Real-Time Attack-Resilient Cyber-Physical Systems

Fanxin Kong
Assistant Professor

Syracuse University
Department of Electrical Engineering and Computer Science

July 07, 2022

Approved for Public Release; Distribution Unlimited: Case # AFRL-2022-3390
Cyber-Physical Systems

We are living in a Cyber-Physical System world!
Cyber-Physical System Attacks

Attacks on Drones and Automobiles

The U.S. government showed just how easy it is to hack drones made by Parrot, DBPower and Cheerson. Researchers took complete control over two of the drones.

High-Tech Pirates: Researchers Hack A Yacht Via GPS

Car hackers use laptop to control standard car

By Zoe Kleinman
Technology reporter, BBC News

SECURITY

Security Researchers Find a Way to Hack Cars

By Nicole Perlroth  July 21, 2015 2:32 PM  101

Attacks on Power Systems

Ukraine power cut ‘was cyber-attack’

Security News This Week: An Unprecedented Cyberattack Hit US Power Utilities

Exposed Facebook phone numbers, an XKCD breach, and more of the week’s top security news.

Syracuse University  Department of Electrical Engineering and Computer Science

Approved for Public Release; Distribution Unlimited: Case # AFRL-2022-3390
CPS Attack Surfaces

- Cyber attack surfaces
  - e.g., communication, networks, computers, ...

- Environmental attack surfaces
  - e.g., GPS signal, electromagnetic interference, ...

- Physical attack surfaces
  - e.g., locks, casings, cables, ...

- Human attack surfaces
  - e.g., phishing, blackmail, ...
What is CPS security?

• A CPS attack whose goal is to (negatively) affect the interaction between a CPS and the physical world
  -- Originates through any attack surface

• CPS security concerns the development of technologies for defending against CPS attacks
  -- Prevention from attacks
  -- Detection of CPS attacks
  -- Reaction to attack effects
Sensor Attacks – Cyber-Physical Vulnerabilities

- Cyber attacks
  - Software attacks
  - Network attacks

- Environmental/Physical attacks
  - Attacking wheel encoder
  - Spoofing GPS

Approved for Public Release; Distribution Unlimited: Case # AFRL-2022-3390
What we study and why?

**Target: Sensor Attacks**

- The attacker can arbitrarily change sensor measurements

![Diagram of sensor attacks](image.png)
What we study and why?

**Target**: Sensor Attacks

- The attacker can arbitrarily change sensor measurements
  - envir./phy. attack surfaces
  - cyber attack surfaces

**Goal**: Attack-Resilience

- To ensure control performance with sensor attacks
## Outline

<table>
<thead>
<tr>
<th>Data-driven (Machine learning)</th>
<th>Model-based (Formal methods)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensor attack detection</td>
<td>DAC’20</td>
</tr>
<tr>
<td></td>
<td>RTAS’21</td>
</tr>
<tr>
<td>Recovery from sensor attacks</td>
<td>RTCSA’21</td>
</tr>
<tr>
<td></td>
<td>RTSS’20</td>
</tr>
<tr>
<td></td>
<td>EMSOFT’21</td>
</tr>
<tr>
<td>DAC’22</td>
<td></td>
</tr>
</tbody>
</table>
# Outline

<table>
<thead>
<tr>
<th></th>
<th>Data-driven (Machine learning)</th>
<th>Model-based (Formal methods)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensor attack detection</td>
<td>DAC’20 RTAS’21</td>
<td>DAC’22</td>
</tr>
<tr>
<td>Recovery from sensor attacks</td>
<td>RTCSA’21</td>
<td>RTSS’20 EMSOFT’21</td>
</tr>
</tbody>
</table>
## Outline

<table>
<thead>
<tr>
<th></th>
<th>Data-driven (Machine learning)</th>
<th>Model-based (Formal methods)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensor attack detection</td>
<td>DAC’20</td>
<td>DAC’22</td>
</tr>
<tr>
<td></td>
<td>RTAS’21</td>
<td></td>
</tr>
<tr>
<td>Recovery from sensor attacks</td>
<td>RTCSA’21</td>
<td>RTSS’20 EMSOFT’21</td>
</tr>
</tbody>
</table>
Adaptive Window-Based Sensor Attack Detection for Cyber-Physical Systems
Window-Based Sensor Attack Detection

- Usually, detectors detect attacks by monitoring residuals between observed sensor measurements and predicted values within a detection window.
- The existing detector treat the detection window as a fixed hyper-parameter, which faces a dilemma.

How to trade off between these two metrics?

Adaptive: dynamically adjust the detection window according to detection deadline that is computed by online safety analysis.
Overview

**System Model:** A discrete linear time-invariant (LTI) model

\[ x_{t+1} = Ax_t + Bu_t + v_t \]

**Threat Model:** An attacker can manipulate sensor measurement, thus compromise state estimates.

**Detection Deadline Estimator**
- **Reachability-based** technique for future potential behaviours
- Estimates the detection deadline after which the physical system may touch unsafe states.

**Adaptive Detector**
- A window-based detection
- Dynamically adapt its detection delay according to the deadline
- Do not miss any data points

**Data Logger**
- A sliding-window based data logging protocol
- Keeps trustworthy data for the deadline estimation and attack detection

---

**System Model:**

\[ x_{t+1} = Ax_t + Bu_t + v_t \]

**Threat Model:**

An attacker can manipulate sensor measurement, thus compromise state estimates.

**Detection Deadline Estimator**

- **Reachability-based** technique for future potential behaviours
- Estimates the detection deadline after which the physical system may touch unsafe states.

**Adaptive Detector**

- A window-based detection
- Dynamically adapt its detection delay according to the deadline
- Do not miss any data points

**Data Logger**

- A sliding-window based data logging protocol
- Keeps trustworthy data for the deadline estimation and attack detection
Detection Deadline Estimation

**Reachable set:** $\mathcal{R}$ contains all possible system states evolving from initial state, and it is easier to compute its over-approximation $\overline{\mathcal{R}}$

$$x_t \subseteq \overline{\mathcal{R}}(x_0, t) = A^i x_0 \bigoplus_{j=0}^{i-1} A^j B B U \bigoplus_{k=0}^{i} A^k B_e$$

where $\bigoplus$ denotes the Minkowski sum

**Safety Analysis:** If reachable set over-approximation does not intersect with unsafe set, i.e. $\overline{\mathcal{R}} \cap \mathcal{F} = \emptyset$, the system is guaranteed to be safe.

We can compute the upper and lower bound of $\overline{\mathcal{R}}$ through **support function** method efficiently:

$$\rho_{\overline{\mathcal{R}}} = l^T (A^t x_0) + \sum_{i=0}^{t-1} \rho_{B_U} ((A^i B)^T l) + \sum_{i=0}^{t-1} \rho_{A^i B_e} (l)$$
Adaptive Window Based Attack Detection

average residual in the detection window

\[ z_t^{avg} = \frac{1}{w_c} \sum_{i \in [t-w_c,t]} |\tilde{x}_i - \bar{x}_i| \]

where \( \tilde{x}_t = A\tilde{x}_{t-1} + Bu_{t-1} \) is predicted state, and \( \bar{x}_t \) is observed state at time \( t \).

shorter window \( \Rightarrow \) false alarm

longer window \( \Rightarrow \) deadline miss

Decreasing the Detection Window Size

Increasing the Detection Window Size
Data Logging Protocol

- **Record historical data:**
  - **Residual** between predicted and observed states
  - System **state estimations**
- **Keeps trustworthy data** for the deadline estimation and sufficient data points for attack detection

**Buffer.**

- **Within** the current detection window $w_c$
- Whether they are intact is still **unknown** since they are still being checked by the detector

**Hold.**

- Moved **outside** the current detection window
- Data are regarded **trustworthy** and thus held

**Release.**

- Historical data before $t - w_m - 1$ are outside the sliding window and **not needed** anymore
- Can be **released** to save storage space

Illustration of the Data Logger
Simulation Setting

CPS simulators:

Sensor attack scenarios:
- **Bias attack** replaces sensor data with arbitrary values.
- **Delay attack** delays sensor measurements sent to the controller for a certain time period, so that the controller cannot update the current state estimate in time.
- **Replay attack** replaces sensor data with previously recorded ones.

Simulation Settings. legends: No.: simulator number, $\delta$: control stepsize (in second), PID: PID control parameters, U: control input range, $\epsilon$: uncertainty bound, S: safe state set, $\tau$: detection threshold

<table>
<thead>
<tr>
<th>No.</th>
<th>Simulator</th>
<th>$\delta$</th>
<th>PID</th>
<th>U</th>
<th>$\epsilon$</th>
<th>$S$</th>
<th>$\tau$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Aircraft Pitch</td>
<td>0.02</td>
<td>14,0,8,5,7</td>
<td>$[-7,7]$</td>
<td>7.8e-3</td>
<td>$z \in [-\infty, -\infty, -2.5, \infty, \infty, 2.5]$</td>
<td>[0.012, 0.012, 0.012]</td>
</tr>
<tr>
<td>2</td>
<td>Vehicle Turning</td>
<td>0.02</td>
<td>0.5,7,0</td>
<td>$[-3,3]$</td>
<td>7.5e-2</td>
<td>$z \in [-2,2]$</td>
<td>[0.07]</td>
</tr>
<tr>
<td>3</td>
<td>Series RLC Circuit</td>
<td>0.02</td>
<td>5,5,0</td>
<td>$[-5,5]$</td>
<td>1.7e-2</td>
<td>$z \in [-3.5, -5, 3.5, 5]$</td>
<td>[0.04, 0.01]</td>
</tr>
<tr>
<td>4</td>
<td>DC Motor Position</td>
<td>0.1</td>
<td>11,0,5</td>
<td>$[-20,20]$</td>
<td>1.5e-1</td>
<td>$z \in [-4, -\infty, -\infty, 4, \infty, \infty]$</td>
<td>[0.118, 0.118, 0.118]</td>
</tr>
<tr>
<td>5</td>
<td>Quadrotor</td>
<td>0.1</td>
<td>0.8,0,1</td>
<td>$[-2,2]$</td>
<td>1.56e-15</td>
<td>$z \in [-5,5]$</td>
<td>[0.018, ..., 0.018]</td>
</tr>
</tbody>
</table>

Approved for Public Release; Distribution Unlimited: Case # AFRL-2022-3390
Simulation Results

• Our adaptive detector can raise alerts before the detection deadline, i.e., in-time detection, while the detector with a fixed window size finds attacks after the deadline, i.e., untimely detection.

• Note that our adaptive detector may raise some false alarms before real attacks are launched. This is because that our adaptive detector chooses a smaller window size to catch up with the detection deadline while increasing the false positives.
**Simulation Results**

- **Our adaptive detector** can raise alerts **before** the detection deadline, i.e., in-time detection, while the detector with a **fixed** window size finds attacks **after** the deadline, i.e., untimely detection.

- Note that our adaptive detector may raise some **false alarms** before real attacks are launched. This is because that our adaptive detector chooses a smaller window size to catch up with the detection deadline while increasing the false positives.
Simulation Results

- Our adaptive detector tends to have larger false positive numbers, but with minimal deadline misses.
- Note that our adaptive detector may miss the detection deadline in just 3 out of 100 experiments for one case, because those attacks have a negligible effect on the physical system.
Testbed Results

- System model is obtained by system identification
- **Our** detector alert in the **first step** after the attack, but the **fixed** window-based detection alerts **after** the vehicle reaching the **unsafe** state, which may already cause damages.
- Note that our adaptive detector detects the alert in the first step because the estimator computes the tightest deadline and shrinks the window size, making the average residual within the window larger than the threshold.

Attack detection in vehicle testbed. x-axis represents time, y-axis represents state $x$. Purple horizontal line = unsafe set boundary (below is unsafe state set). Orange circle marker = first alarm of adaptive detector. Purple square marker = first alarm of detector with fixed window.
Outline

<table>
<thead>
<tr>
<th></th>
<th>Data-driven (Machine learning)</th>
<th>Model-based (Formal methods)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensor attack detection</td>
<td>DAC’20 RTAS’21</td>
<td>DAC’22</td>
</tr>
<tr>
<td>Recovery from sensor attacks</td>
<td>RTCSA’21</td>
<td>RTSS’20 EMSOFT’21</td>
</tr>
</tbody>
</table>
Real-Time Attack-Recovery for Cyber-Physical Systems
Defending Sensor Attacks

Prevention | Detection
---|---

Response
Defending Sensor Attacks

Prevention    Detection    Response
Defending Sensor Attacks

Real-Time Attack-Recovery
- Extend the benefits of detection
- Discontinue the deviation
- Remove the negative impact before a deadline
Defending Sensor Attacks

- Prevention
- Detection
- Response

Real-Time Attack-Recovery
- Extend the benefits of detection
- Discontinue the deviation
- Remove the negative impact before a deadline
Existing Attack Recovery Works

Why not use original controller?
- Mild policy
  - Safe
  - Deadline
- Bounded disturbance → attacks

Initial solution:
Linear-Programming-based Real-Time Recovery
(Zhang, RTSS 2020)

- Which important issues are to be solved?
  - Control performance
  - Maintain after recovery
  - Computational overhead
  - Varying detection delay
Overview of the Recovery Framework

- **State Reconstructor**
- **Deadline Estimator**
- **Recovery Control Calculator**
- **Recovery Controller**
- **Original Controller**
- **Checkpointer**
- **Attack Detector**
- **Actuator**
- **Plant**
- **Sensor**

- **Computing** $X_r$ **at the time** $t_r$
- **Computing** safe deadline $t_d$ **and maintainable time** $t_l$
- **Building** the quadratic programming problem **based on** $X_r, t_d, t_l$
- **Finding** the recovery control sequence

---

Approved for Public Release; Distribution Unlimited: Case # AFRL-2022-3390
Recovery Control Calculator

Discrete LTI system: \[ x_{t+1} = Ax_t + Bu_t + v_t \]

Recovery Problem Formulation:

Objective: \[ J(x_r, \ldots, x_l, u_r, \ldots, u_{l-1}) = \sum_{i=r}^{l-1} (x_i^T Q x_i + u_i^T R u_i) + x_f^T Q_f x_l \]

(Based on LQR method – smoothness)

Constraints:

\[ (x_r = \text{center}(X_r)) \quad \text{Initial state} \]

\[ \wedge \bigwedge_{i=r}^{l} (x_i \oplus B_i \subseteq X_S \ominus A^{i-r} I_R) \quad \text{Safe} \]

\[ \wedge \bigwedge_{i=d}^{l} (x_i \oplus B_i \subseteq X_T \ominus A^{i-r} I_R) \quad \text{Target} \]

\[ \wedge \bigwedge_{i=r}^{l-1} (u_i \in U) \quad \text{Control limit} \]

\[ \wedge \bigwedge_{i=r}^{l-1} (x_{i+1} = Ax_i + Bu_i) \quad \text{Dynamics} \]

Aside:

Minkowski sum / difference:

\[ X \oplus Y = \{ x + y \mid x \in X, y \in Y \} \]

\[ X \ominus Y = \bigcap_{y \in Y} \{ x - y \mid x \in X \} \]

Over-approximation of state \( x_r \):

\[ X_r \subseteq \{ x_r \} \oplus I_R \]

Accumulation of Uncertainties: \( B_i \)
Recovery Control Calculator

Discrete LTI system: \[ x_{t+1} = Ax_t + Bu_t + v_t \]

Recovery Problem Formulation:

Objective: \[ J(x_r, \ldots, x_l, u_r, \ldots, u_{l-1}) = \sum_{i=r}^{l-1} (x_i^T Q x_i + u_i^T R u_i) + x_l^T Q_f x_l \]

(Based on LQR method – smoothness)

Constraints:
- Initial state: \( x_r = \text{center}(X_r) \)
- Safe: \( \land_{i=r}^{l} (x_i \oplus B_i \subseteq X_S \ominus A^{i-r} I_R) \)
- Target: \( \land_{i=r}^{l} (x_i \oplus B_i \subseteq X_T \ominus A^{i-r} I_R) \)
- Control limit: \( \land_{i=r}^{l-1} (u_i \in U) \)
- Dynamics: \( \land_{i=r}^{l-1} (x_{i+1} = Ax_i + Bu_i) \)

Alternating Direction Method of Multipliers (ADMM)
- separable objective
- decompose a large global problem into small local subproblems
- reduce the total solving time
Supporting Components

Checkpoint:

- $t_0 - 3 \rightarrow t_0 - 2 \rightarrow t_0 - 1 \rightarrow t_0 \rightarrow t_\alpha$
  - **deleted** (trustworthy)
  - **stored**
  - **buffered** (may be attacked)

**nominal window**

**State Reconstructor:**

\[
X_R(v_0, \ldots, v_{N_r-1}) = A^{N_r} x_w + \sum_{i=0}^{N_a-1} A^i B u_i + \sum_{i=N_a}^{N_r-1} A^i B u(t_a) + \sum_{i=0}^{N_r-1} A^i v_i
\]

- **last trustworthy state**
- **historical control inputs**
- **anticipated future control inputs**
- **uncertainties**

\[
\bar{x}(t)
\]

Department of Electrical Engineering and Computer Science

Syracuse University

Approved for Public Release; Distribution Unlimited: Case # AFRL-2022-3390
Supporting Components

Deadline Estimator:

- Estimation of safe deadline $t_d$:

When the system reaches unsafe states if we maintain the current control input?

verify safety for the reachable set over-approximation $X_k$:

$$X_k = A^k x_0 + \sum_{i=0}^{k-1} A^i B u(t_a) + \sum_{i=0}^{k-1} A^i v_{i+N_r} \quad x_0 = X_T(v_0, \ldots, v_{N_r-1})$$

check until there exists nonempty intersection between $X_k$ and the unsafe set

- Estimation of maintainable time $t_l$:

When the system states inevitably exceeds the target set considering the uncertainties?

control envelope: $E_j = \left\{ \sum_{i=0}^{j-1} A^i B \ u_{i+N_r} \ | \ u_{N_r}, \ldots, u_{N_r+j-1} \in U \right\}$ (all possible control effect)

constrained target set: $D_j = X_T \ominus \left\{ A^j x_0 + \sum_{i=0}^{j-1} A^i v_{i+N_r} \ | \ v_{N_r}, \ldots, v_{N_r+j-1} \in V \right\}$ (considering uncertainties)

$E_j \cap D_j = \emptyset \implies$ no feasible control input to maintain the states in target set
Evaluation - Benchmarks

1. Vehicle Turning
\[ \dot{x} = -\frac{25}{3}x + 5u \]

2. Series RLC Circuit
\[
\begin{bmatrix}
\dot{x}_1 \\
\dot{x}_2 \\
\dot{x}_3
\end{bmatrix} = \begin{bmatrix}
0 & \frac{1}{C} & -\frac{R}{L} \\
-\frac{1}{L} & 0 & \frac{1}{L} \\
0 & -\frac{1}{L} & \frac{1}{L}
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2 \\
x_3
\end{bmatrix} + \begin{bmatrix}
0 \\
0 \\
1
\end{bmatrix} u
\]

3. DC Motor Position
\[
\begin{bmatrix}
\dot{x}_1 \\
\dot{x}_2 \\
\dot{x}_3
\end{bmatrix} = \begin{bmatrix}
0 & 1 & 0 \\
0 & -\frac{1}{K} & \frac{1}{K} \\
0 & -\frac{1}{L} & -\frac{1}{L}
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2 \\
x_3
\end{bmatrix} + \begin{bmatrix}
0 \\
0 \\
1
\end{bmatrix} u
\]

4. Aircraft Pitch
\[
\begin{bmatrix}
\dot{x}_1 \\
\dot{x}_2 \\
\dot{x}_3
\end{bmatrix} = \begin{bmatrix}
-0.313 & 56.7 & 0 \\
-0.0139 & -0.426 & 0 \\
0 & 56.7 & 0
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2 \\
x_3
\end{bmatrix} + \begin{bmatrix}
0.232 \\
0.0203 \\
0
\end{bmatrix} u
\]

5. Quadrotor
\[
\begin{aligned}
\dot{\phi} &= p \\
\dot{\theta} &= q \\
\dot{\psi} &= r \\
\dot{p} &= \tau_x + \tau_w x \\
\dot{q} &= \tau_y + \tau_w y \\
\dot{r} &= \tau_z + \tau_w z \\
\dot{u} &= -g\theta + \frac{f_w x}{m} \\
\dot{v} &= g\phi + \frac{f_w y}{m} \\
\dot{w} &= \frac{f_w z - f_l}{m} \\
\dot{x} &= u \\
\dot{y} &= v \\
\dot{z} &= w
\end{aligned}
\]

Reference: https://ctms.engin.umich.edu/CTMS/index.php

Approved for Public Release; Distribution Unlimited: Case # AFRL-2022-3390
Recovery Results

- Recovery mechanism is needed during sensor attack, or the system may touch the unsafe states (red).
- ICCPS 2018: Non-real-time recovery (gold) cannot steer system state back to normal within a deadline.
- RTSS 2020: LP recovery (blue) may make the system oscillate before the deadline.
- EMSOFT 2021: LQR-based recovery (green) can recover systems smoothly within the deadline and maintain the states in target set before maintainable time.
More Recovery Results

(d) RLC Circuit & bias attack

(e) RLC Circuit & delay attack

(f) RLC Circuit & replay attack

(g) DC Motor Position & bias attack

(h) DC Motor Position & delay attack

(i) DC Motor Position & replay attack

(j) Aircraft Pitch & bias attack

(k) Aircraft Pitch & delay attack

(l) Aircraft Pitch & replay attack

(m) Quadrotor & bias attack

(n) Quadrotor & delay attack

(o) Quadrotor & replay attack
## Overhead Analysis

<table>
<thead>
<tr>
<th></th>
<th>Vehicle turning</th>
<th>RLC Circuit</th>
<th>DC Motor Position</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>bias</td>
<td>delay</td>
<td>replay</td>
</tr>
<tr>
<td>$T_{LP}$</td>
<td>0.83</td>
<td>1.04</td>
<td>1.03</td>
</tr>
<tr>
<td>%$_{LP}$</td>
<td>4.15%</td>
<td>5.20%</td>
<td>5.15%</td>
</tr>
<tr>
<td>$T_{solver}$</td>
<td>24.05</td>
<td>24.54</td>
<td>26.80</td>
</tr>
<tr>
<td>%$_{solver}$</td>
<td>120.25%</td>
<td>122.70%</td>
<td>134.00%</td>
</tr>
<tr>
<td>$T_{ADMM}$</td>
<td>1.69</td>
<td>1.97</td>
<td>1.85</td>
</tr>
<tr>
<td>%$_{ADMM}$</td>
<td>8.45%</td>
<td>9.85%</td>
<td>9.25%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Aircraft Pitch</th>
<th>Quadrotor</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>bias</td>
<td>delay</td>
</tr>
<tr>
<td>$T_{LP}$</td>
<td>10.79</td>
<td>6.10</td>
</tr>
<tr>
<td>%$_{LP}$</td>
<td>53.95%</td>
<td>30.50%</td>
</tr>
<tr>
<td>$T_{solver}$</td>
<td>81.10</td>
<td>62.02</td>
</tr>
<tr>
<td>%$_{solver}$</td>
<td>405.50%</td>
<td>310.10%</td>
</tr>
<tr>
<td>$T_{ADMM}$</td>
<td>27.37</td>
<td>22.80</td>
</tr>
<tr>
<td>%$_{ADMM}$</td>
<td>136.85%</td>
<td>114.00%</td>
</tr>
</tbody>
</table>
Testbed and Demos
Autonomous Vehicle Testbed Demos (Version 1)
Autonomous Vehicle Testbed Demos (Version 2)
Self-balanced Two-Wheel Car
A Demo on Attack-Resilient Sensor Fusion

Kalman Filter (KF), Interval Fusion (IF), Interval Fusion with Distribution (IFD)
Conclusion

- Untimely defense = No defense!
  - Detection of and recovery from an attack before any irreparable consequences occur

- Real-time Detection and Recovery
  - Formal Methods and Machine Learning

Thank you!  Q&A