Model-Based Design for Resilient Cyber-Physical Systems

Xenofon Koutsoukos, Janos Sztipanovits, and Gabor Karsai

Institute for Software Integrated Systems
Vanderbilt University

FORCES Kickoff Meeting
Washington, D.C., April 12, 2013
Overview

- **Objective:** Tools for multi-layer RC+EI integrated model-based design
  - Validation using multi-scale CPS simulation/emulation
  - Co-design and co-experimentation
- **Example:** Resilient cooperative control of networked systems
  - EI-aware RC design
- **Research Plan**
  - Multi-scale CPS simulation: Command and control wind tunnel (C2WT)
  - Multi-scale CPS emulation: DETERLab
  - Reference implementation using an open software infrastructure
  - FORCES Modeling Integrated Language (MIL)
Validation Using Multi-Scale CPS Simulation/Emulation

Multi-scale CPS Simulation / Emulation

- Operations toolbox
- Network events
- Network capacities
- Network Demand
- Historic Traffic
- Network Layout

Data processor
- Automated calibration
- Automated imputation
- Data collector

Fast & trusted simulator
- Scenario simulation
- Short-term predictions
- Dynamic filter

FORCES
- Real-time assurance via
  1. Flow control
  2. Demand response
  3. Incident management
  4. State awareness

RC + EI
- Attack & fault diagnosis
- Security strategy selection
- Predictive management plans
- Resilient control actions

Multi-source, untrusted, real-time data

Communication network

Attacks
Faults

Transportation / Electricity network
Co-Experimentation

{Security Levels} (EI)
(availability/integrity/confidentiality)
(restrictions on Information Flow)

{Control modalities} (EI)

Real-world data

Security & Reliability experimental scenarios

Apparatus (DETER/C2WT)  Experiment procedure

Topology
- Regular graphs
- Random graphs
- T & D networks

Traffic
- Background (Internet)
- Foreground (CPS)
- Adversarial (DDoS/Botnet)

CPS design
- Plant controller locations
- Comm. network protocols
- IT security tools

Workflow
- CPS start-stop events (EI)
- Human intervention (EI)
- Response & reconfig (RC)

Invariants
- CPS dynamics (RC)
- Sensor-actuator constraints (RC)
- Robustness margins (RC)

Experiment execution, interpretation, and RC+EI validation
Co-Design

[Diagram showing the process of co-design, including steps such as system dynamics, component code, SW timing model, and implementation model.]
Example: Resilient Cooperative Control in the Presence of Adversaries

- Resilient Asymptotic Consensus
  - Continuous Time [HSCC2011, HSCC2012]
  - Discrete Time [HiCoNS 2012, JSAC 2013]
- Resilient Asymptotic Synchronization
  - LTI Systems [HSCC 2013]
Adversary Models

- **Crash Adversary**
- **Malicious Adversary**
  - Must convey the same information to all neighbors
    - Local broadcast model
- **Byzantine Adversary**
  - Can convey different information to different neighbors
- **All adversaries are omniscient**
  - Topology of the network
  - States and algorithms of the other nodes
  - Other adversaries (can collude)

- **$F$-Total Model**
  - At most $F$ adversaries in the entire network

- **$F$-Local Model**
  - At most $F$ adversaries in the neighborhood of any normal node

- **$f$-Fraction Local Model**
  - At most a fraction $f$ of adversaries in the neighborhood of any normal node

3-total, 3=local, (3/5)-fraction local
Resilient Asymptotic Consensus

- **Hybrid system dynamics**
  \[ x_i(t + 1) = f_{i,\sigma(t)}(t, x_i(t), \{x_{(j,i)}(t)\}), \quad i \in \mathcal{N}, j \in \mathcal{N}_i^{\text{in}}, t \in \mathbb{Z}_{\geq 0}, D_{\sigma(t)} \in \Gamma_n \]

- **Agreement Condition**
  \[ \lim_{t \to \infty} \Psi(t) = 0 \quad \text{where} \quad \Psi(t) = M_{\mathcal{N}}(t) - m_{\mathcal{N}}(t) \]

- **Safety Condition**
  \[ x_i(t) \in \mathcal{I}_t = [m_{\mathcal{N}}(t), M_{\mathcal{N}}(t)], \quad \forall t \in \mathbb{Z}_{\geq 0}, \forall i \in \mathcal{N} \]

- **Weighted Mean-Subsequence-Reduced (W-MSR) Algorithm**
  \[ x_i(t + 1) = w_{(i,i)}(t)x_i(t) + \sum_{j \in \mathcal{N}_i^{\text{in}}(t) \setminus \mathcal{R}_i(t)} w_{(j,i)}(t)x_{(j,i)}(t) \]
Robust Networks

<table>
<thead>
<tr>
<th>Threat</th>
<th>Scope</th>
<th>Necessary</th>
<th>Sufficient</th>
</tr>
</thead>
<tbody>
<tr>
<td>Crash &amp; Malicious</td>
<td>$F$-Total</td>
<td>$(F+1,F+1)$-robust</td>
<td>$(F+1,F+1)$-robust</td>
</tr>
<tr>
<td>Crash &amp; Malicious</td>
<td>$F$-Local</td>
<td>$(F+1,F+1)$-robust</td>
<td>$(2F+1)$-robust</td>
</tr>
<tr>
<td>Crash &amp; Malicious</td>
<td>$f$-Fraction local</td>
<td>$f$-fraction robust</td>
<td>$p$-fraction robust, where $2f &lt; p \leq 1$</td>
</tr>
<tr>
<td>Byzantine</td>
<td>$F$-Total &amp; $F$-Local</td>
<td>Normal Network is $(F+1)$-robust</td>
<td>Normal Network is $(F+1)$-robust</td>
</tr>
<tr>
<td>Byzantine</td>
<td>$f$-Fraction local</td>
<td>Normal Network is $f$-robust</td>
<td>Normal Network is $p$-robust where $p &gt; f$</td>
</tr>
</tbody>
</table>

- [IEEE JSAC April 2013]
- Normal network is the network induced by the normal nodes
- Necessary Conditions for $F$-Total and $F$-Local are necessary for any successful DTRAC algorithm
Let $D=(V, E)$ be a nontrivial $(r,s)$-robust digraph. Then, $D'=(V \cup \{v_{\text{new}}\}, E \cup E_{\text{new}})$, where $v_{\text{new}}$ is a new node added to $D$ and $E_{\text{new}}$ is the directed edge set related to $v_{\text{new}}$, is $(r,s)$-robust if

\[ d_{v_{\text{new}}}^{\text{in}} \geq r + s - 1 \]

**Preferential-attachment model**

- Initial graph: $K_5$
- $K_5$ is $(3,2)$-robust
- Num edges / round: 4
- End with $(3,2)$-robust graph
EI-Aware RC Design

- **Attacker strategy**
  - $F$-local model
  - Compromise a node with probability $p_i$

- **Defender strategy**
  - Select neighbors to cooperate

- **Objective**
  - Determine equilibria within the class of mixed strategies

- **Resilient control**
  - Implement the outcome using W-MSR

$$x_i(t + 1) = w_{(i,i)}(t)x_i(t) + \sum_{j \in N_{i}^{in}(i) \setminus R_i(t)} w_{(j,i)}(t)x_{(j,i)}(t)$$

**Research Challenge:**
Can we characterize robust network topologies?
C2W integration models
(data flow, timing, parameters)

Based on C2WT models configuration files are generated for the various simulation components. Configure how the component is connected to the simulation (input-output binding)

Federates have to have a common data model to be able to share data.
- Data model can be imported from domain specific models
- Domain specific models can be generated from data models

C2W data models
(interaction and object models)

Domain specific C2W simulation components
- OMNET component
- CPN component
- Simulink component
- Delta3D component

Domain specific simulation models
- Omnet models
- CPN models
- Simulink models
- ...
Network Control System Simulation

Model-based Design and Integration (Design-Time Modeling Environment)

HLA-based Run-Time Simulation Environment
- **Network interactions**: Information exchange that flows through the communication network (e.g., sensor and control signals)
- **Controller interactions**: Information exchange by means other than the communication network (e.g., a proximity sensor on a UAV for obstacle detection)
- **Cross-Layer interactions**: Information exchange between the network and application layers of a network protocol stack (e.g., network conditions such as bit rate and loss rate)
Model-Based Design and Integration
Multi-Scale CPS Emulation

- Integrated Simulation and Emulation Platform for Cyber-Physical System Security Experimentation (iSEE)
  - Greater realism and accuracy with truthful protocol implementation and real network traffic delivery
  - Providing a computing platform where prototypes of software components can be deployed
- Modeling environment for system specification and experiment configuration
  - System model of CPS
  - Security experiment scenario configuration
- Run-time environment that supports experiment execution
  - DETERlab: large number of tools available for emulate network attacks
iSEE Framework

Modeling Environment

- Network Interaction Model
- Deployment Model
- Topology Model

Model Interpreter

Run-Time Environment

- NCSWT Simulation Environment
  - RTI
  - EmuGateway Federate
  - Simulink Federate
  - Tap Server
  - Tap Client
  - Network Application Code
  - Network File System
  - TCL script

Deployment Model

Configuration/Control Environment

DETERlab Emulation Environment
Impact of Security Attacks

10% loss rate

20% loss rate

30% loss rate
Research Challenges

- Integration of resilient control design and validation
  - Threat assessment
  - Attack diagnosis
  - Attack resilient control

- Integration of hierarchical EI+RC design and validation
Networked node with local processing and storage, sensors, actuators, and propulsion system:

Nodes for an ad-hoc network that has 1+ ground-link and performs a coordinated sensing/control function

Examples:
- Swarm of UAVs performing tornado damage surveillance
- Fleet of UUVs performing collecting climate change data from oceans
- DARPA System F6: Fractionated Spacecraft – A Space Global Common Challenges:
  - Networked, distributed control
  - Fault-resilience
  - Applications with different trust and security levels must run side-by-side

Open Sensing/Computing/Actuation Platform where various customer applications can run, side-by-side.
An Open Software Infrastructure

Model-driven Development

Software toolchain for modeling, synthesis, analysis, and verification

Software platform with support for resource sharing, security, and fault tolerance

The F6 IAP Reference Implementation will be released under the MIT/X Open Source License.
### Research Challenges

<table>
<thead>
<tr>
<th>Area</th>
<th>Background</th>
<th>Challenge</th>
</tr>
</thead>
<tbody>
<tr>
<td>System/Security Co-Design</td>
<td>The model-based development framework supports Multi-Level Security on software components</td>
<td>How to verify the security of information flows for a suite of component-based applications?</td>
</tr>
<tr>
<td></td>
<td>The software platform APIs provide robust and protected interfaces via generated code</td>
<td>How to verify that the APIs are robust and resilient to security attacks?</td>
</tr>
<tr>
<td>Robust Networked Control</td>
<td>The software platform has the open interfaces to implement various distributed control schemes for vehicle flight control</td>
<td>Given that malicious applications and network interference may be deployed, how to ensure that critical control functions are available even under attacks?</td>
</tr>
<tr>
<td>Threat assessment and diagnostics</td>
<td>The software platform has the open interfaces to add security/threat monitors</td>
<td>How to implement a threat monitor for a CPS Cloud?</td>
</tr>
<tr>
<td>Mechanism design</td>
<td>The physical platform is operated by an owner who makes it available to (not necessarily completely trusted) customers and their software applications</td>
<td>How to balance economic incentives and resilience techniques to achieve an economical operation of the cloud platform even under fault scenarios and security attacks?</td>
</tr>
</tbody>
</table>
Co-Design Requires Modeling Languages

- Stochastic hybrid system modeling
- Network modeling
- Secure Architecture Modeling
- Attacker/defender modeling
- RT SW modeling

Domain Specific Modeling Languages
How Should We Choose Modeling Language(s)?

- Define yet another modeling language?
- Choose one that already exists and broad enough to cover the design domain?
- Create a new standard or update an old one?

- What are the implications on tools?
- How about “my freedom of abstractions”?
- What is the language agility and evolution trajectory?
Research Challenge: FORCES MIL

**Model-Based Design**

**Domain Specific Design Automation Environments:**
- Power
- Transportation

**Tools:**
- Modeling
- Analysis
- Verification
- Synthesis

**Challenges:**
- Cost of tools
- Benefit only narrow domains
- Islands of Automation

---

**Key Idea:** Use models in domain-specific design flows and ensure that final design models are rich enough to enable production of artifacts with sufficiently predictable properties.

**Impact:** significant productivity increase in design technology

---

Mathematical and physical foundations

---

Domain-Specific Environments

---

Production Facilities

---

Design Requirements
doTransition (fsm as FSM, s as State, t as Transition) =
require s.active
step exitState (s)
step if t.outputEvent <> null then emitEvent (fsm, t.outputEvent)
step activateState (fsm, t.dst)

**Key Idea:** Ensure reuse of high-value tools in domain-specific design flows by introducing a metaprogrammable tool infrastructure.

**VU-ISIS implementation:** Model Integrated Computing (MIC) tool suite ([http://repo.isis.vanderbilt.edu/downloads/](http://repo.isis.vanderbilt.edu/downloads/))

**Domain Specific Design Automation Environments:**
- Power
- Transportation

**Metaprogrammable Tool Infrastructure**
- Model Building
- Model Transf.
- Model Mgmt.
- Tool Integration

**Explicit Semantic Foundation**
- Structural
- Behavioral

**Semantic Foundation Component Libraries**

**Layer**

**Meta**

**Domain-Specific Environments**

**Design Requirements**

**Production Facilities**
Summary

- Tools for model-based integrated EI+RC design
- Multi-scale CPS simulation and emulation
- FORCES modeling integration language