A Virtual Environment for Industrial Control Systems: A Nonlinear Use-Case in Attack Detection, Identification, and Response

Andrés F. Murillo
Joint work with
Luis F. Cómbita, Andrea Calderón, Sandra Rueda, Álvaro Cárdenas, Nicanor Quijano.

Colombia, 2018
Problem – Attacks to critical infrastructures:
- Critical infrastructures interact with the real world
- Attacks have huge financial, economical, political and even military impact

Credit: Office of the Presidency of the Islamic Republic of Iran

Credit: Steag/VGB Power Tech GmbH, CC BY-SA
We need:
• Environments for testing security of industrial control systems
• Fidelity of testing environments is important
Idea:
• Virtual environments offer a high fidelity environment without the cost associated to physical environments

Proposal:
• Using a virtualized open source platform to test security approaches for Industrial Control Systems (ICS)
Previous work:

- Mininet: light virtualization tool to emulate communication networks (http://mininet.org/)

- Mininet enables to emulate network topologies, in which each host and switch is represented by a container
Previous work:

- **MiniCPS**: mininet extension that enables the emulation of industrial control systems ([https://github.com/scy-phy/minicps](https://github.com/scy-phy/minicps))

- MiniCPS emulated plants with very simplistic models
  - MiniCPS does not represent plant-sensor and actuator-plant behavior
Our previous work:

- Virtual Incident Response Functions in Control Systems
- Topology with three control loops
- Emulation of Sensor-PLC-Actuator communication

Plant model was still too simple.
- In such systems is difficult to grasp network impact in security attacks
In this paper, we extend our previous work by using a non-linear plant for experiments:

- One control loop
- Behavior of the three tanks is interdependent,
- Non-linear
- Tank 301 is not controlled
Objective:
• Keep the plant at a desired setpoint

Flow:
1. Controller receives reading from sensor
2. Controller calculates action control
3. Controller sends commands to actuators
Plant is represented by:

\[ X_{k+1} = AX_k + Bu_k \]
\[ Y_k = CX_k \]

X \rightarrow \text{Plant State}
Y \rightarrow \text{Sensor Output}

Sensor reading **might not** represent accurately plant state!
- Malfunction
- Attack
Integrity Attacks:
An integrity attack tampers sensor readings

1. Attacker hampers sensor reading
2. PLC applies control with false information
3. Plant behavior can be erratic or dangerous
• Integrity attacks on ICS networks are akin to malware on traditional IT environment

• In both cases, the message can be authentic and have integrity
  • Payload is designed with deep knowledge of application
  • Payload “tricks” main application to perform desired behavior

• In both cases, defenses require to inspect the packet payload
Defenses to this type of attacks could be applied at different network points:

a) IDS at supervisory level

b) IDS at field level

c) IDS integrated into the PLC
• Also an integrity attack
• A bias attack adds an F value to the original sensor reading
Anomaly Detection: Unknown Input Observer (UIO)

- Known plant behavior
- Plant behavior must follow physical laws
- Anomaly detection can be performed using physical laws
- If at time \( k \), the systems deviate from expected physical model, an anomaly is detected
Anomaly Detection: Unknown Input Observer (UIO)

- Plant behavior must follow physical laws

- The residue of the plant without an attack is measured, this is used as a base value

- If at time $k$, the systems deviate from expected physical model, an anomaly is detected
Anomaly Detection: Unknown Input Observer (UIO)
Sensor reading

Sensor reports malicious reading to PLC

PLC calculates UIO residue

Is residue above threshold?

Yes

PLC computes mitigation control action

PLC sends – Actuator applies control action

No
Setup:
- Mininet - MiniCPS
- Physical plant
  - Three tank water plant
  - Python odeint differential equation system solver
Setup:
- Plant equations:

\[
S \frac{d}{dt} L_1(t) = Q_1(t) - q_{13}(t),
\]
\[
S \frac{d}{dt} L_2(t) = Q_2(t) + q_{32}(t) - q_{20}(t),
\]
\[
S \frac{d}{dt} L_3(t) = q_{13}(t) - q_{32}(t),
\]
\[
q_{13}(t) = \mu_{13} S_n \text{sgn}(L_1(t) - L_3(t)) \sqrt{2g|L_1(t) - L_3(t)|}
\]
\[
q_{32}(t) = \mu_{32} S_n \text{sgn}(L_3(t) - L_2(t)) \sqrt{2g|L_3(t) - L_2(t)|}
\]
\[
q_{20}(t) = \mu_{20} S_n \sqrt{2gL_2(t)}.
\]
Setup:
• Mininet - MiniCPS
• PLC – Sensors – Actuators
  • Each one a mininet node, running a python script
Set point (expected behavior)
Bias Attack:

1. Time $k=200$. Attacker starts the attack
2. Attack value: 0.02m
3. Controller thinks that the plant level is below the desired setpoint and applies the control algorithm
4. Attack ends at $k=350$
Tank overflows
(Damaged product – Physical Harm)
Experiment

Defense mitigates impact
Bias Attack:

- Other experiments to further test behavior of our defense

- Five experiments
  - All of them: with and without defense
  - Duration: 500 seconds
  - Attacks start at 200 and ends at 350

- Change bias attack from 0.01m to 0.05m

- Calculate mean error between desired behavior and current behavior of the tank
Plant Mean Error with Bias Attack

- Mean Error without Defense
- Mean Error with Defense
- Mean Error without Attack

Average Error (m) vs. Attack Value of Bias Attack (m) $\times 10^{-3}$
## Related Work

<table>
<thead>
<tr>
<th>Testbed</th>
<th>Controlled Plant</th>
<th>Industrial Equipment</th>
<th>Network Representation</th>
<th>Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pipeline virtual testbed</td>
<td>Simulation (Simulink)</td>
<td>Emulation (Open PLC)</td>
<td>Physical</td>
<td>Physical network is difficult to scale</td>
</tr>
<tr>
<td>Lancaster ICS Testbed</td>
<td>Physical Equipment</td>
<td>Physical Equipment</td>
<td>Physical Equipment</td>
<td>Expensive testbed and difficult to scale</td>
</tr>
<tr>
<td>Emulab ICS Testbed</td>
<td>Simulation (Matlab)</td>
<td>Simulation (PLC Code)</td>
<td>Emulation (Emulab Network)</td>
<td>Uses proprietary software</td>
</tr>
</tbody>
</table>
• Virtual environments may be used to emulate the behavior of a non-linear plant and a networked control loop *(better scalability)*

• Emulating and controlling a non-linear plant is much more challenging than emulating a linear plant *(more testing setups – and more interesting)*

• Unknown Input Observer (UIO) can be used on Networked Industrial Control Systems to protect the plant from harmful behavior *(in the same way that anti-malware software is used today)*
• Extend our virtual environments to consider Real-Time Operating Systems  
  • Enables to evaluate real time constrains (RTOSs are actually used in ICS environments)

• Deploy environment in a cloud environment  
  • Offers a testbed that more closely resembles a virtualized industrial control system
Thanks
Questions and comments are always welcome

Andrés Felipe Murillo
Github: https://github.com/afmurillo/
af.murillo225@uniandes.edu.co

COMIT Github: https://github.com/ComitUniandes
UTD Cy-Phy Security lab: https://github.com/Cyphysecurity