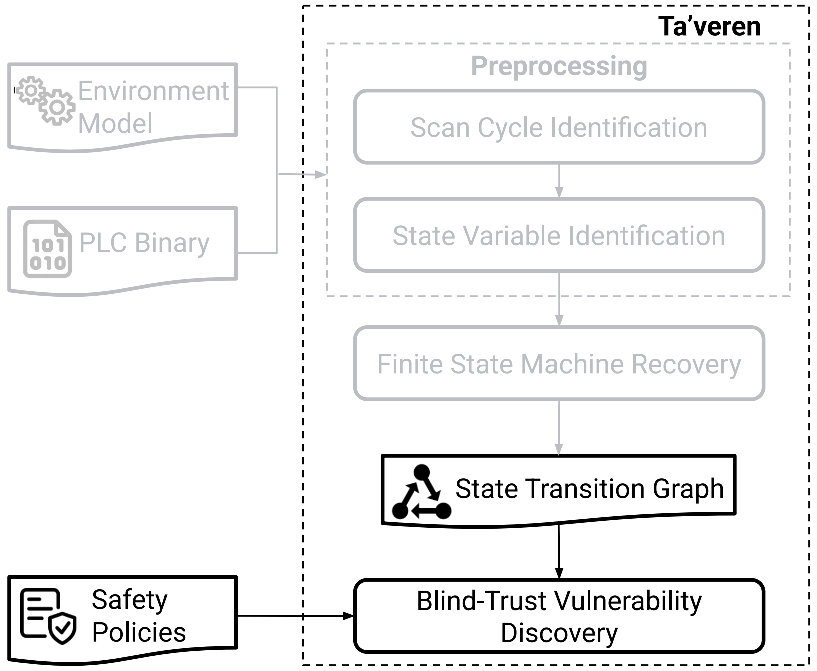
Concrete state	Input constraints	Prev abs state
S_7	level < LOW	STOP_PUMP, pump off
S_7	level ≥ LOW	STOP_PUMP, pump off

security engineering for future computing

Ta’veren: BTV Discovery

Model checking the generated FSM

- Comply or violate
- Input sequence to the violation state



Ta’veren
Preprocessing
Environment Model
Scan Cycle Identification
PLC Binary
State Variable Identification
Finite State Machine Recovery
State Transition Graph
Safety Policies
Blind-Trust Vulnerability Discovery

Evaluation: Dataset

Build dataset from real examples and synthesized programs

- 22 binaries
- 9 categories
  - Warehouse Lifter, Water Tank, Packaging System, Car Wash, Traffic Light, Launch Abort System, Oven, Vending Machine, and Elevator
- 4 toolchains
  - OpenPLC, Beremiz, Simulink, and Arduino
- 5 architectures
  - x86-64, ARM, MIPS, PowerPC, and AVR8

sefcom
security engineering for future computing

Evaluation: Effectiveness of BTV Discovery

17 BTVs (23 violations)

Evaluation: Effectiveness of BTV Discovery

17 BTVs (23 violations)

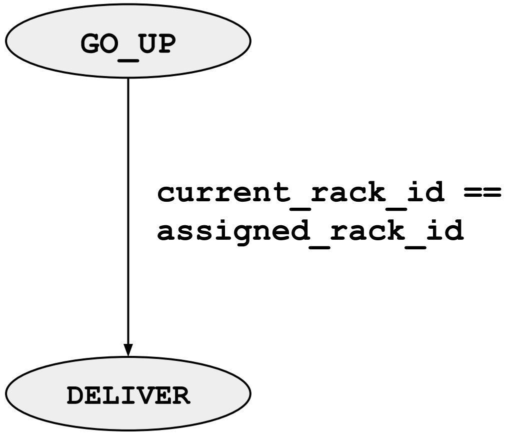
Root causes:

- C.1. Incomplete range handling
- C.2. Incorrect input check
- C.3. Unhandled input combination
- C.4. Unchecked input acceptance
- C.5. Wrong action
- C.6. Missing input handling

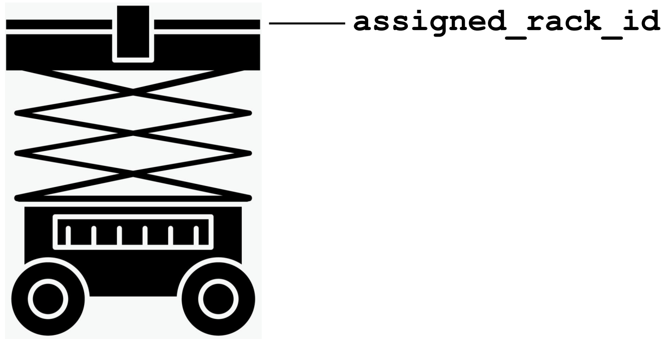
sefcom
security engineering for future computing

C.1 Incomplete Range Handling

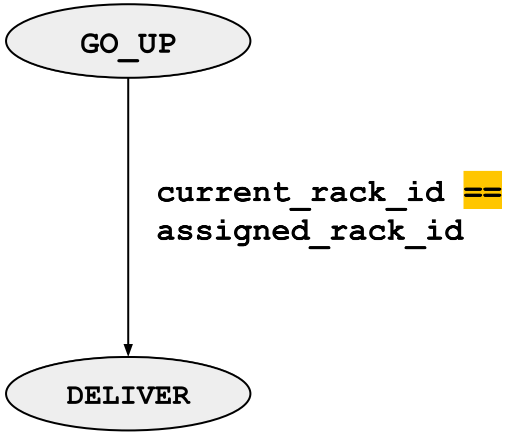
Warehouse Lifter



C.1 Incomplete Range Handling

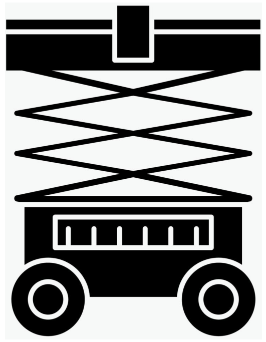


Warehouse Lifter

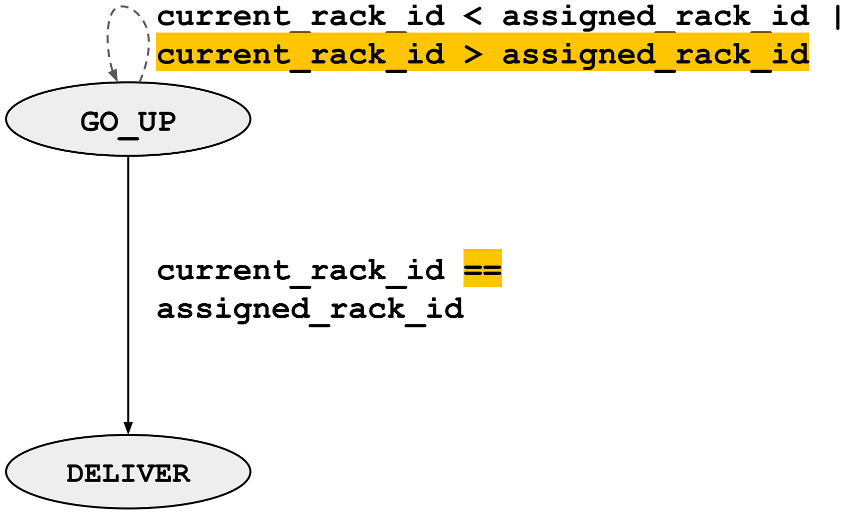


sefcom
security engineering for future computing

C.1 Incomplete Range Handling



Warehouse Lifter


current_rack_id < assigned_rack_id |
current_rack_id > assigned_rack_id
GO_UP
current_rack_id ==
assigned_rack_id
DELIVER

Evaluation: Effectiveness of BTV Discovery

17 BTVs (23 violations)

Root causes:

C.1. Incomplete range handling

C.2. Incorrect input check

C.3. Unhandled input combination

C.4. Unchecked input acceptance

C.5. Wrong action

C.6. Missing input handling

sefcom
security engineering for future computing

Evaluation: Efficiency

Pack.1	Preprocessing	FSM Recovery	BTV Discovery
Time Spent (seconds)	27	42	0.0008

- FSM recovery takes the most amount of time
- Time $\propto$ size of FSM $\times$ size of scan cycle function

sefcom security engineering for future computing

Evaluation on Robotic Vehicle Binaries

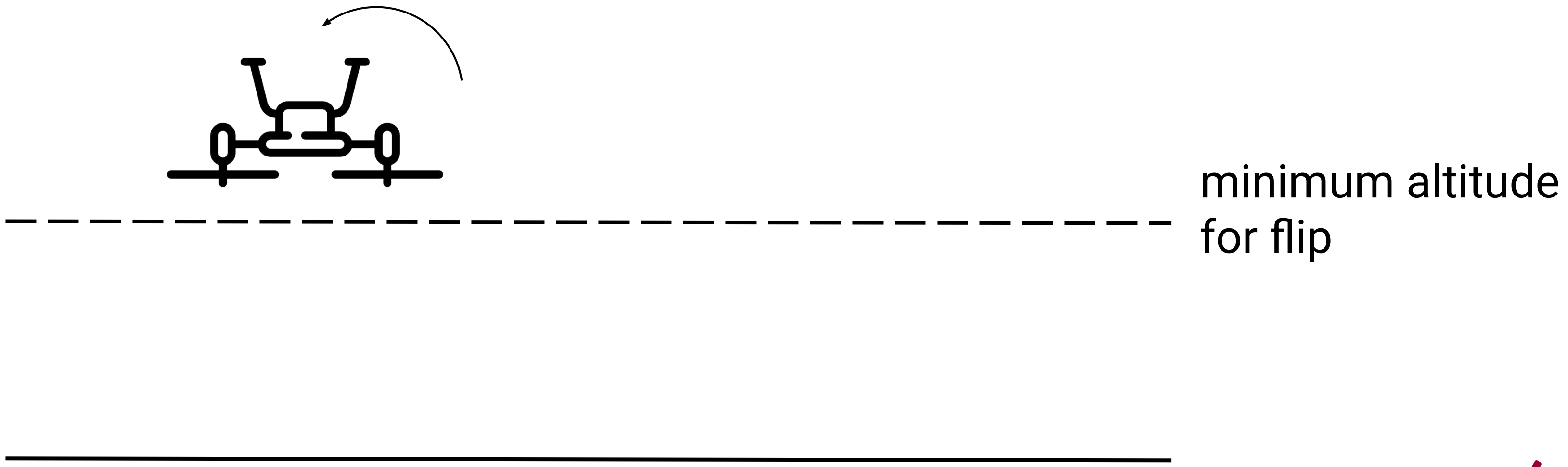
Arducopter (mode flip)

Rover

Evaluation on Robotic Vehicle Binaries

Arducopter (mode flip)

- Missing input handling (C.6) of altitude.

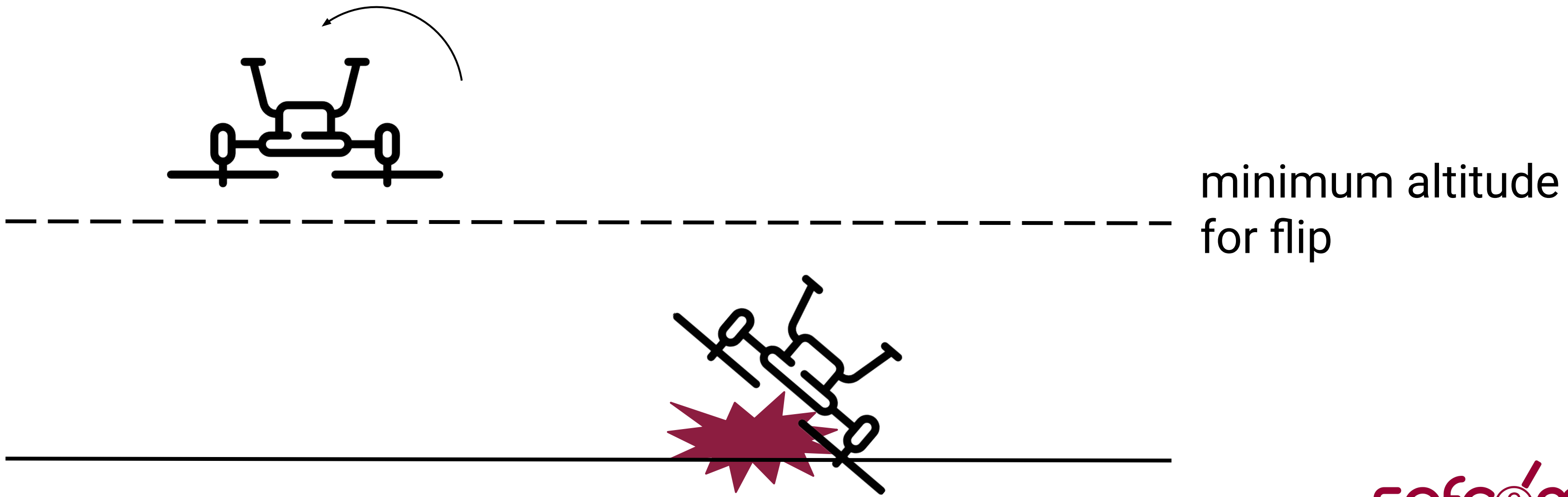


sefcom
security engineering for future computing

Evaluation on Robotic Vehicle Binaries

Arducopter (mode flip)

- Missing input handling (C.6) of altitude.



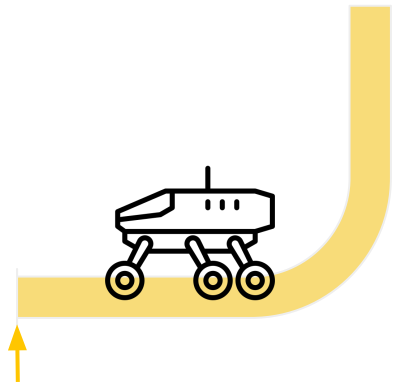
Evaluation on Robotic Vehicle Binaries

Arducopter (mode flip)

- Missing input handling (C.6) of altitude.

Rover

- Unchecked input acceptance (C.4) of a user command.



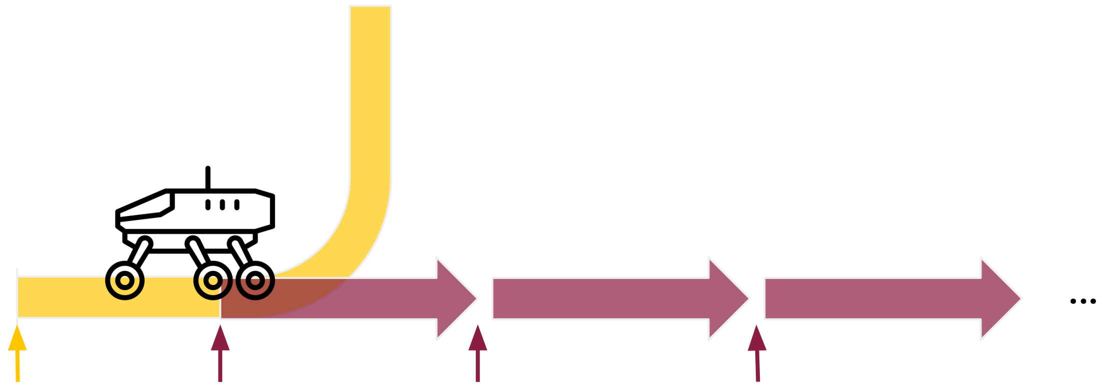
Evaluation on Robotic Vehicle Binaries

Arducopter (mode flip)

- Missing input handling (C.6) of altitude.

Rover

- Unchecked input acceptance (C.4) of a user input.



Thank you!

Ta’veren | https://github.com/sefcom/taveren

Fangzhou “Bonnie” Dong | bonniedong@asu.edu | bonnie1256.github.io

Arizona State University

BE BOLD. Shape the Future.® New Mexico State University

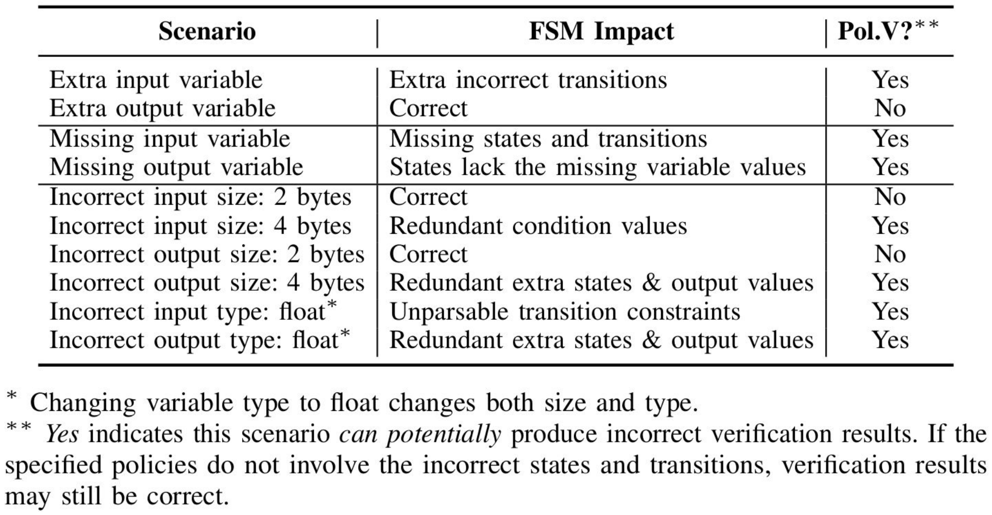
Backup Slides

Summary

- BTVs are unassumed inputs causing safety violations.
- Ta’veren finds BTVs by FSM recovery and model checking.
- We build a realistic dataset and show Ta’veren’s capabilities.

Evaluation: Impact of Inaccurate Environment Models

- Robust against output variable inaccuracies
- Sensitive to input variable inaccuracies
- Tolerates variable configuration errors, but not out-of-bound accesses



Scenario	**FSM Impact**	**Pol.V?**[**]
Extra input variable	Extra incorrect transitions	Yes
Extra output variable	Correct	No
Missing input variable	Missing states and transitions	Yes
Missing output variable	States lack the missing variable values	Yes
Incorrect input size: 2 bytes	Correct	No
Incorrect input size: 4 bytes	Redundant condition values	Yes
Incorrect output size: 2 bytes	Correct	No
Incorrect output size: 4 bytes	Redundant extra states & output values	Yes
Incorrect input type: float[*]	Unparsable transition constraints	Yes
Incorrect output type: float[*]	Redundant extra states & output values	Yes

[*] Changing variable type to float changes both size and type.
[**] *Yes* indicates this scenario *can potentially* produce incorrect verification results. If the specified policies do not involve the incorrect states and transitions, verification results may still be correct.