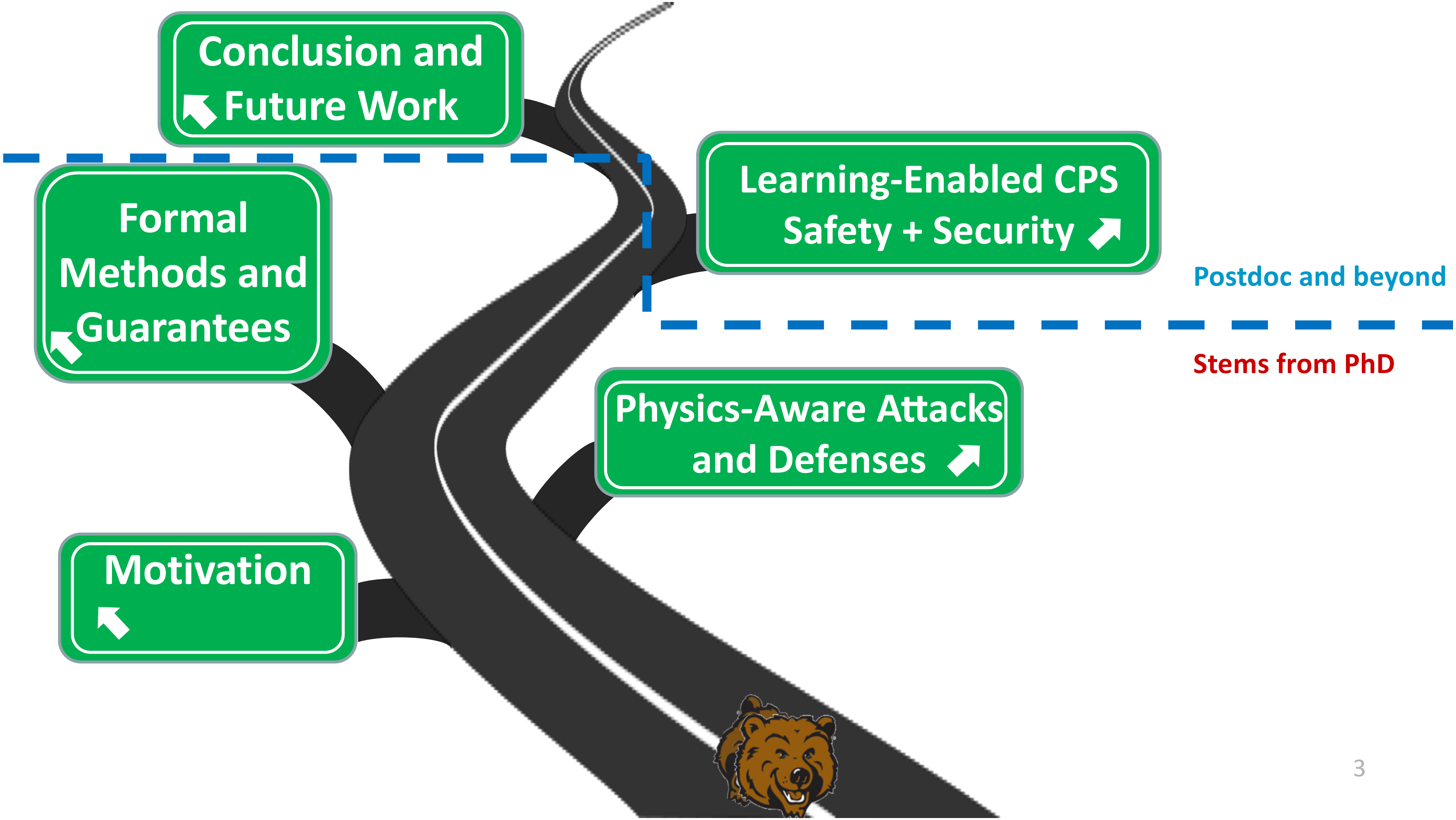
Feeling Safe and Secure in a Learning-Enabled Cyber-physical World

Luis Garcia
UCLA

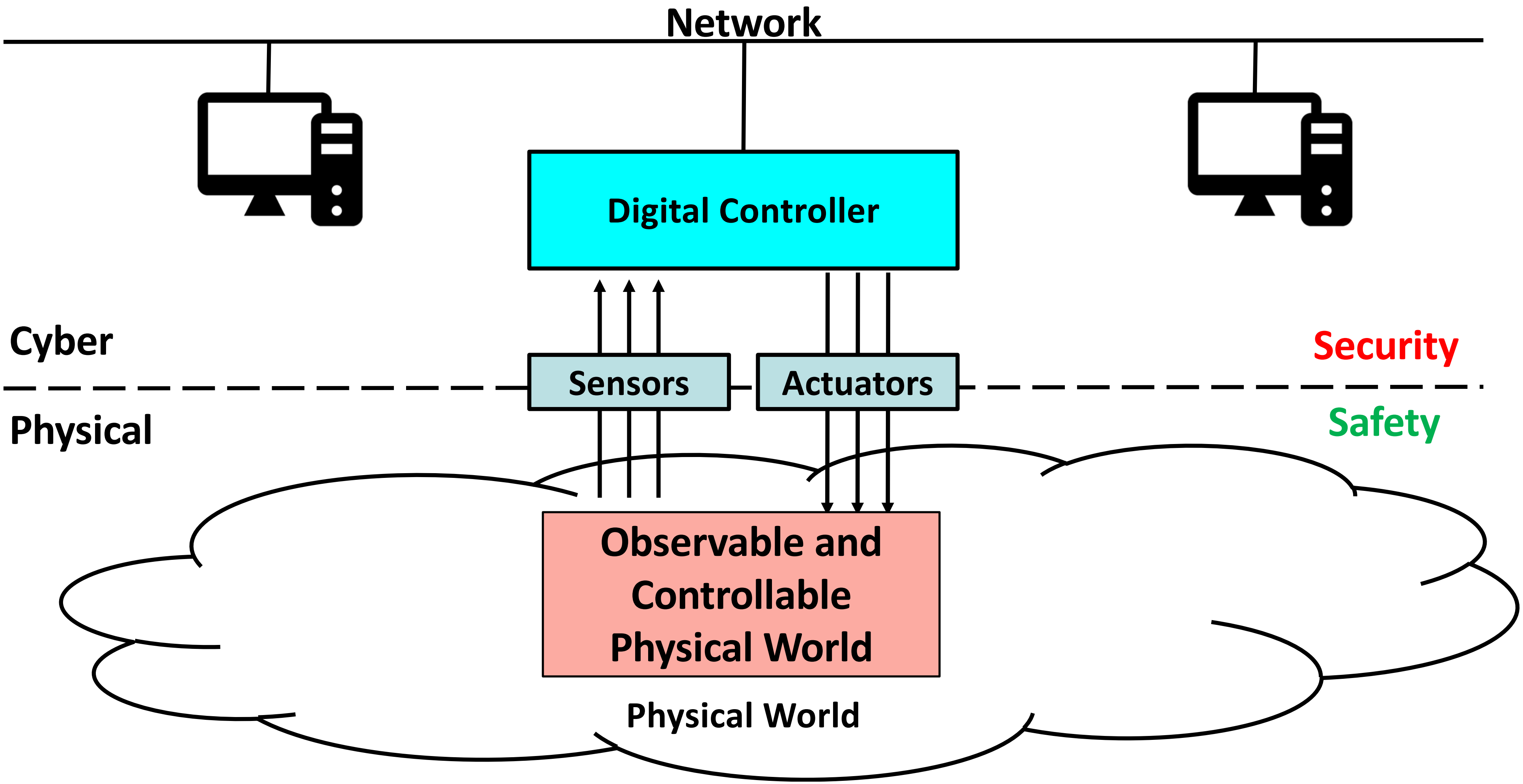
IFIP WG10.4 WINTER 2020
1/29/20

About myself…

- Postdoctoral Scholar in ECE/CS at UCLA
  - NESL Research Group
- PhD in Computer Engineering with Cybersecurity Track at Rutgers
  - 4N6 Research Group
  - GAANN Fellow (Graduate Assistance in Areas of National Need)
- BS and MS in ECE from the University of Miami
- Academic Intern at CMU with André Platzer
- Previous Intern at Siemens Corporate Research


Conclusion and Future Work
Learning-Enabled CPS Safety + Security
Formal Methods and Guarantees
Postdoc and beyond
Stems from PhD
Physics-Aware Attacks and Defenses
Motivation

What does it mean to feel "safe" and "secure"?



Cyber-Physical Industrial Control Systems (ICS)

Water Treatment

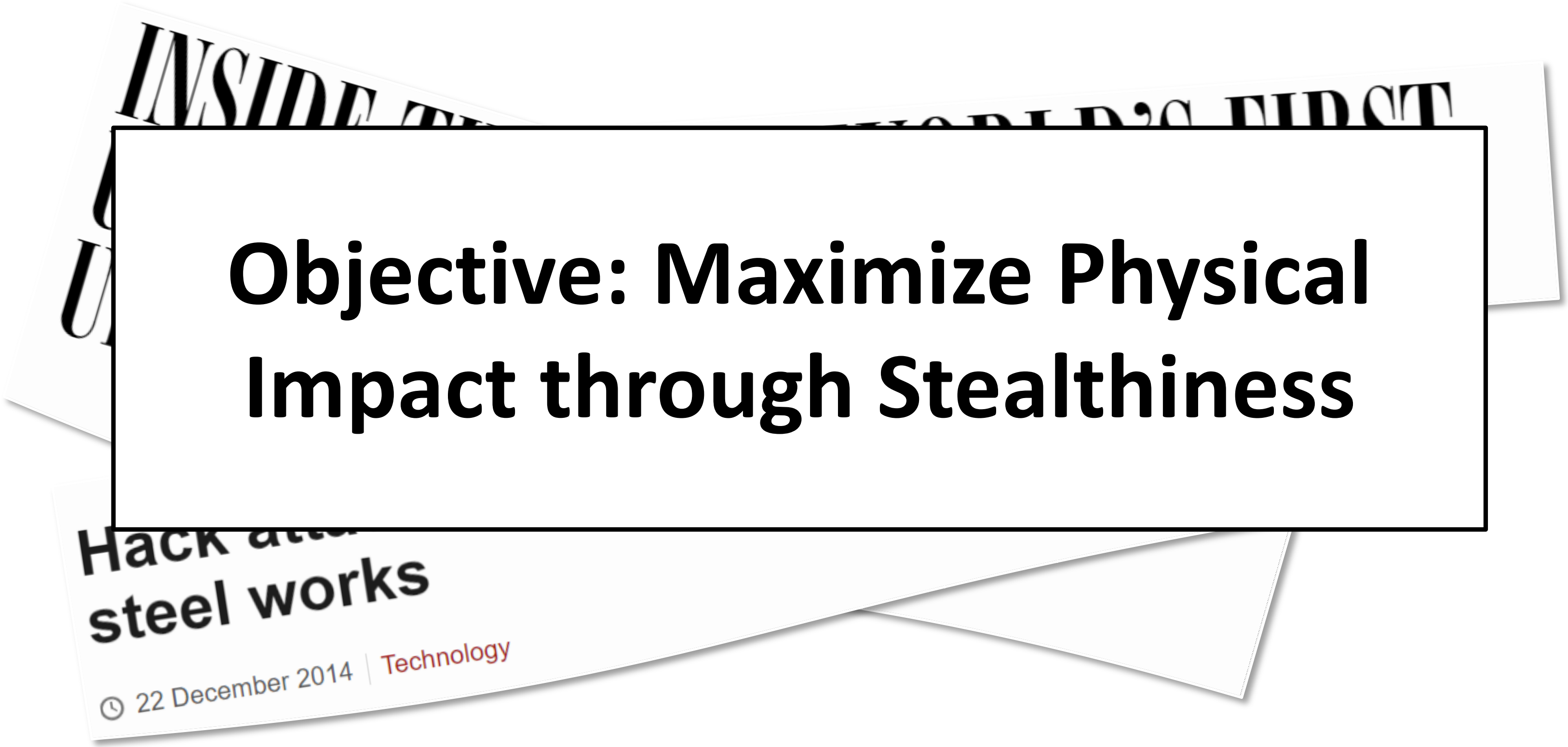
Factory Automation

Great targets to attack!

Electric Power Grid

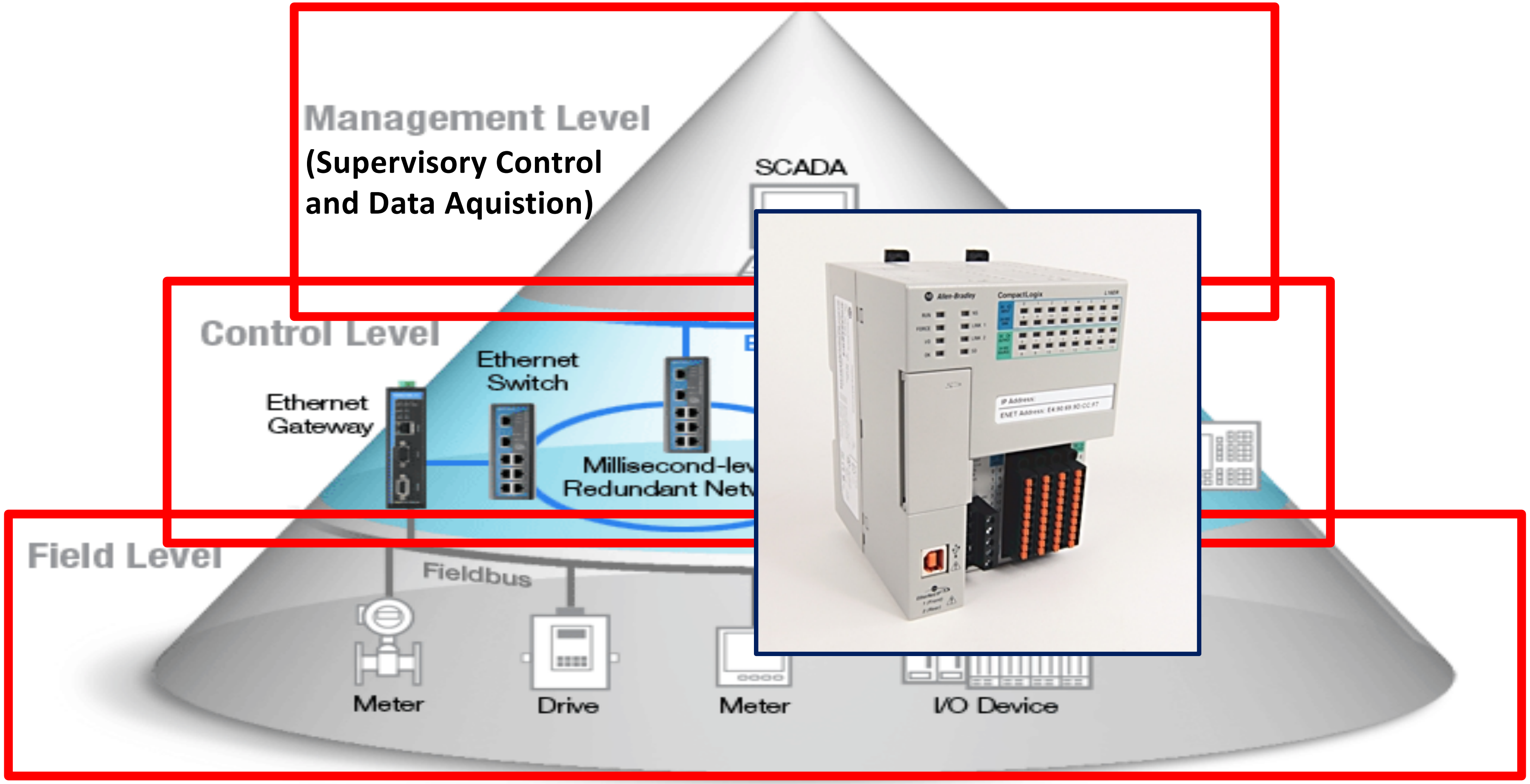
Nuclear Reactor

Industrial Control System (ICS) Attacks



Objective: Maximize Physical Impact through Stealthiness

Programmable Logic Controllers (PLCs) and Industrial Control Systems (ICS)



Stuxnet/BlackEnergy Attack Overview

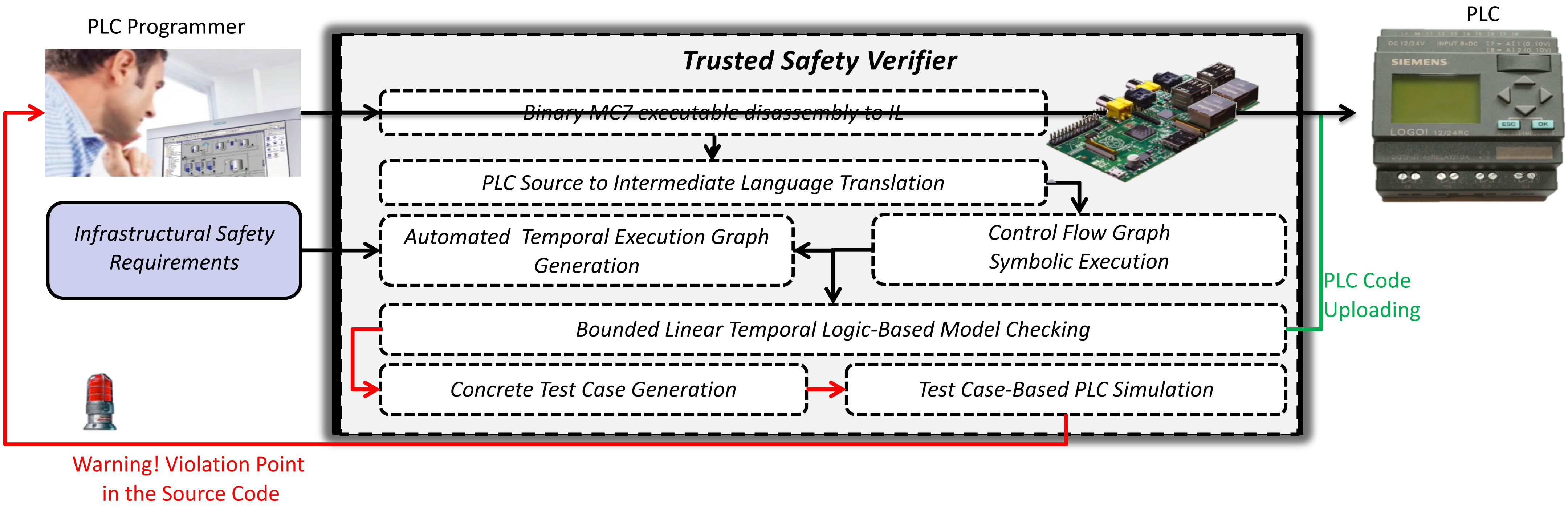
BlackEnergy3 plugins	functionality
fs.dll	File system operations
jn.dll	Parasitic infector with a given payload
ps.dll	Password stealer
si.dll	System information
ss.dll	screenshots
vs.dll	Network discovery and remote execution
up.dll	Update malware
tv.dll	TeamViewer
dc.dll	List Windows accounts
bs.dll	Query system HW, BIOS, Windows info.
dstr.dll	Destroy system
kl.dll	Key-logger
scan.dll	Network host port scan
rd.dll	Simple pseudo 'remote desktop'
grc.dll	Back comm. channel using plus.google.com
cert.dll	certificate stealer
sn.dll	Logs traffic, extracts login-passwords
usb.dll	gathers info. on connected USB devices

PLC

STL code block

Observation: BlackEnergy3 – compared to Stuxnet - included many more reconnaissance plugins!

Trusted Safety Verifier (TSV) Overview



Infrastructural Safety Requirements

- Formulated using linear temporal logic (LTL) expressions

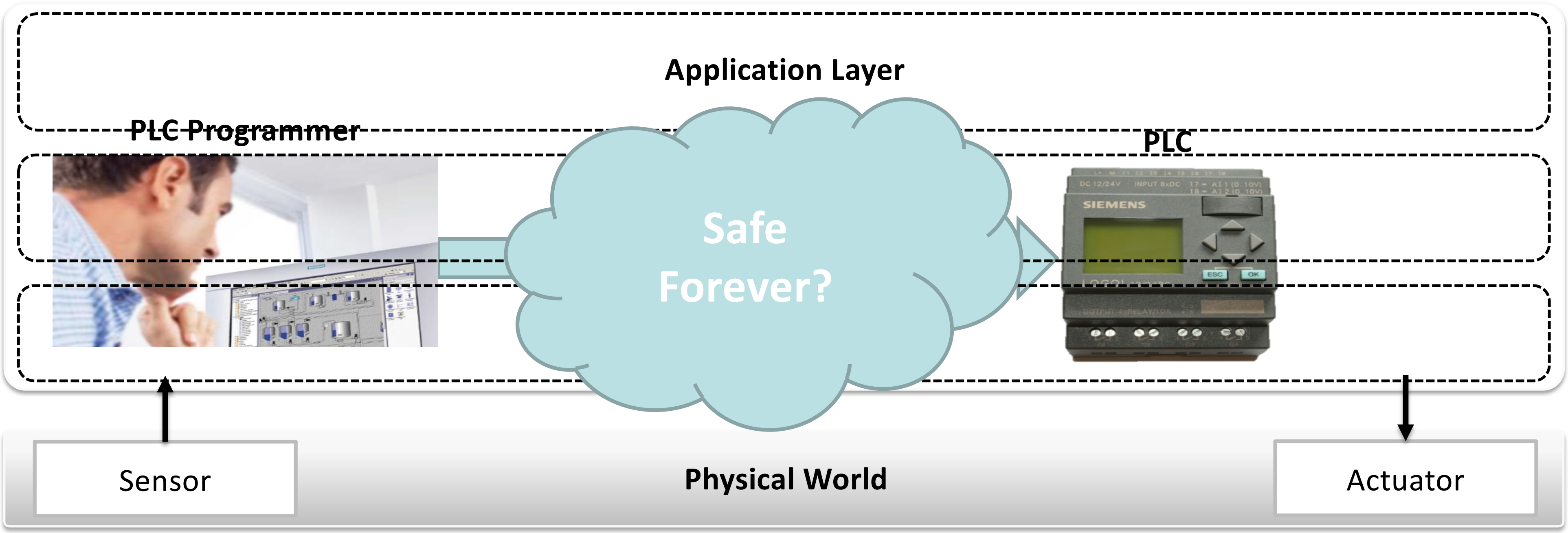
- Example safety requirement
  - *English expression*
    - *Relay R_1 should **NOT** open **UNTIL** Generator G_2 turns on*
  - *Logical expression*
    - *Atomic propositions*
      - a_1: "Relay R_1 is open"
      - a_2: "Generator G_2 is on"

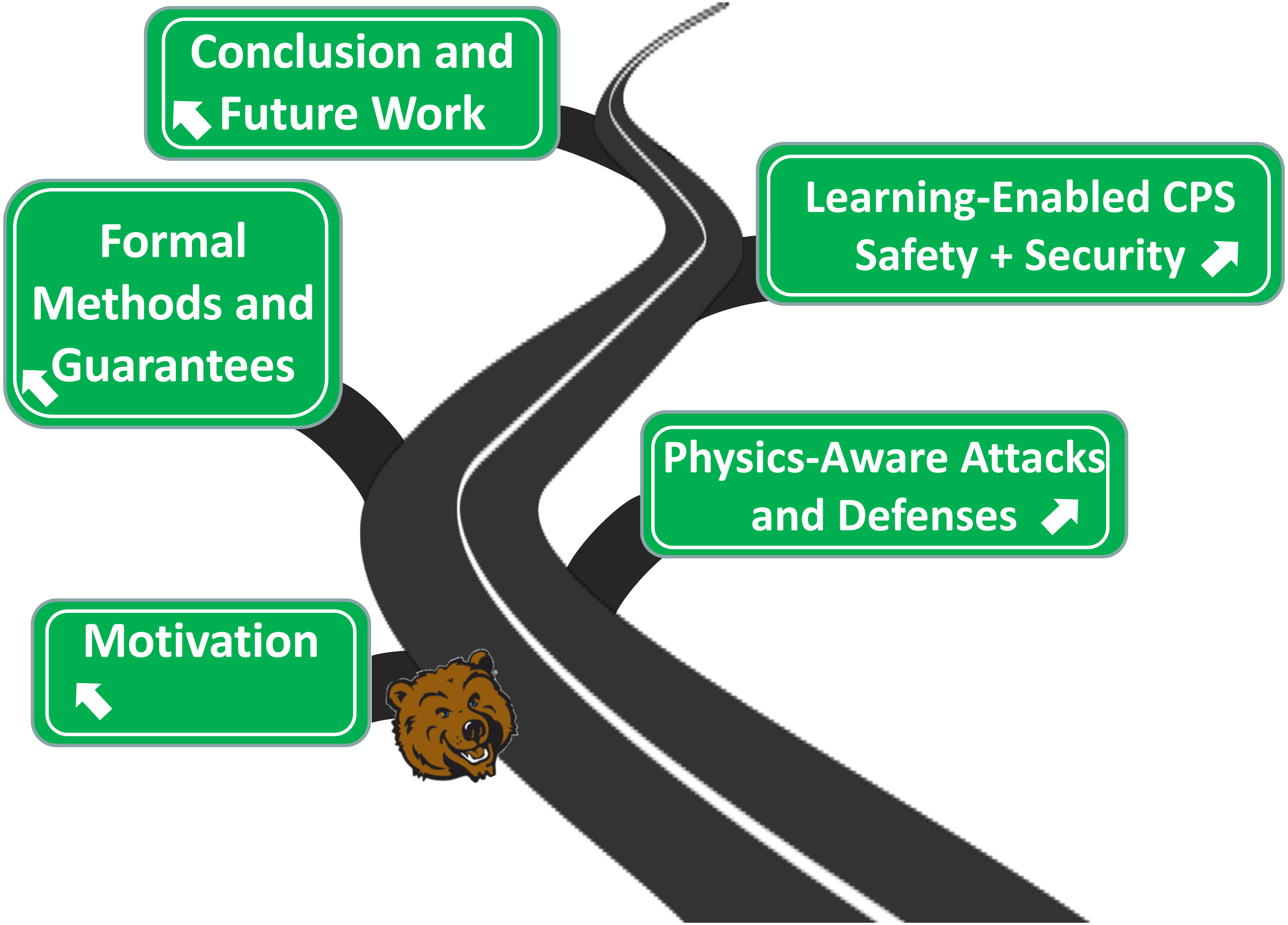
LTL: **!**a_1 **U** a_2

Limitations of Previous Solutions

- Application layer protection
  - No means of protecting lower levels of abstraction
- Physical safety properties are assumed to be provided

PLC Architecture




Conclusion and Future Work
Learning-Enabled CPS Safety + Security
Formal Methods and Guarantees
Physics-Aware Attacks and Defenses
Motivation

Hey, My Malware Knows Physics! Attacking PLCs with Physical Model Aware Rootkit

NDSS 2017

Harvey: Model-Aware Rootkit (NDSS 2017)

- A rootkit that takes into account the physical topology of the ICS
- Model
  - Uses physical models to optimize control commands for an adversarial objective function
- PLC infection: compromising the PLC's firmware
  - Utilize the firmware update mechanism to replace firmware over the network
  - Local firmware modifications, e.g., SD card or JTAG implantation
  - Run-time attacks, e.g., network exploits or remote code execution vulnerabilities (FrostyURL)

Stuxnet vs. Harvey

Stuxnet:

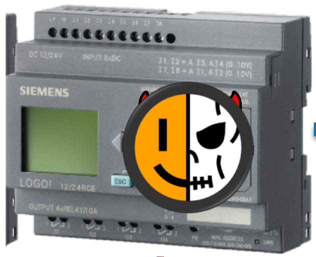
1. Compromised PLC control logic
2. Modified HMI to prevent detection of PLC modifications
3. Replayed benign measurements to HMI

Harvey:

1. Does not require SCADA compromise to remain stealthy
2. Can calculate fake data for dynamic systems, Stuxnet only replays recorded measurements

Actuator/Sensor

PLC



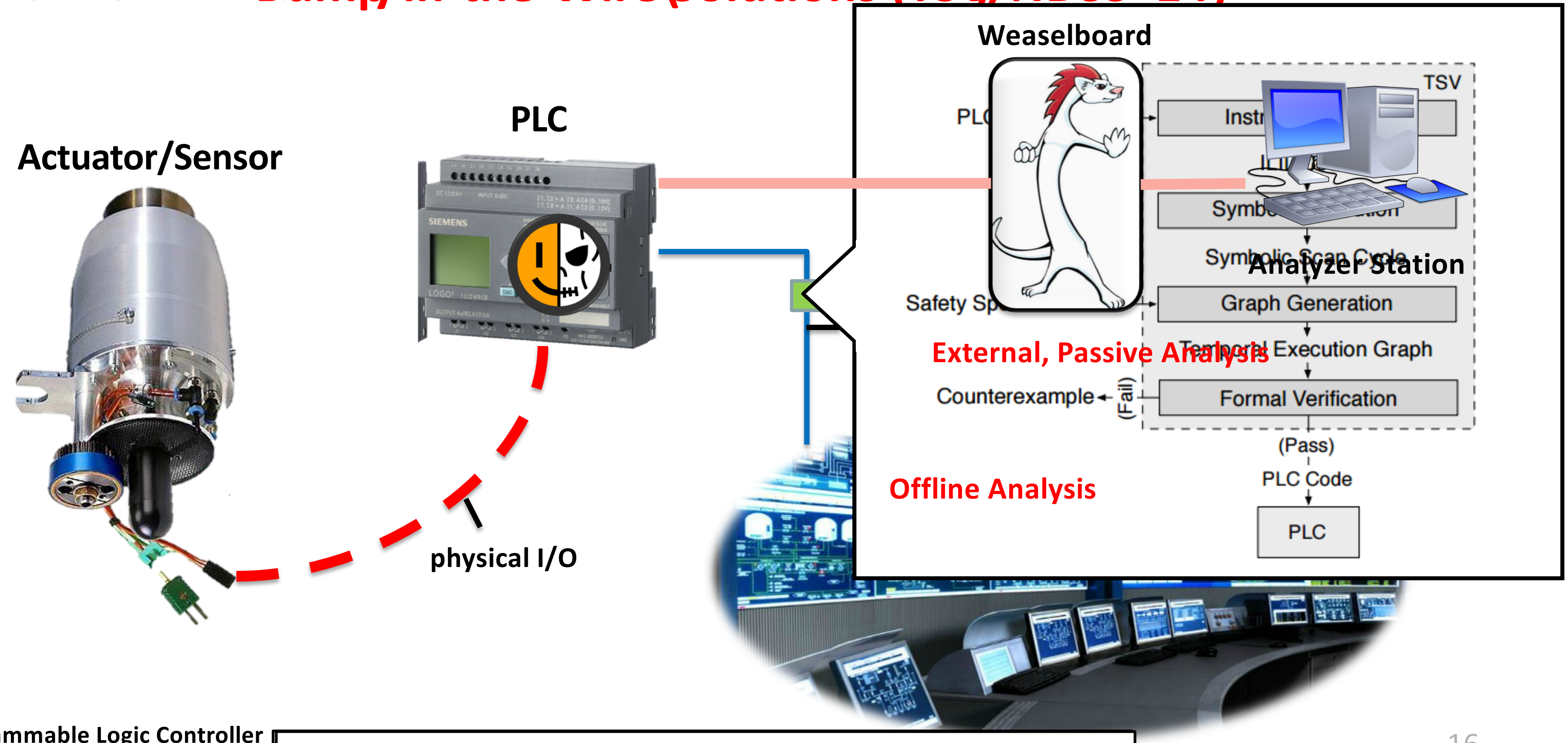
physical I/O

cyber

Supervisory Control (SCADA)

PLC: Programmable Logic Controller
HMI: Human Machine Interface

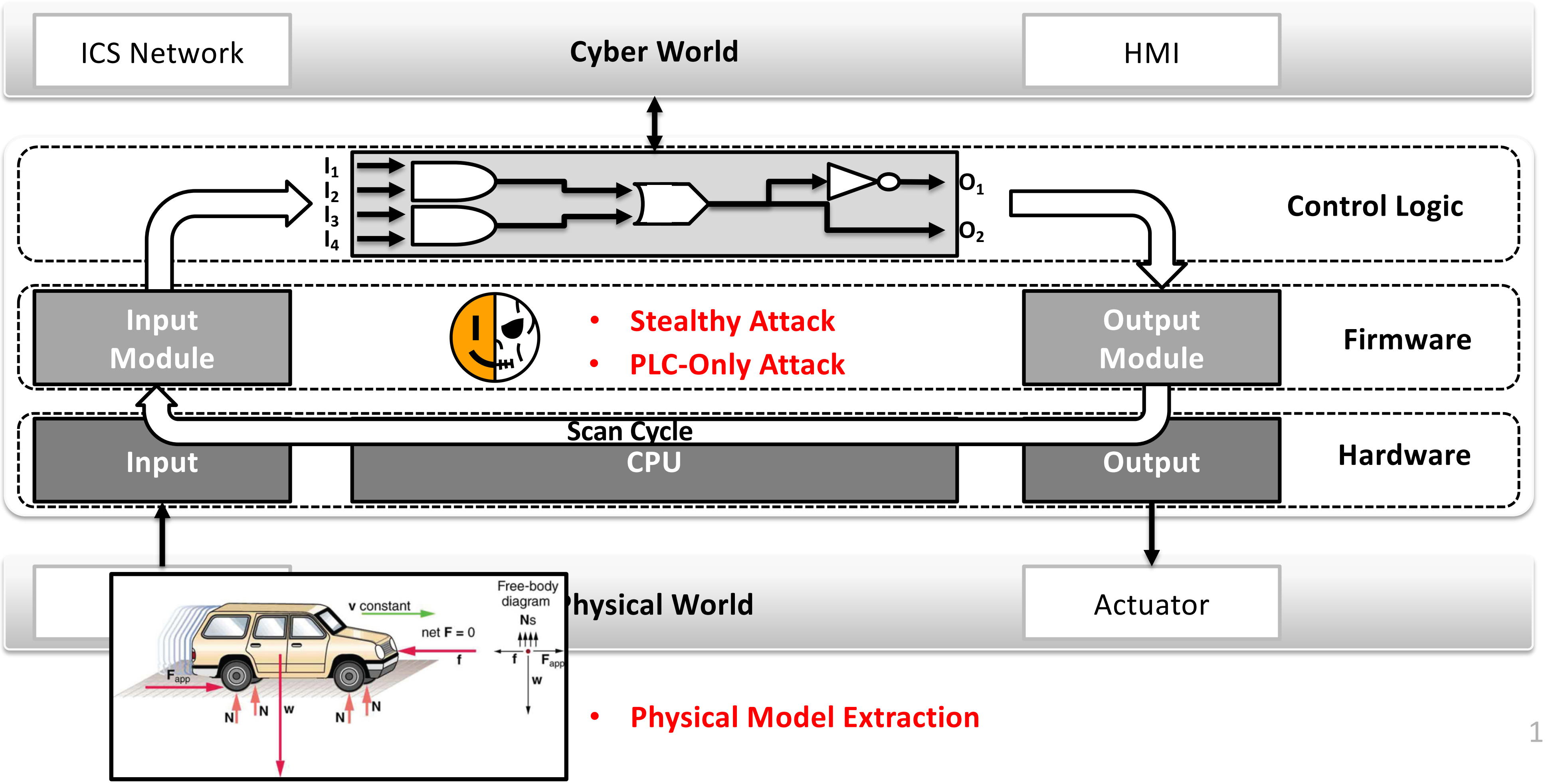
Stuxnet vs. Bump-in-the-Wire (Weaselboard) (TSV), *NDSS '14*)



PLC: Programmable Logic Controller
HMI: Human Machine Interface

McLaughlin, Stephen E., et al. "A Trusted Safety Verifier for Process Controller Code." *NDSS*. Vol. 14. 2014.

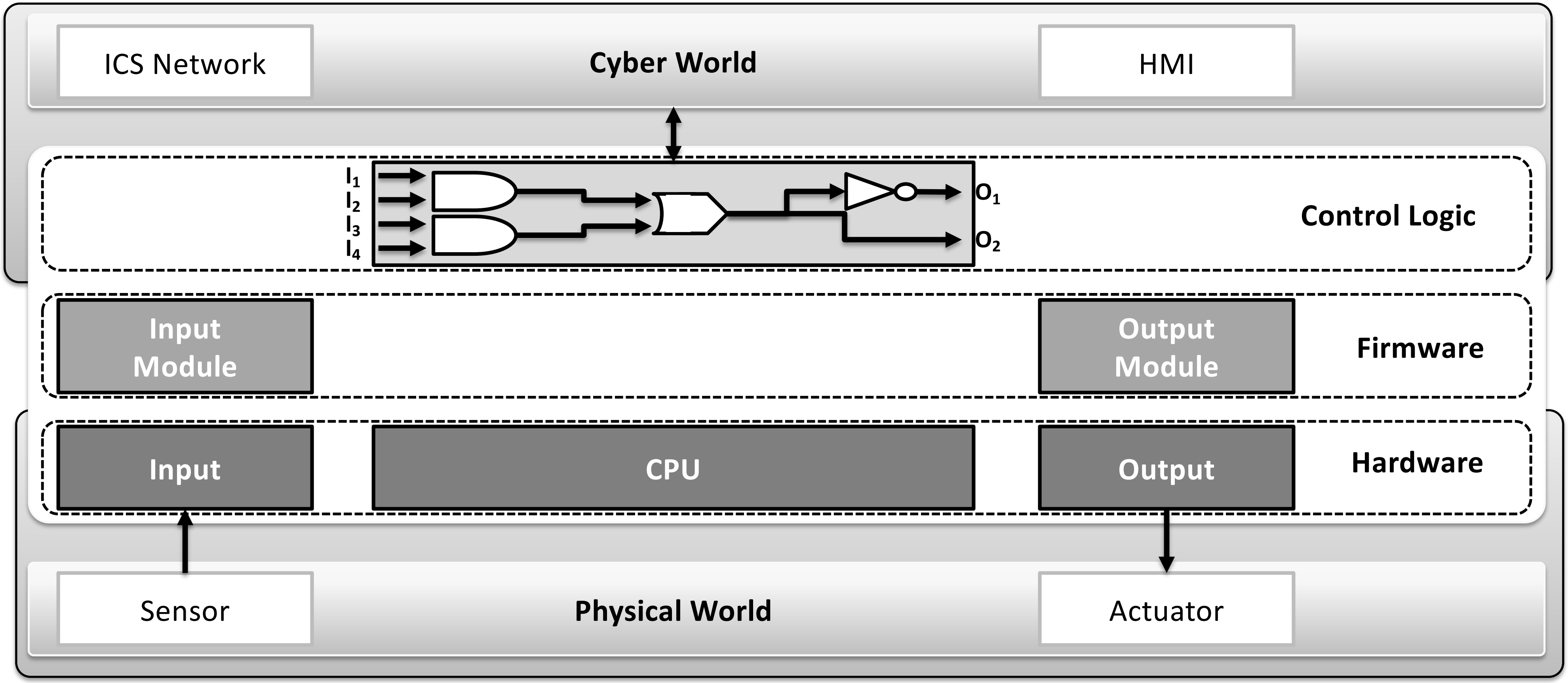
PLC Architecture & Adversary Model



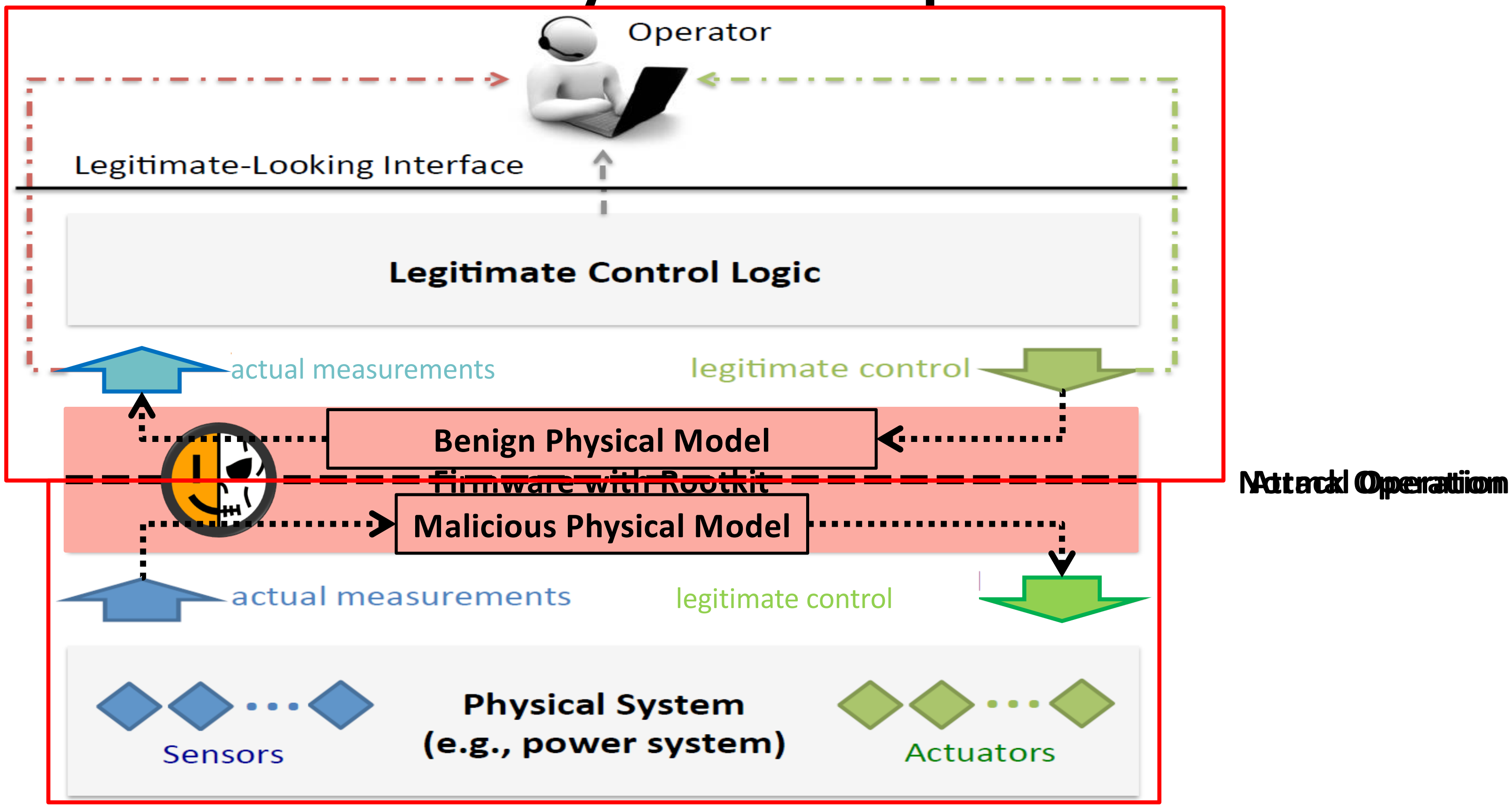
- Physical Model Extraction

Physics-Awareness: 2-Way Data Manipulation

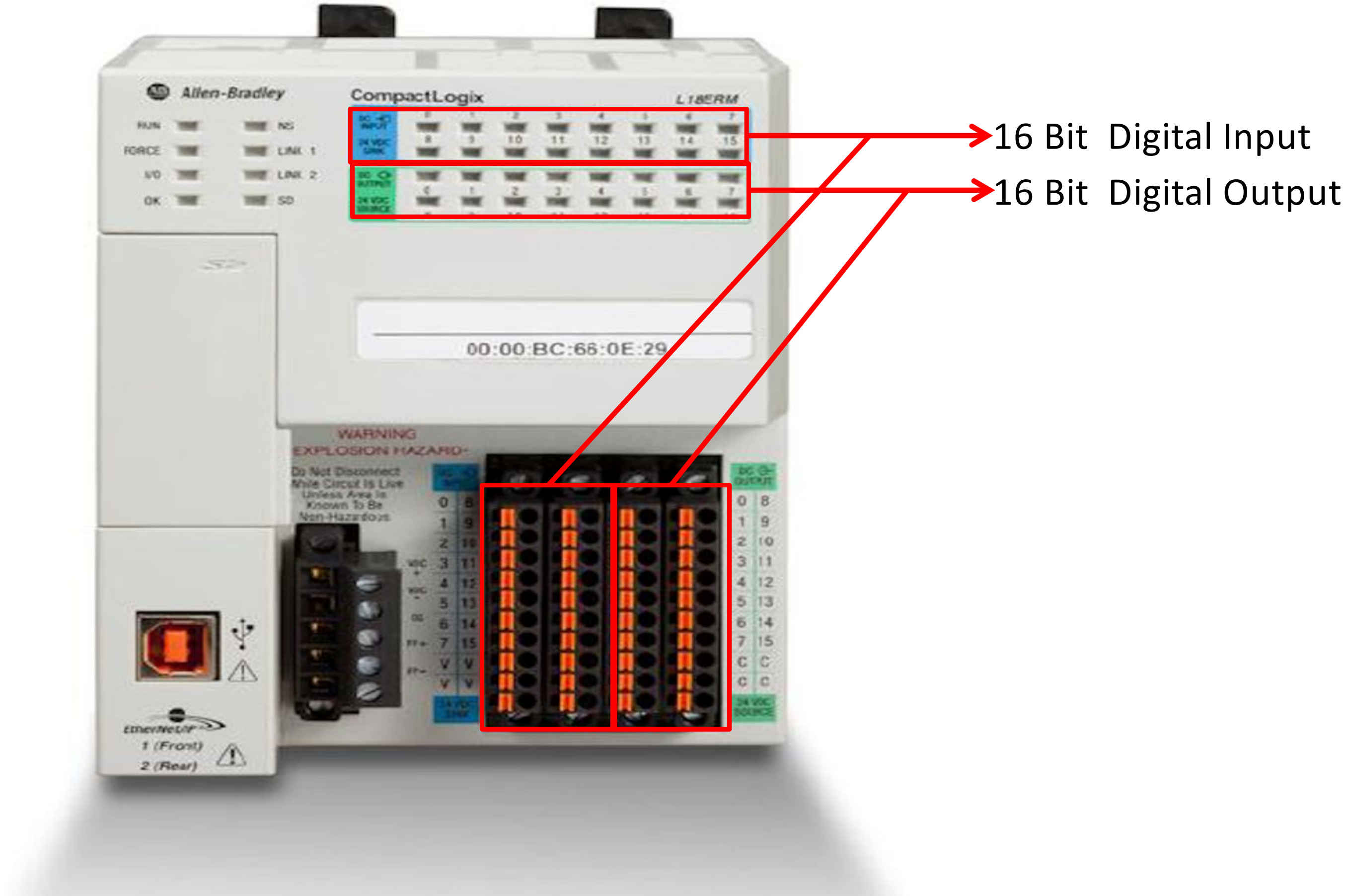
PLC Architecture & Adversary Model



Physics-Awareness: 2-Way Data Manipulation

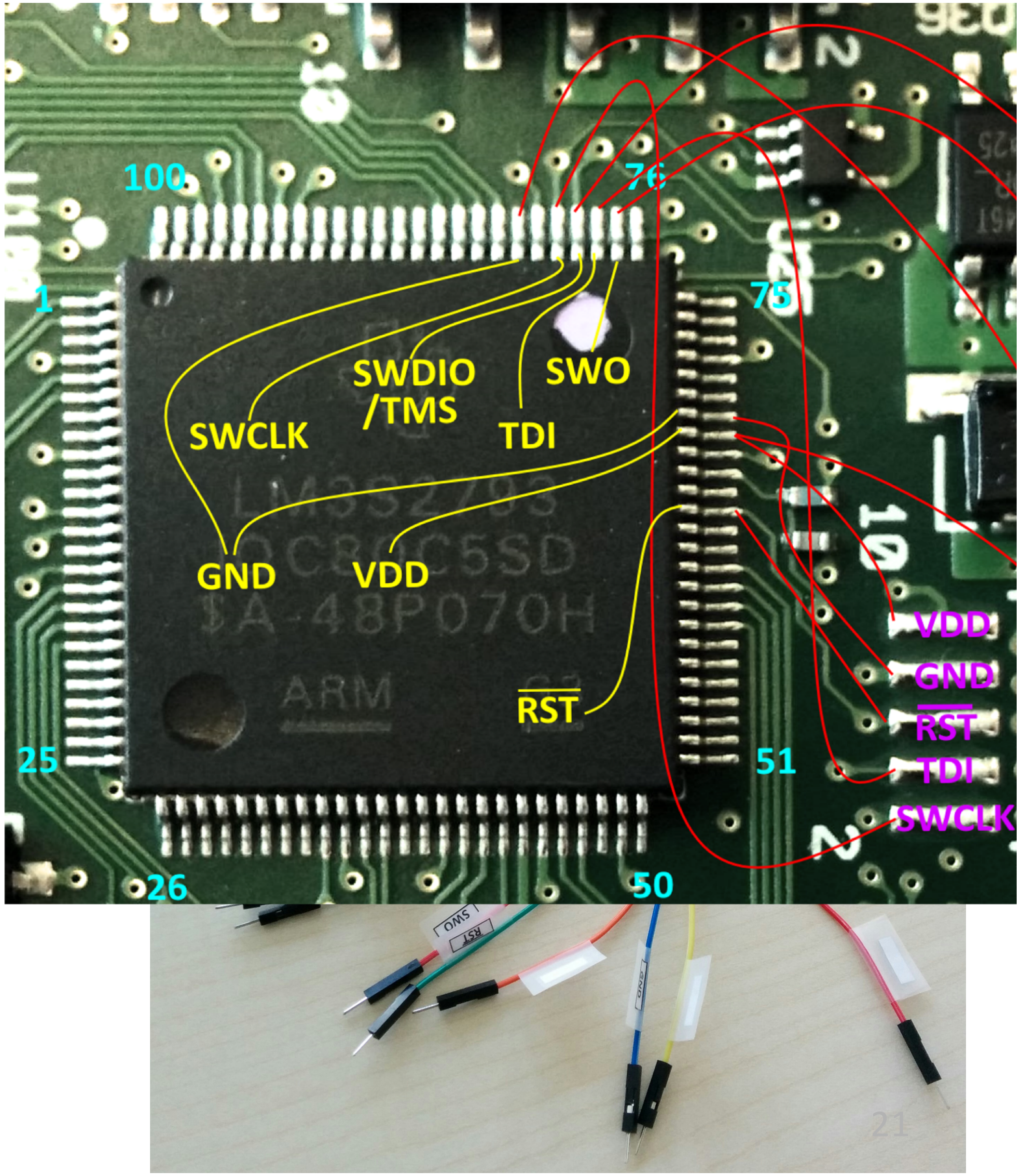


CompactLogix L1 PLC



- High Value (1) ~ 24 V DC
- Low Value (0) ~ 8 V DC

Memory Analysis with JTAG



Memory Analysis with JTAG

- JTAG interface to dump memory for code disassembly
- TI Stellaris LM3S2793 data sheet to find memory layout and built-in ROM functions


TEXAS INSTRUMENTS-PRODUCTION DATA

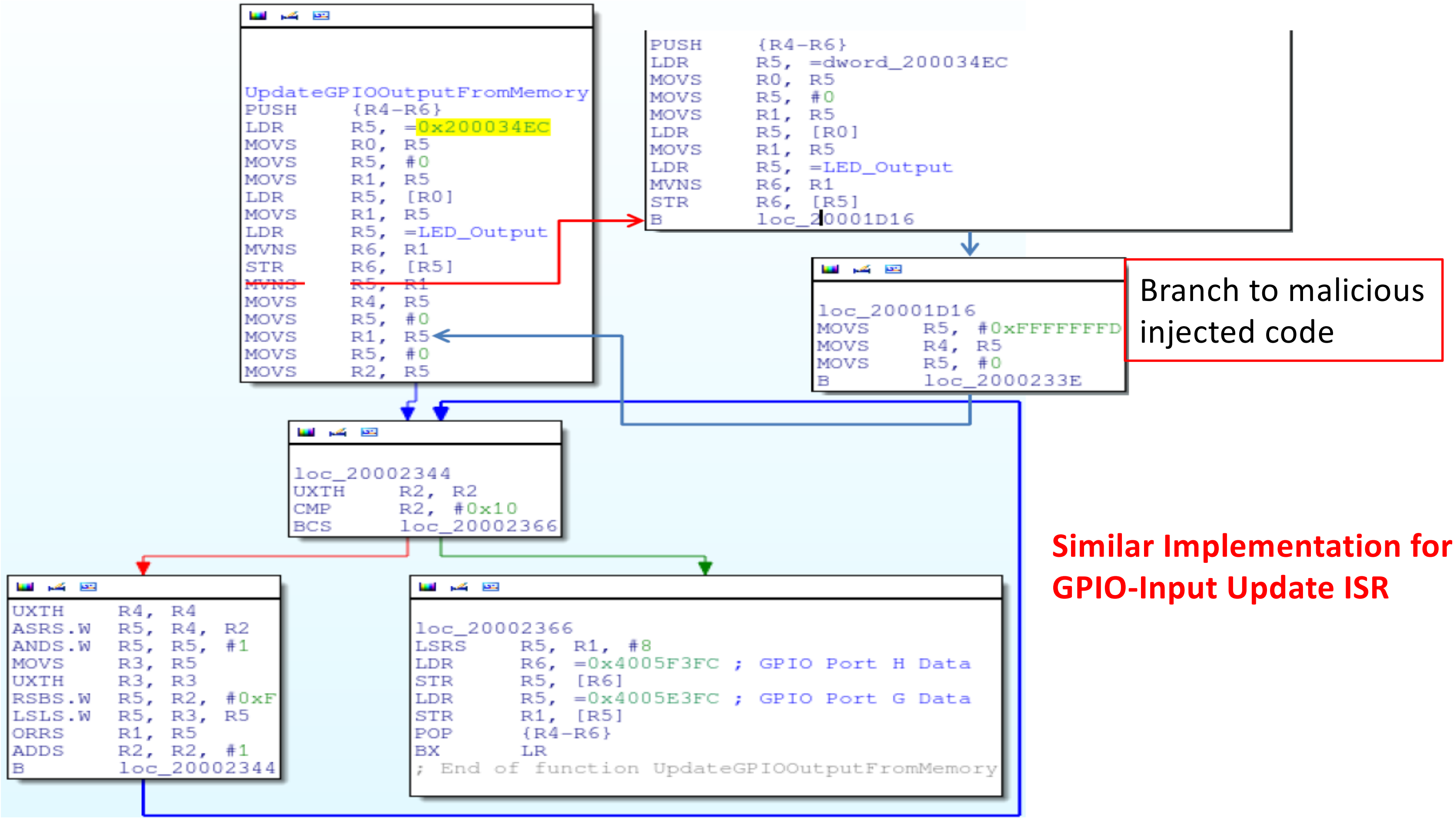
TEXAS INSTRUMENTS

Stellaris® LM3S2793 Microcontroller

DATA SHEET

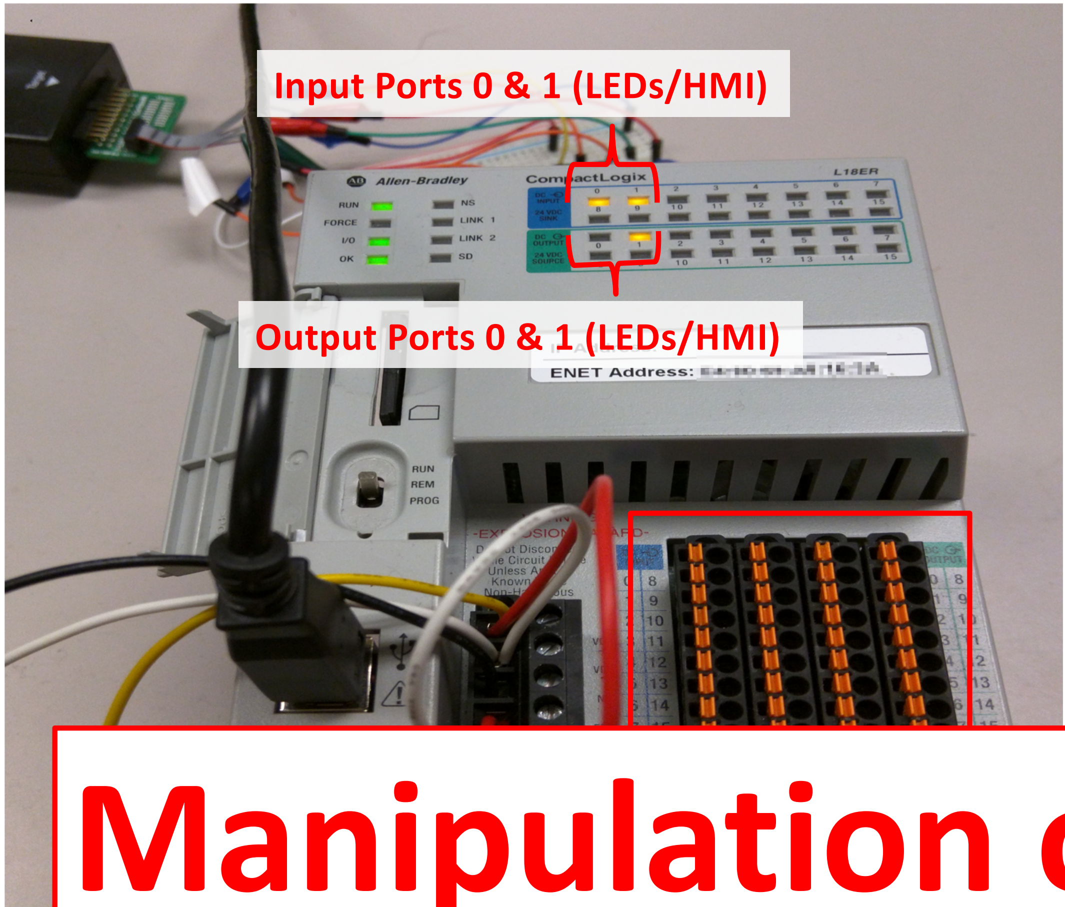
Start	End	Description
0x00000000	0x0001FFFF	On-chip Flash
0x00020000	0x00FFFFFF	Reserved
0x01000000	0x1FFFFFFF	ROM
0x20000000	0x2000FFFF	On-chip SRAM
0x20010000	0x21FFFFFF	Reserved
0x22000000	0x221FFFFF	Bit-band alias of SRAM
...	...	
0x4005C000	0x4005CFFF	GPIO Port E (AHB)
0x4005D000	0x4005DFFF	GPIO Port F (AHB)
0x4005E000	0x4005EFFF	GPIO Port H (AHB)
0x4005F000	0x4005FFFF	GPIO Port G (AHB)
...	...	

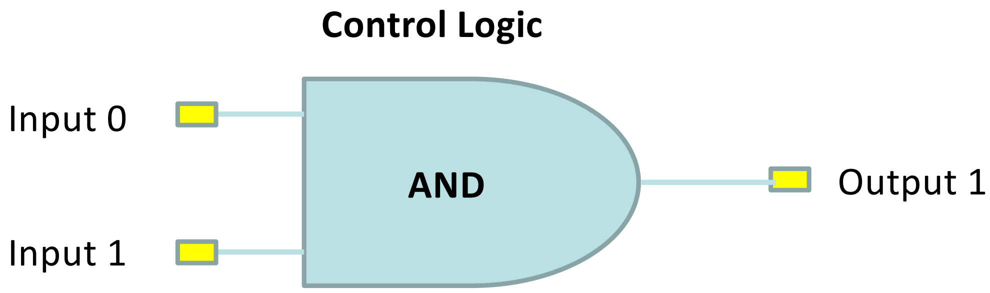
Modified GPIO-Output Update ISR



Similar Implementation for GPIO-Input Update ISR

Harvey Spoofing Inputs



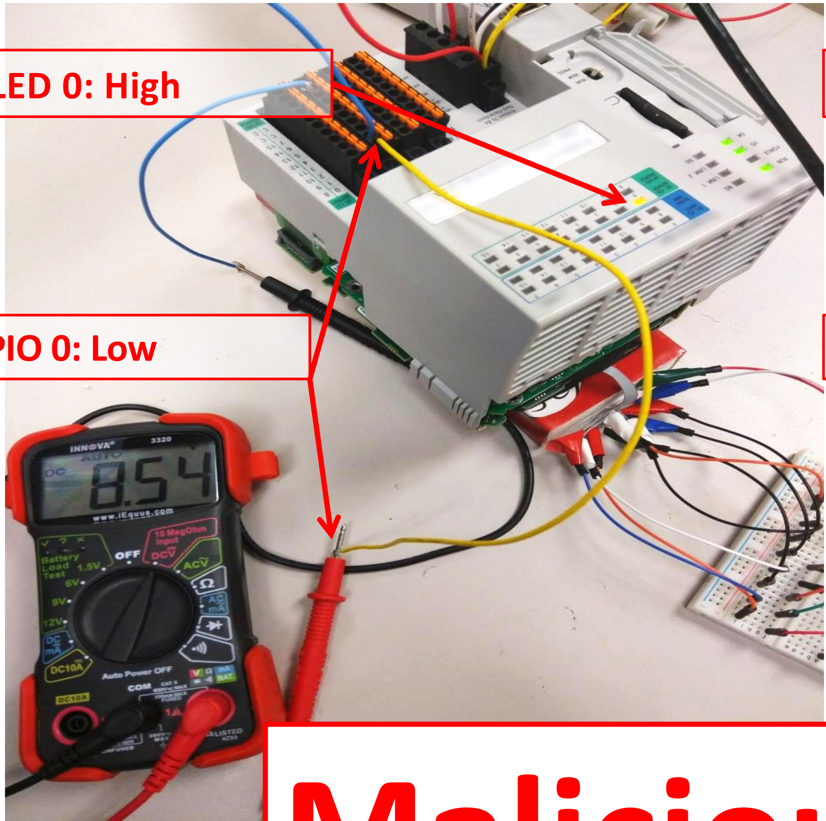


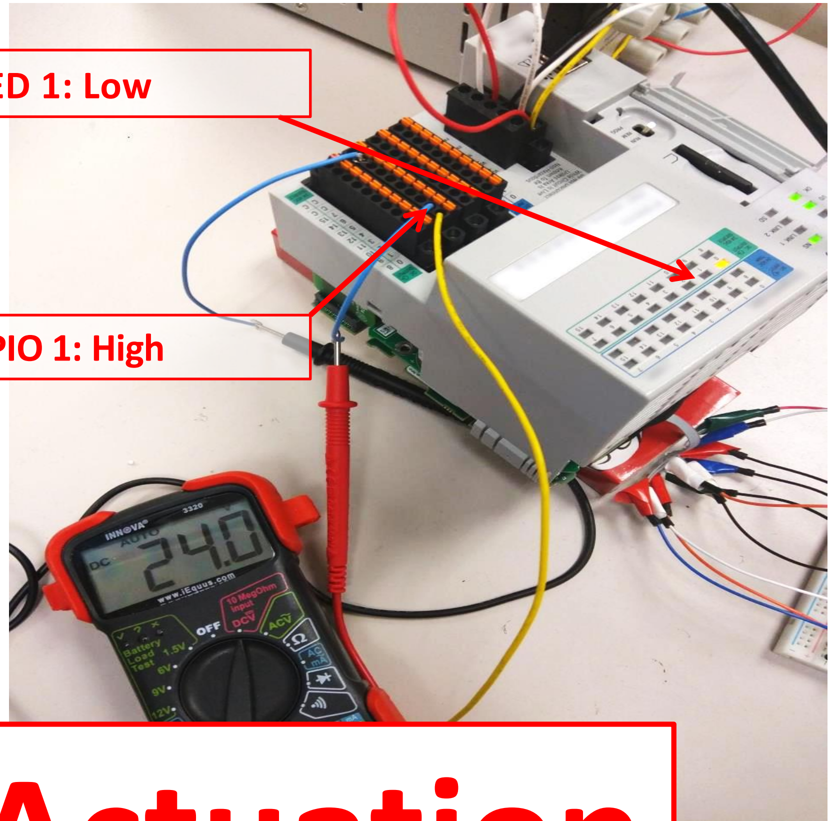
LEDs/HMI see:

Inputs 0 & 1=High

Manipulation of Sensor Data

Harvey Spoofing Outputs

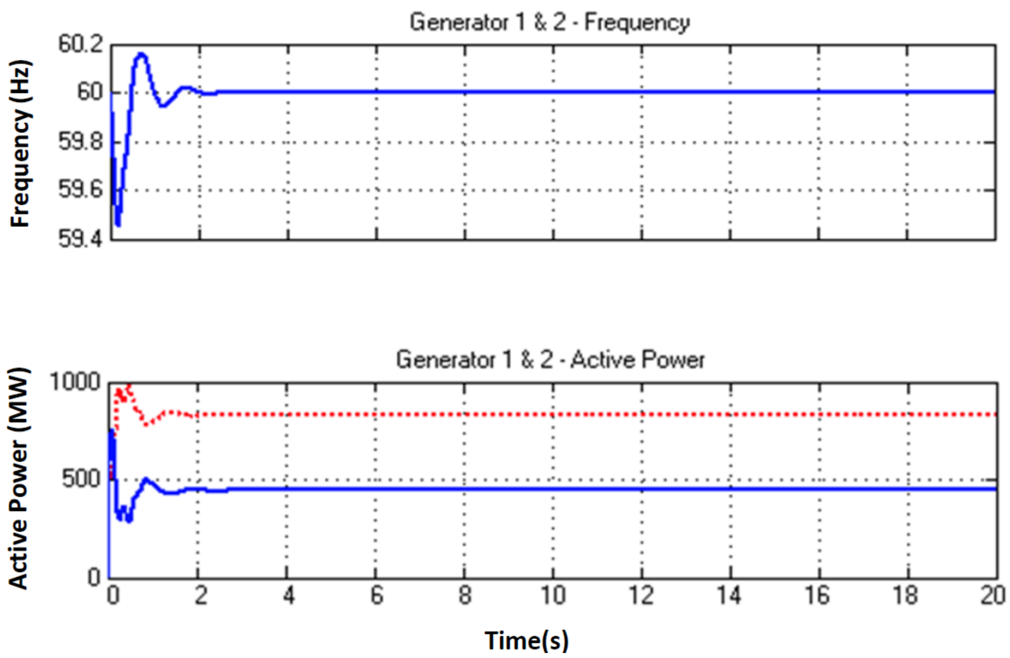




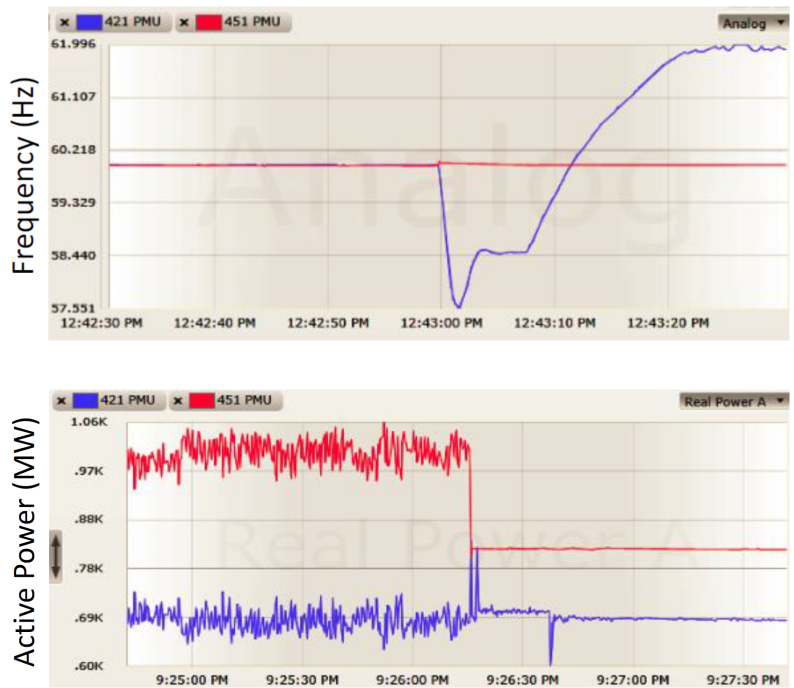
Malicious Actuation

Real-world Attack Demo: Attacking a Power System

Real-world Attack Demo: Attacking a Power System



HMI Measurements



Actual System Measurements

Let’s Get Defensive

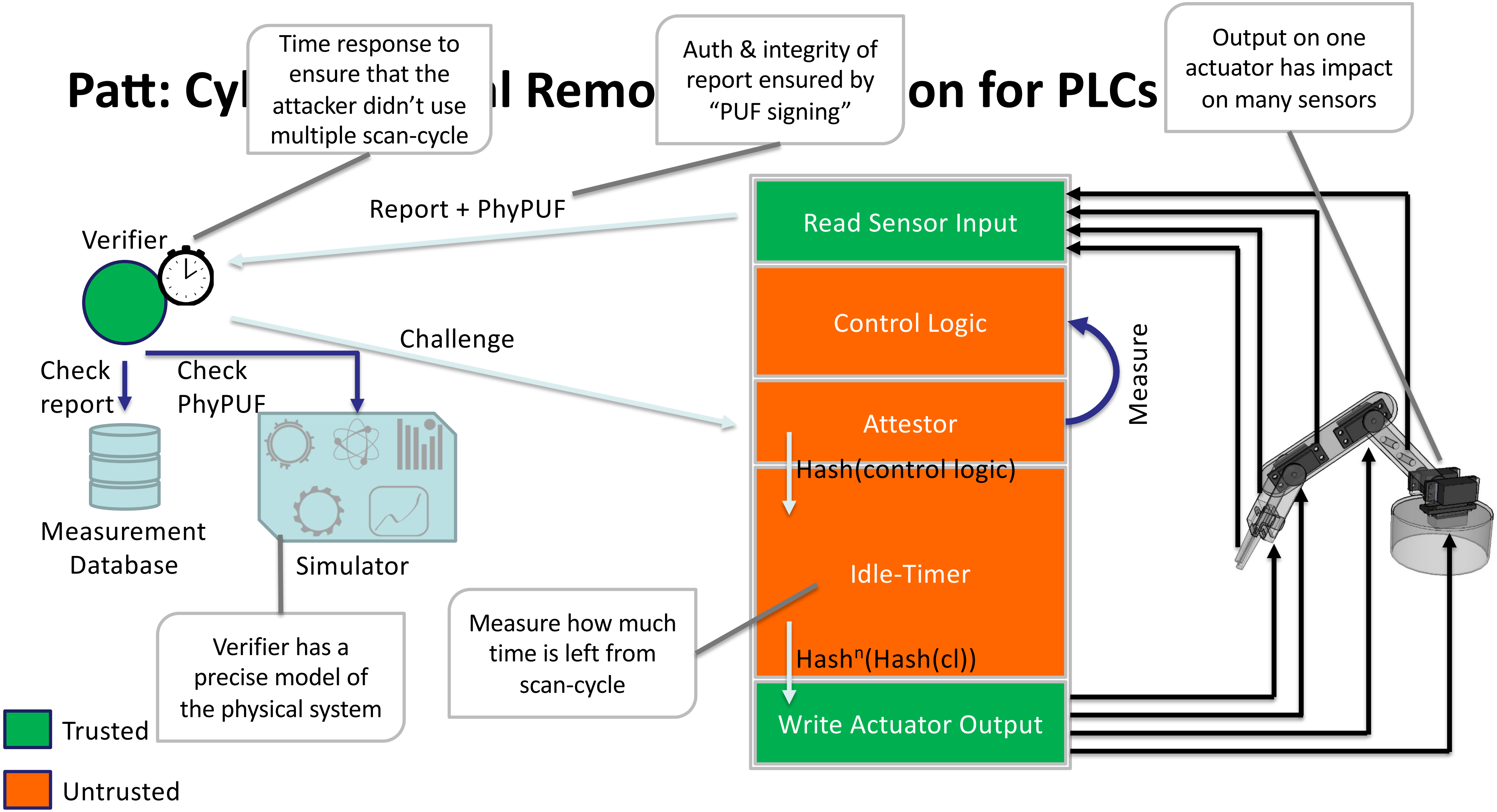
- **Problem:** Legacy devices (such as PLCs) don’t support a hardware trusted computing base (TCB) for remote attestation

→ **Patt: Physics-based Attestation of Control Systems**

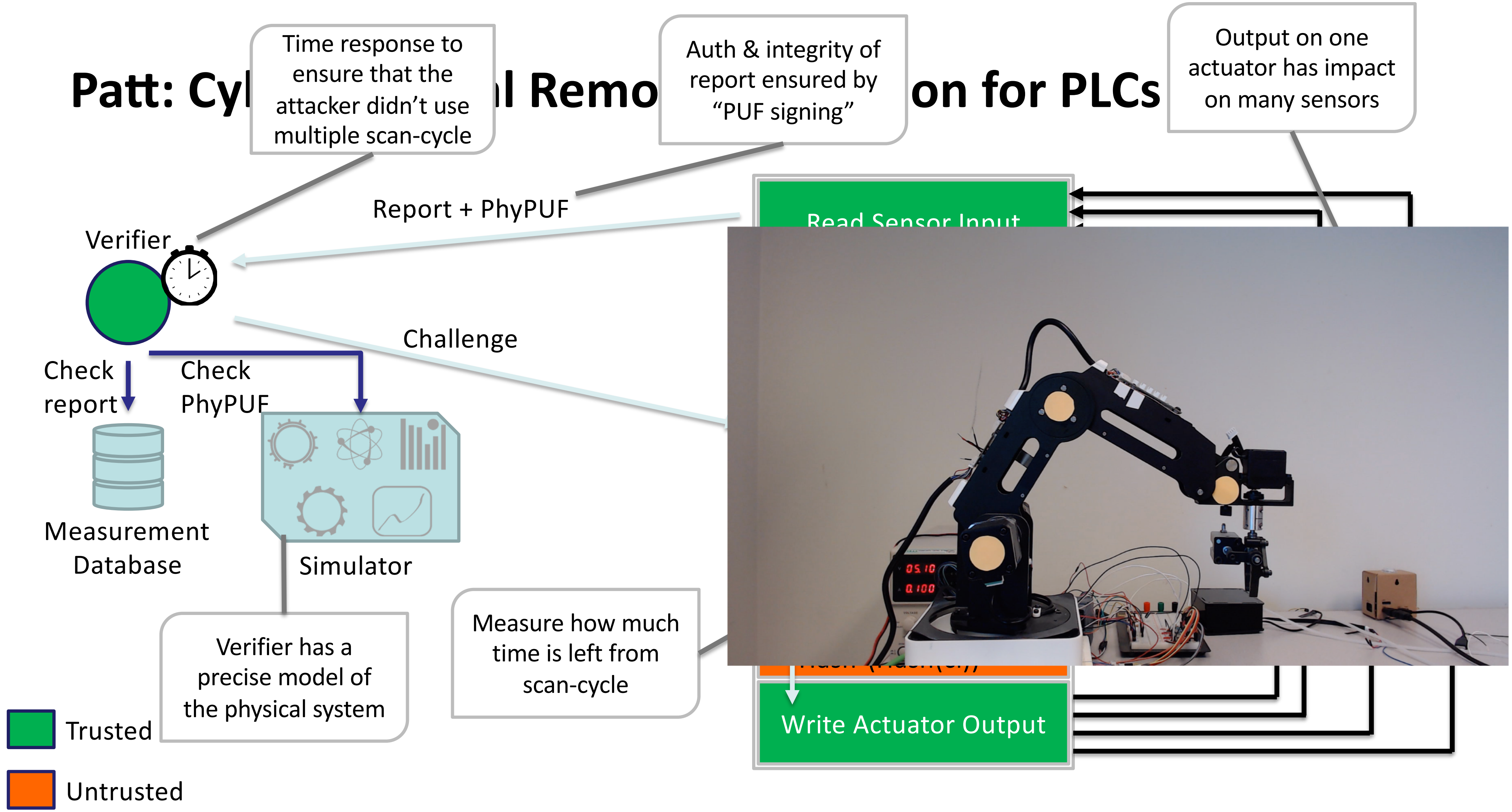
Patt: Physics-based Attestation of Control Systems

(RAID 2019)

TECHNISCHE UNIVERSITÄT DARMSTADT



Patt: Cyl
l Remo
on for PLCs
Time response to ensure that the attacker didn't use multiple scan-cycle
Auth & integrity of report ensured by "PUF signing"
Output on one actuator has impact on many sensors
Report + PhyPUF
Verifier
Challenge
Check report
Check PhyPUF
Measurement Database
Simulator
Verifier has a precise model of the physical system
Measure how much time is left from scan-cycle
Read Sensor Input
Control Logic
Attestor
Measure
Hash(control logic)
Idle-Timer
Hash^n(Hash(cl))
Write Actuator Output
Trusted
Untrusted


Patt: Cyl
l Remo
on for PLCs
Time response to ensure that the attacker didn't use multiple scan-cycle
Auth & integrity of report ensured by "PUF signing"
Output on one actuator has impact on many sensors
Report + PhyPUF
Verifier
Challenge
Check report
Check PhyPUF
Measurement Database
Simulator
Verifier has a precise model of the physical system
Measure how much time is left from scan-cycle
Read Sensor Input
Write Actuator Output
Trusted
Untrusted

Let's Get Defensive

- **Problem:** Legacy devices (such as PLCs) don't support a hardware trusted computing base (TCB) for remote attestation

  → **Patt: Physics-based Attestation of Control Systems**

- **Problem:** The complexity of these CPS involves several domain-specific physical processes that need to be verified

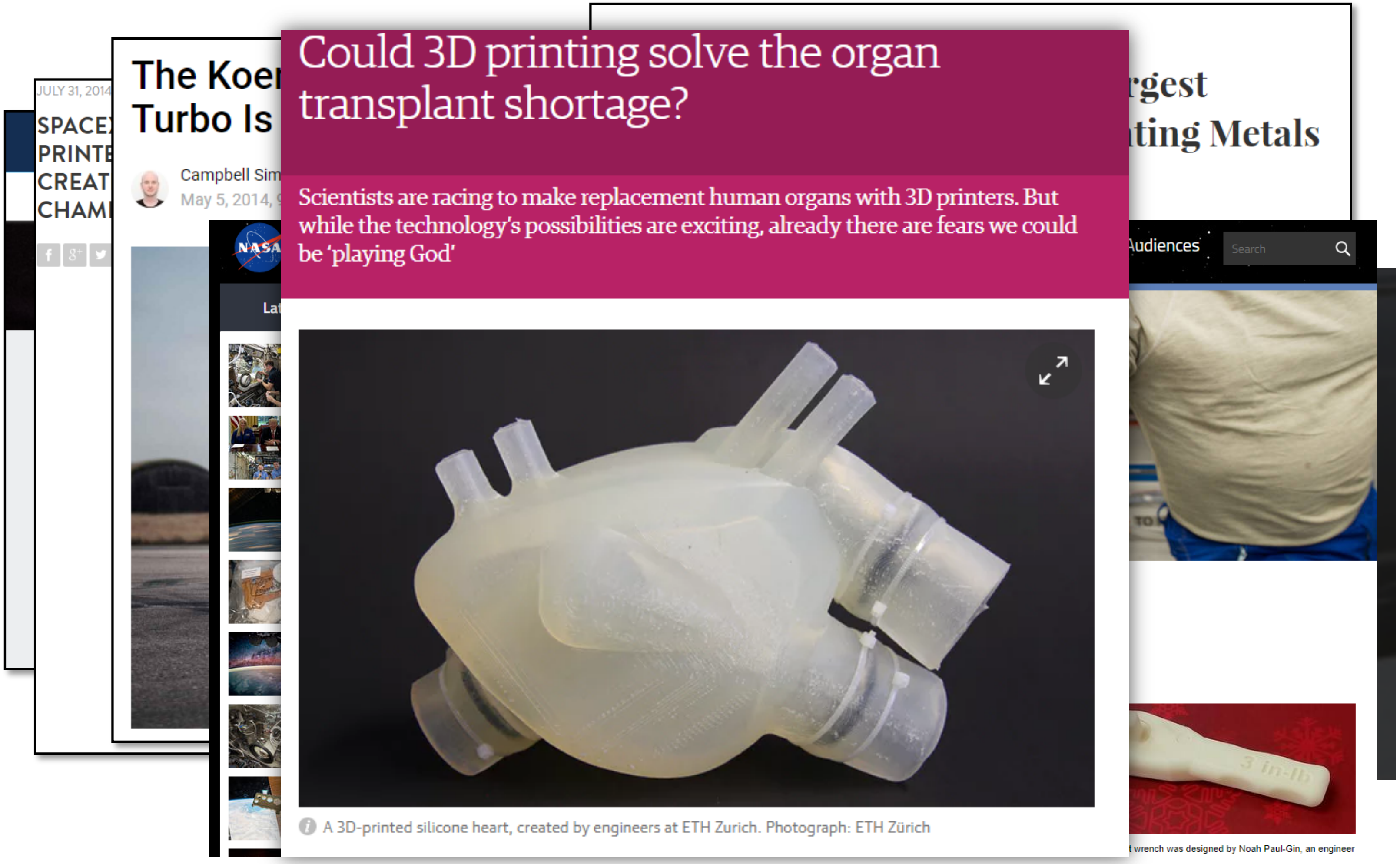
  → **Domain-specific Physics-based Verification of ICS Processes**

See No Evil, Hear No Evil, Feel No Evil… *Print* No Evil?

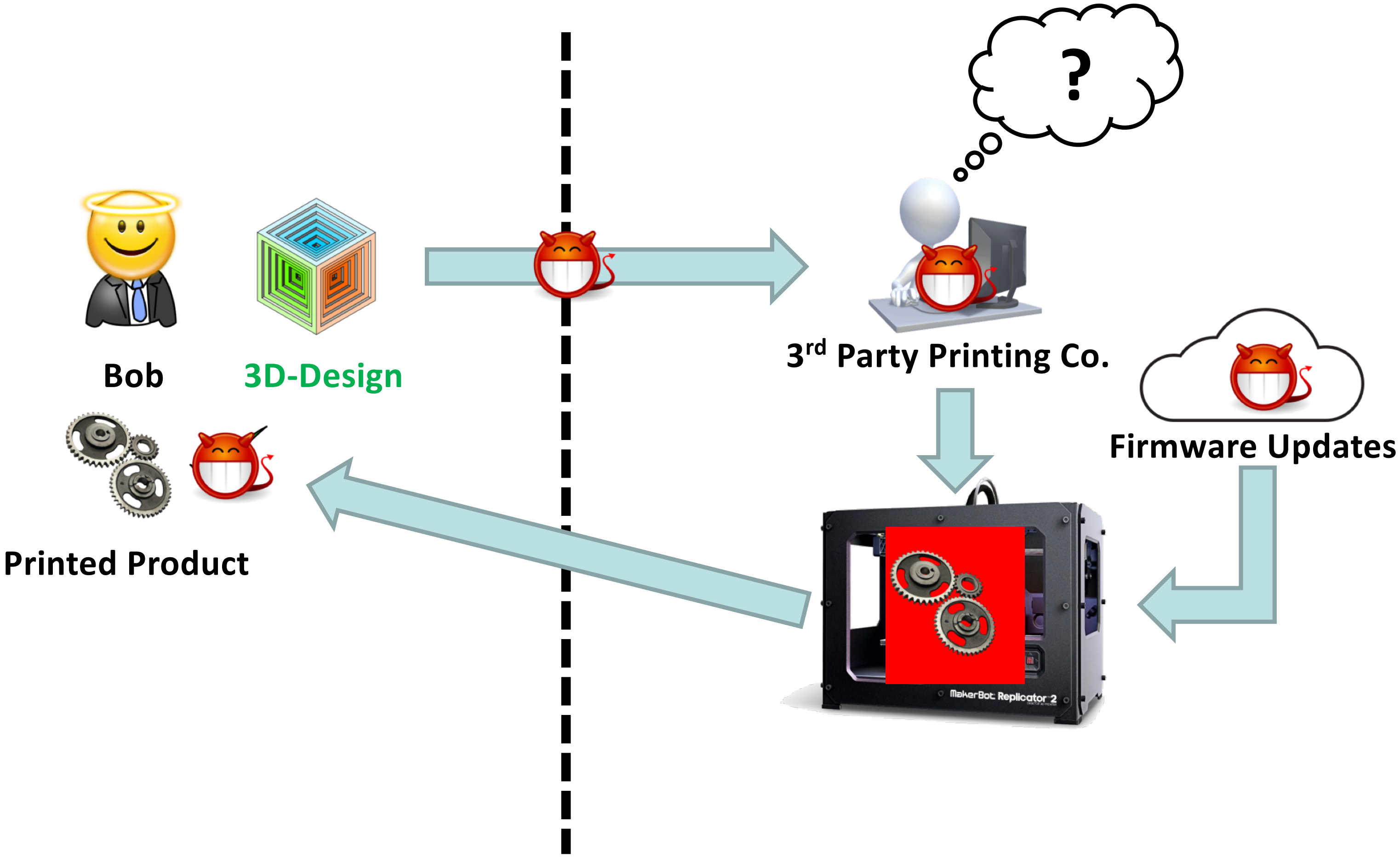
Malicious Fill Pattern Detection in Additive Manufacturing

USENIX Security 2017

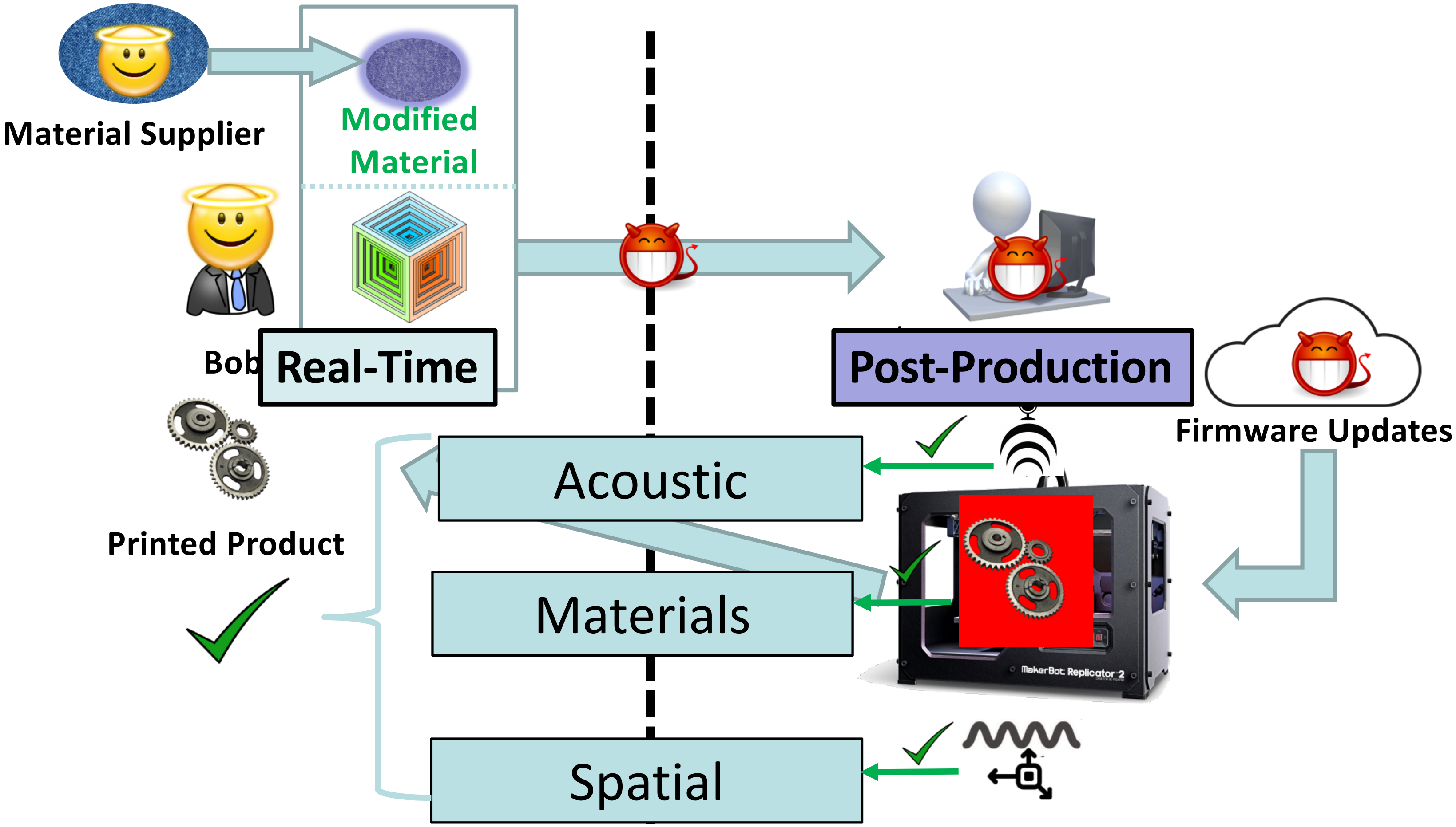
Industrial 3D Printing



Third Party 3D Printing



Malicious Fill Pattern Detection (USENIX Security 2017)



Let’s Get Defensive

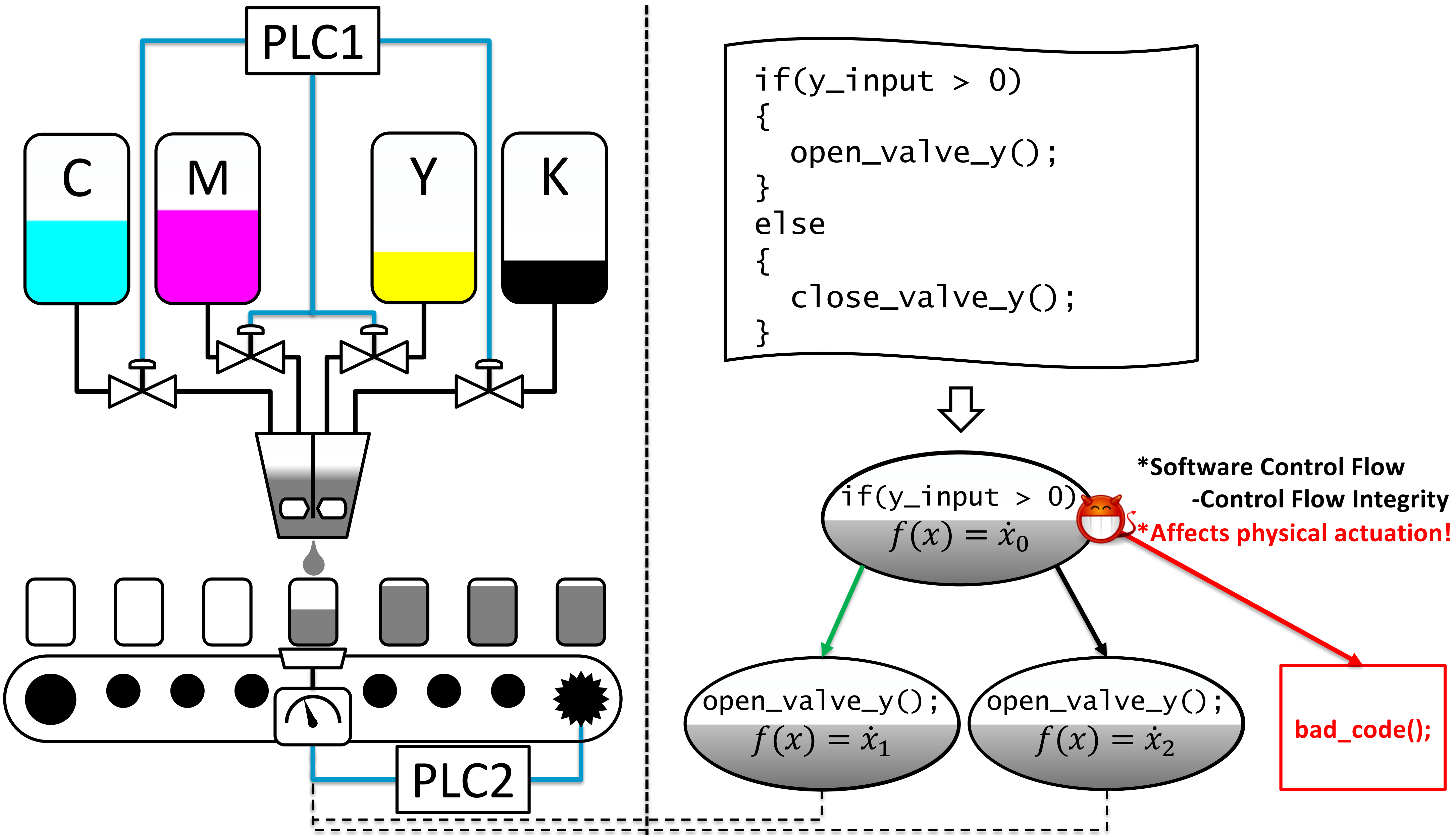
- **Problem:** Legacy devices (such as PLCs) don’t support a hardware trusted computing base (TCB) for remote attestation → **Patt: Physics-based Attestation of Control Systems**
- **Problem:** The complexity of these CPS involves several domain-specific physical processes that need to be verified → **Domain-specific Physics-based Verification of ICS**
- **Problem:** How can we verify these cyber-physical properties in distributed settings? → **Cyber-physical Control Behavior Integrity**

Control Behavior Integrity for Distributed Cyber-Physical Systems

ICCPS 2020

TECHNISCHE UNIVERSITÄT DARMSTADT

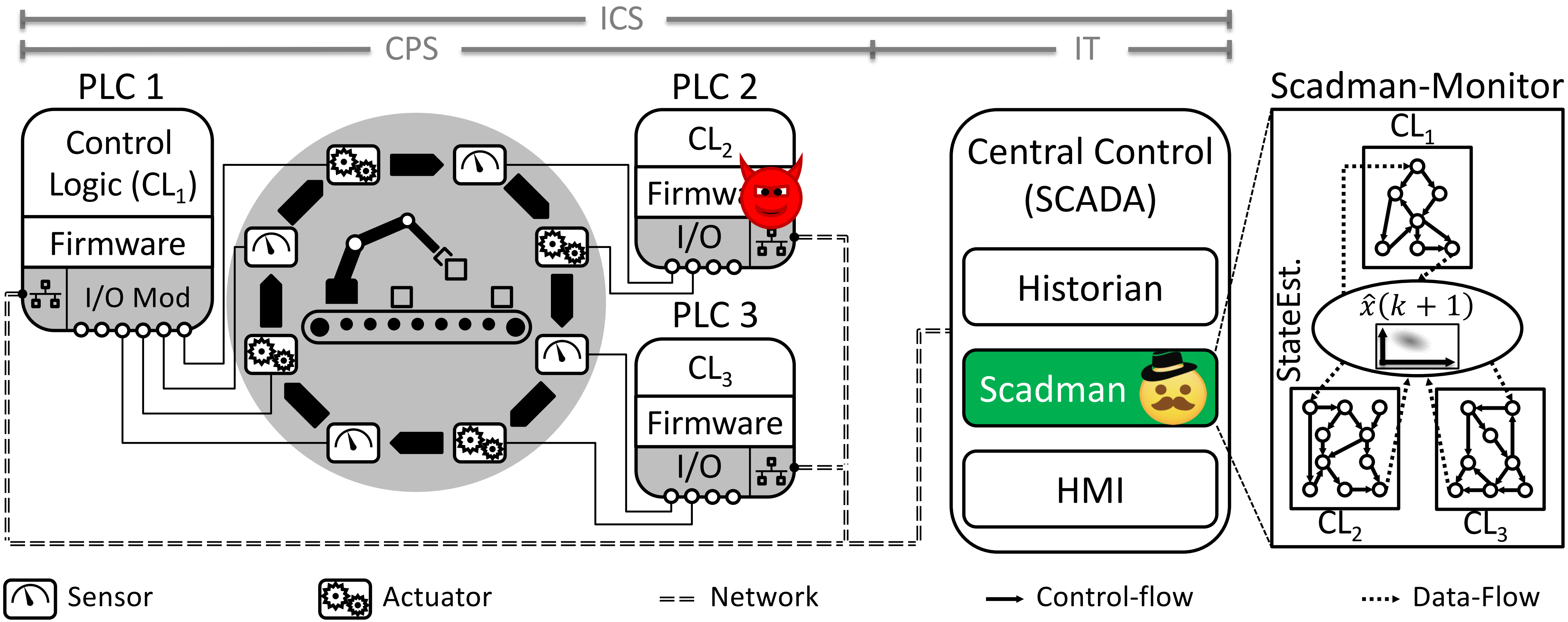
SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN

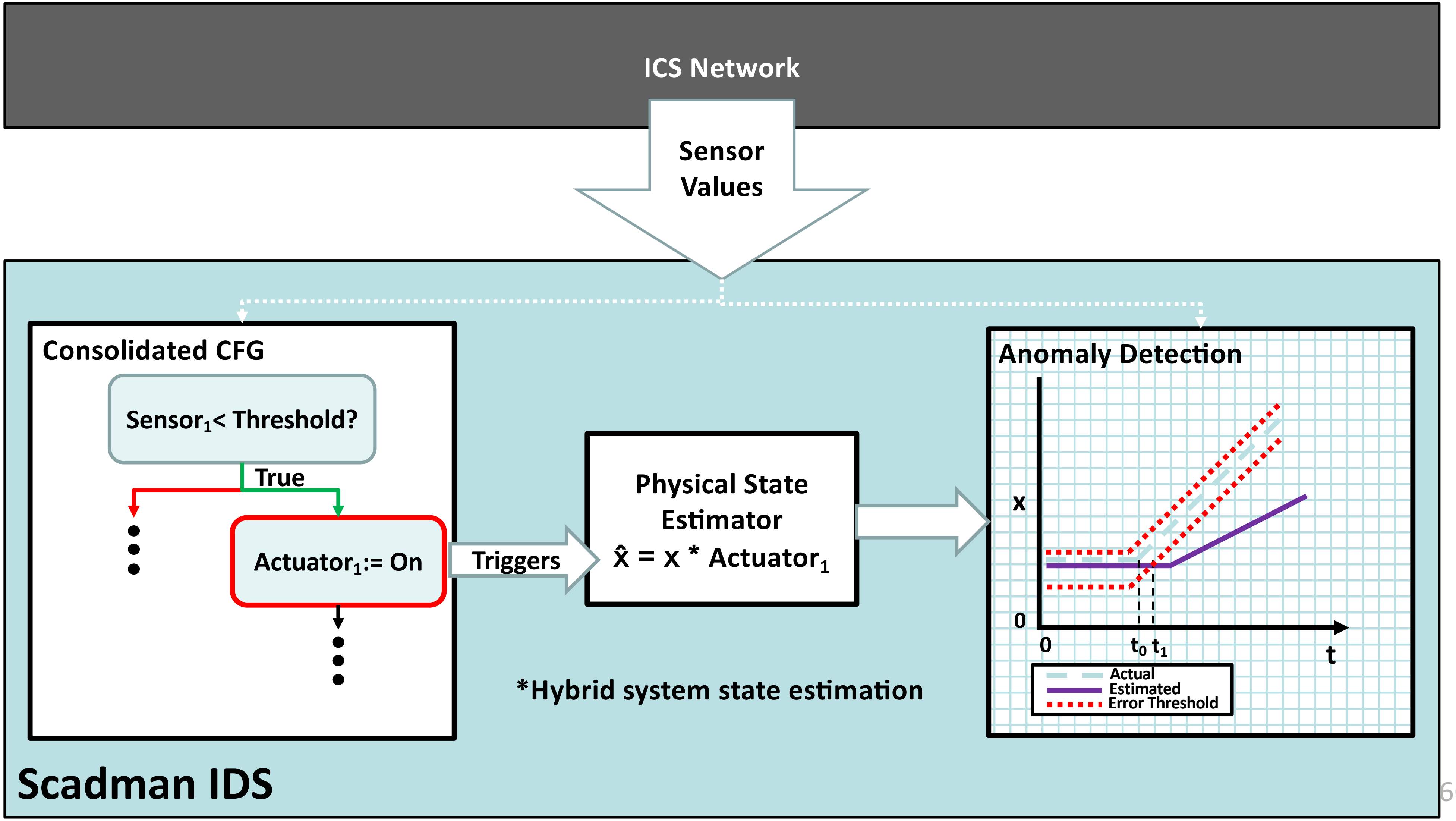

PLC1
C
M
Y
K
PLC2
if(y_input > 0)
{
open_valve_y();
}
else
{
close_valve_y();
}
*Software Control Flow
-Control Flow Integrity
*Affects physical actuation!
if(y_input > 0)
$f(x) = \dot{x}_0$
open_valve_y();
$f(x) = \dot{x}_1$
open_valve_y();
$f(x) = \dot{x}_2$
bad_code();

Scadman: Control Behavior Intrusion Detection

- An intrusion detection solution for distributed ICS
- Hybrid model
  - Uses physical state estimation for IDS
  - Updates physical state estimation based on software control flow

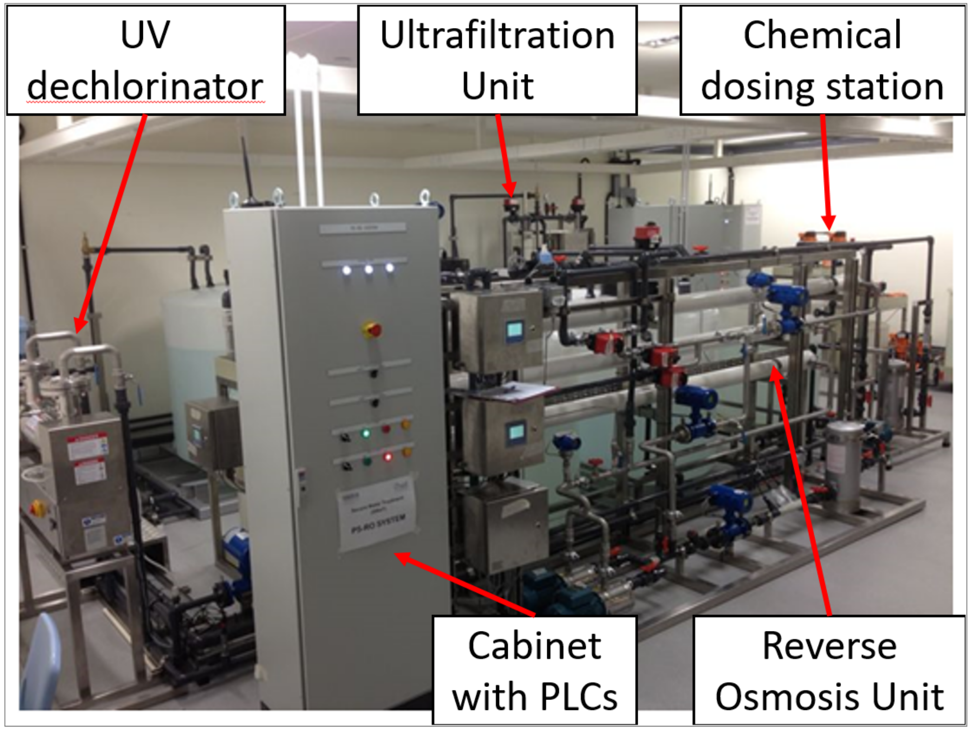
Scadman Overview

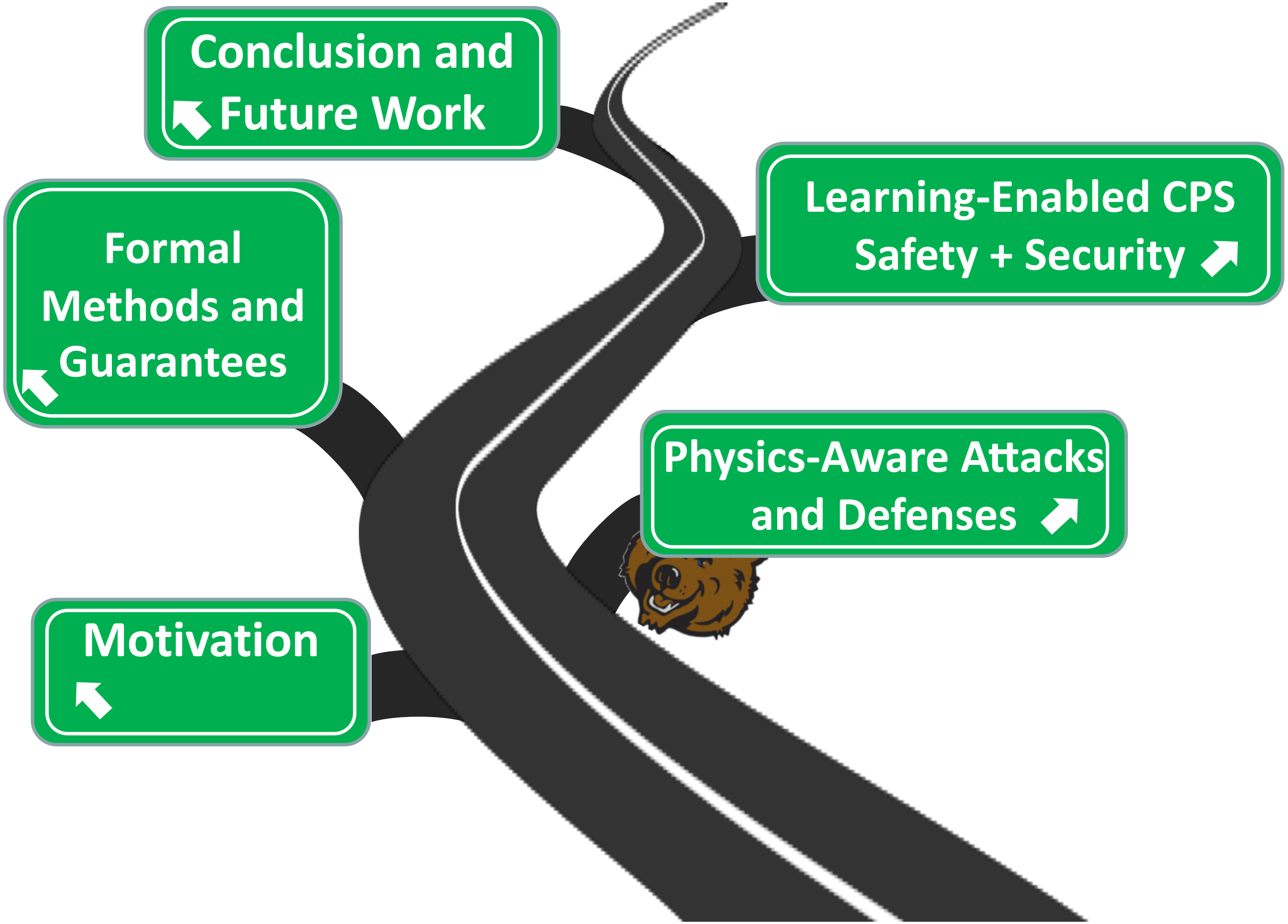



ICS Network
Sensor Values
Consolidated CFG
Sensor1< Threshold?
True
Actuator1:= On
Triggers
Physical State Estimator
x̂ = x * Actuator1
Anomaly Detection
x
0
0
t0 t1
t
Actual
Estimated
Error Threshold
*Hybrid system state estimation
Scadman IDS

Evaluation: Water Treatment Testbed

- Evaluated against known set of ICS attacks from
  - 7 days worth of data
  - Multi-point attacks included
- Detected all attacks
  - Also detected faulty sensor data
  - Zero false positives
- No overhead on ICS operation
  - Scadman utilizes historian data



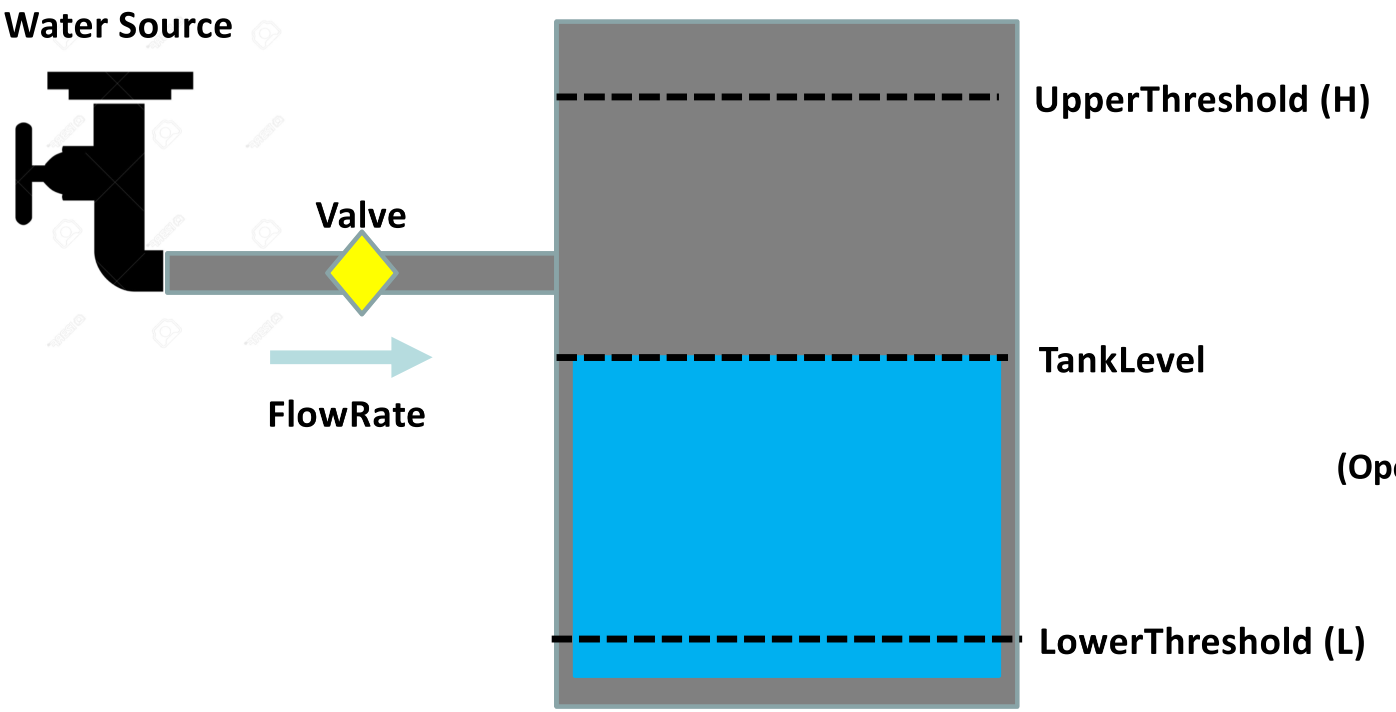

Conclusion and Future Work
Formal Methods and Guarantees
Motivation
Learning-Enabled CPS Safety + Security
Physics-Aware Attacks and Defenses

HyPLC: Hybrid PLC Program Translation for Verification

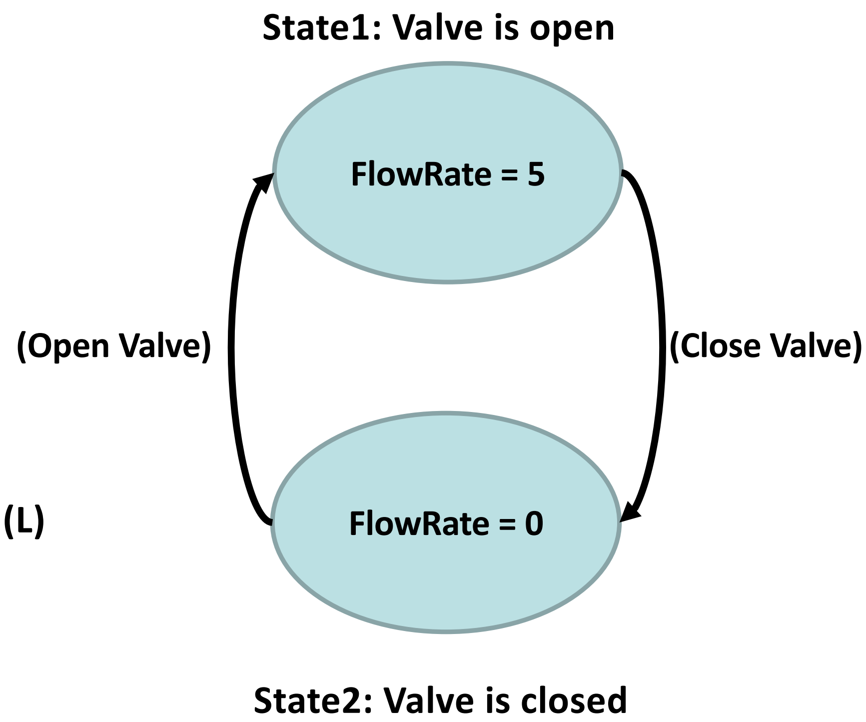
ICCPS 2019 (Best Paper Finalist)

Hybrid Systems Modeling of CPS

Physical Plant:
TankLevel = TankLevel + FlowRate * Δt



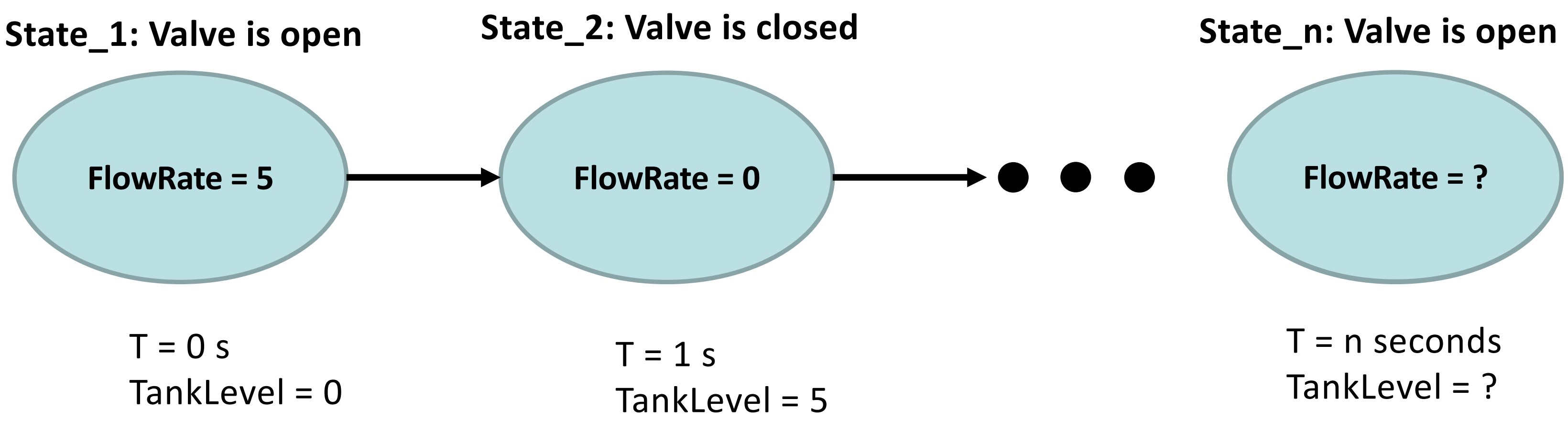
Safety: L<TankLevel<H



Moving Away from Linear Temporal Logic (LTL)

Consider previous example

- **LTL expression:** **Globally(L<TankLevel<H)**



- Infinite amount of states for real-time systems
  - Can only provide runtime checks
- Real systems have non-deterministic states

How can we guarantee safety forever for complex CPS?

Safety: L<TankLevel<H

Verification Using Differential Dynamic Logic

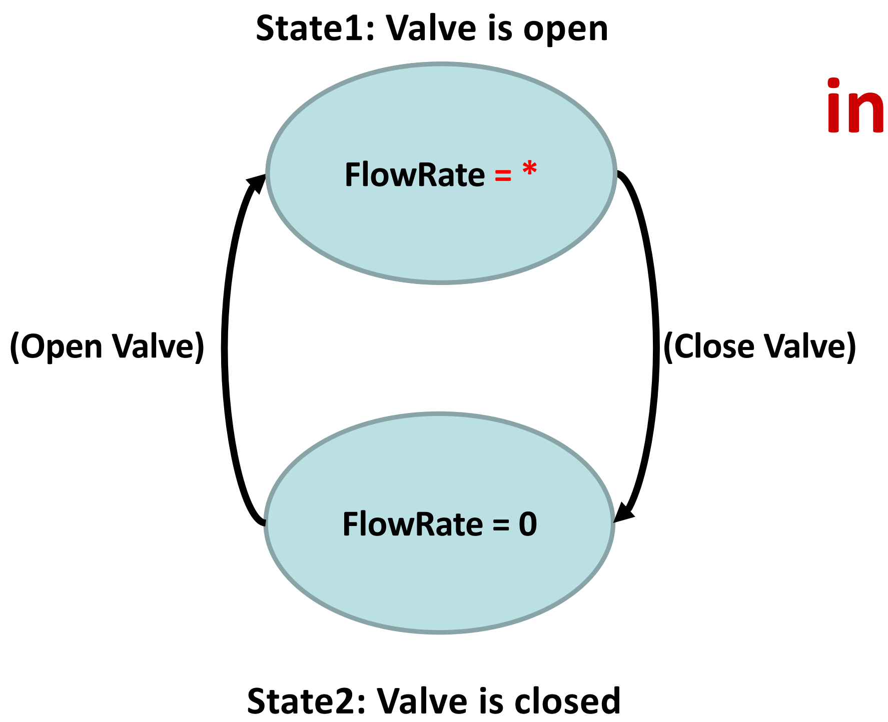
[α]ψ	For all states in **α, ψ** holds true (safety)
<α>ψ	There exists a state **α** where **ψ** is true (liveness)

Definition (Hybrid program α)

$x' = f(x)$	(continuous evolution)
$x := \theta$	(discrete jump)
$?\chi$	(conditional execution)
$\alpha; \beta$	(seq. composition)
$\alpha \cup \beta$	(nondet. choice)
α^*	(nondet. repetition)

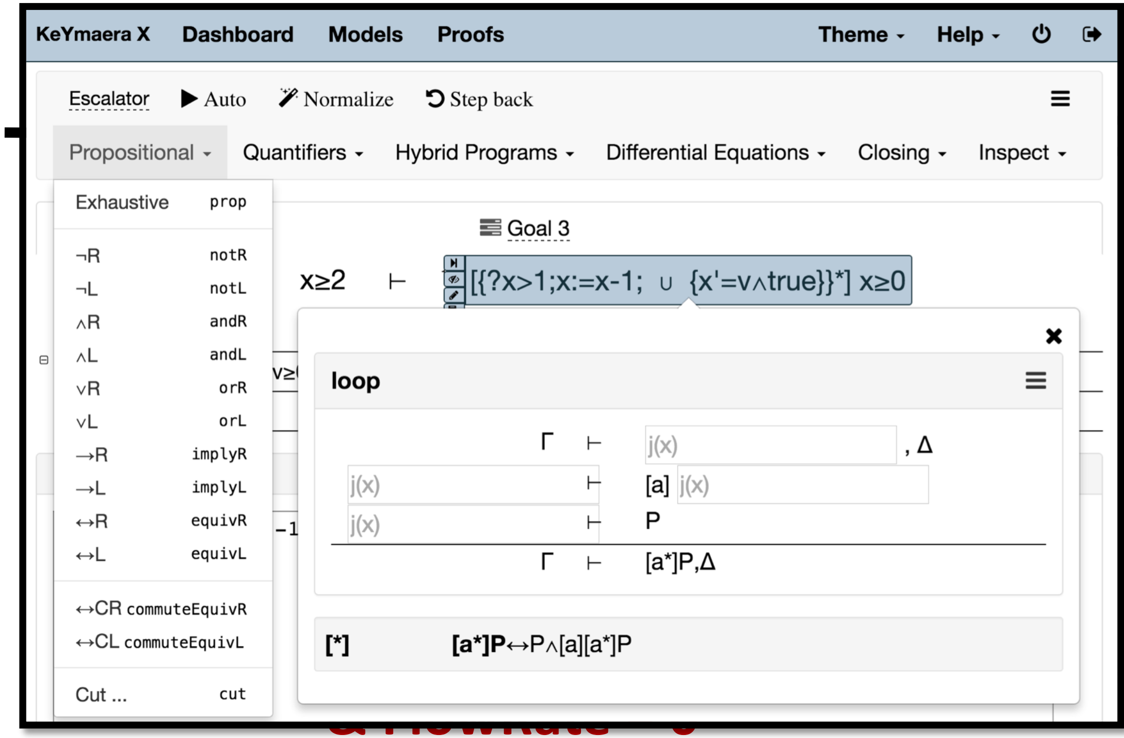
Verification Using Differential Dynamic Logic

$[\alpha]\psi$ For all states in α, ψ holds true (safety)



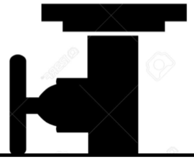
init

H)



HyPLC: Hybrid Programmable Logic Controller Program Translation for Verification

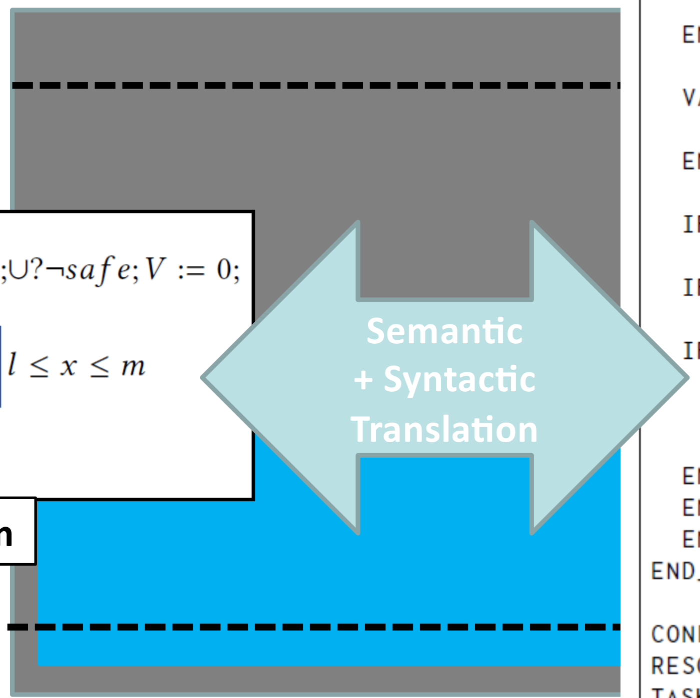
Raw Water Source



$$l \le x \le m \land \epsilon > 0 \rightarrow \Big[\Big(f := *; ?safe; V := 1; \cup ?\neg safe; V := 0;$$
$$t := 0; \big(x' = f * V, t' = 1 \,\&\, x \ge 0 \land t \le \epsilon\big)\Big)^*\Big] l \le x \le m$$

Where $safe \equiv \left(\frac{l-x}{\epsilon} \le f \le \frac{m-x}{\epsilon}\right)$.

Verified Hybrid Program

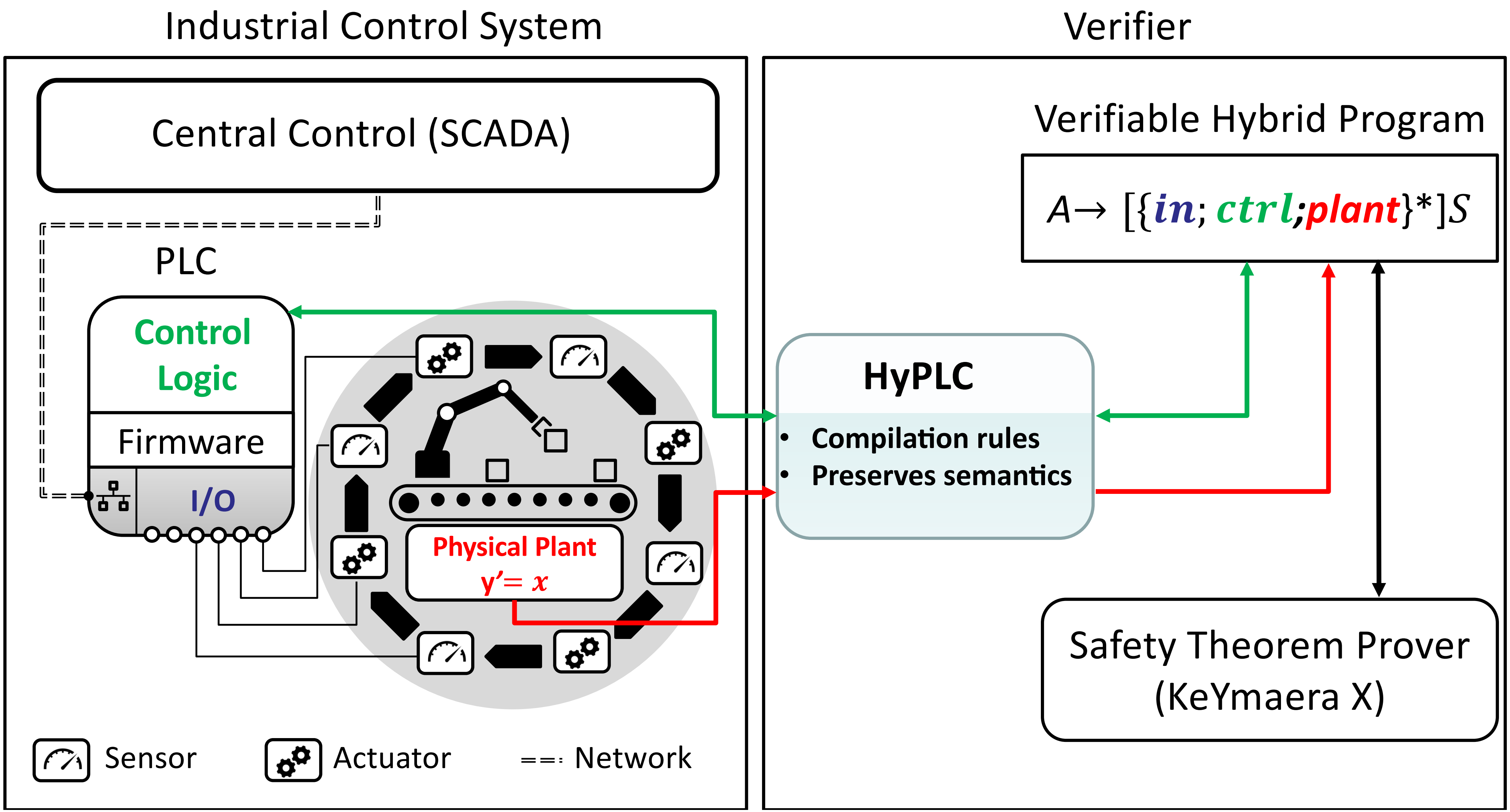


PLC

Safety: L<TankLevel

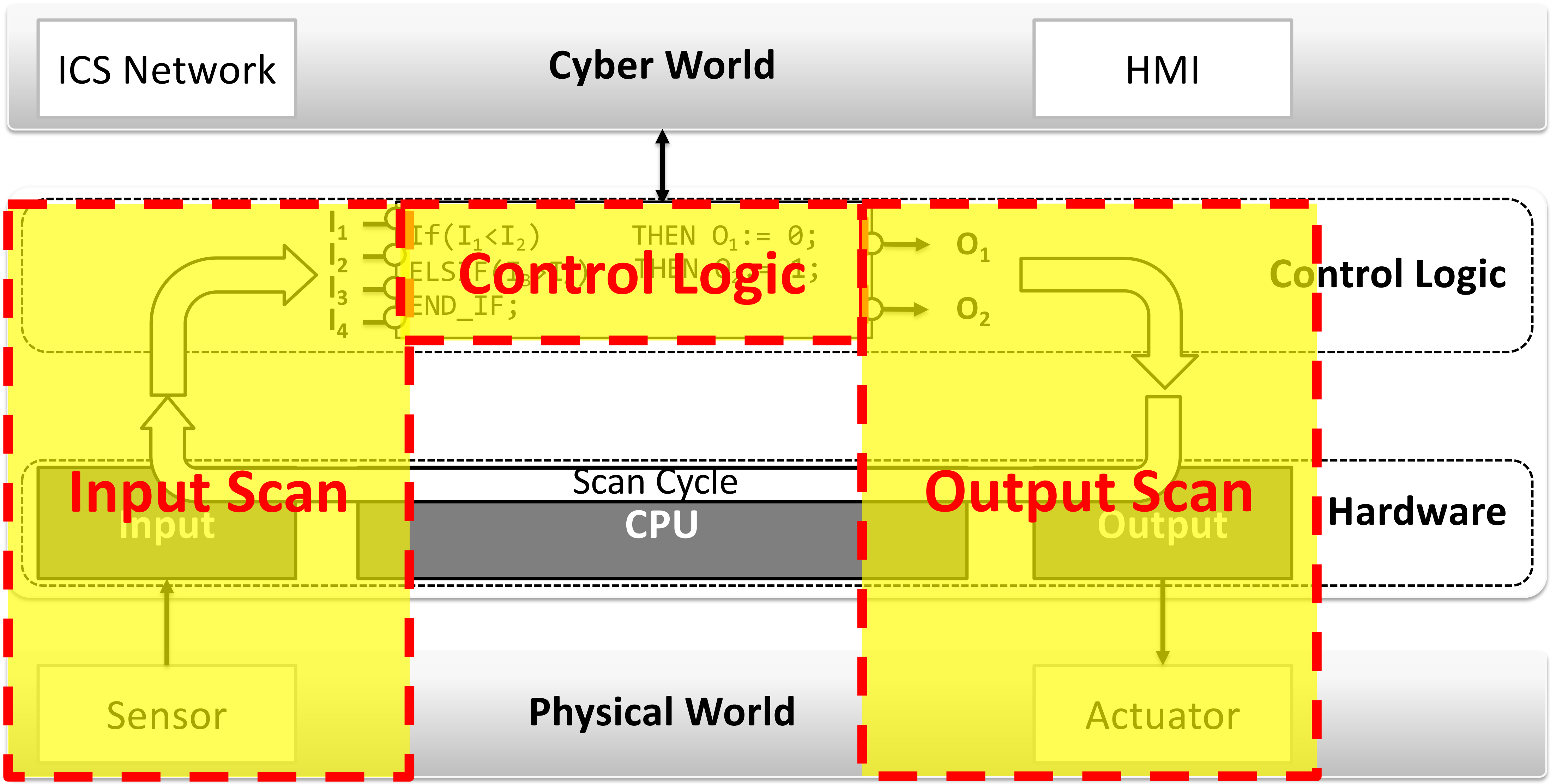
```
PROGRAM prog0
  VAR_INPUT
    f : REAL;
    x : REAL;
  END_VAR

  VAR_OUTPUT
    V : BOOL;
  END_VAR

  IF((f<((L-x)/(Tsample+Tplc)))) THEN
    V:=0; ELSE
  IF((f>((H-x)/(Tsample+Tplc)))) THEN
    V:=0; ELSE
  IF((f>=((L-x)/(Tsample+Tplc)))) THEN
    IF((f<((H-x)/(Tsample+Tplc)))) THEN
      V:=1;
    END_IF;
  END_IF;
  END_IF;
  END_IF;
END_PROGRAM

CONFIGURATION Config0
RESOURCE Res0 ON PLC
TASK Main(INTERVAL:=T#Tsamplems,PRIORITY:=0);
PROGRAM Inst0 WITH Main : prog0;
END_RESOURCE
END_CONFIGURATION
```

PLC Structured Text



Industrial Control System
Verifier
Central Control (SCADA)
PLC
Control Logic
Firmware
I/O
Physical Plant
y′= x
HyPLC
• Compilation rules
• Preserves semantics
Verifiable Hybrid Program
A→ [{in; ctrl;plant}*]S
Safety Theorem Prover
(KeYmaera X)
Sensor
Actuator
Network

Revisiting the PLC Scan Cycle

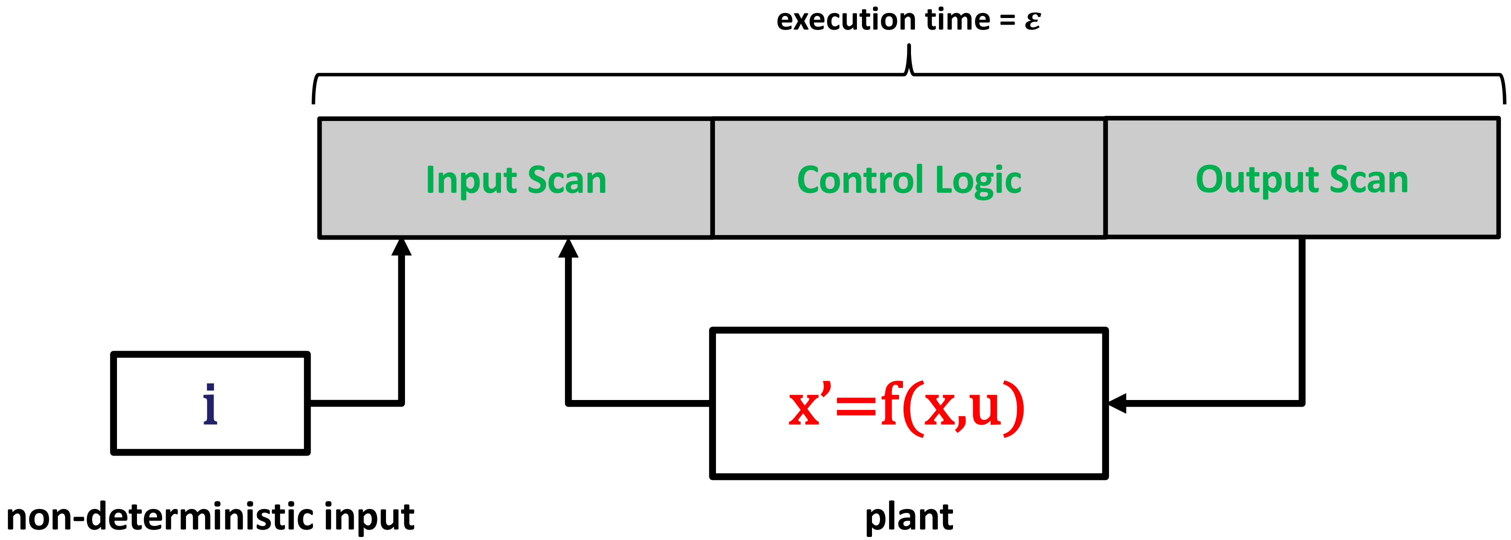


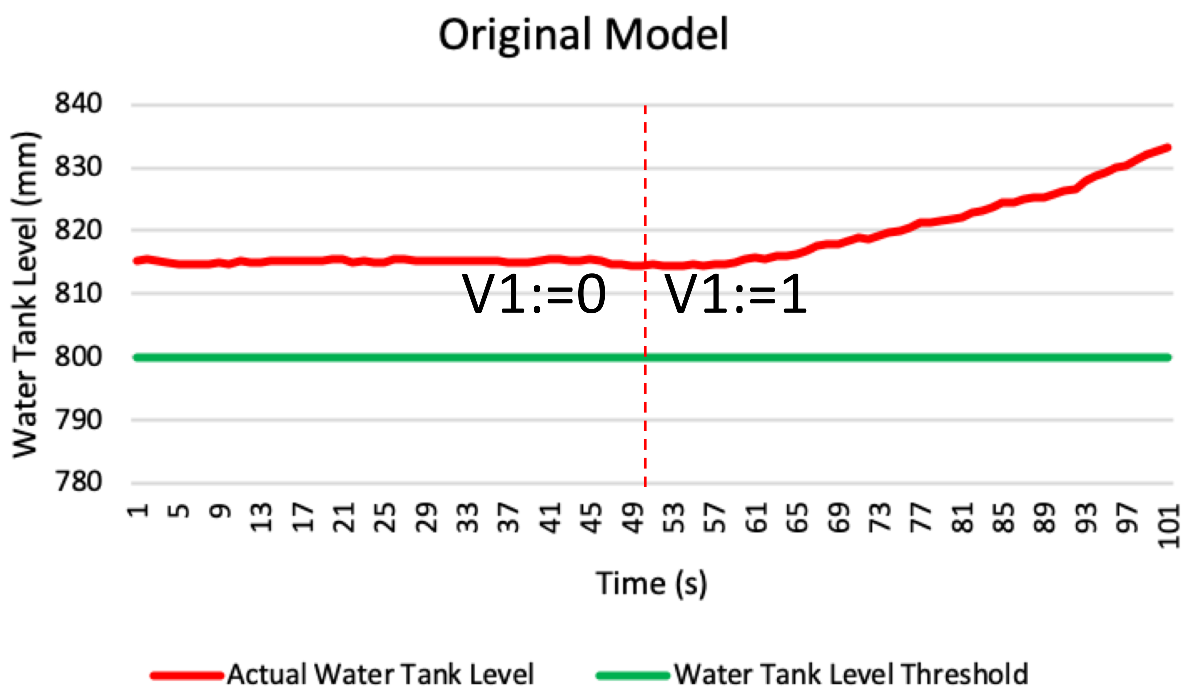
Revisiting the PLC Scan Cycle

Hybrid Program Scan Cycle:
(Normal Form)

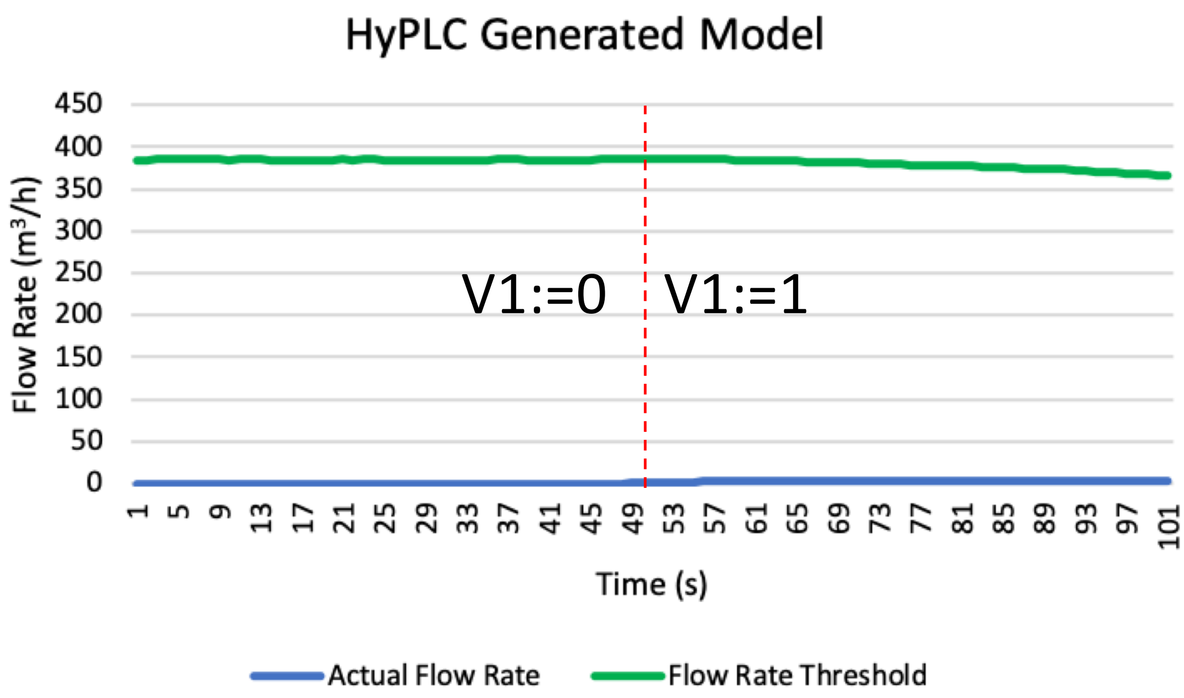
$$A \longrightarrow [(i := *;\ u \in ctrl(x,i);\ t := 0;\ \{x' = f(x,u);\ t' = 1\ \&\ t \le \varepsilon\})^*]\ S$$

PLC Scan Cycle:

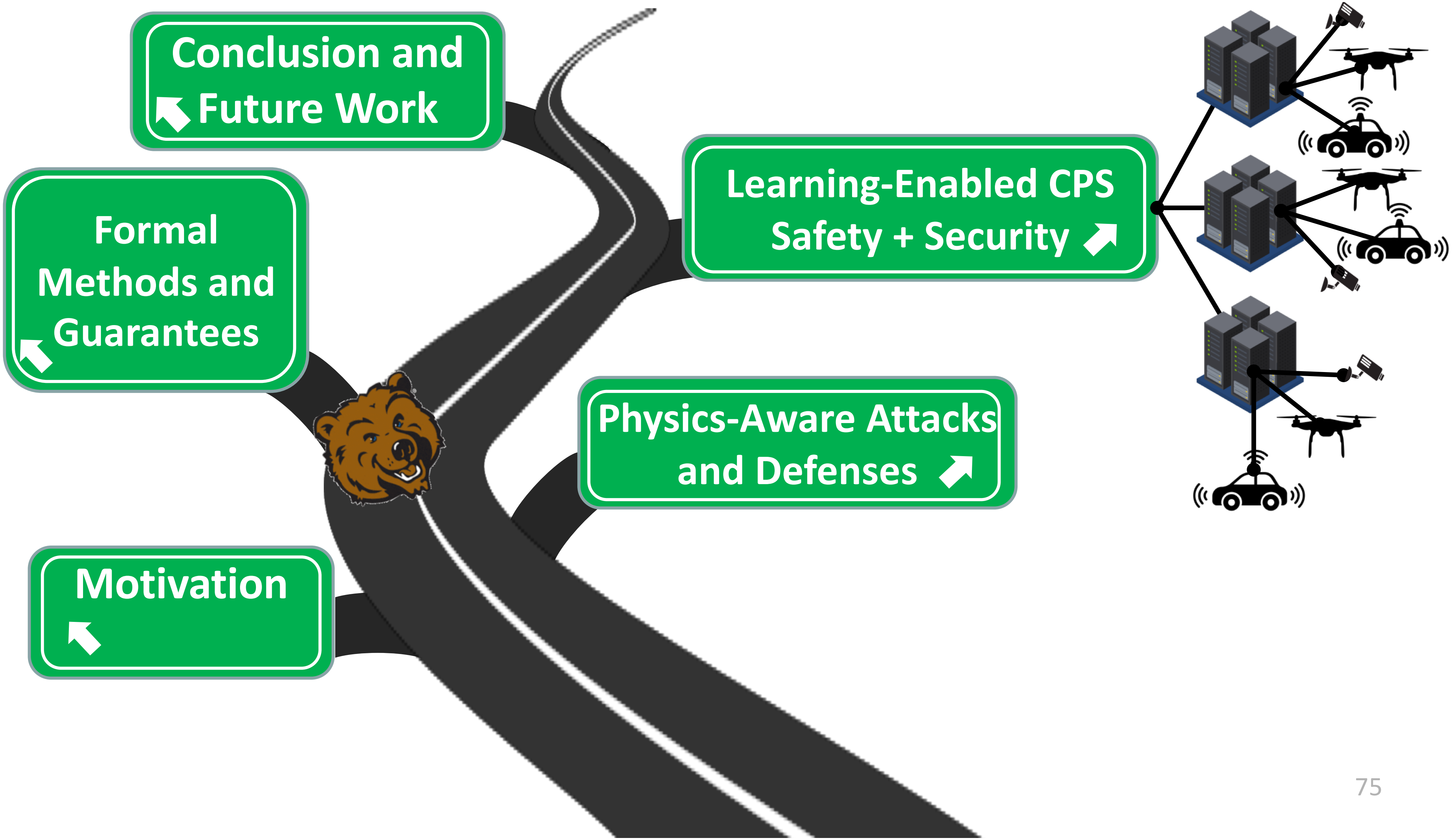


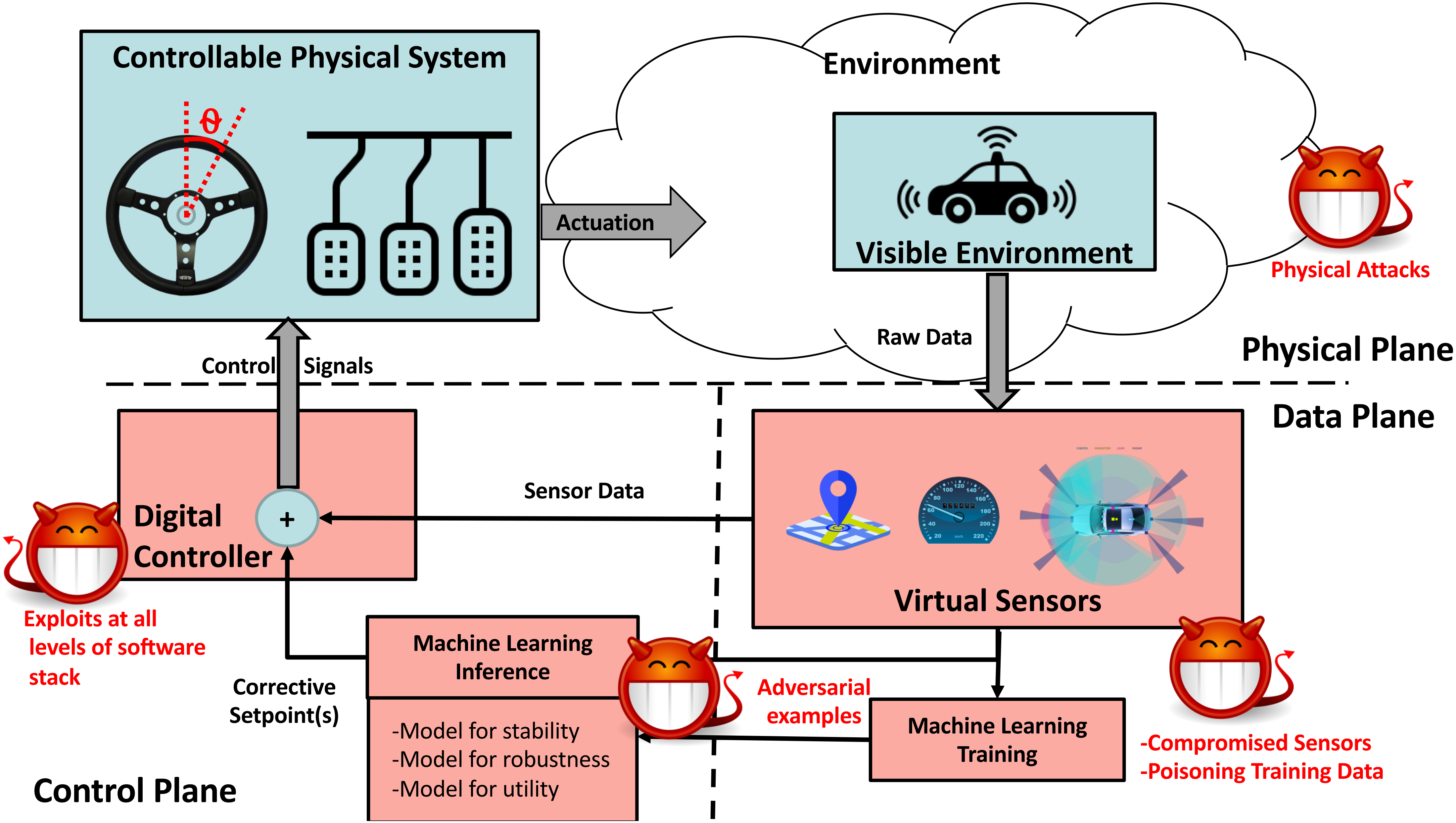


*** Original model would have reported a violation for this entire sequence based on the water tank level**



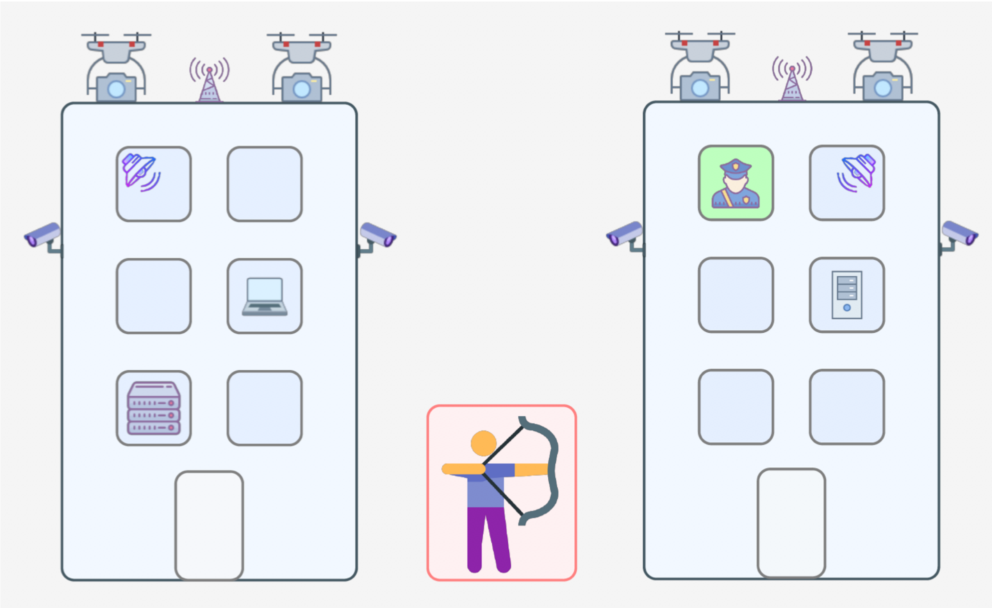
*** HyPLC model reports this same sequence as safe operation based on flow rate**


Conclusion and Future Work
Formal Methods and Guarantees
Motivation
Learning-Enabled CPS Safety + Security
Physics-Aware Attacks and Defenses


Controllable Physical System
θ
Actuation
Environment
Visible Environment
Physical Attacks
Raw Data
Control
Signals
Physical Plane
Data Plane
Digital
Controller
+
Sensor Data
Virtual Sensors
Exploits at all
levels of software
stack
Machine Learning
Inference
Corrective
Setpoint(s)
Adversarial
examples
Machine Learning
Training
-Model for stability
-Model for robustness
-Model for utility
-Compromised Sensors
-Poisoning Training Data
Control Plane

Formal Guarantees for Macroprogramming Heterogeneous IoT Networks

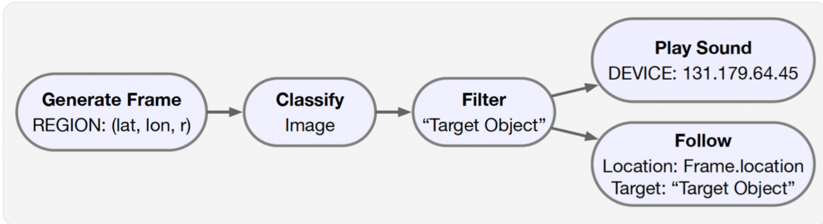
Smart City Environment

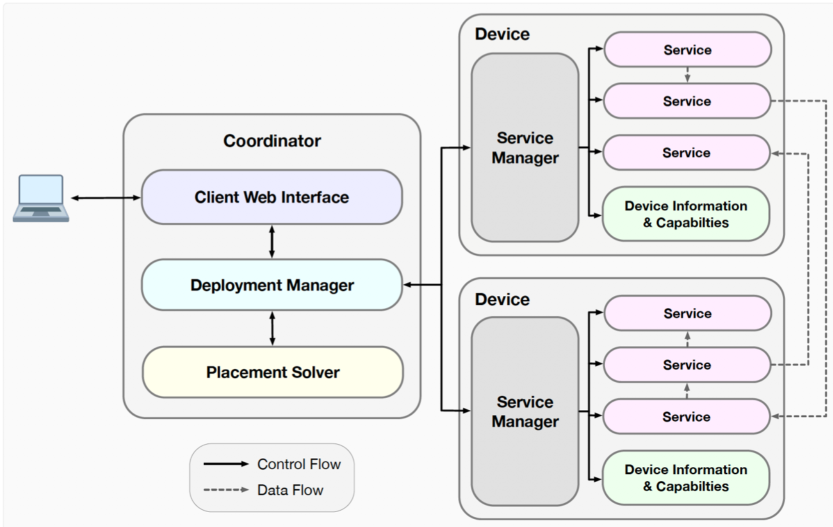


Officer to IoT Network: "Identify shooter and follow"

*How do we expose device services safely?

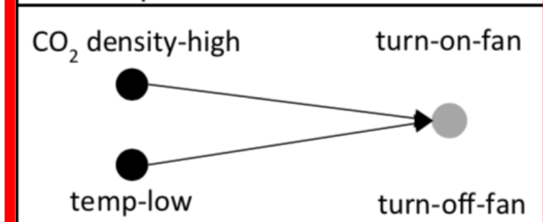
DDFlow: Visualized Declarative Programming for Heterogenous IoT Networks (IoTDI 2019)

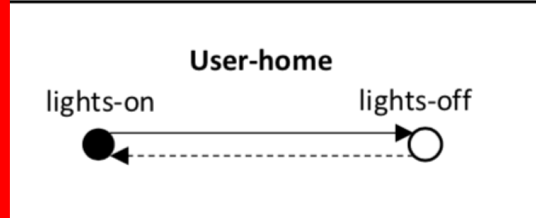


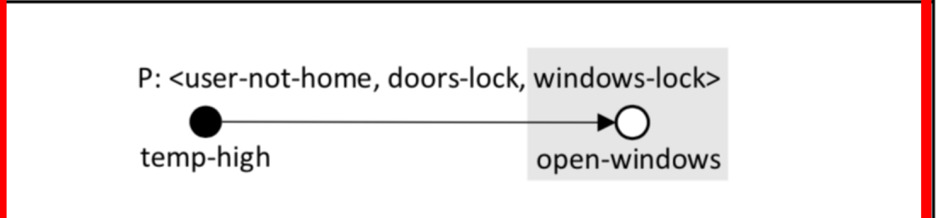


Formal Guarantees for Macroprogramming Heterogeneous IoT Networks

Types of Policies		
P1: Mutually exclusive states must not exist in the environment.		**P2: User-defined rules.**
Racing Events	Cyclic Events	**E.g.: Doors and windows must be locked is user is not home**
E1: CO_2 density-high -> turn-on-fan E2: temp-low -> turn-off-fan	E1: user-home, lights-on -> lights-off E2: user-home, lights-off -> lights-on	E1: user-away -> user-away-mode-on E2: user-away-mode-on, temp-high -> windows-on

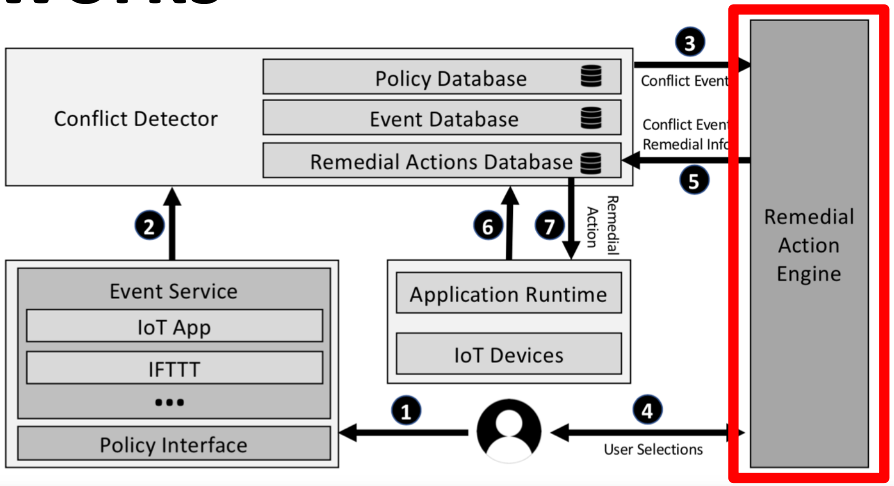


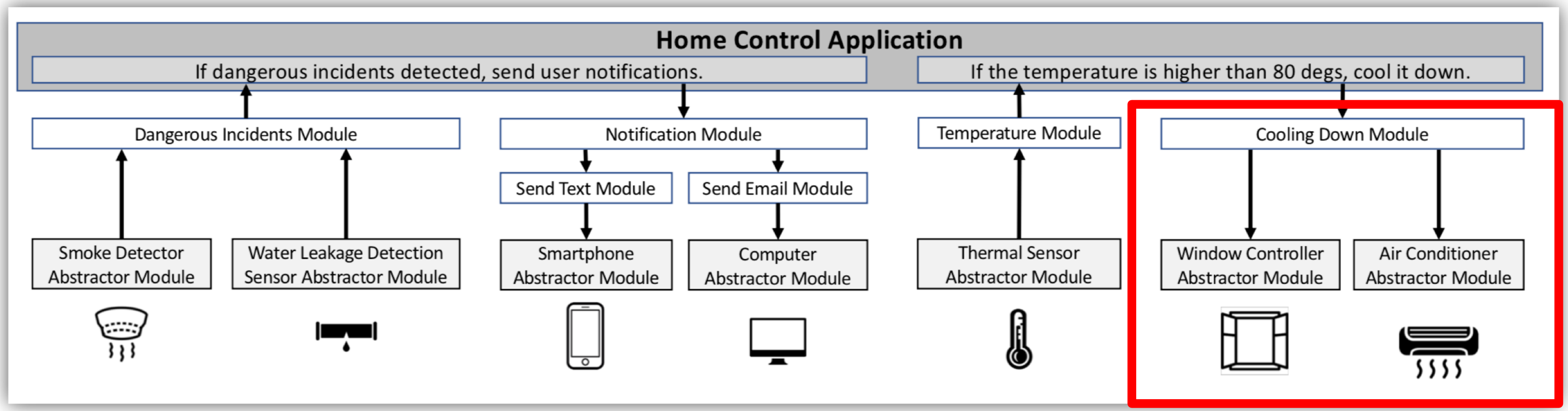




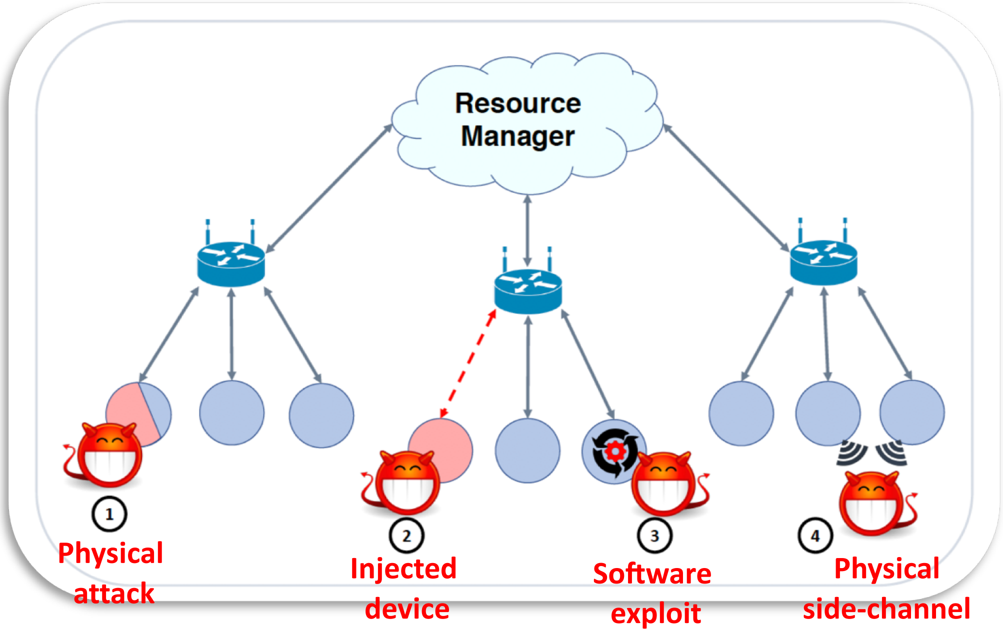
Formal Guarantees for Macroprogramming Heterogeneous IoT Networks

RemedIoT: Remedial Actions for Internet-of-Things Conflicts (BuildSys 2019)

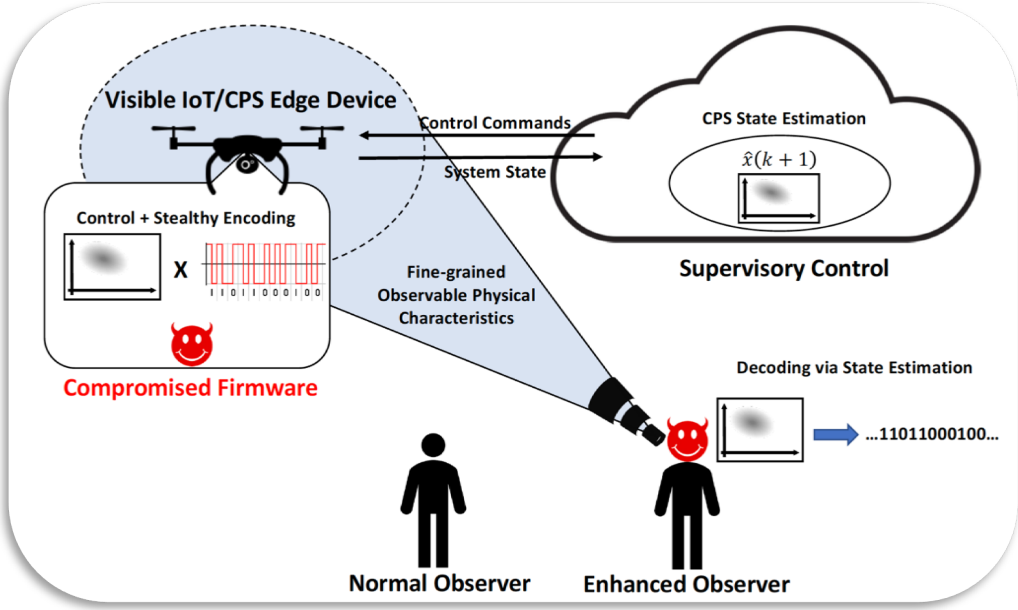




Characterizing Security in IoT Macroprogramming Environments

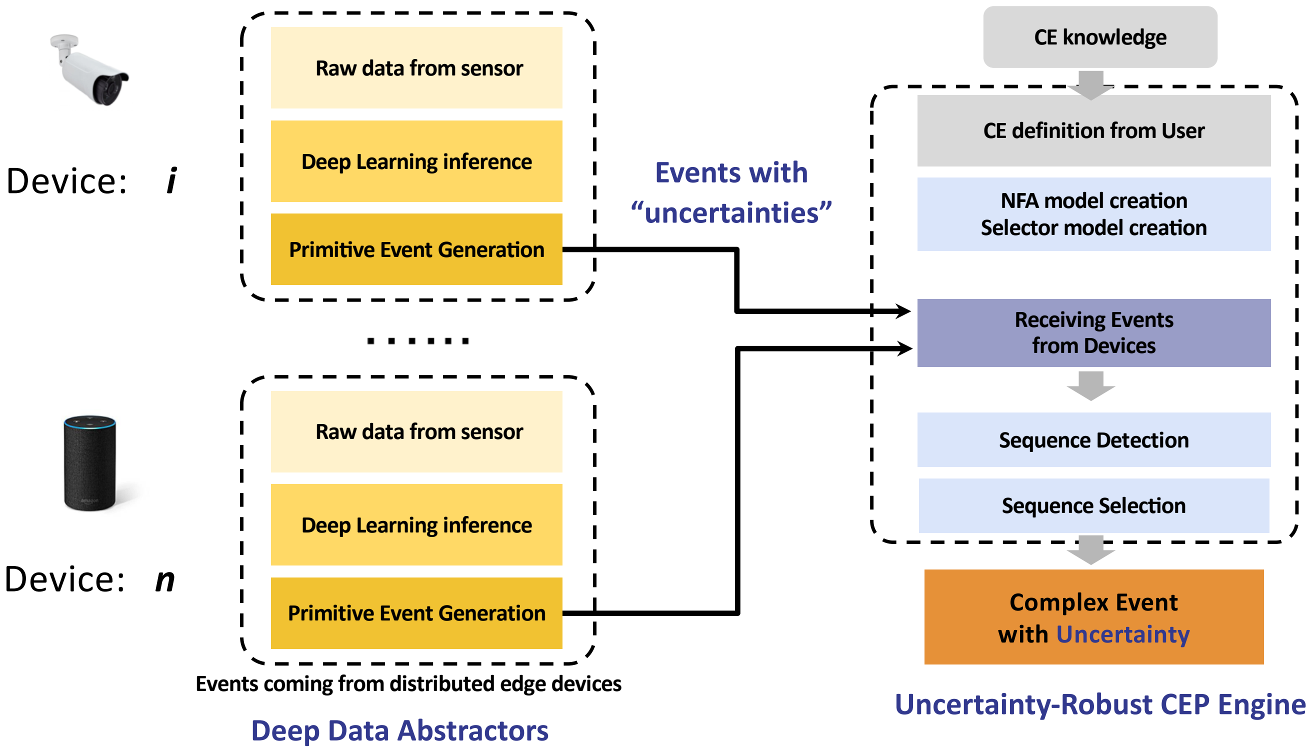


The Case for Robust Adaptation: Autonomic Resource Management is a Vulnerability (MILCOM 2019)



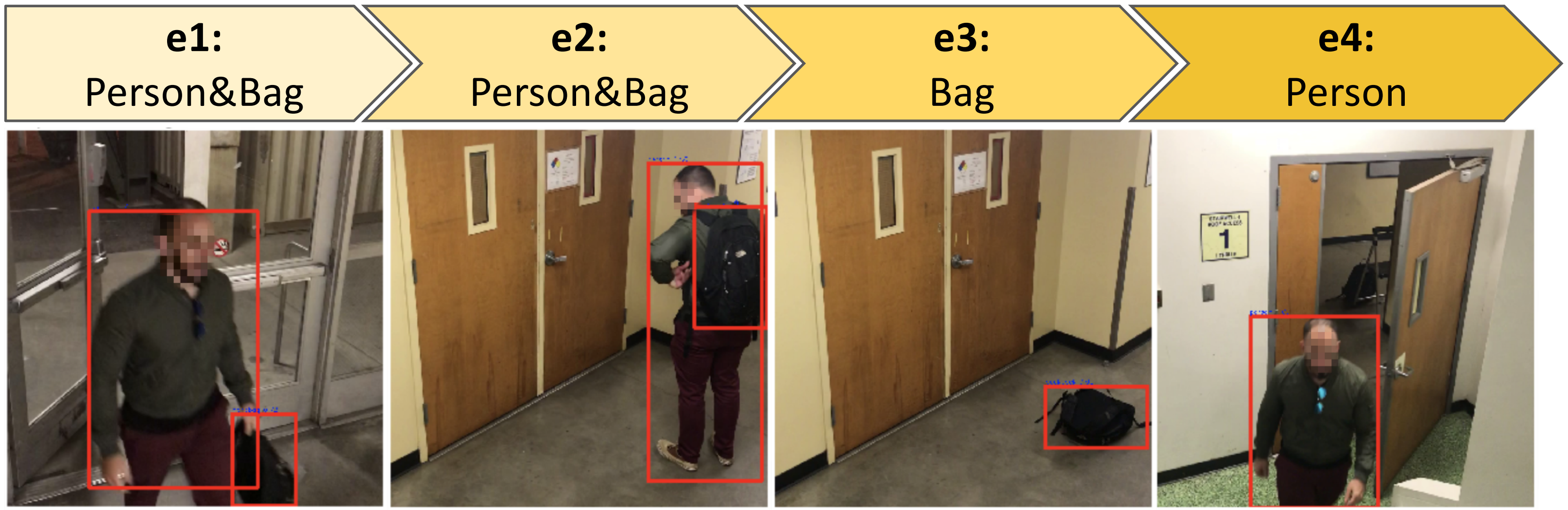
Let's Talk Through Physics! Covert Cyber-Physical Data Exfiltration on Air-Gapped Edge Devices (Submitting to USENIX 2020)

Robust Multi-modal Inferencing in Heterogenous IoT Environments



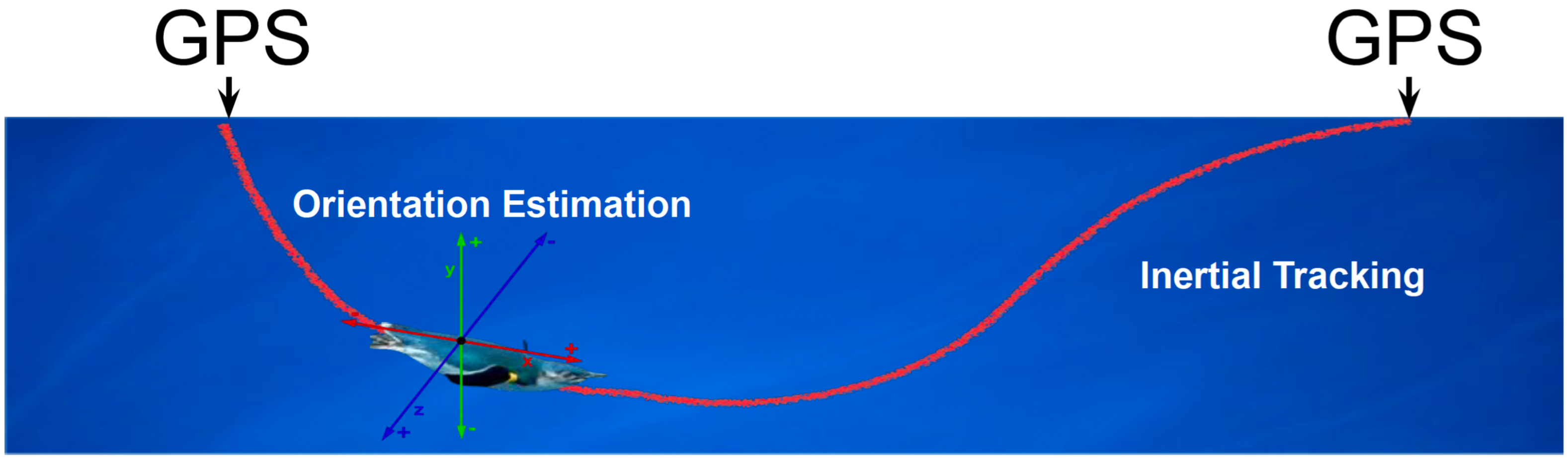
DeepCEP: Deep Complex Event Processing Using Distributed Multimodal Information (SMARTCOMP 2019)

Robust Multi-modal Inferencing in Heterogenous IoT Environments

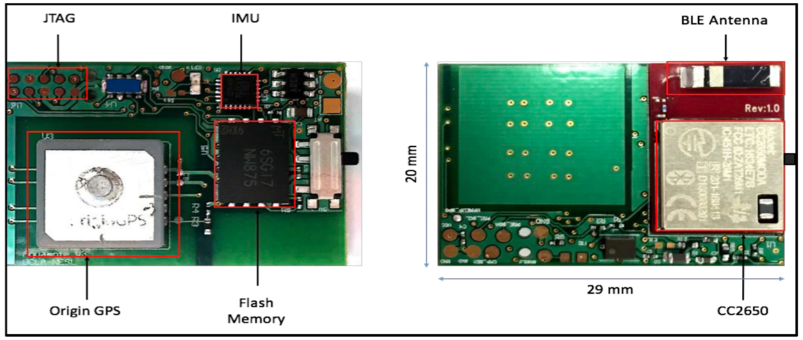


Example: Detecting an unattended bag

Some future applications…

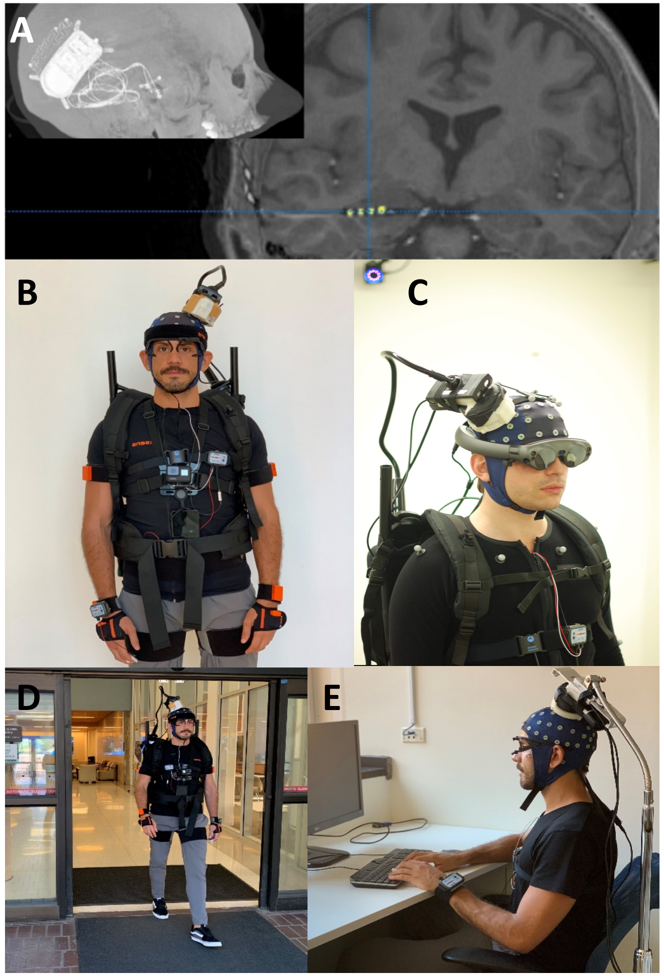


3D Trajectory and Orientation Estimation of Marine Animals

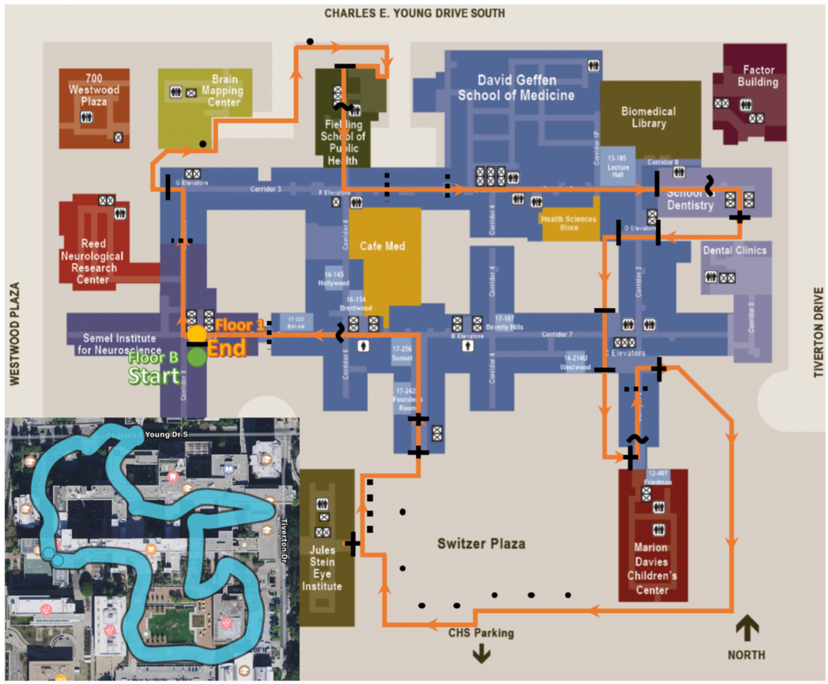


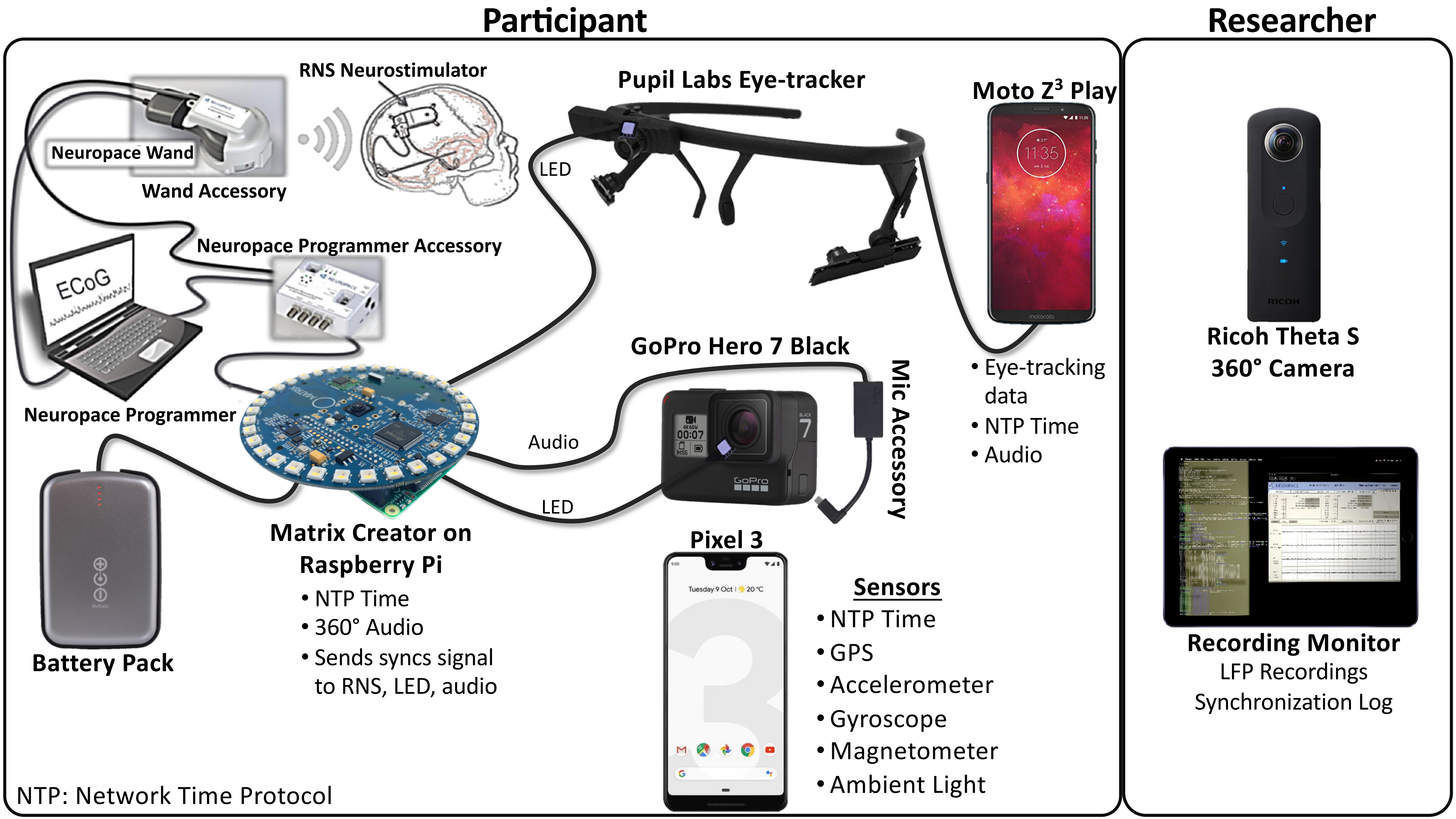
Aquamote v3

Some future applications…

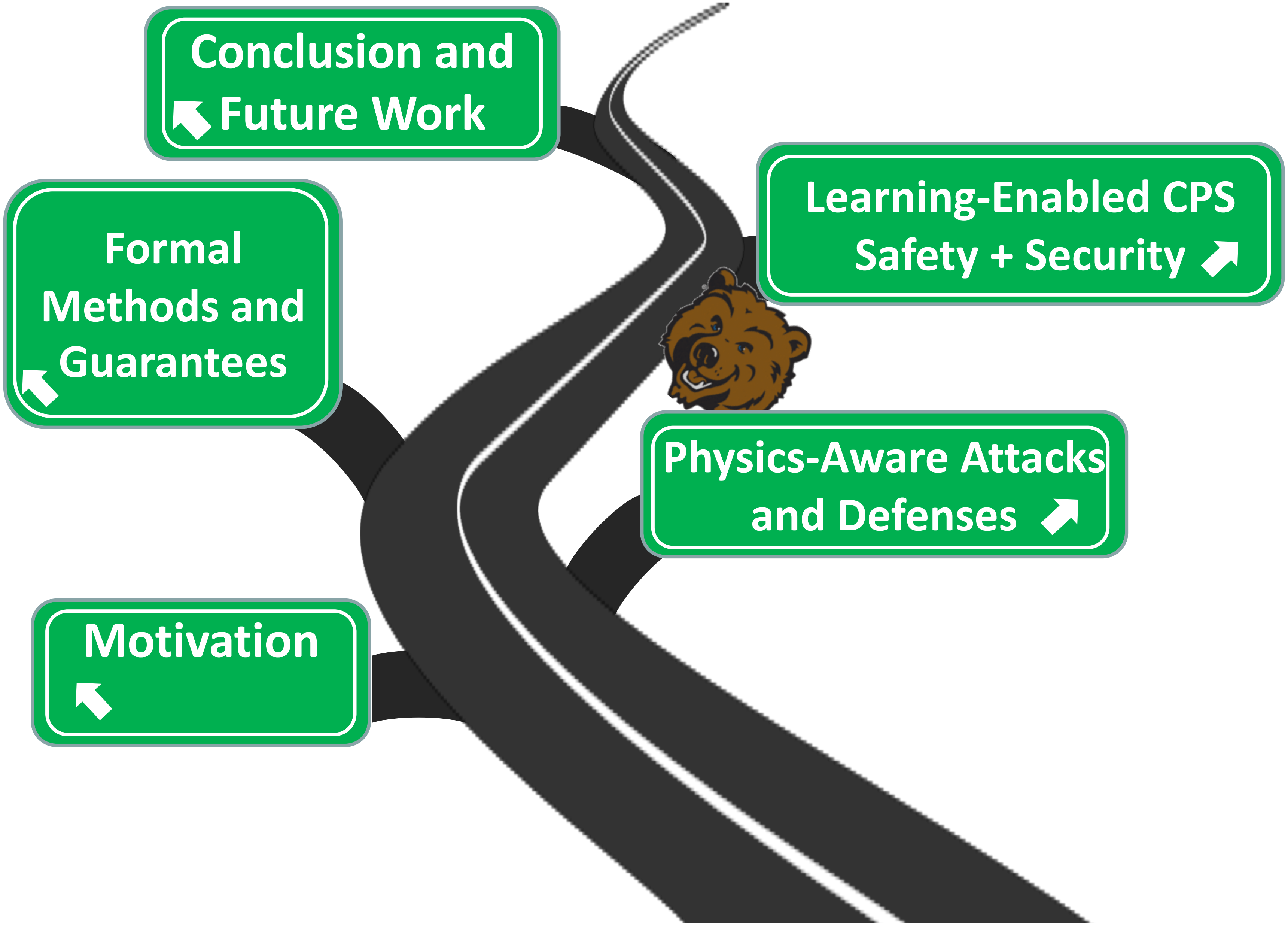


Decoding How Humans Encode Memory





Participant
Researcher
RNS Neurostimulator
Neuropace Wand
Wand Accessory
Pupil Labs Eye-tracker
LED
Moto Z3 Play
ECoG
Neuropace Programmer Accessory
Neuropace Programmer
GoPro Hero 7 Black
Mic Accessory
Audio
LED
• Eye-tracking data
• NTP Time
• Audio
Ricoh Theta S 360° Camera
Matrix Creator on Raspberry Pi
• NTP Time
• 360° Audio
• Sends syncs signal to RNS, LED, audio
Battery Pack
Pixel 3
Sensors
• NTP Time
• GPS
• Accelerometer
• Gyroscope
• Magnetometer
• Ambient Light
Recording Monitor
LFP Recordings
Synchronization Log
NTP: Network Time Protocol


Conclusion and Future Work
Learning-Enabled CPS Safety + Security
Formal Methods and Guarantees
Physics-Aware Attacks and Defenses
Motivation

Conclusion

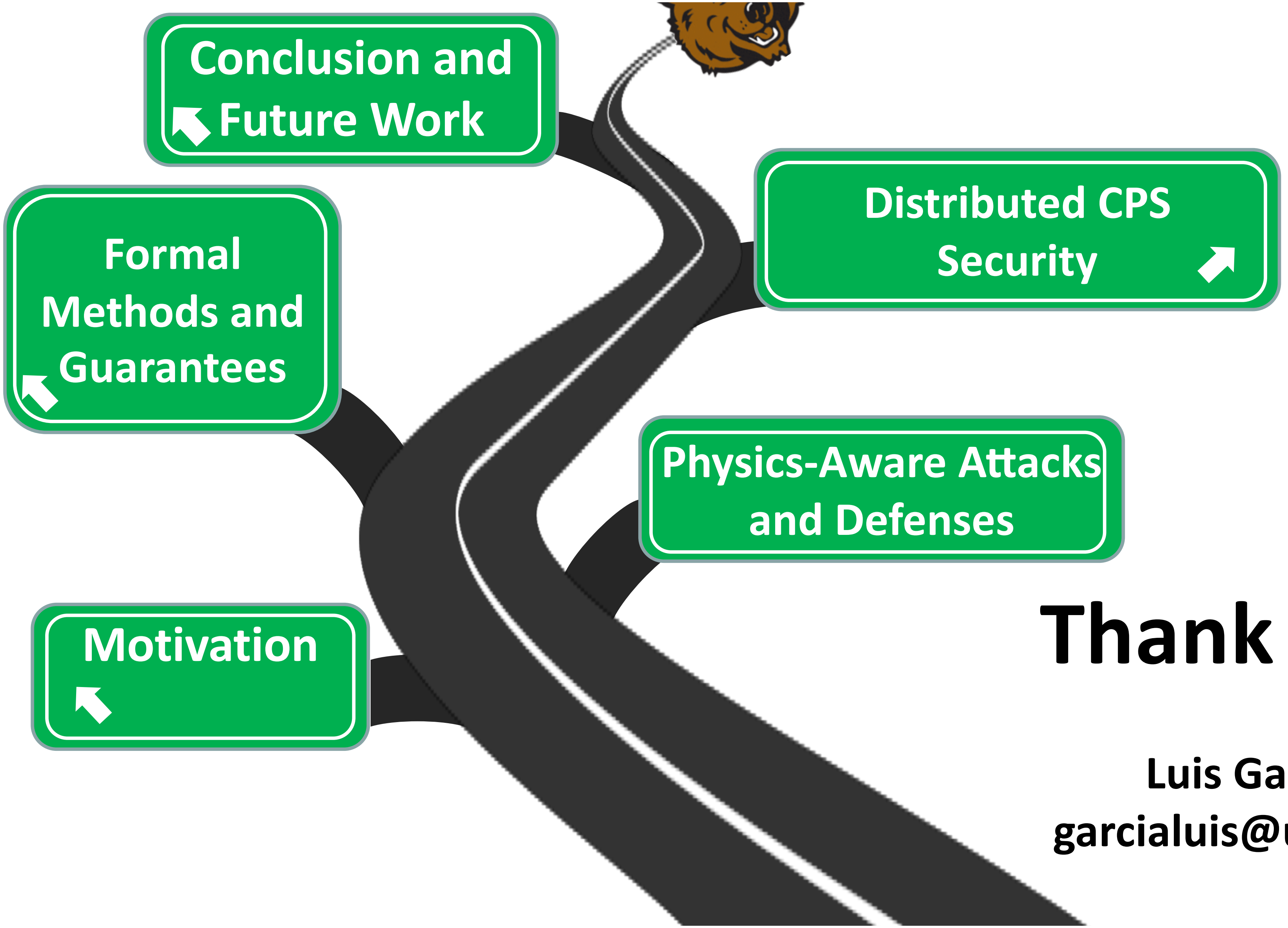
- Practical security analysis based on **physical properties** of CPS
  - *Harvey: a physics-aware, two-faced rootkit malware*
  - *Physics-aware defenses for at various levels of distribution as well as in domain-specific scenarios*

- Applicability of **formal deductive verification** techniques in complex CPS
  - *HyPLC: a bi-directional translation of PLC controller code and differential dynamic logic hybrid programs*

- Practical security and safety considerations for **learning-enabled IoT/CPS**
  - *Formal guarantees for macroprogramming heterogenous IoT networks*
  - *Multi-modal inferencing in distributed and heterogenous IoT networks*

- **Future Work**
  - *Scalable deductive security verification via compositional verification and safety contracts*
  - *Robust multi-modal inferencing in distributed and contested environments*
  - *Security detection and intervention in IoT macroprogramming environments*

Questions?



Thank You!

Luis Garcia
garcialuis@ucla.edu

Publications

- Data Flow Security for Mobile Devices
  - Context Aware Information-Flow-Based Micro-Security Perimeters for Mobile Devices. *DSN, 2016*
- Cyber-physical Vulnerability Assessment
  - A Cyber-Physical Modeling and Assessment Framework for Power Grid Infrastructures. *IEEE Transactions on Smart Grid, 2015*
  - Covert Channel Communication Through Physical Interdependencies in Cyber-Physical Infrastructures. *IEEE SmartGridComm, 2014*
  - Threat Model Quantification in Smart Grid Critical Infrastructures. *IEEE SmartGridComm 2014*
- Embedded Systems Verification and Controller Security
  - Hey, My Malware Knows Physics! Attacking PLCs with Physical Model Aware Rootkit. *NDSS, 2017*
  - Detecting PLC Control Corruption via On-Device Runtime Verification. *IEEE Resilience Week 2016*
- Cyber-physical Control Flow Integrity
  - Cyber-Physical Control Flow Integrity for Distributed Controllers. *Submitting to ICCPS 2019*
  - PAtt: Physics-based Attestation of Control Systems. *RAID 2019*
  - Tell Me More than Assembly Instructions! Reversing Semantics of IoT Software Binaries. *DSN 2019*
  - See No Evil, Hear No Evil, Feel No Evil, Print No Evil? Malicious Fill Patterns Detection in Additive Manufacturing. *USENIX Security, 2017*
- Hybrid Systems Modeling and Verification
  - HyPLC: Hybrid Programmable Logic Controller Program Translation for Verification. *ICCPS 2019 (Best Paper Finalist)*
- Macroprogramming of Distributed and Heterogeneous IoT/CPS
  - Let's Talk Through Physics! Covert Cyber-Physical Data Exfiltration on Air-Gapped Edge Devices. *Submitting to USENIX Security, 2020*
  - DDFlow: Visualized Declarative Programming for Heterogeneous IoT Networks. *IoTDI 2019*
  - RemedIoT: Remedial Actions for Internet-of-Things Conflicts. *BuildSys 2019*
- Robust Multi-modal Inferencing
  - PhysioGAN: Training High Fidelity Generative Model for Physiological Sensor Readings. *Submitted to TPAMI 2019*
  - DeepCEP: Deep Complex Event Processing Using Distributed Multimodal Information. *SmartComp 2019*
  - RadHAR: Human Activity Recognition from Point Clouds Generated through a Millimeter-wave Radar. *MMNets 2019*

Formal Verification of Hybrid Controller Logic for Transient Stability

(Case Study)

Transient Stability of a Simple Power System

Final Complete SMIB Hybrid Program

$$init \Rightarrow [\{ctrl; plant\&H\}*](req)$$

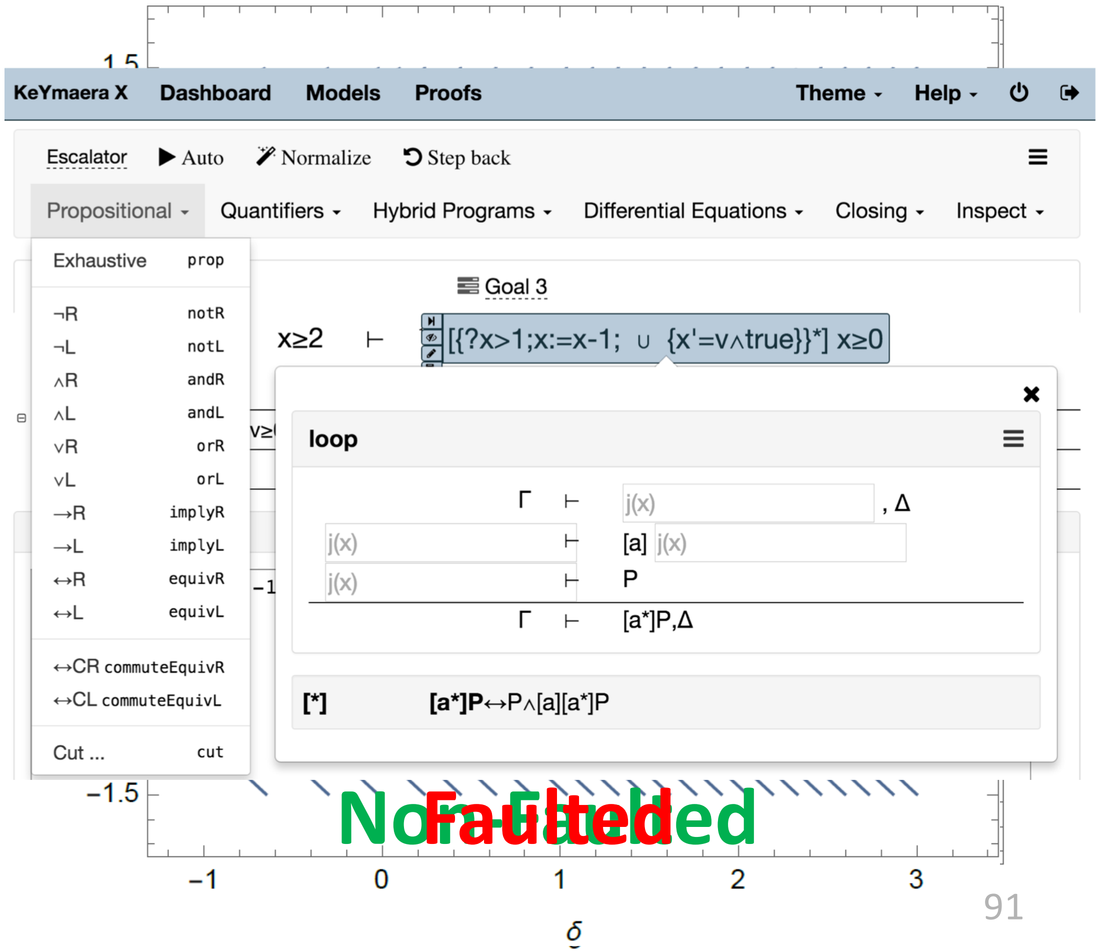
$$init \equiv P_M = 1 \bigwedge P_{e,max} = \tfrac{3}{2} \bigwedge \omega = 0 \bigwedge \theta = \arcsin(\tfrac{P_M}{P_{e,max}})$$
$$\bigwedge \theta_{max} = \pi - \theta \bigwedge \sin\theta = \tfrac{P_M}{P_{e,max}} \bigwedge \cos\theta = \tfrac{\sqrt{P_{e,max}^2 - P_M}}{P_{e,max}}$$
$$\bigwedge c = 2P_M\theta_{max} - 2P_{e,max}\cos(\theta)$$

$$ctrl \equiv \{(a := P_M - Pe,max\sin(\theta)) \bigcup (a := P_M)\};$$
$$t := 0;$$
$$d0 := d;$$
$$v0 := v;$$
$$T := *;$$
$$?T \geq 0 \wedge \theta_f(T) \leq \theta_e$$
$$\wedge\, 360c \leq 720\theta P_M + 720P_{e,max} - 360\theta_f(T)^2 P_{e,max}$$
$$+ 30\theta_f(T)^4 P_{e,max} - \theta_f(T)^6 P_{e,max} - 360\omega_f(T)^2$$

$$plant \equiv \theta' = \omega, \omega' = a, \sin\theta' = \omega\cos\theta, \cos\theta' = -\omega\sin\theta, t' = 1$$

$$H \equiv \sin^2\theta + \cos^2\theta = 1$$

$$req \equiv \theta \leq \theta_{max}$$

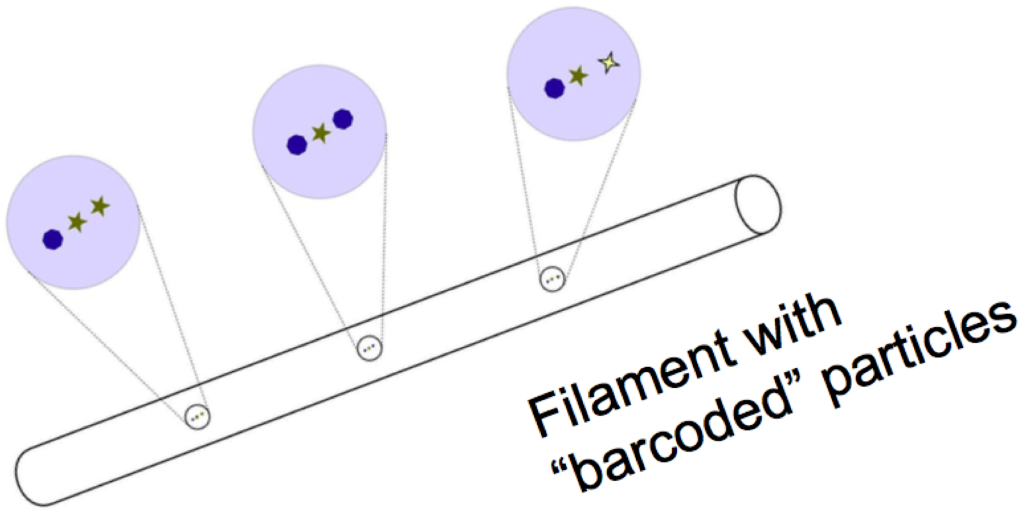


Embedding material as identification markers

Embedded materials are unknown to third party

- ➢ Micro/nano materials are invisible to naked eyes

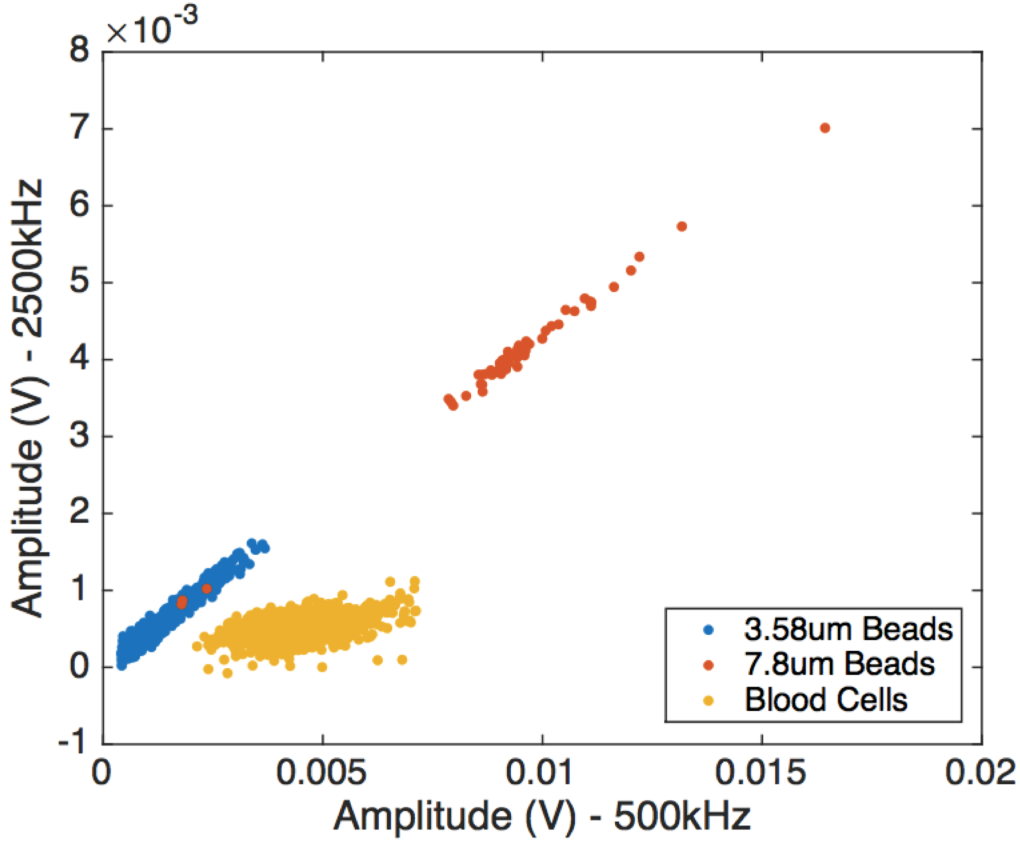
- ➢ Unknown materials averts the scanning attempts without correct modality
- ➢ Embedding material as "barcode" for unique identification



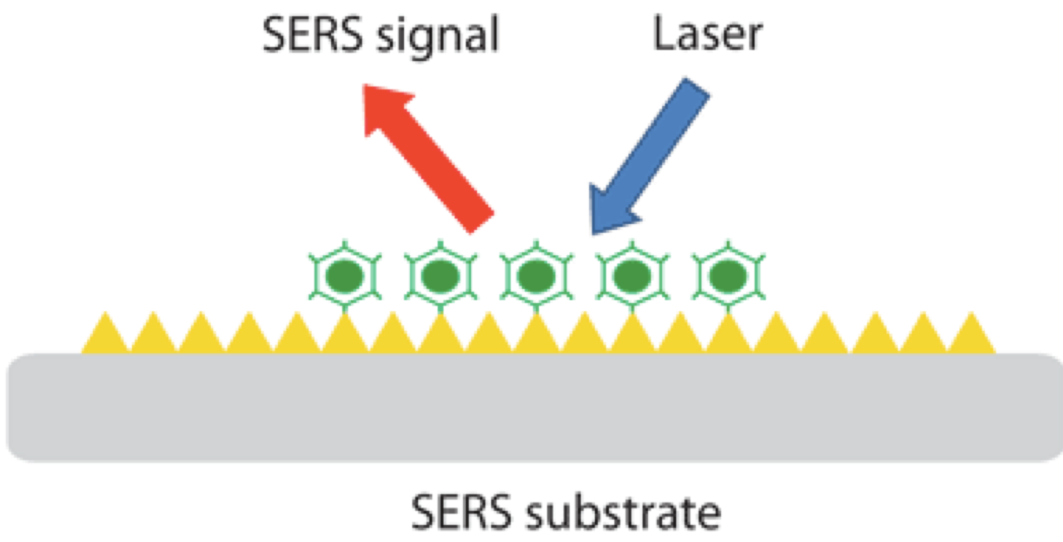
RT PP

Previous study: identification markers in blood tests

Microbeads with different size and concentration can identify multiple test results



Embedding markers



Raman spectroscopy

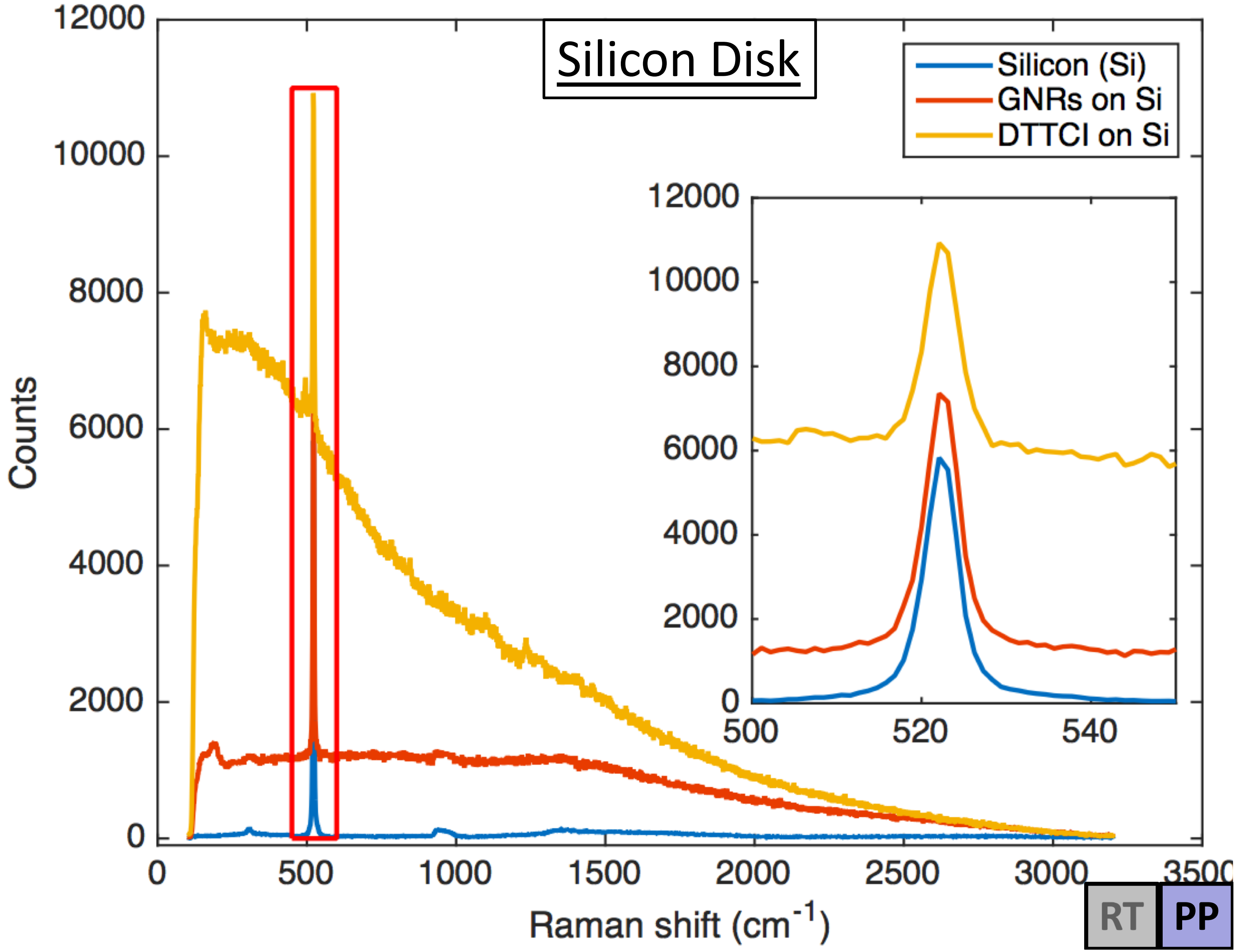
- Monochromatic light source
- Measure the shifted frequency of incident light

Surface-enhanced Raman spectroscopy (SERS) using gold nanorods (GNRs) or 3,3’-Diethylthiatricarbocyanine iodide (DTTCI) as enhancers

Embedded markers selection based on the availability of scanning modalities

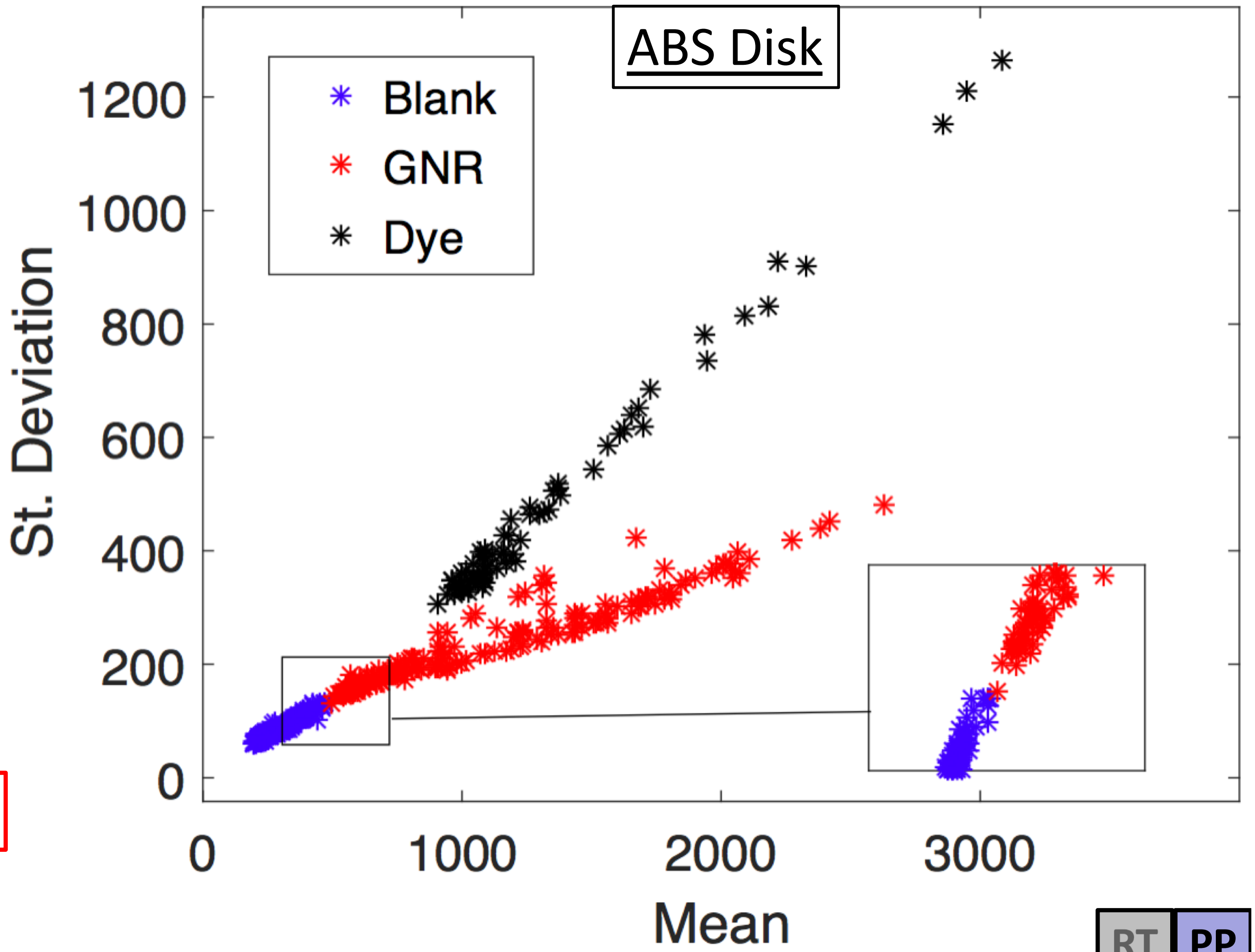
Raman Spectroscopy

GNRs and dye (DTTCI) in SERS result in **enhanced spectral response**



RT PP

Raman Spectroscopy



Spectral response of markers are easily distinguishable.

300um depth limitation

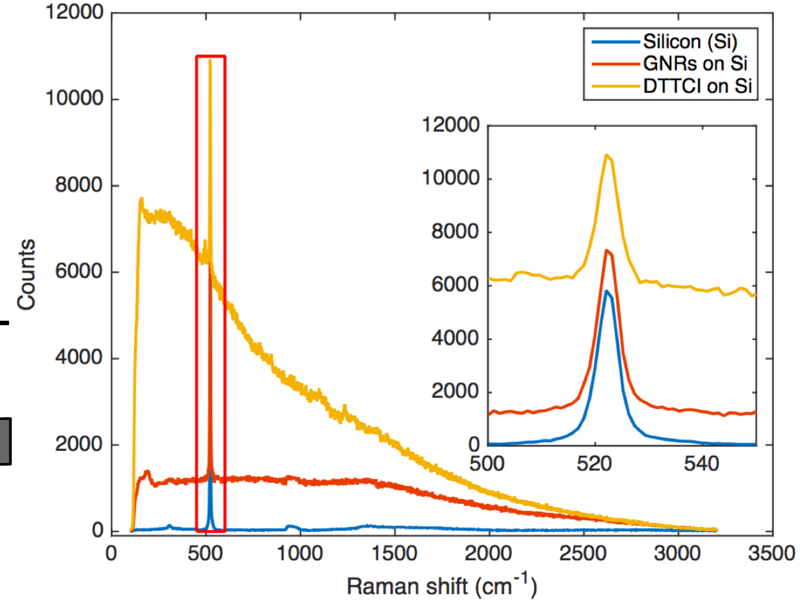
RT PP

Raman spectroscopy

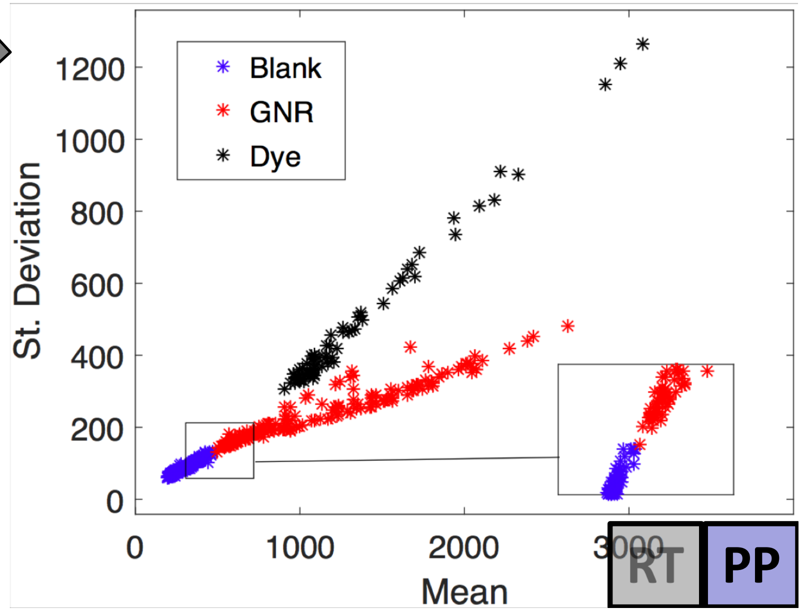
- ➢ Enhanced spectral response of Silicon using GNRs and dye (DTTCI) in SERS
- ➢ Spectral responses of control ABS and embedded ABS show
  - ○ Unadulterated ABS has more concentrated spectral response
  - ○ Spreading response due to uncontrolled concentration of embedded markers

300um depth limitation

SERS initial results of Silicon spectrum.



SERS response of control and embedded 3D disks

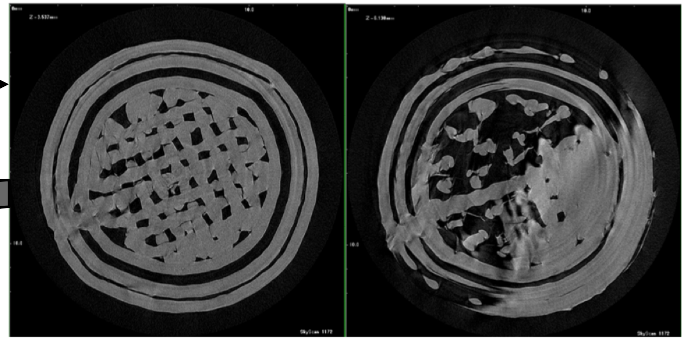


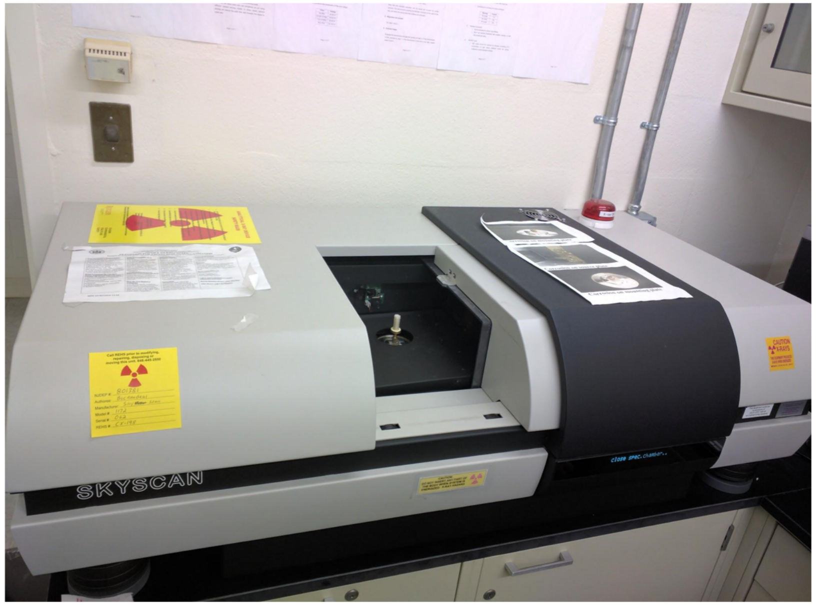
RT PP

MicroCT

- ➢ Higher depth scan for 3D object reconstruction
- ➢ Steel saturated filament is used for higher contrast

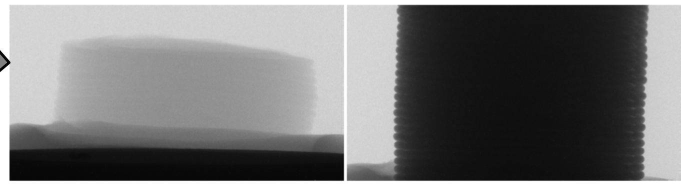
MicroCT scan of ABS cylindrical tube with embedded GNRs





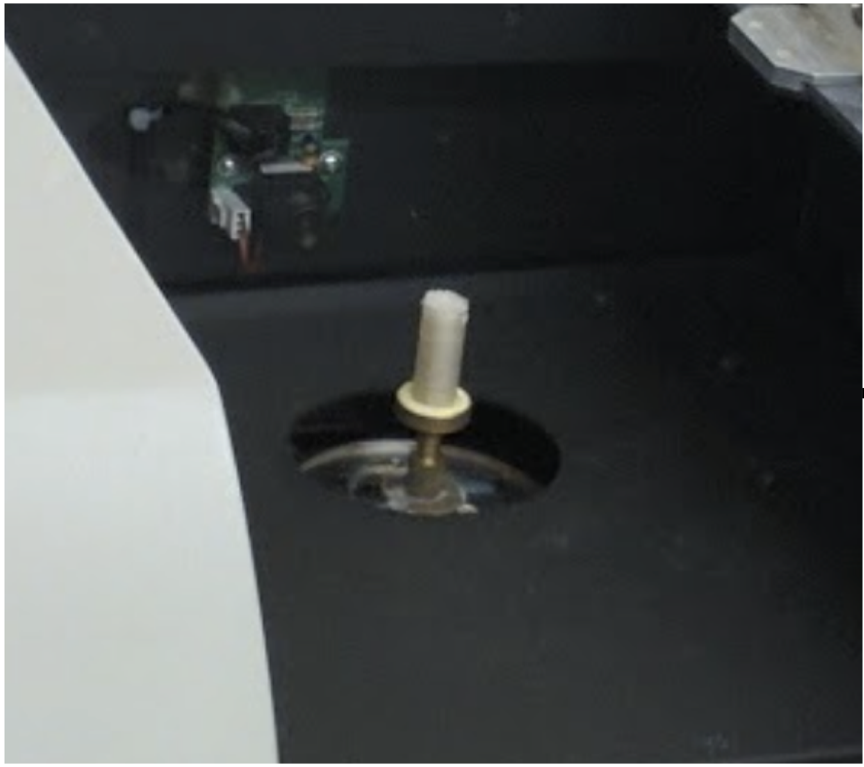
Skyscan MicroCT 1172 scanner

Changing filament and markers for higher contrast

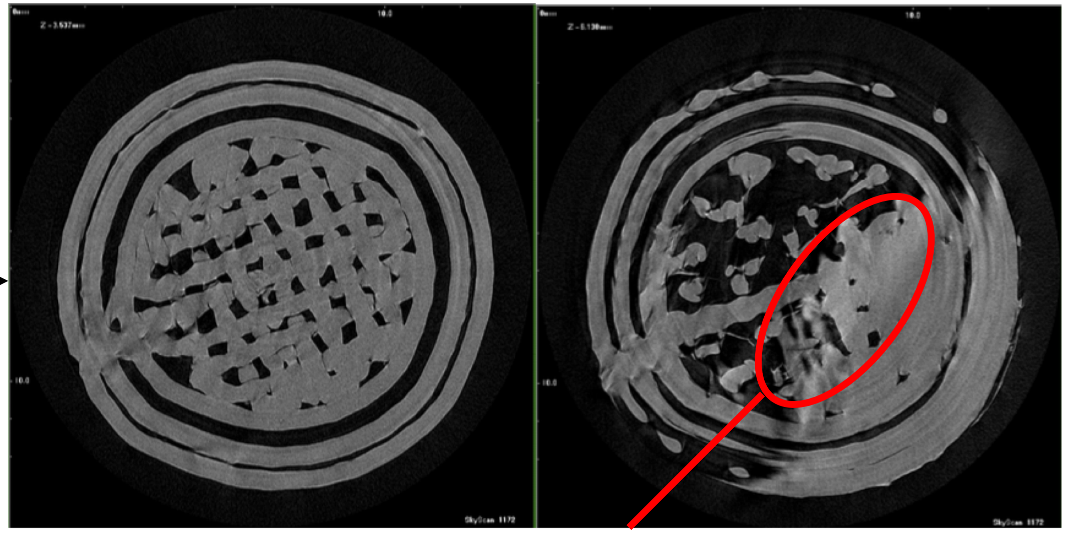


MicroCT scan of PLA (left) and steel embedded

MicroCT scanning: ABS filament



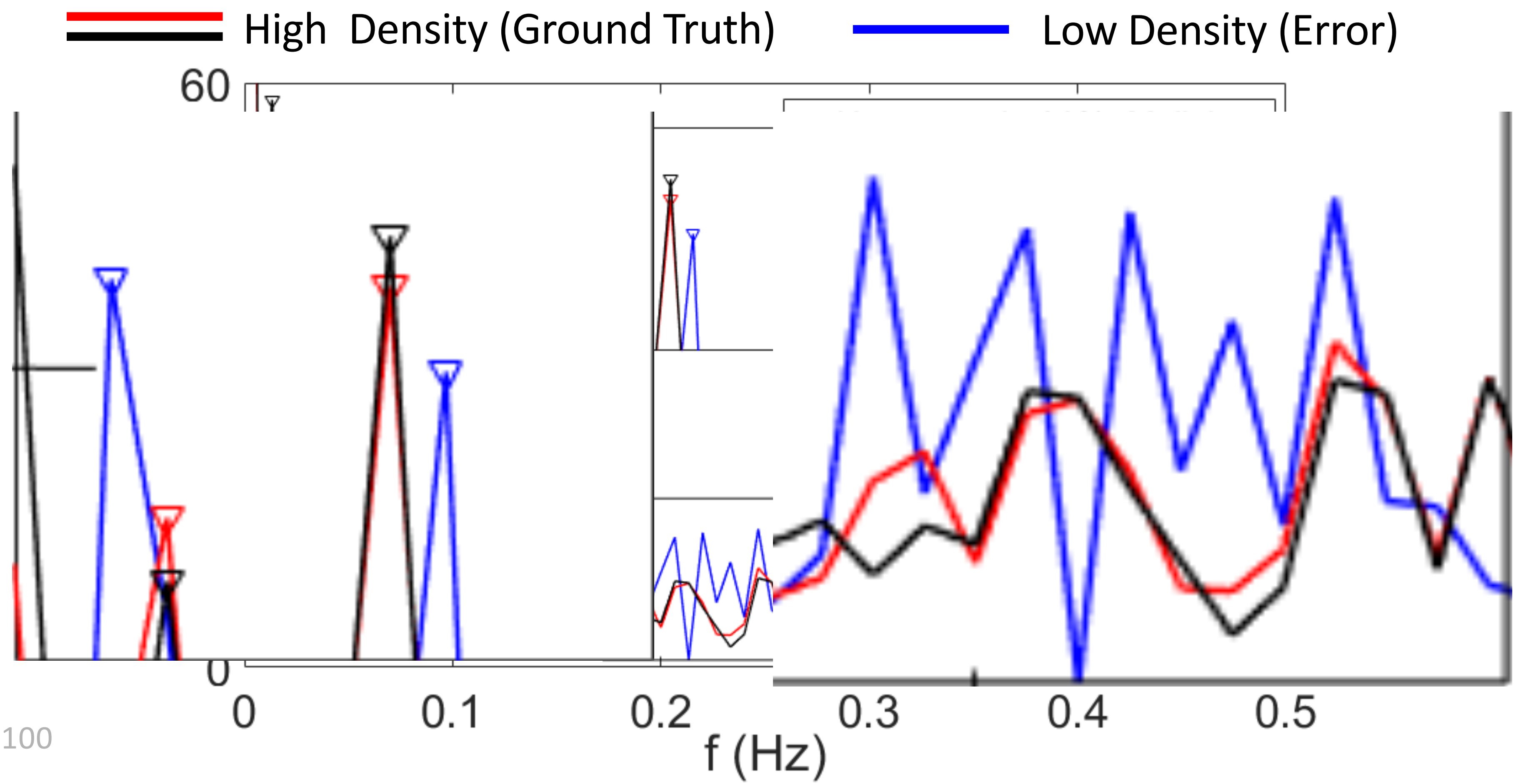
MicroCT scan of ABS cylindrical tube with embedded GNRs



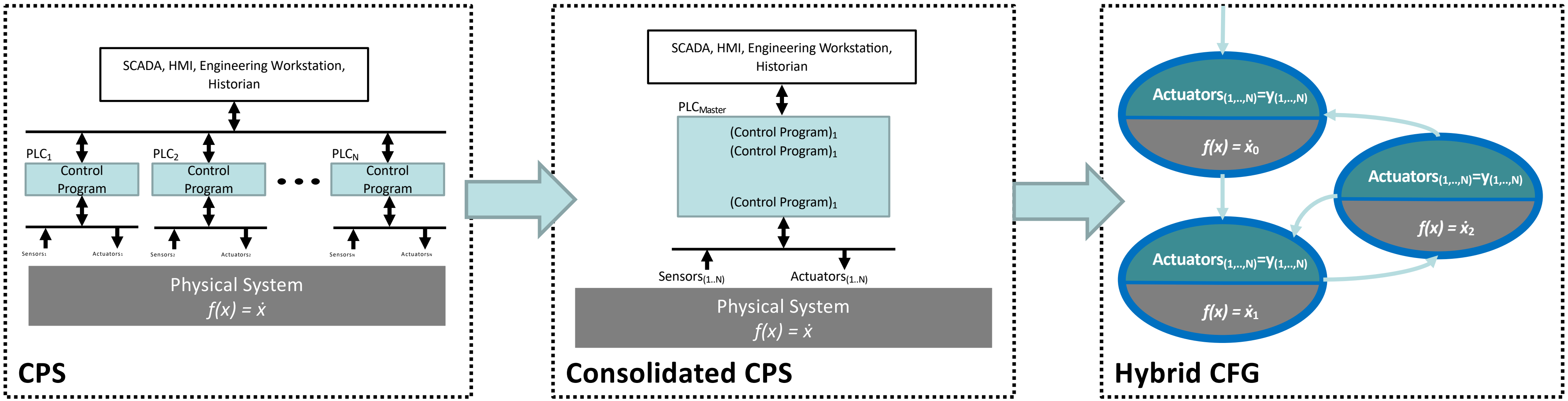
Embedded GNRs with high reflection

The precise placement of GNRs is out of scope for the current work

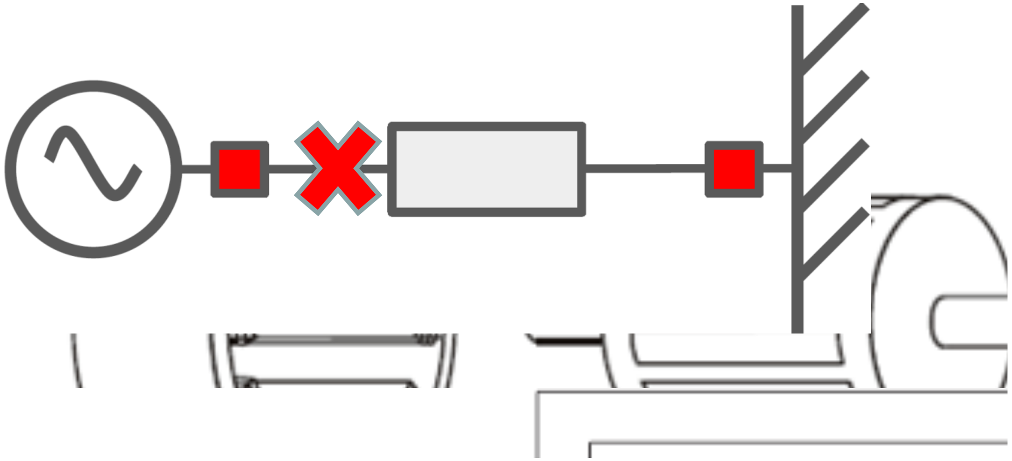
Quantitative spatial analysis also shows error



Scadman: Cyber-physical Control Flow Integrity (Submitting to USENIX Security 2018)



Rotor Angle Stability of Synchronous Machine

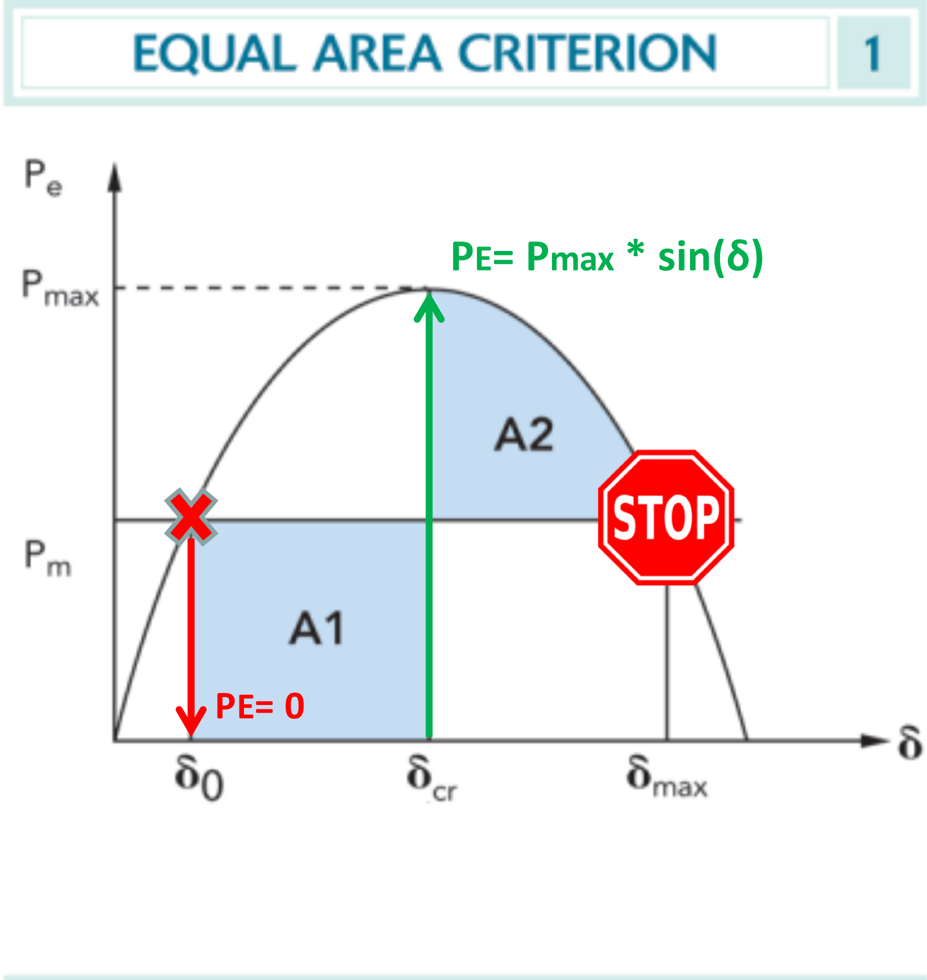


$$\frac{d^2\delta}{dt^2} = P_M$$

P_S

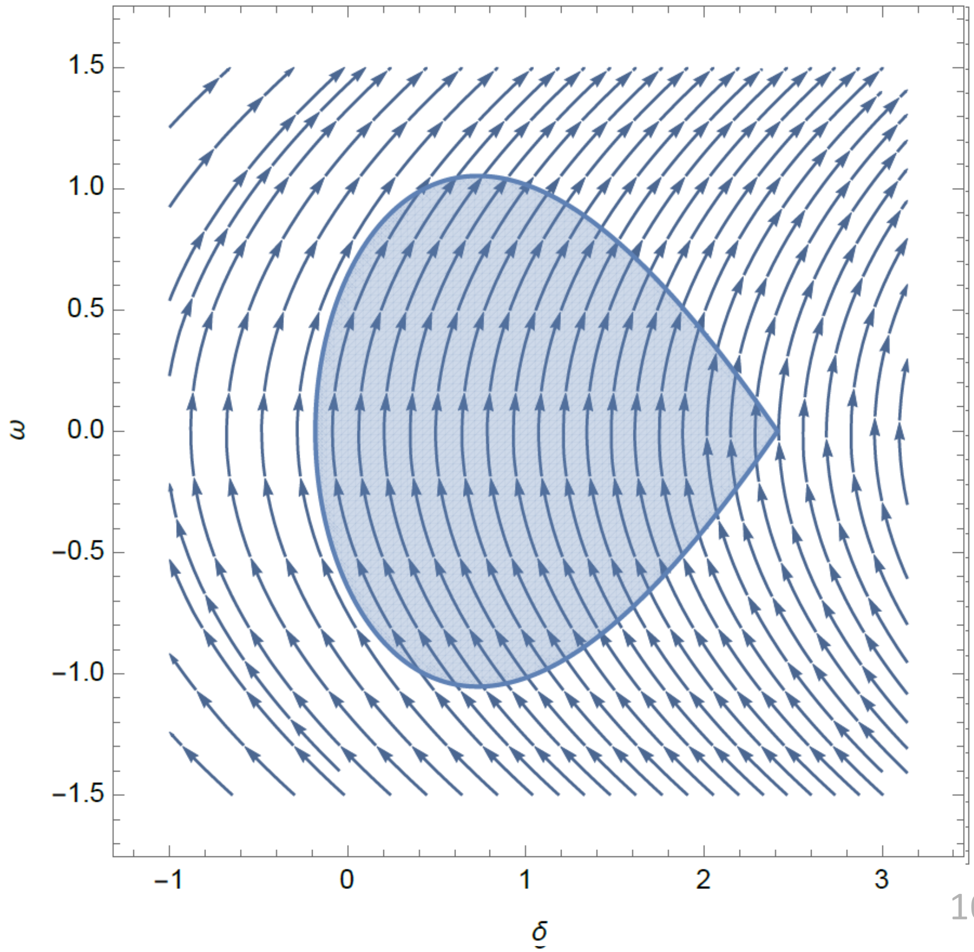
'S W

$P_E = P_{max} * \sin(\boldsymbol{\delta})$

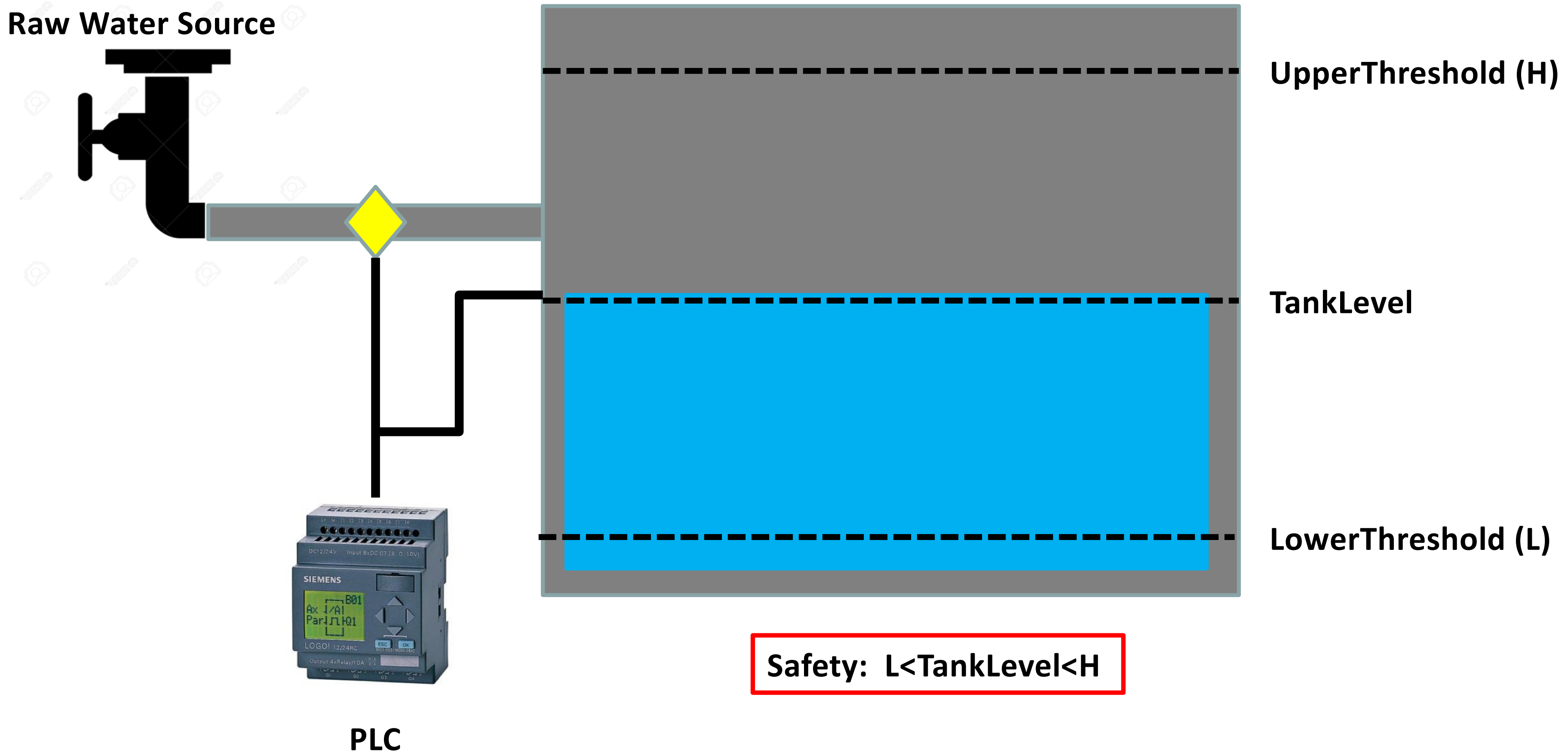


Hybrid Program Representation

Final Complete SMIB Hybrid Program
$init \Rightarrow [\{ctrl; plant\&H\}*](req)$
$init \equiv P_M = 1 \bigwedge P_{e,max} = \frac{3}{2} \bigwedge \omega = 0 \bigwedge \theta = \arcsin(\frac{P_M}{P_{e,max}})$ $\bigwedge \theta_{max} = \pi - \theta \bigwedge \sin\theta = \frac{P_M}{P_{e,max}} \bigwedge \cos\theta = \frac{\sqrt{P_{e,max}^2 - P_M}}{P_{e,max}}$ $\bigwedge c = 2P_M\theta_{max} - 2P_{e,max}\cos(\theta)$
$ctrl \equiv \{(a := P_M - Pe,max\sin(\theta)) \bigcup (a := P_M)\};$ $t := 0;$ $d0 := d;$ $v0 := v;$ $T := *;$ $?T \geq 0 \wedge \theta_f(T) \leq \theta_e$ $\wedge 360c \leq 720\theta P_M + 720P_{e,max} - 360\theta_f(T)^2 P_{e,max}$ $+30\theta_f(T)^4 P_{e,max} - \theta_f(T)^6 P_{e,max} - 360\omega_f(T)^2$
$plant \equiv \theta' = \omega, \omega' = a, \sin\theta' = \omega\cos\theta, \cos\theta' = -\omega\sin\theta, t' = 1$
$H \equiv \sin^2\theta + \cos^2\theta = 1$
$req \equiv \theta \leq \theta_{max}$

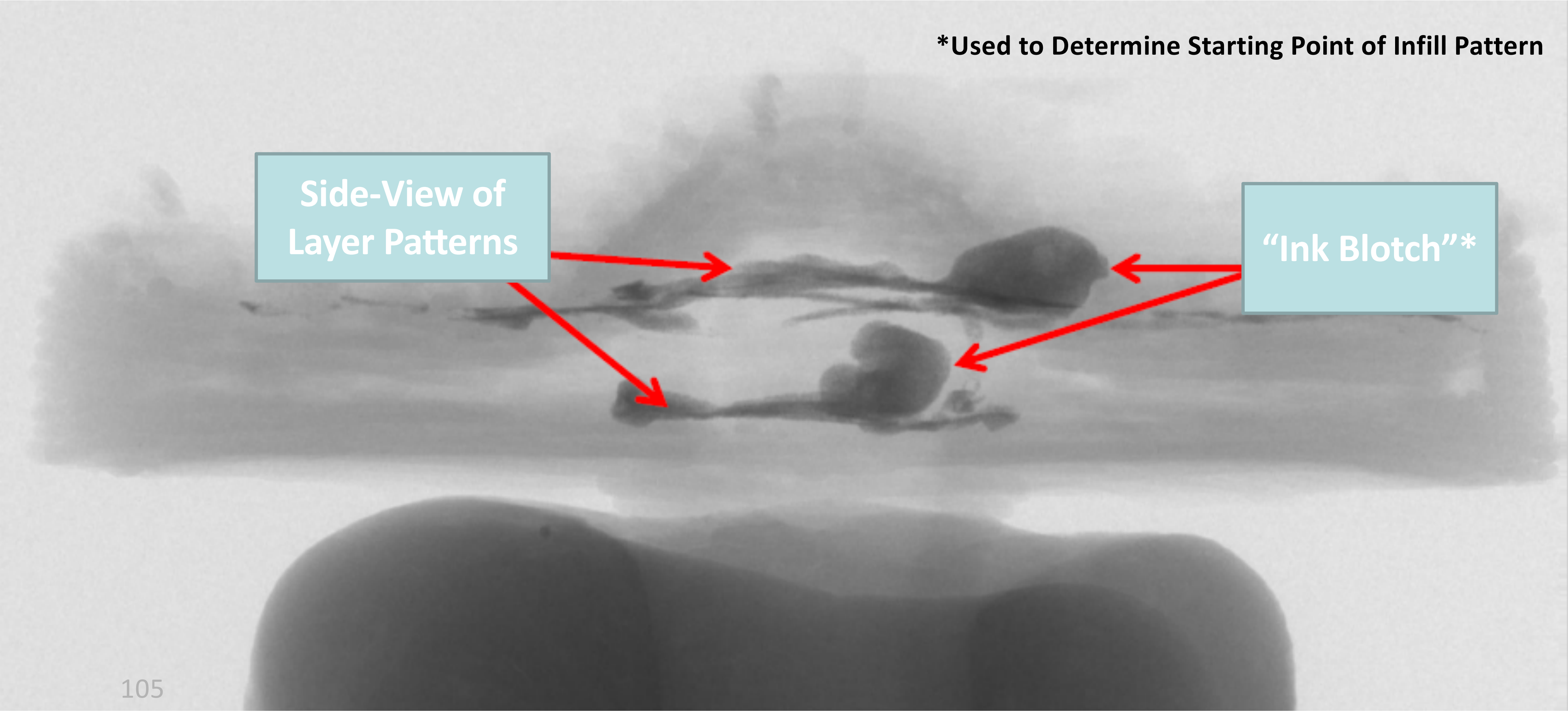


HyPLC: Hybrid PLC Program Translation for Verification



Safety: L<TankLevel<H

Tibial Implant MicroCT Scan



Tell Me More Than Just Assembly! Reversing Cyber-physical Execution Semantics of Embedded IoT Controller Software Binaries

Pengfei Sun†, Luis Garcia∗ and Saman Zonouz†
†Rutgers University, ∗University of California, Los Angeles

We live in a cyber-physical world…

* https://bricz.com/oms-integration/internet-of-thingsiot-and-cloud-computing-concept-smart-city-cyber-physical-systemscps/

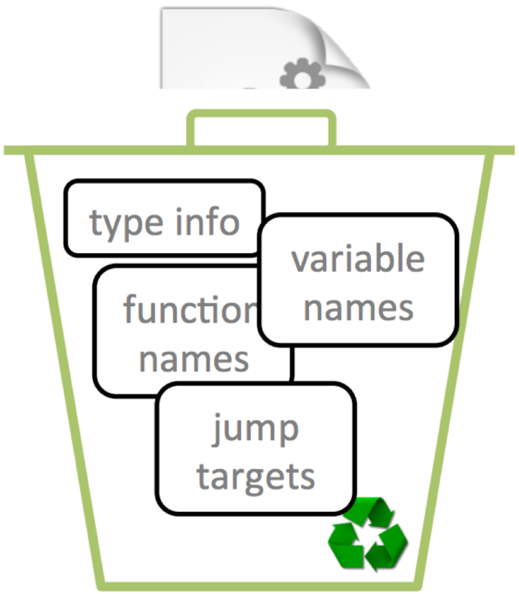
…and attacks are increasingly cyber-physical

Analyze and reverse engineer the system from security perspective

```
int gcd(int a, int b){
    int com_divisor;
    if(b!=0){
        com_divisor = gcd (b, a % b);
    }
    else{
        com_divisor = a;
    }
    return com_divisor;
}
```

Compile and Link

Strip



Reverse Engineering

```
8048094:  push   ebp
8048095:  mov    ebp, esp
8048097:  sub    esp, 0x18
804809a:  cmp    [ebp + 0xc]:32, 0x0
804809e:  jnz    0x80480a5
80480a0:  mov    eax, [ebp + 0x8]:32
```

Rever
Engine

Domain knowledge

```
int32_t sub_8048094 (int32_t arg1, int32_t arg2){
    int32_t eax1;
    if(arg2!=0){
        eax1=sub_8048094(arg2, arg1 % arg2);
    }
    else{
        eax1=arg1;
    }
    return eax1;
}
```

???

```
int32_t
    int32_t eax1;
    if(arg2!=0){
        eax1=sub_8048094(arg2, arg1 % arg2);
    }
    else{
        eax1=arg1;
    }
    return eax1;
}
```

didWeLose(int gauntLetFound, int infinityStoneCount)

Mismo: Domain-specific Reverse Engineering Framework

- Propose a **domain-specific reverse engineering** framework to extract **high-level algorithmic control- and data-flow semantics** from embedded binary executables.

- Introduce a **semantic mapping** using **dynamic binary analysis** and symbolic comparison of the **mathematical and binary expressions** to fill the semantic gap between high-level algorithm descriptions and low-level stripped binary segments.

Domain Knowledge (Cyber-physical System)

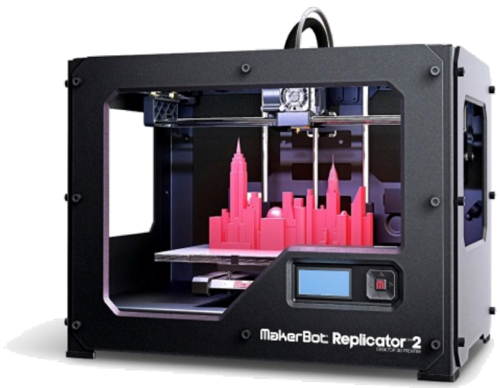
Input

Proportional-Integral-Derivate (PID)

Pulse-width modulation (PWM)

Kalman-Filters

Output



Current Solutions

- IDA PRO
- OllyDbg
- Ninja
- Snowman

Disassembler,
Decompiler,
Static and dynamic analysis

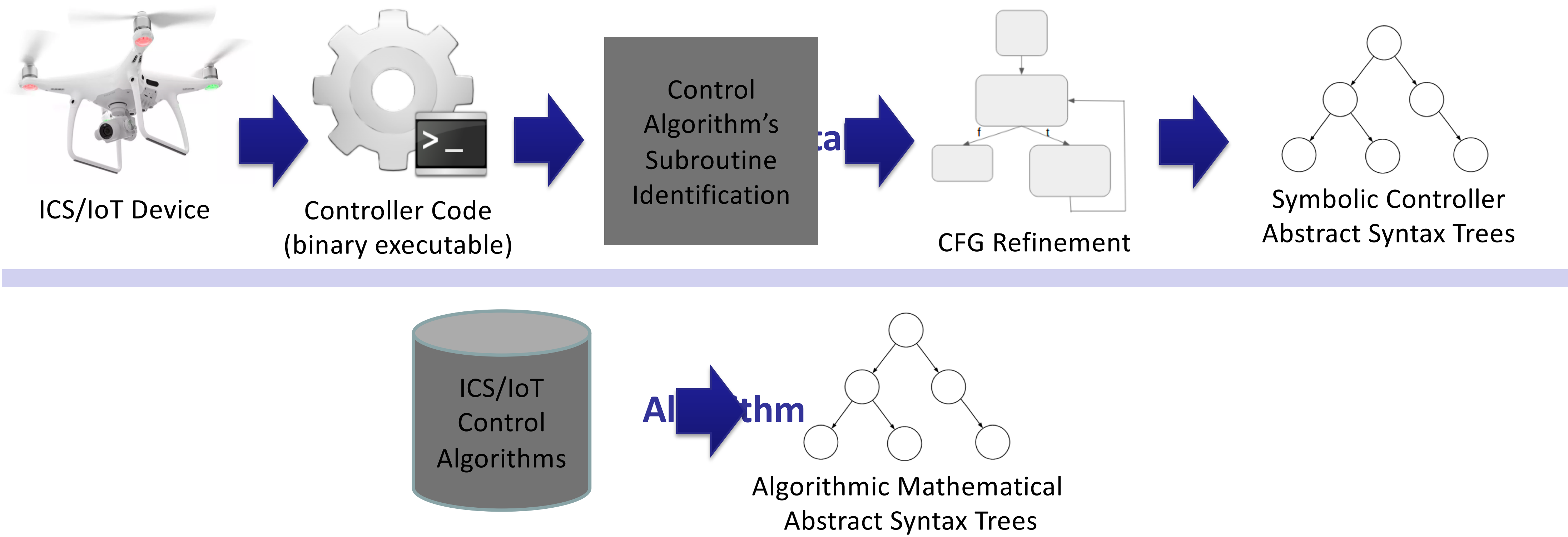
- Reward (NDSS’10)
- TIE (NDSS’11)
- Howard (NDSS’11)
- ReViver (ACSAC’16)

Data structure definition,
Data structure memory instance

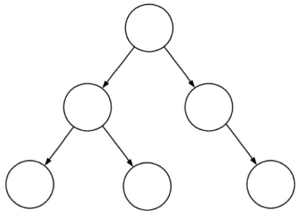
- BinDiff (SSTIC'05)
- BinJuice (PPREW '13)
- Blex (Usenix’14)

Binary similarity checking

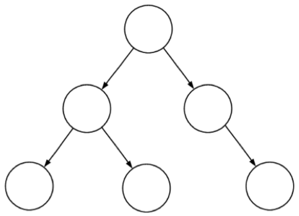
Mismo Overview



Mismo Overview

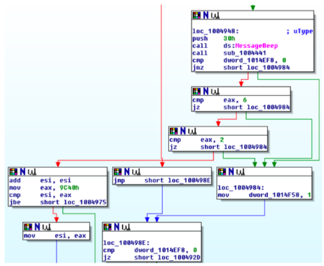


Symbolic Controller
Abstract Syntax Trees



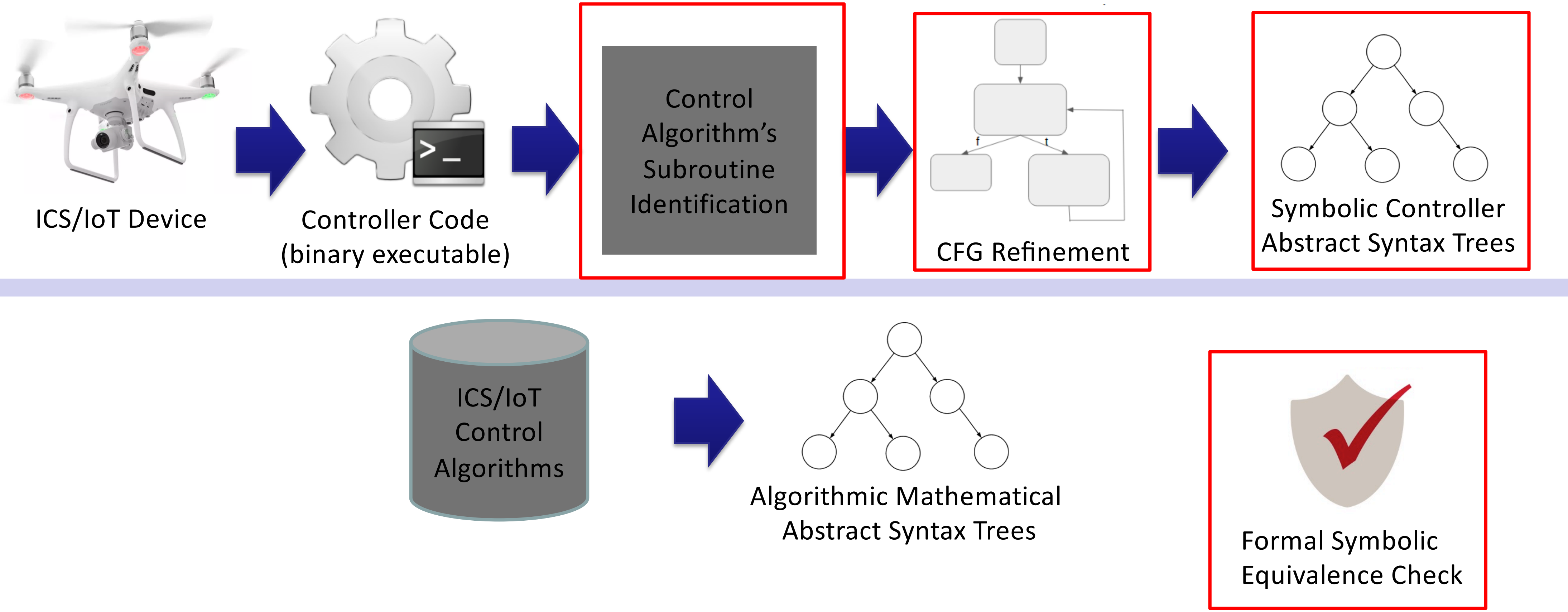
Algorithmic Mathematical
Abstract Syntax Trees

Formal Symbolic
Equivalence Check



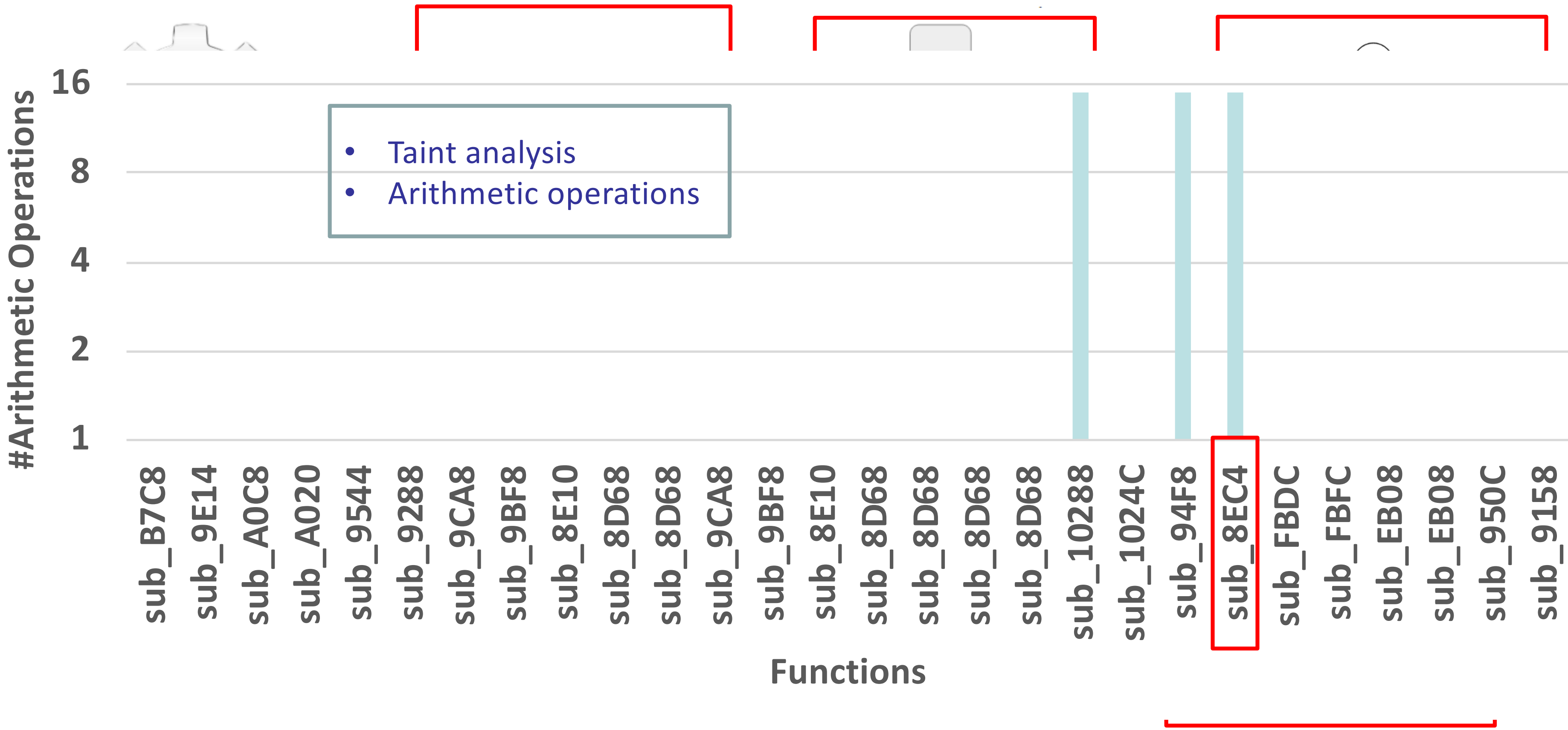
Annotated
Disassembly

Mismo Overview



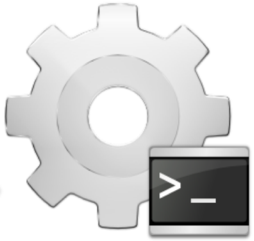
Case Study

ICS/IoT Device

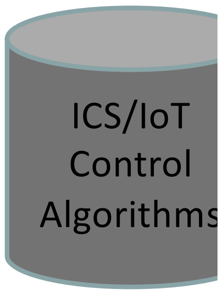


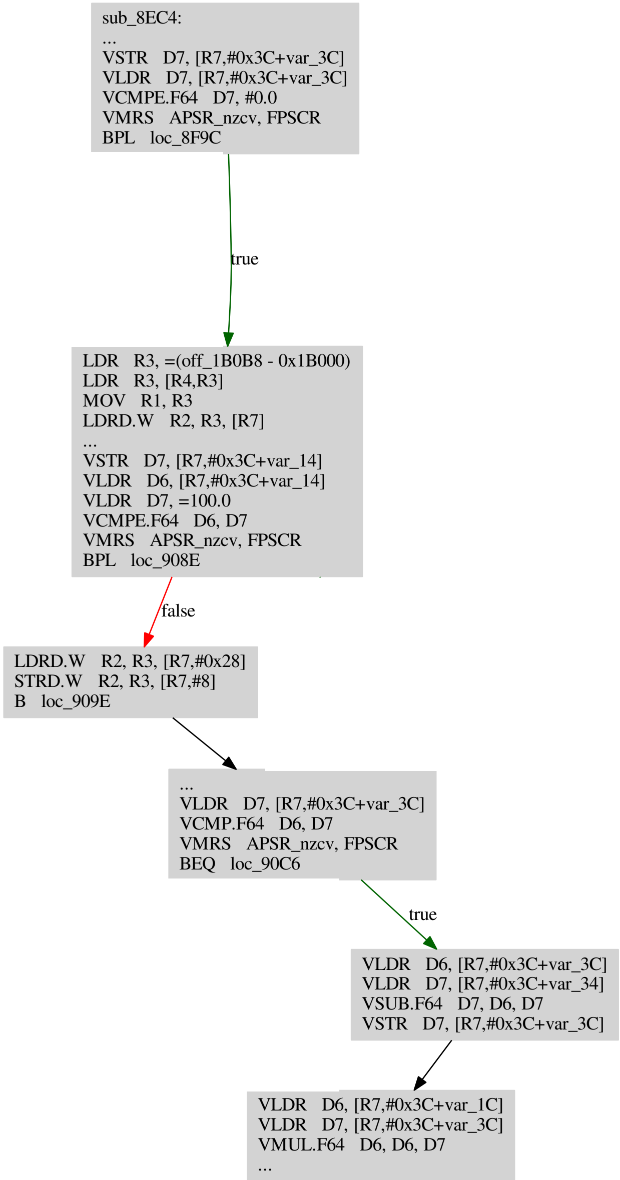
Case Study

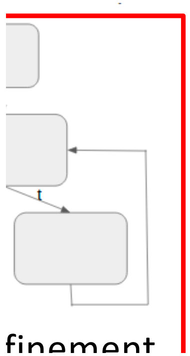
ICS/IoT Device



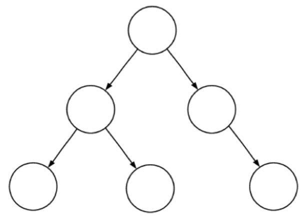
Controller Code
(binary executable)







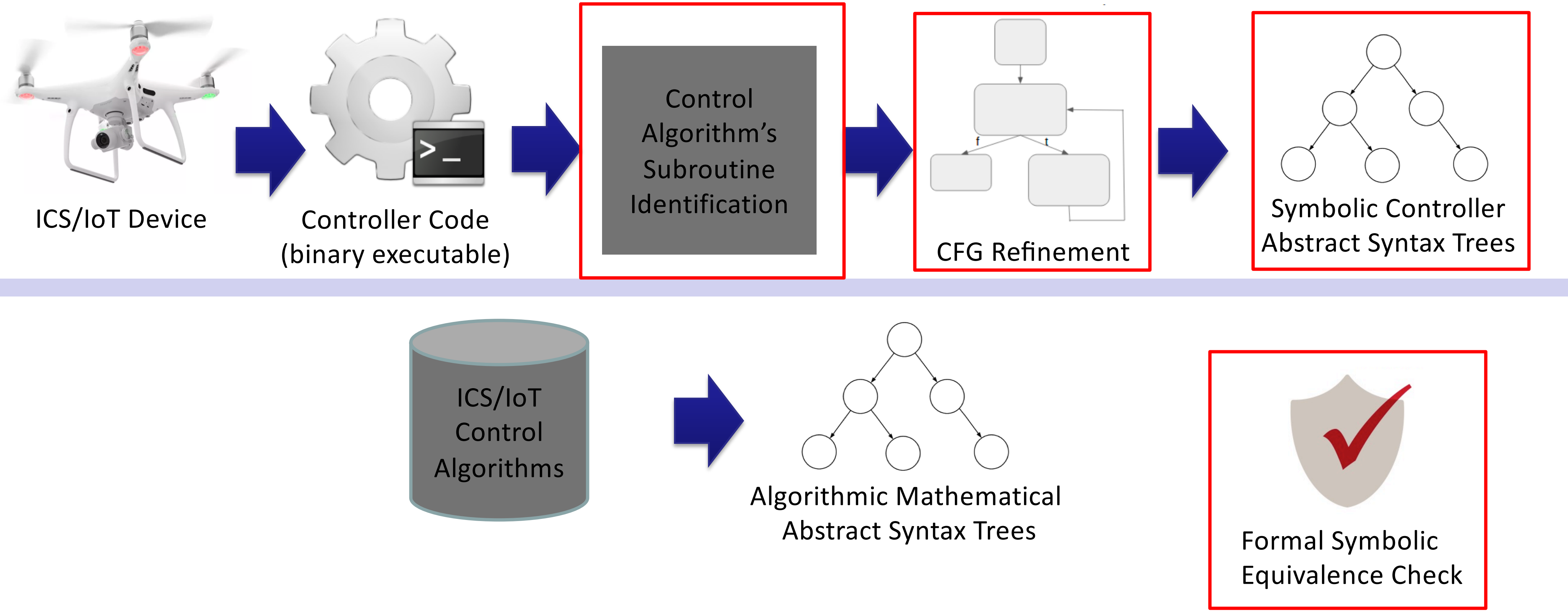
ical
s



Symbolic Controller
Abstract Syntax Trees

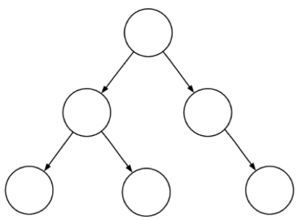
Formal Symbolic
Equivalence Check

Case Study


ICS/IoT Device
Controller Code
(binary executable)
Control Algorithm’s Subroutine Identification
CFG Refinement
Symbolic Controller Abstract Syntax Trees
ICS/IoT Control Algorithms
Algorithmic Mathematical Abstract Syntax Trees
Formal Symbolic Equivalence Check

Case Study

Symbolic expression



Symbolic Controller
Abstract Syntax Trees

Symbolic execution

Symbolic output variable

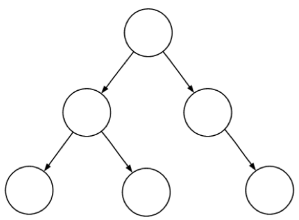
```
…
0x90DE: VMUL.F64   D6, D6, D7
        -> 0xc0175999f54482ee
0x90E2: VLDR   D7, [R7,#0x3C+var_2C]
        -> 0x4028000000000000
0x90E6: VADD.F64   D7, D6, D7
        -> 0x4018a6660abb7d12
…
```

Symbolic input variables

Backward slicing analysis

Case Study

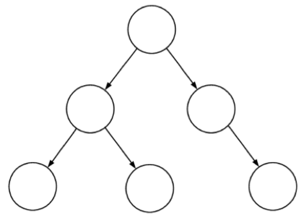
Symbolic expression



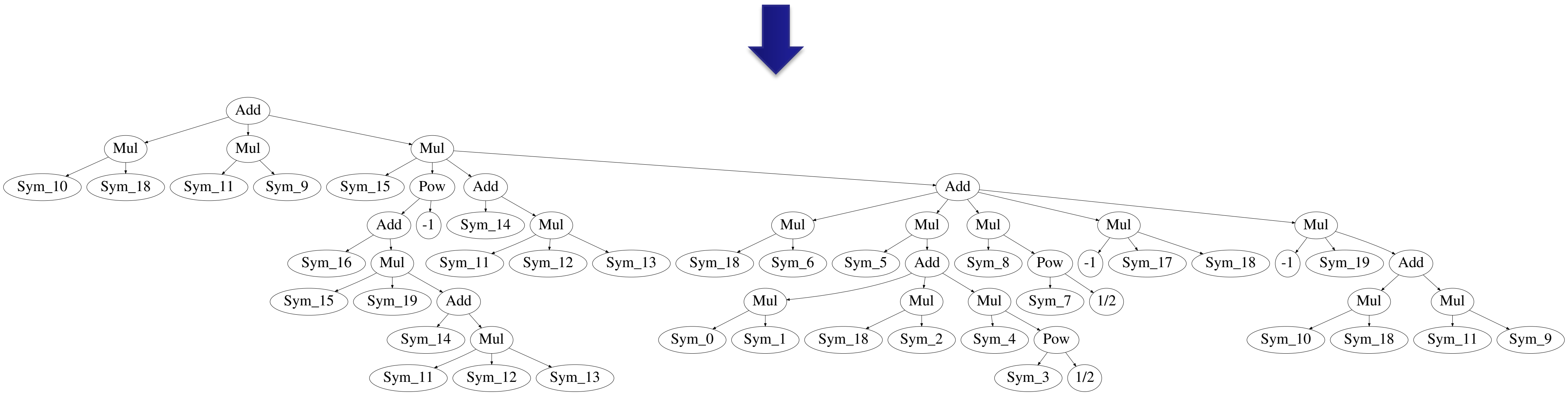
Symbolic Controller
Abstract Syntax Trees

((((Sym_15 * (((Sym_12 * Sym_11) * Sym_13) +
Sym_14)) / ((Sym_19 * (Sym_15 * (((Sym_12 * Sym_11) *
Sym_13) + Sym_14))) + Sym_16)) * ((((Sym_7) ^ (-1) *
Sym_8) + ((Sym_5 * (((Sym_3) ^ (-1) * Sym_4) +
((Sym_0 * Sym_1) + (Sym_18 * Sym_2)))) + (Sym_18 *
Sym_6))) – ((Sym_19 * ((Sym_18 * Sym_10) + (Sym_11*
Sym_9))) + (Sym_17 * Sym_18)))) + ((Sym_18 * Sym_10)
+ (Sym_11 * Sym_9)))

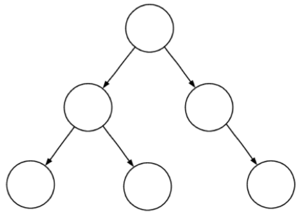
Case Study



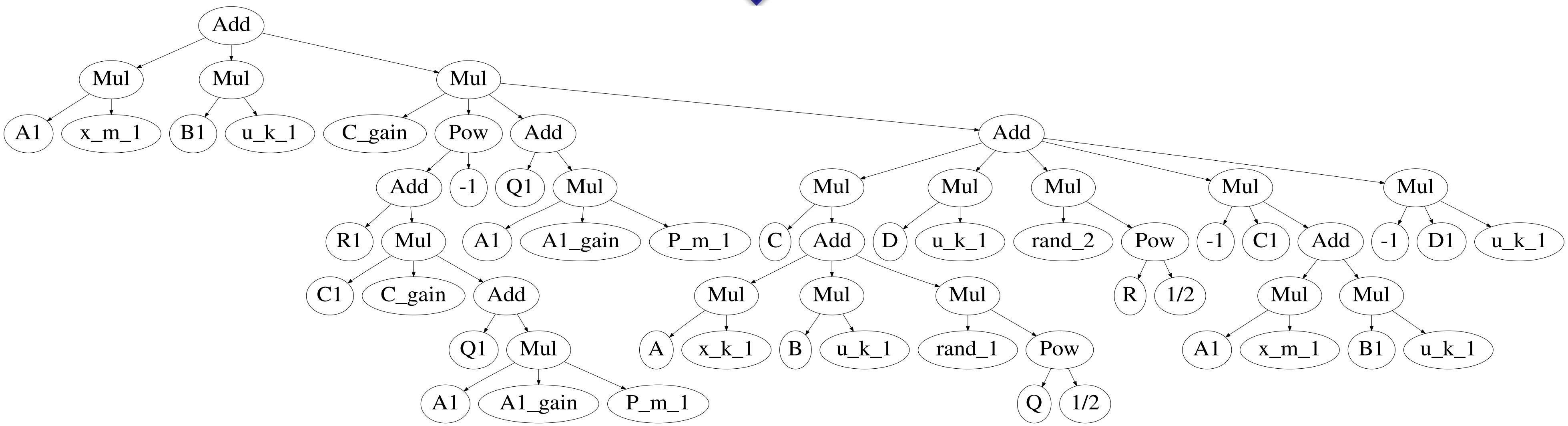
Symbolic Controller
Abstract Syntax Trees



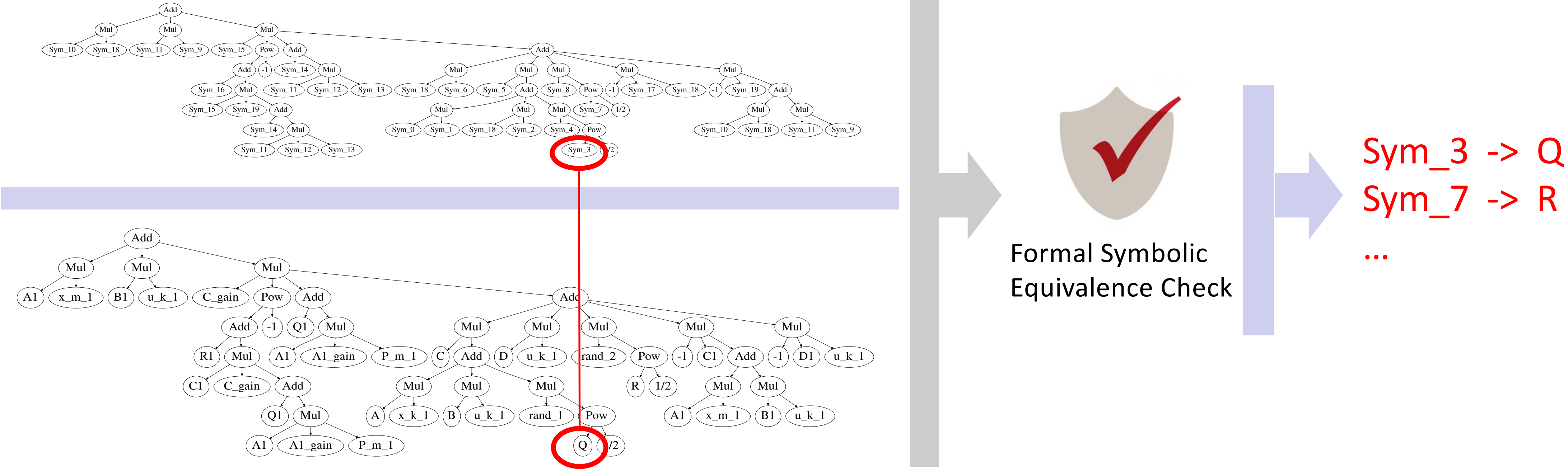
Case Study



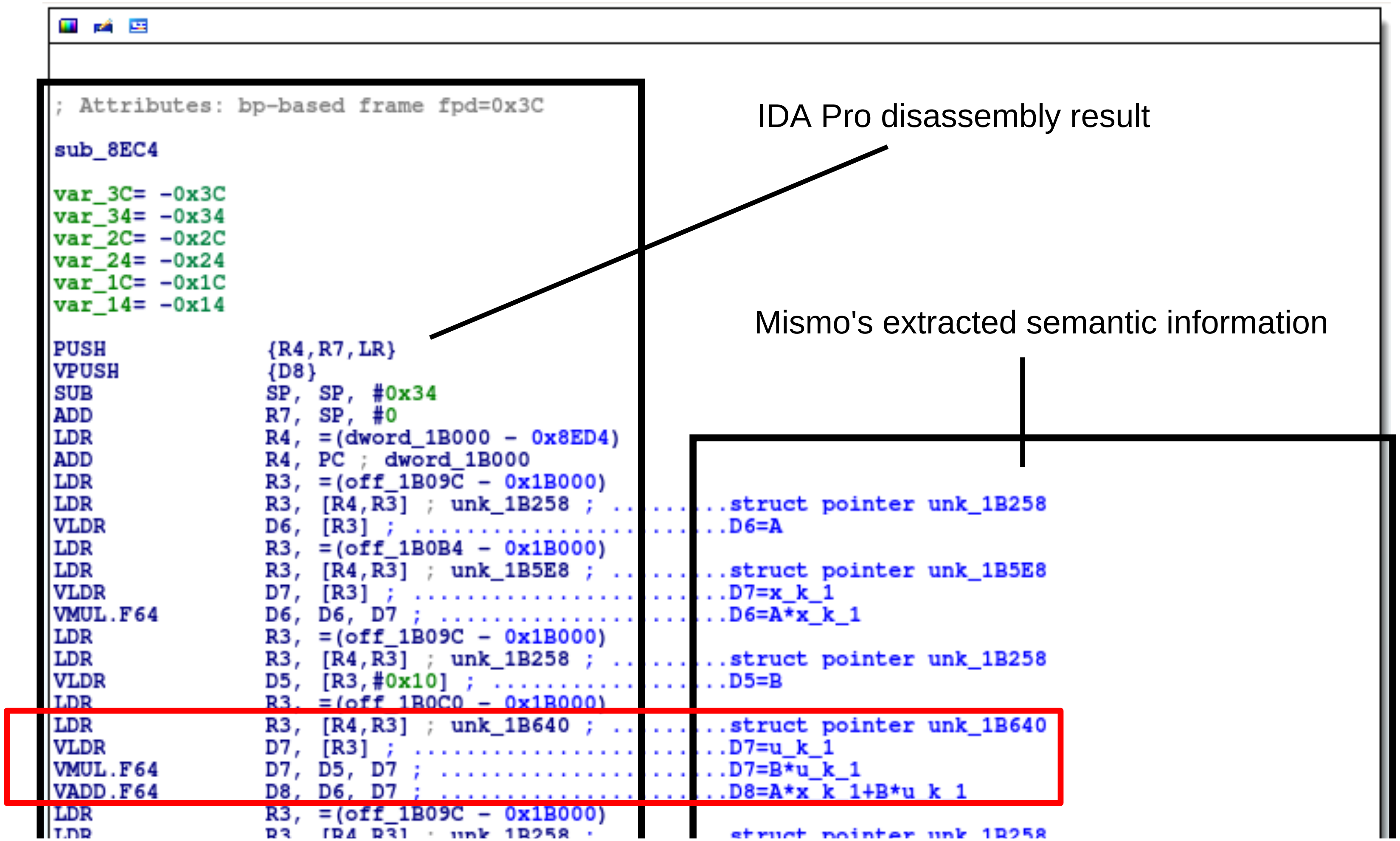
Algorithmic Mathematical
Abstract Syntax Trees



Mismo Overview



Annotated Disassembly



Evaluation

c variables

Accuracy

:ions

Category	Vendor	Control Algorithm					#firmwares	Data Accuracy (%)	Code Accuracy (%)
		BB	KF	PF	PID	PWM			
Drone	Bitcraze		✓		✓	✓	38	100.00	96.40
	Ardupilot		✓		✓	✓	168	78.57	86.96
	DJI		✓		✓	✓	66	100.00	93.69
	3D Robotics		✓		✓	✓	327	78.57	86.96
	Cleanflight				✓		48	71.43	50.26
	Fluoreon				✓		1	77.78	48.70
	Eachine				✓		1	77.78	48.70
	Paparazzi				✓		53	77.78	86.14
	Cheerson		✓		✓	✓	169	84.29	91.56
Automotive	Baidu		✓		✓		2	100.00	93.67
	PolySync				✓		3	100.00	97.01
	Microsoft				✓		1	100.00	100.00
	Tier IV		✓		✓		11	100.00	89.47
	Udaticy		✓	✓	✓		2	100.00	97.14
3D Printer	LulzBot	✓					22	90.91	92.86
	Makerbot				✓		19	88.89	63.81
	Repetie	✓			✓	✓	6	100.00	82.96
	Printrbot	✓			✓		22	90.91	92.86
	BCN3D	✓			✓		15	81.82	50.26
	Robo3D				✓		1	90.91	92.86
	Teacup	✓			✓	✓	1	100.00	93.24
	Solidoodle				✓		2	90.91	92.86
Robotics	ROS		✓	✓	✓	✓	62	88.89	94.20
	Robotiq				✓		1	100.00	98.64
	LinuxCNC				✓		145	53.85	43.34
	Drake		✓		✓		8	85.71	87.38
Smart Home	SmartPID				✓		2	100.00	100.00
	Particle				✓		87	100.00	96.81
	MBED				✓	✓	147	100.00	100.00
Linux Kernel	Linux Kernel				✓		833	100.00	100.00
Total/Average	**30**						**2,263**	**89.82**	**84.96**

Potential Use-cases

Potential Use-cases

- Binary vulnerability assessment

Potential Use-cases

- Binary vulnerability assessment
- Memory forensics analysis

Potential Use-cases

- Binary vulnerability assessment
- Memory forensics analysis
- Sensitive code and data segment protection

Potential Use-cases

- Binary vulnerability assessment
- Memory forensics analysis
- Sensitive code and data segment protection
- Correct algorithm implementation verification

Potential Use-cases

- Binary vulnerability assessment
- Memory forensics analysis
- Sensitive code and data segment protection
- Correct algorithm implementation verification
- Binary-level software similarity measures

Compare with Snowman

Source Code	Snowman Reversed Result	MISMO Reversed Result

`typedef struct`
`{`
`    double windup_guard;`
`    double proportional_gain;`
`    double integral_gain;`
`    double derivative_gain;`
`    double prev_input;`
`    double int_error;`
`    double control;`
`    double prev_steering_angle;`
`} PID;`
	`signed int v6; // r3@2`
	`double v19; // [sp+0h] [bp+0h]@1`
	`double v20; // [sp+8h] [bp+8h]@8`
	`int v21; // [sp+1Ch] [bp+1Ch]@1`
		`struct`
		`{`
		`    0: double SymVar;`
		`    8: double Kp;`
		`    10: double Ki;`
		`    18: double Kd;`
		`    20: double prev_measured_value;`
		`    28: double integral;`
		`    30: double output;`
		`}`

`diff = ((input - pid->`
`    prev_input)/dt);`
	`_R3 = v21;`
	`__asm`
	`{`
	`    VLDR D7, [R3,#0x20]`
	`    VLDR D6, [R7,#0x4C+var_44]`
	`    VSUB.F64 D6, D6, D7`
	`    VLDR D7, [R7,#0x4C+var_4C]`
	`    VDIV.F64 D7, D6, D7`
	`    VSTR D7, [R7,#0x4C+var_24]`
	`}`
		`reg_D6 = measured_value`
		`    - previous_measured_value;`
		`reg_D7 = reg_D6/dt;`

Linux Kernel Bug

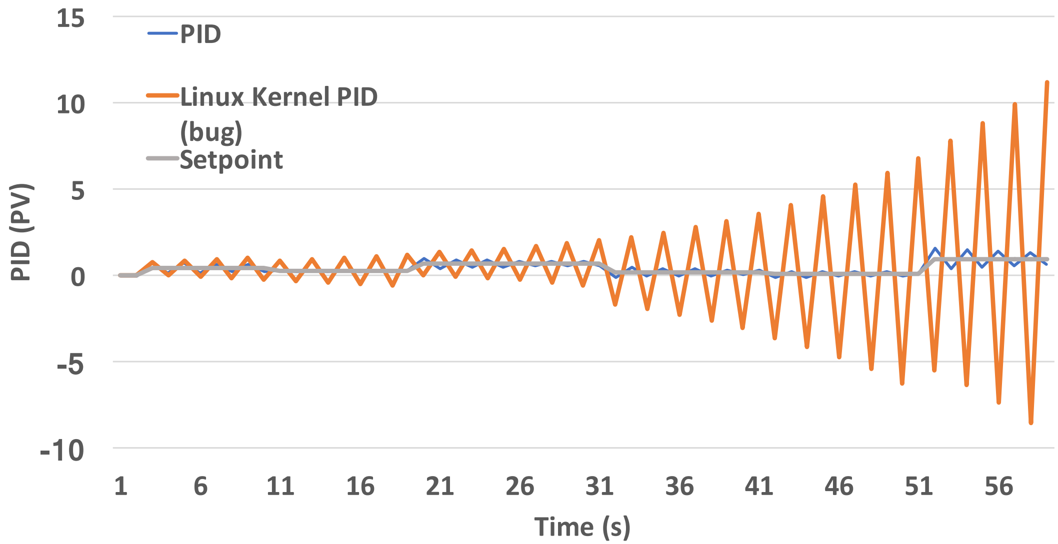
$$y_k = y_{k-1} - K_p * (x_k - x_{k-1}) + K_i * T_s * e_k$$
$$-K_d * (x_k - 2 * x_{k-1} + x_{k-2})/T_s$$

Algorithm

Mismo

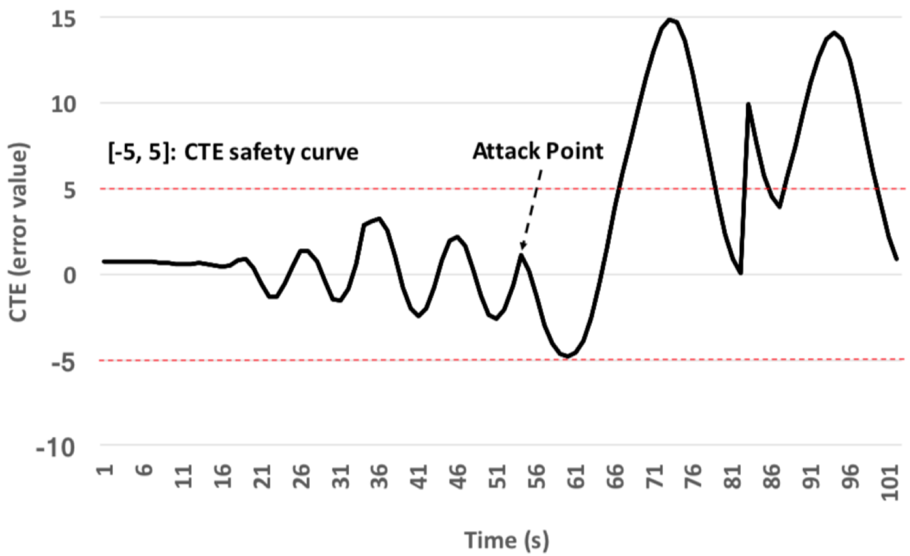
Program Code

```
…
/* compute intermediate PID terms */
p_term = -p_param.kp * (xk - xk_1);
i_term = p_param.kp * p_param.ki * p_param.ts * ek;
d_term = -p_param.kp * p_param.kd * (xk - 2 * xk_1
            +xk_2) / p_param.ts;

/* compute output */
*yk += p_term + i_term + d_term;
…
```



Attack Control Algorithm

(a) Car crash visualization using the autonomous controller.



(b) Modified gain parameter of the controller causes the crash.

Conclusions & QA

- A general framework to extract semantic information of an embedded firmware binaries with respect to its associated high-level control algorithm.
- Using dynamic binary analysis and symbolic comparison of the mathematical and binary expressions to fill the semantic gap between high-level algorithm descriptions and low-level stripped binary segments.

Thank You

Questions?

Email: **garcialuis@ucla.edu**

PID

$$u(t) = \mathrm{MV}(t) = K_\mathrm{p} e(t) + K_\mathrm{i} \int_0^t e(\tau)\, d\tau + K_\mathrm{d} \frac{de(t)}{dt},$$

where

K_p is the proportional gain, a tuning parameter,

K_i is the integral gain, a tuning parameter,

K_d is the derivative gain, a tuning parameter,

$e(t) = \mathrm{SP} - \mathrm{PV}(t)$ is the error (SP is the setpoint, and PV(t) is the process variable),

t is the time or instantaneous time (the present),

τ is the variable of integration (takes on values from time 0 to the present t).

Kalman Filter

Predict [edit]

Predicted (*a priori*) state estimate	$\hat{\mathbf{x}}_{k\mid k-1} = \mathbf{F}_k \hat{\mathbf{x}}_{k-1\mid k-1} + \mathbf{B}_k \mathbf{u}_k$
Predicted (*a priori*) error covariance	$\mathbf{P}_{k\mid k-1} = \mathbf{F}_k \mathbf{P}_{k-1\mid k-1} \mathbf{F}_k^\mathsf{T} + \mathbf{Q}_k$

Update [edit]

Innovation or measurement pre-fit residual	$\tilde{\mathbf{y}}_k = \mathbf{z}_k - \mathbf{H}_k \hat{\mathbf{x}}_{k\mid k-1}$
Innovation (or pre-fit residual) covariance	$\mathbf{S}_k = \mathbf{R}_k + \mathbf{H}_k \mathbf{P}_{k\mid k-1} \mathbf{H}_k^\mathsf{T}$
Optimal Kalman gain	$\mathbf{K}_k = \mathbf{P}_{k\mid k-1} \mathbf{H}_k^\mathsf{T} \mathbf{S}_k^{-1}$
Updated (*a posteriori*) state estimate	$\hat{\mathbf{x}}_{k\mid k} = \hat{\mathbf{x}}_{k\mid k-1} + \mathbf{K}_k \tilde{\mathbf{y}}_k$
Updated (*a posteriori*) estimate covariance	$\mathbf{P}_{k\mid k} = (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k) \mathbf{P}_{k\mid k-1} (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k)^\mathsf{T} + \mathbf{K}_k \mathbf{R}_k \mathbf{K}_k^\mathsf{T}$
Measurement post-fit residual	$\tilde{\mathbf{y}}_{k\mid k} = \mathbf{z}_k - \mathbf{H}_k \hat{\mathbf{x}}_{k\mid k}$

The formula for the updated (*a posteriori*) estimate covariance above is valid for any gain $\mathbf{K}_k$ and is sometimes called the **Joseph form**. For the optimal Kalman gain the formula further simplifies to $\mathbf{P}_{k\mid k} = (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k) \mathbf{P}_{k\mid k-1}$, in which form it is most widely used in applications. However, one must keep in mind, that it is valid only for the optimal gain that minimizes the residual error. Proof of the formulae is found in the *derivations* section.