WASHINGTON STATE
UNIVERSITY

Cyber-physical system recovery from sensor attacks

Brief Outline

- My bio
- What are cyber-physical systems (CPS)?
- What are sensor attacks?
- Why sensor attacks are critical to CPS?
- How to recover CPS from sensor attacks?
  - Model-based
  - Model-free
- Future directions

My bio

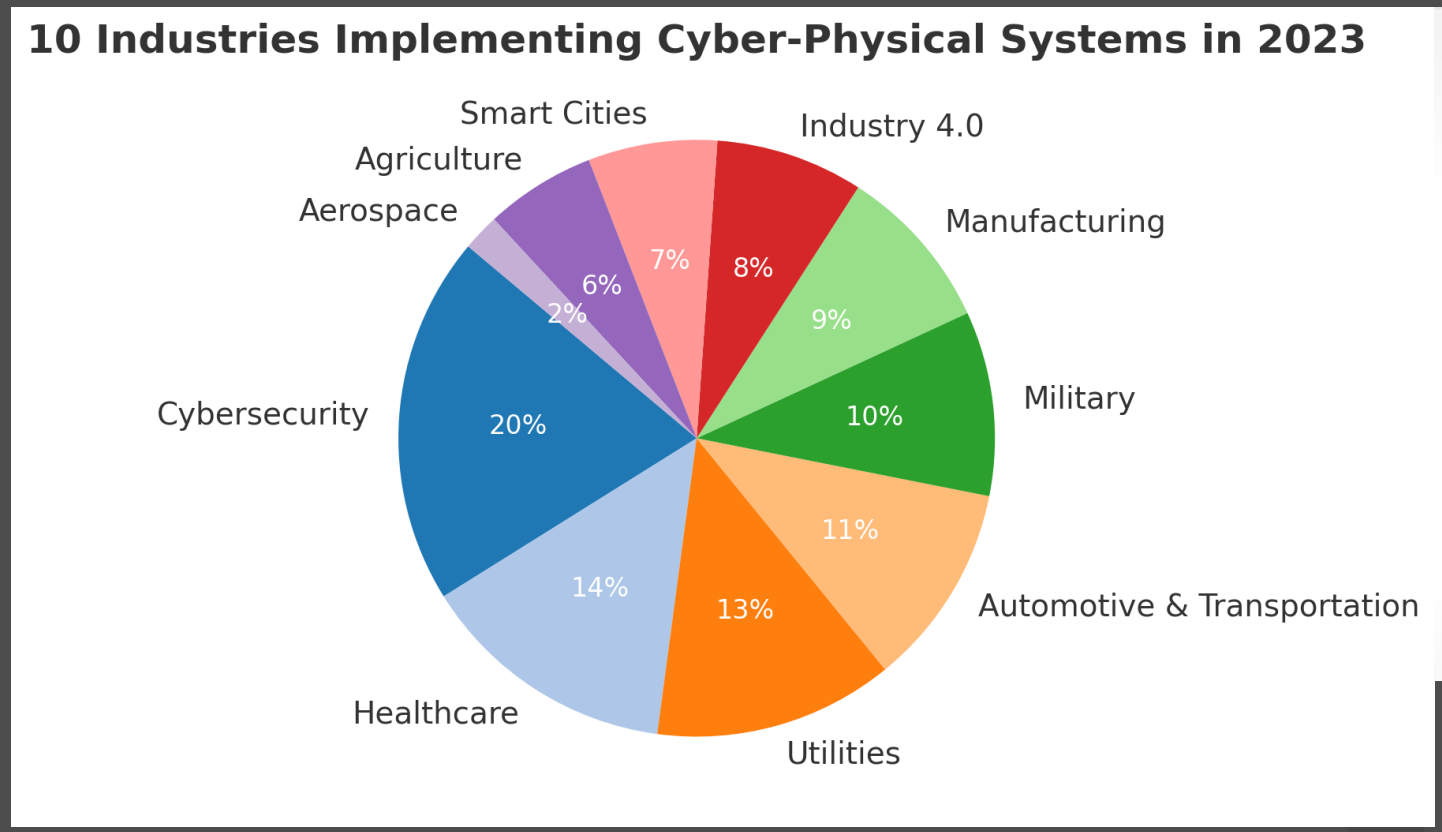
- Assistant professor of computer science and cybersecurity, WSU Tri-Cities
- PhD, University of Notre Dame
- MS, Syracuse University
- BS, University of Toronto

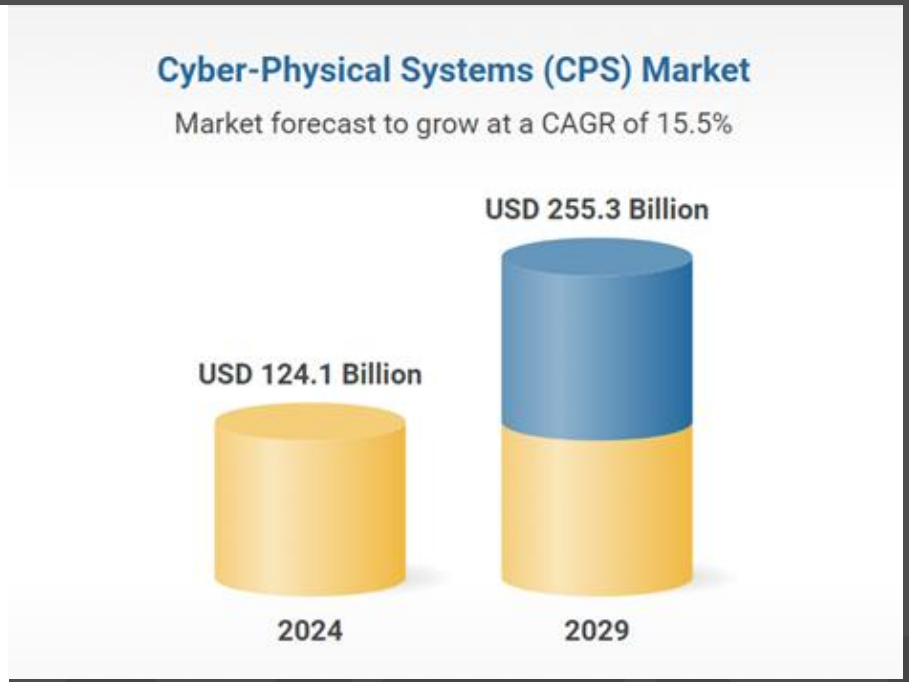
My research focuses on cyber-physical systems (CPS), including safety, security, real-time control, and foundation models. I am particularly interested in reinforcement learning for control synthesis with formal guarantees, and CPS security under adversarial settings.

What are Cyber-Physical Systems (CPS)?

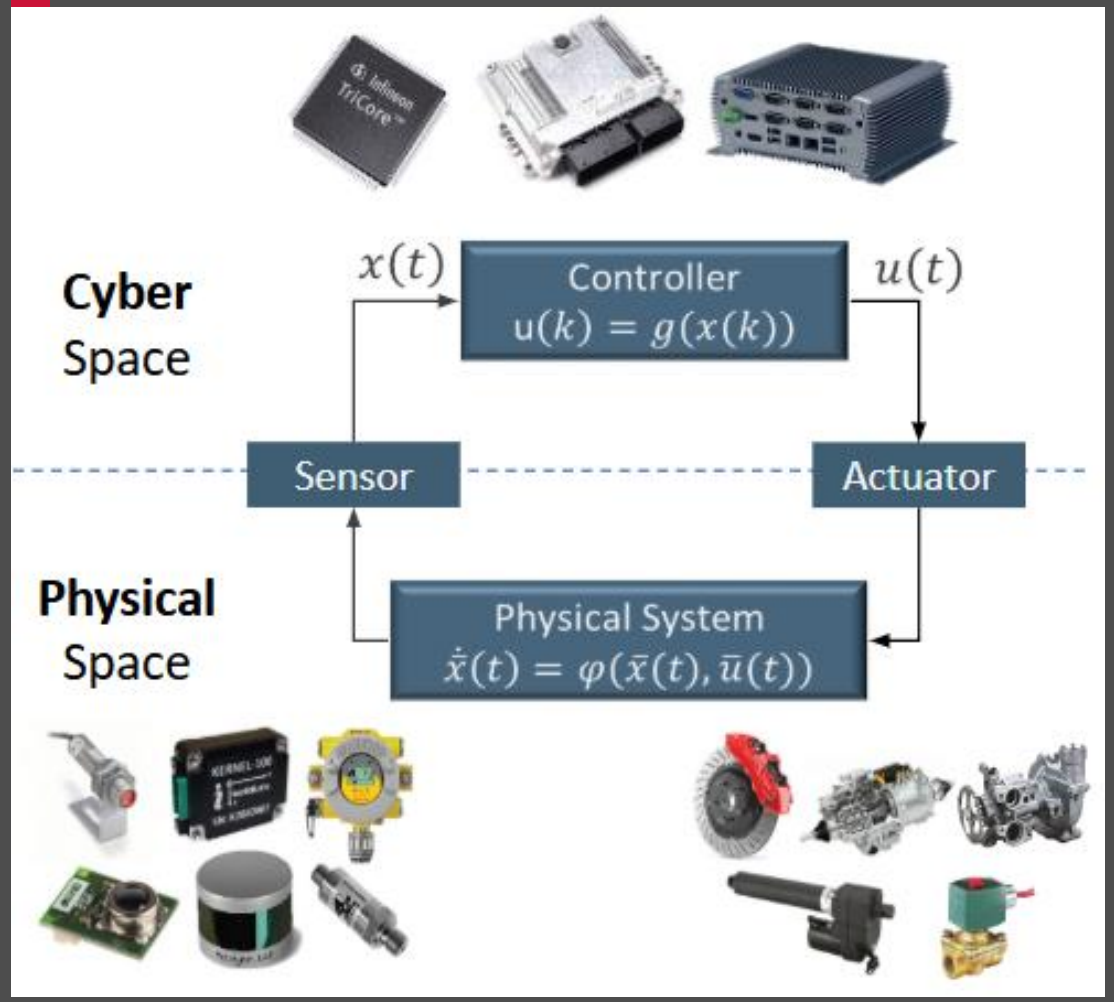
We are living in a Cyber-Physical System world!

CPS Market


10 Industries Implementing Cyber-Physical Systems in 2023
Smart Cities
Industry 4.0
Agriculture
Aerospace
Manufacturing
Military
Cybersecurity
Automotive & Transportation
Healthcare
Utilities
8%
9%
10%
11%
13%
14%
20%
2%
6%
7%

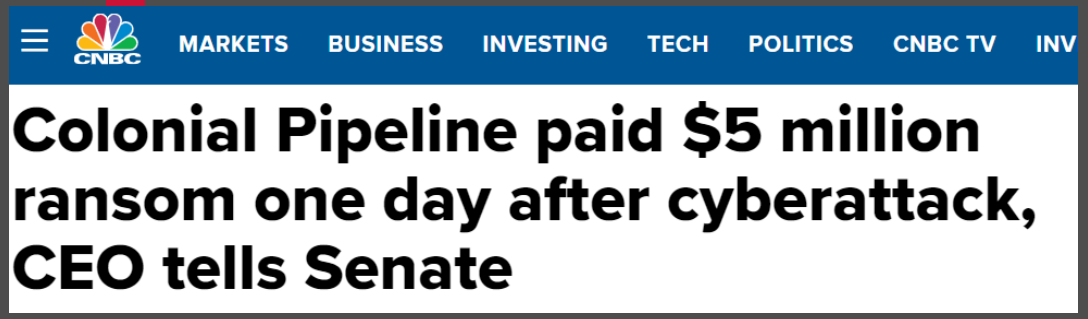

Cyber-Physical Systems (CPS) Market
Market forecast to grow at a CAGR of 15.5%
USD 255.3 Billion
USD 124.1 Billion
2024
2029

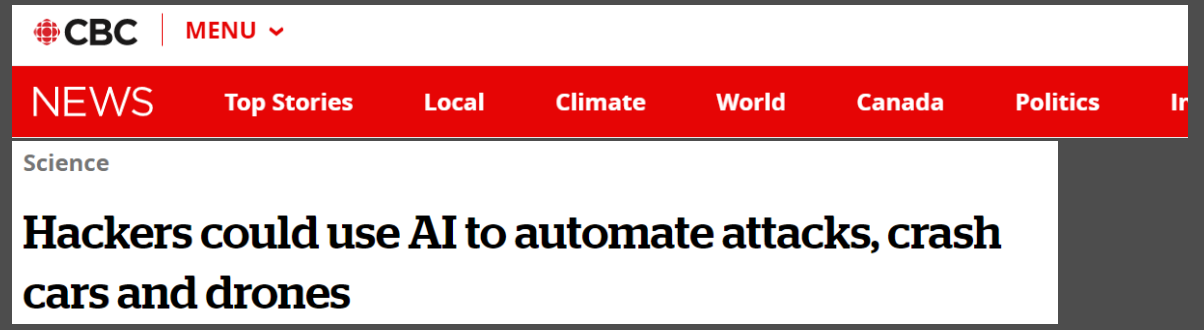
Core of CPS: The feedback Loop

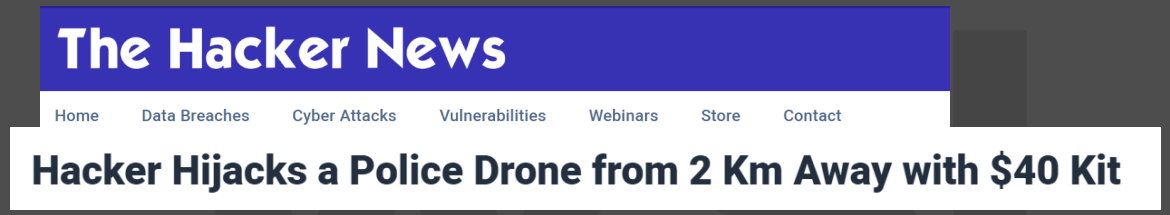


Why it is so important? Because it is automatic! And one cost of this automation is: any component in the system been attacked can cause physical failure!

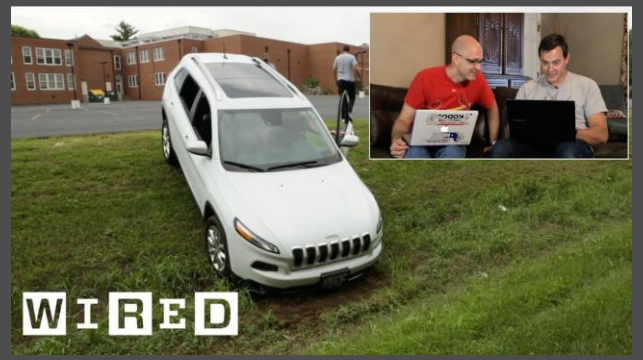
Let’s see some news…





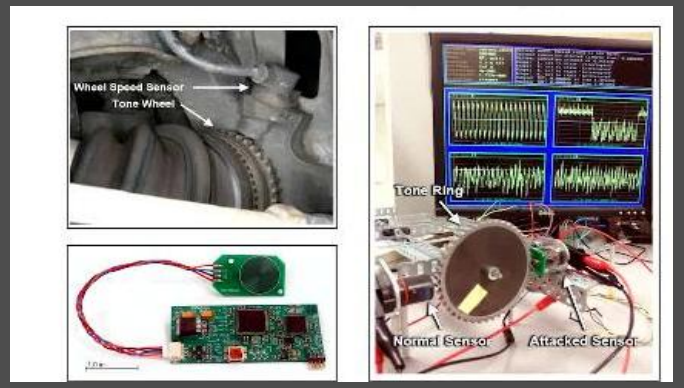


Sensor Attacks – Cyber-Physical Vulnerability

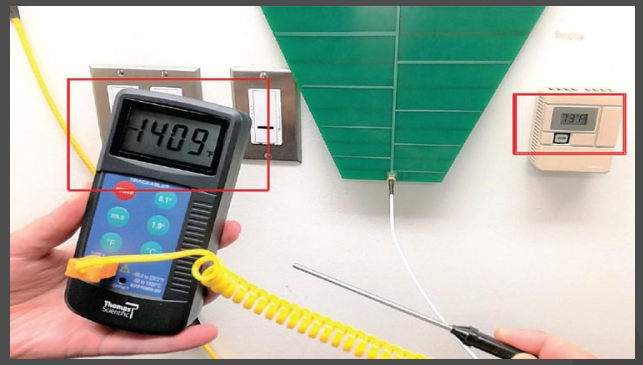


Spoofing GPS

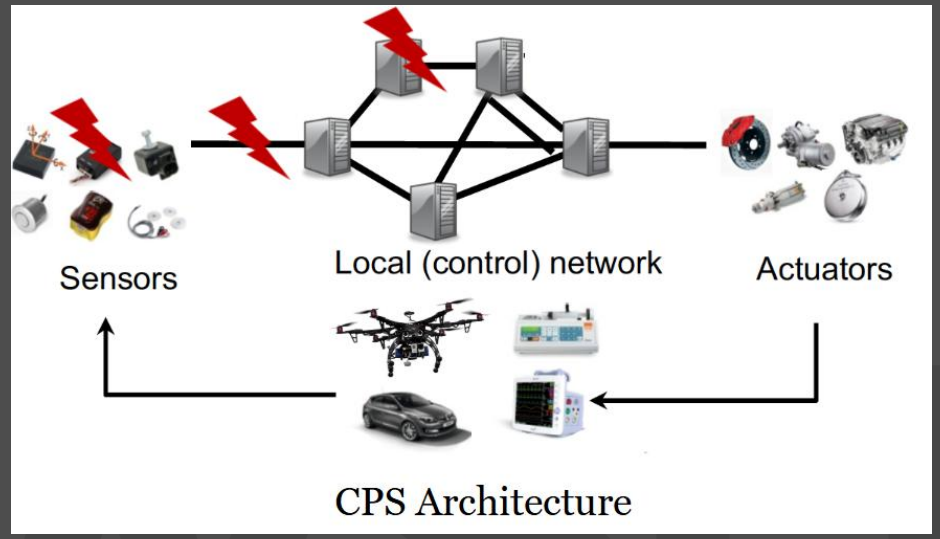
- Alters sensing information to interfere with the physical parts
- Compromises system safety



Attacking Speed Sensor



Attacking Thermocouple



CPS Architecture

What do we care? Safety?

- Of course we care about safety!
- But safety is not the only thing we care about!
- We can just do nothing and stop, that's safe but not what we want.

CPS Safety and Time

In CPS, safety and time are tightly coupled!

My vision is providing safety solutions for CPS in real-time scenarios.

- **When there are attacks**, if we don't make appropriate reaction to attacks in time, the system can be unsafe.
- **When there is no attack**, if we want to make a car to be very agile, sometimes it can be less safe.

Real-time Attack Recovery

- Existing works： Most focuses on attack detection
- Challenges:
  1. **Safety**: avoid unsafe states for recovery
  2. **Timeliness**: should be recovered before irreparable consequences occur

Goal: safely bring CPS back to a safe state in real-time

Reboot is not feasible for emergency

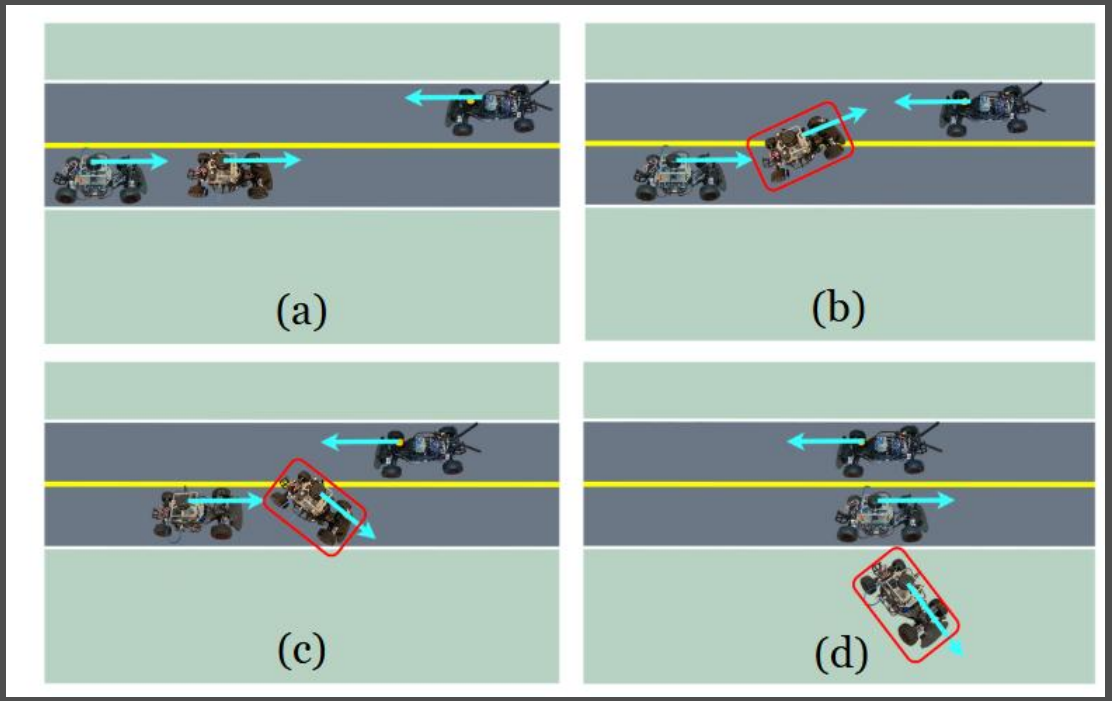
Model-based	Learning-based
[RTAS'23][TCPS'24]	[RTSS'23]

[Conferences][Journals]

Real-time Attack Recovery

How to Recover CPS after Detection of Attacks?

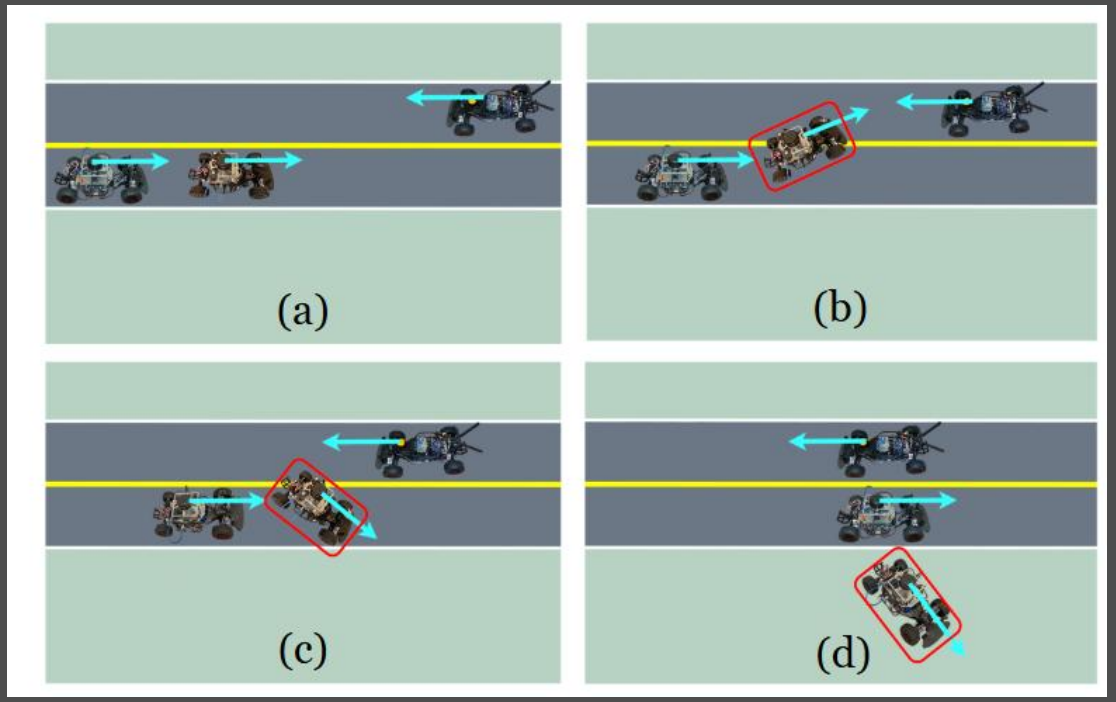
- need to stop drifting
- reduce caused negative impact



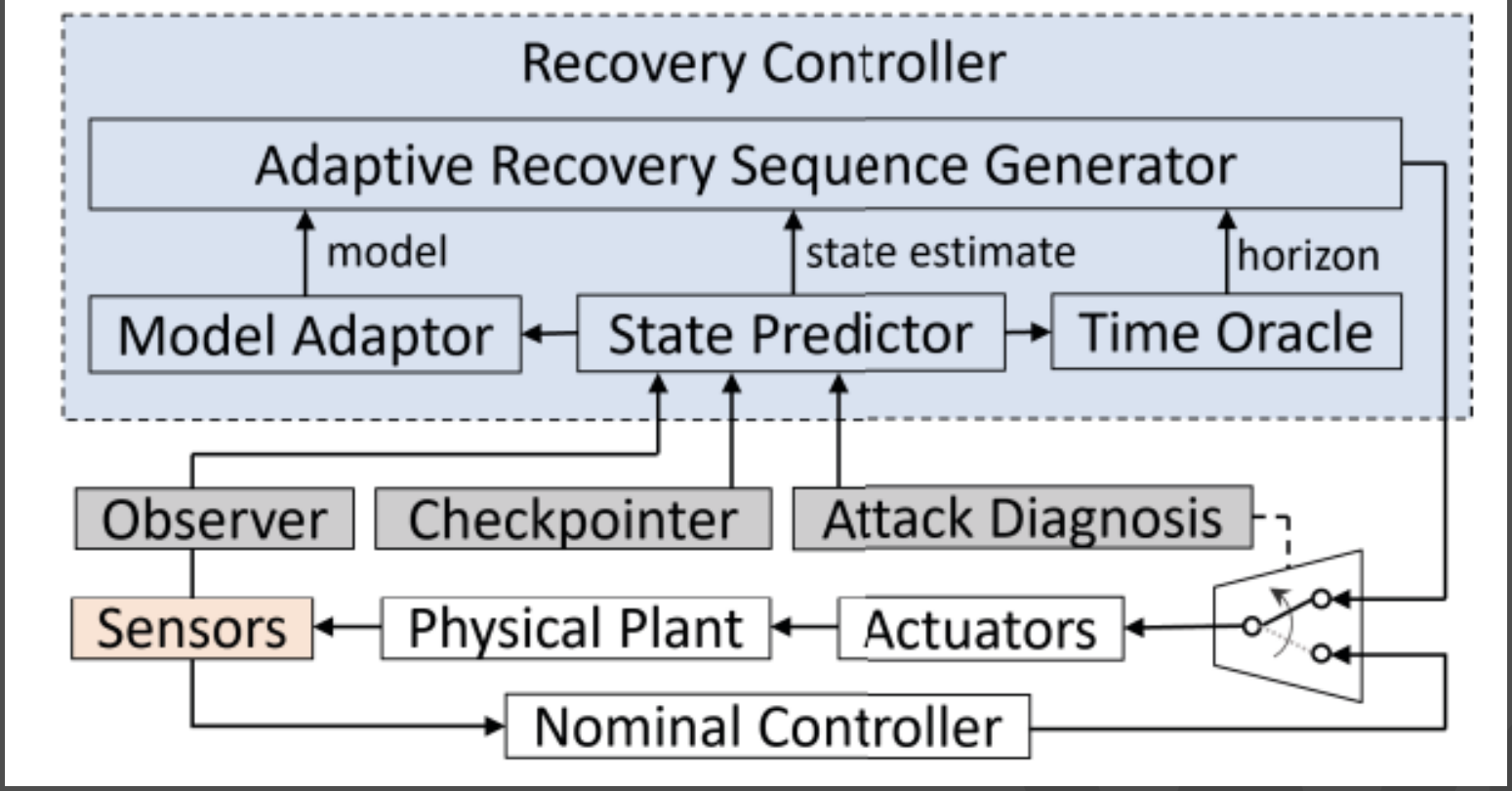
Real-time Attack Recovery

How to Recover CPS after Detection of Attacks?

- need to stop drifting
- reduce caused negative impact



Model-based recovery (RTAS'23)



The recovery workflow

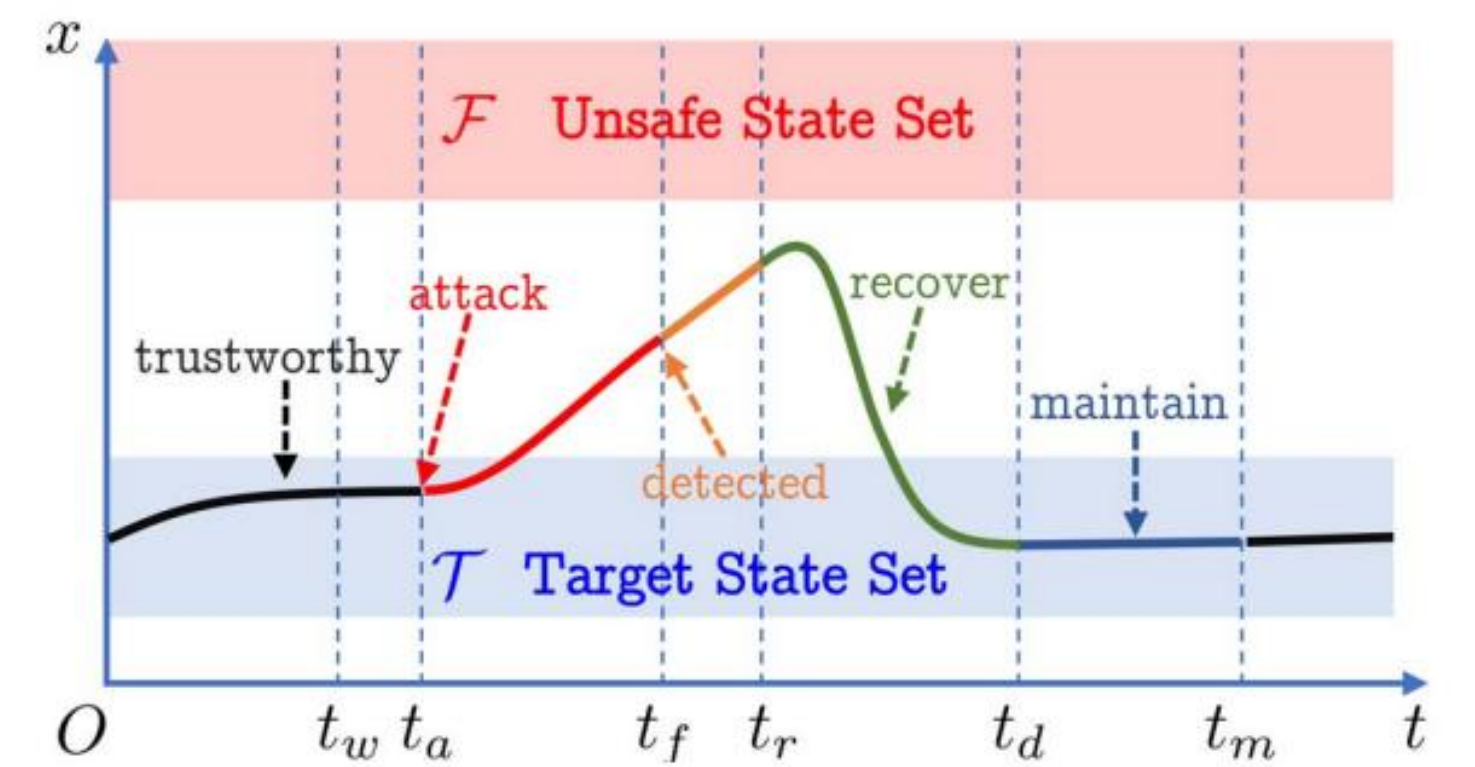
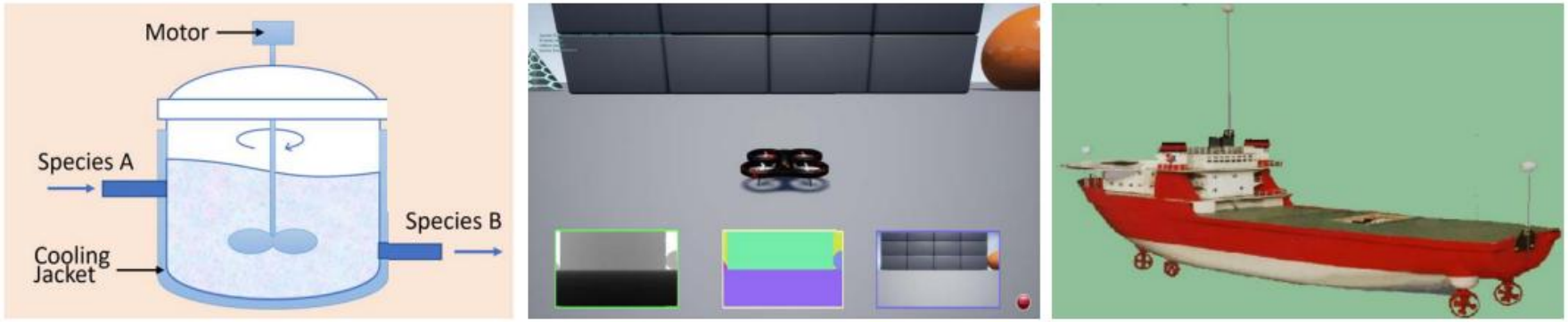


Fig. 3: Illustration of Recovery Timeline

Steps for the model-based recovery

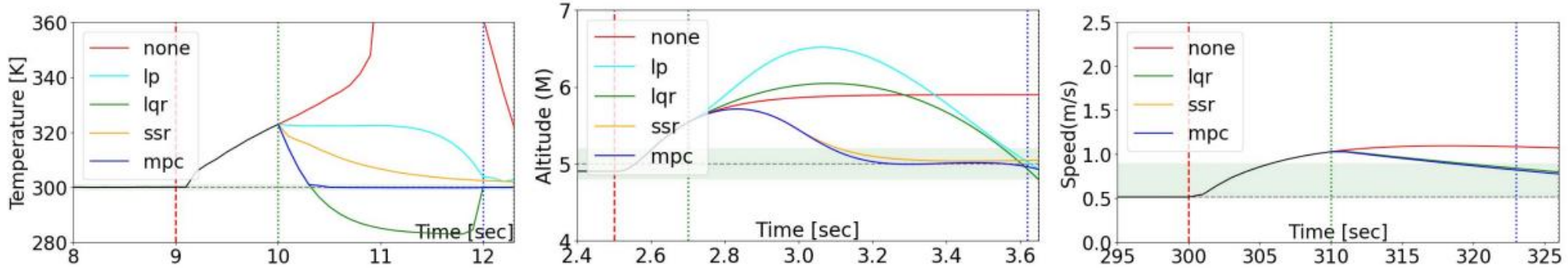
- Identifies local system model
  - Data-driven local system identification
- Using model-predictive control (MPC) to formulate the recovery problem
  - Only the first control is applied, then we reformulate the problem
- Apply the ADMM-based algorithm to solve the problem in real-time
  - Fast solution for online quadratic programming

Some Results



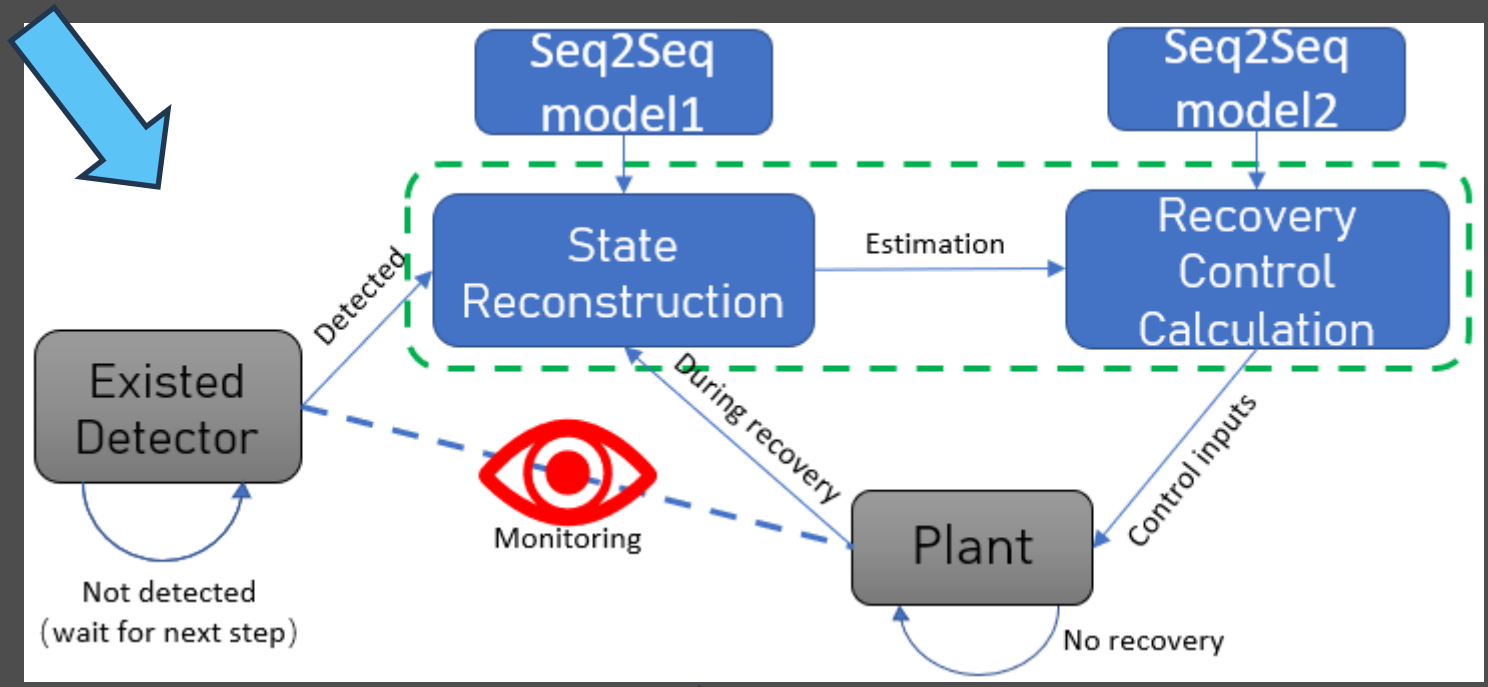
(a) CSTR (b) Drone from AirSim (c) Vessel model 1:70 scale [57]

Fig. 7: Numerical and High-Fidelity CPS Simulators



Model-free Recovery (RTSS'23)

The detector can tell which sensor has been attacked and when the attack started

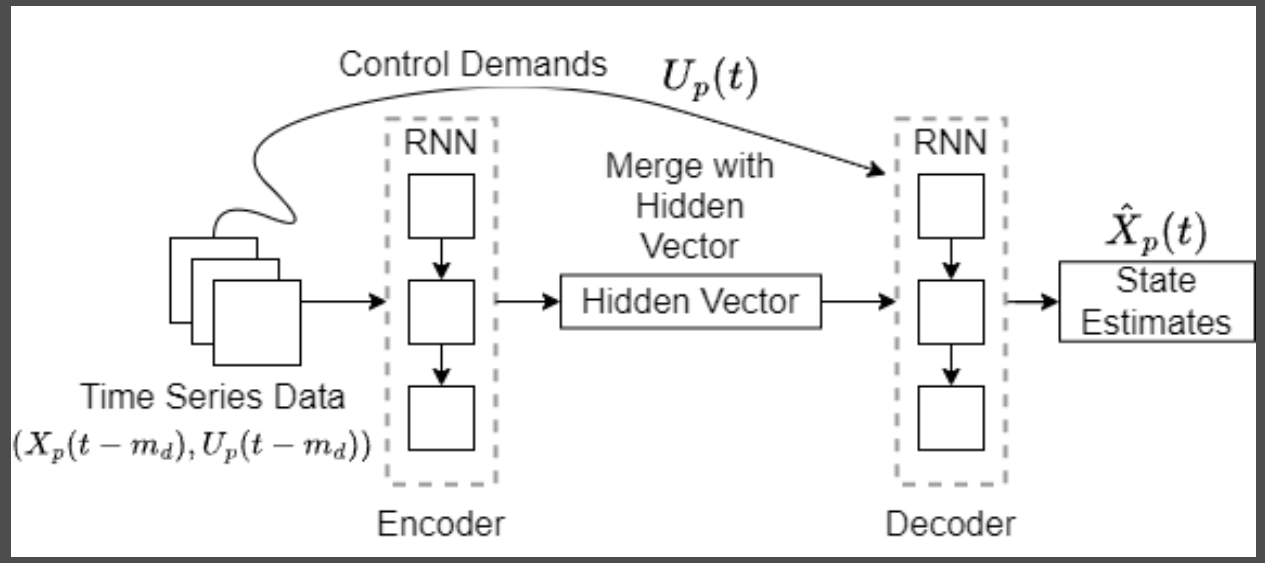


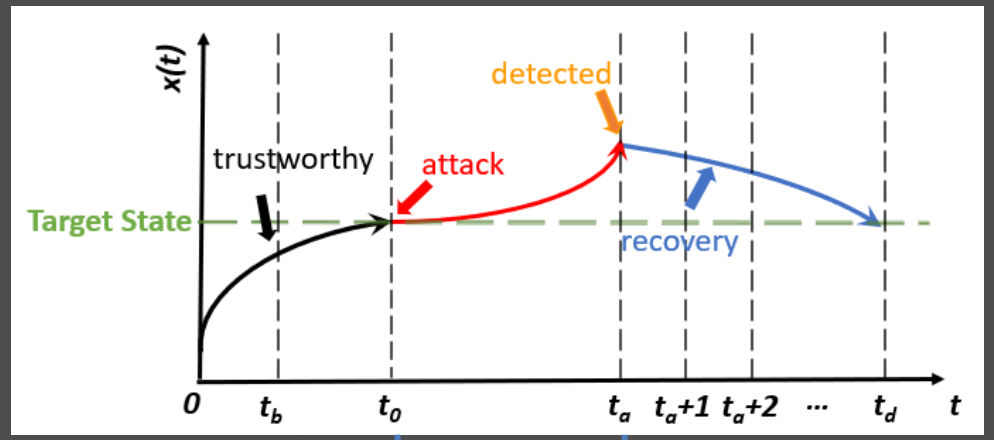
Model-based	Learning-based
[RTAS'23][TCPS'24]	[RTSS'23]

[Conferences][Journals]

There is an original controller control the system when there is no attack detected

State Reconstruction





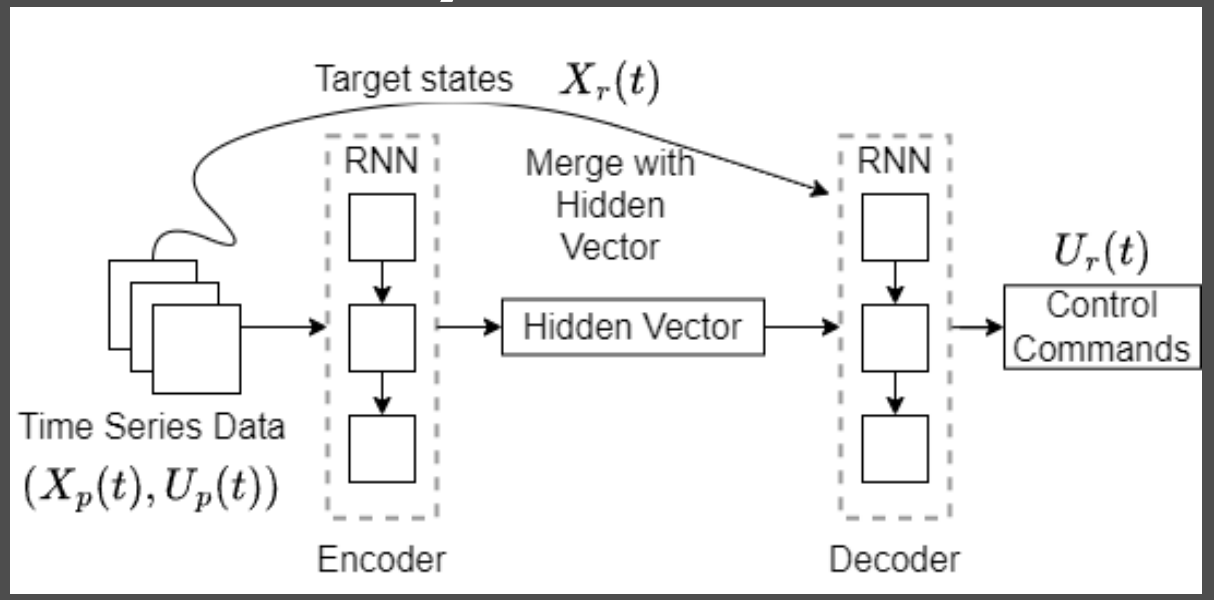
Reconstruct t_a

Reconstruct t_a+1

Reconstruct t_a+2

1. Encoder-Decoder Architecture
2. Reconstruct the state of the system iteratively
3. Keep running until the recovery is over(safe)
4. We don't collect attack samples for training because we can't make assumptions on attack patterns.

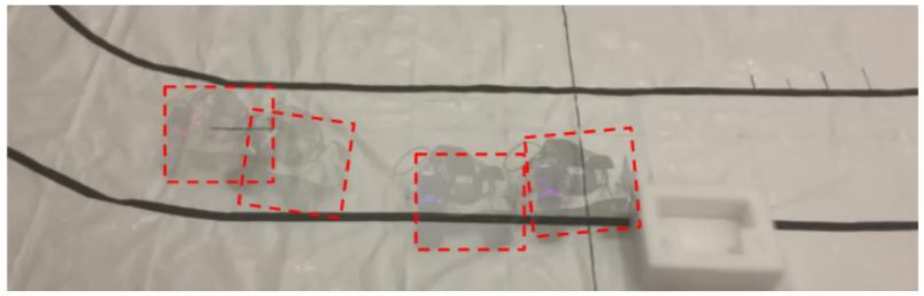
Recovery Control Calculation



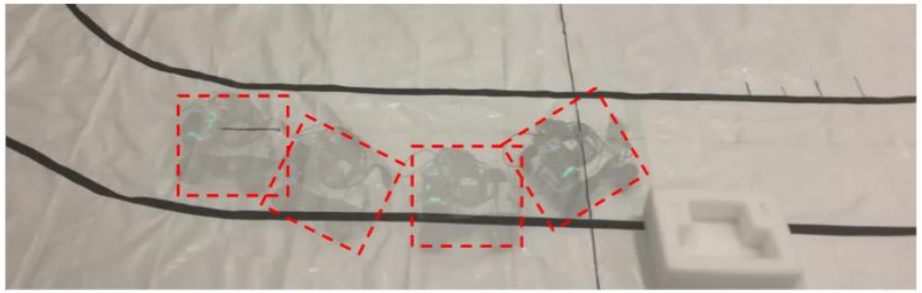
1. Encoder-Decoder Architecture(similar to state reconstruction)
2. Produce a control sequence iteratively, but only the first control is applied to the system
3. Keep running until the recovery is over(safe)

This recovery controller is more agile than the original controller

Testbed Demo



(a) Original Controller



(b) SeqRec

Fig. 9: Recovery Comparison for the 4-wheel testbed turning, using original controller will have a collision with an obstacle due to slow recovery

Untimely recovery is just as no recovery

Future Directions

- Recovery-guided attack detection
- In-context recovery

Recovery-guided attack detection

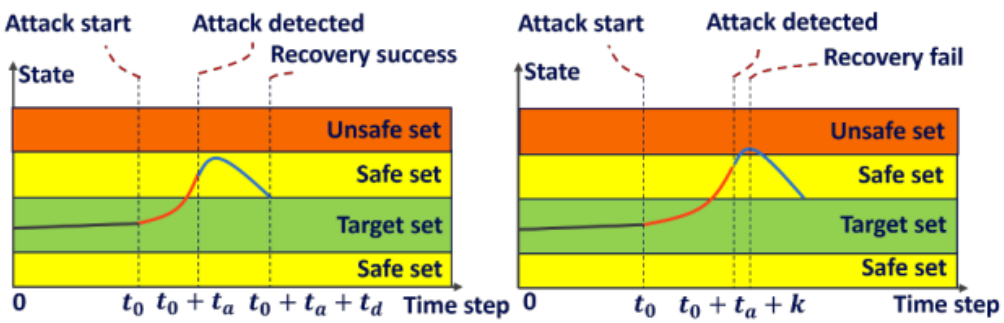
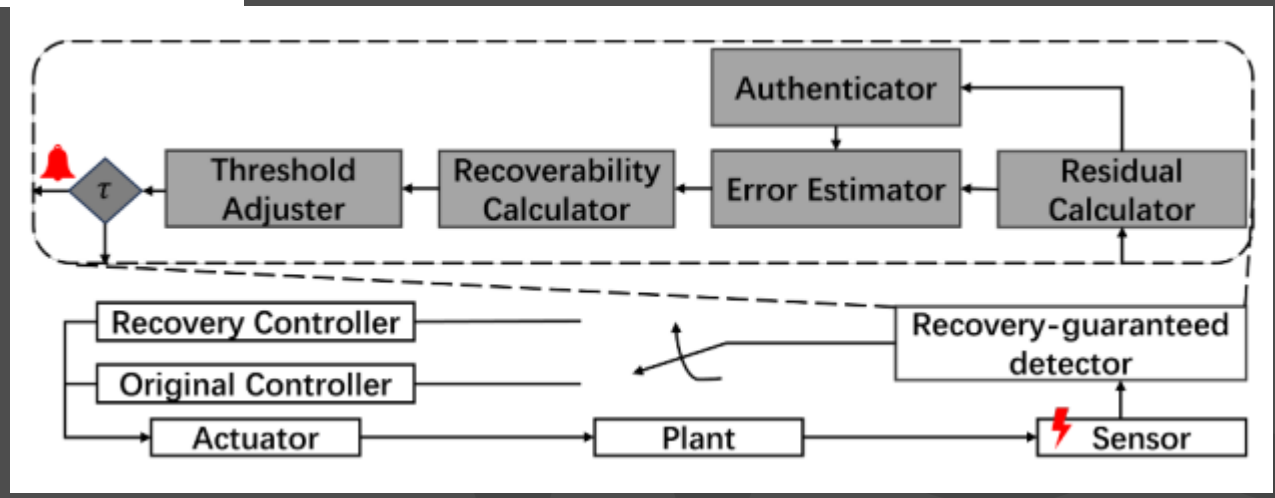


Fig. 1: A motivating example of the recovery-guaranteed detector



In-context Recovery

In different context, we should have different recovery control, how to make the model-free recovery have this adaptability?
Additionally, how to make sure the system is safe and stable if context switching during recovery?