The Cyber-Physical Immune System

Berkeley | EECS
Electrical Engineering and Computer Sciences

Bo Pang*, Ashank Verma*, Jingchao Zhou*, Inigo Incer, Alberto Sangiovanni-Vincentelli (*equal contribution) | {pangb, ashankv, jingchao_zhou, inigo, alberto}@berkeley.edu

Introduction

Cyber-Physical Systems (CPS) are important components of critical infrastructure and must operate with high levels of reliability and security.

We propose a conceptual approach to securing CPSs: the Cyber-Physical Immune System (CPIS), a collection of hardware and software elements deployed on top of a conventional CPS to help identify threats, deploy threat-mitigation strategies, and adapt to the changing environment.

Inspiration from biological immune system

The immune system of mammals is composed of elements that constantly monitor the body for unusual behavior and alert the central immune system of suspicious activity. The central immune system can generate responses to new threats.

Similarly, the CPIS comprises an independent network of distributed computing units that attach onto the existing CPS, as shown in *Figure 1:*

- **CPIS Monitors**: collect execution traces from the computing units present in the CPS, quickly finding known issues or threats using pre-programmed information
- **CPIS Main Computing Unit**: collects data from all monitors and detects anomalies with its system-wide point of view; using data-driven techniques, it learns to recognize new anomalies

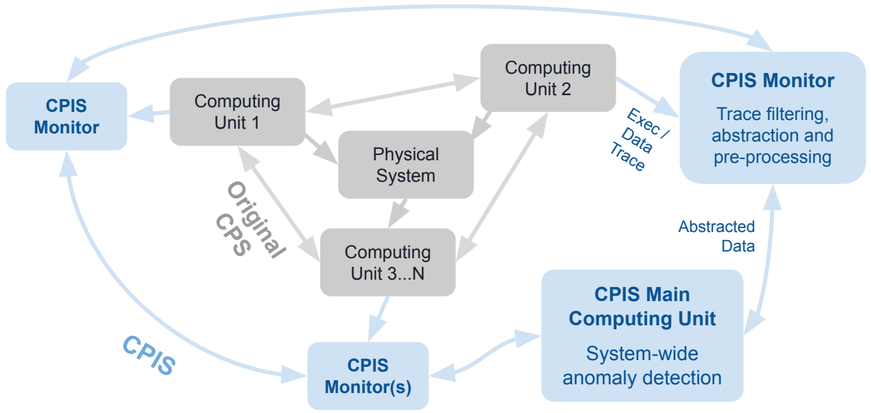

Figure 1: CPIS deployed on top of an existing CPS

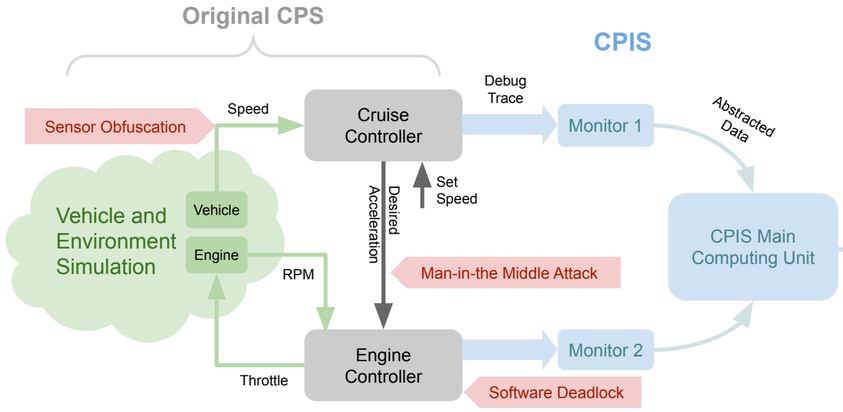

Figure 2: CPIS prototype attached to a cruise control CPS

Implementation

To validate the CPIS concept, we implemented an illustrative vehicular cruise control CPS, consisting of an engine controller, a cruise controller, and a simulated physical environment, shown in the LHS of *Figure 2*. We then implemented a CPIS that attaches to the vehicular CPS, as shown in the RHS of *Figure 2*.

Monitor

- Attaches to each CPS computing unit
- Obtains debug traces from its peer and abstract them into line-counting vectors
- Extracts run-time data (e.g., vehicle speed, throttle, engine RPM) and checks the data against pre-defined system limits

CPIS Main Computing Unit

- Collects line-counting vectors from monitors, mines execution invariants from them, and detects execution-flow anomalies (*Figure 2.1*)
- Collects run-time data from monitors, trains a regression model to learn relationships between heterogeneous data sets, and finds anomalies (*Figure 2.2*)

Simulation of anomalies

We implemented three run-time anomalies to test the CPIS (red pointers in *Figure 2*):

- Sensor obfuscation: randomly overwrites speed data provided by the simulator
- Man-in-the-middle attack: hijacks and randomly modifies the desired acceleration
- Software deadlock: starves engine-controller software and causes it to hang

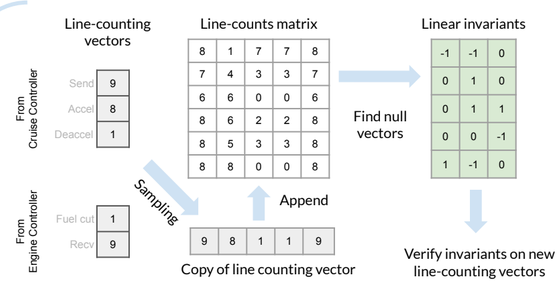

Figure 2.1: Execution-invariant mining workflow (simplified)

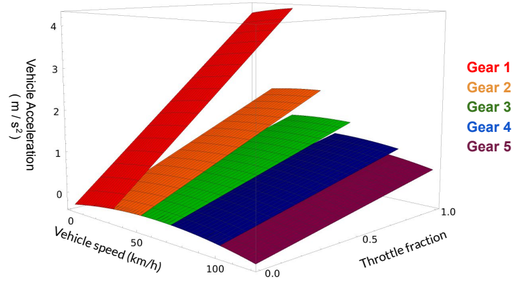

Figure 2.2: Regression model build from run-time data

Results

The CPIS prototype was able to detect our simulated attacks consistently, showing the effectiveness of combining localized and system-wide anomaly detection. For example,

- CPIS monitors detect anomalies during sensor obfuscation quickly and reliably using pre-programmed information;
- The CPIS main computing unit plays an important role during man-in-the-middle attacks by cross-validating data that are seemingly valid to individual monitors.

Future work

As part of our immediate goals for the CPIS, we plan to improve the trace-processing capabilities of the monitors, given that they are exposed to large amounts of trace data, especially on a high-speed, multi-core computing-unit.

Another area of future work is threat-mitigation capabilities, so the CPIS can automatically take countermeasures, just like the immune system does.