Resilience and Security in Cyber-Physical Systems: Self-Driving Cars and Smart Devices

Karthik Pattabiraman
Guanpeng (Justin) Li, Maryam Raiyat, Amita Kamath, Mohammad Rafiuzzaman, Kumseok Jung, Julien Gascon-Samson

with
Siva Hari, Michael Sullivan, Tim Tsai, Joel Emer, Steve Keckler
My Research

• Building error resilient and secure software systems

• Three main areas:
  – Software resilience techniques [SC’17][DSN’17][SC’16][DSN’16][DSN’15][DSN’14][DSN’13][DSN’12]
  – Web applications’ reliability [ASE’17][ICSE’16][ICSE’15][ICSE’14A][ICSE’14B][ASE’14][ASE’15]
  – CPS Security [FSE’17][ACSAC’16][EDCC’15][HASE’14]

• This talk
  – CPS Security and Resilience
Cyber-Physical Systems (CPS)
Cyber-Physical Systems (CPS)
CPS Challenges

Real-time constraints

Resource constraints

Hard to Upgrade

No human-in-the-loop
This Talk

• Motivation

• Resilience of Deep Neural Networks in Self-Driving Cars from Soft Errors [SC’17 – to appear]

• Intrusion Detection Systems for Smart Embedded Devices using Dynamic Invariants [FSE’17]

• Ongoing work and conclusion
DNNs in Self-Driving Cars

DNN applications are widely deployed in safety critical applications
  autonomous-driving cars – specialized accelerators for real-time processing

Silent Data Corruptions (SDCs)
  Results in wrong prediction of DNN application
  Safety standard requires SoC FIT<10 overall (ISO 26262)
Soft Errors

\[ M = 0001 \]

\[ M = 0101 \]
Soft Error Problem

- Soft errors are increasing in computer systems

Source: Shekar Borkar (Intel) - Stanford talk
Current Solutions

Traditional Solutions

- DMR for all latches in execution units
- ECC/Parity on all storage elements

Incurs high overhead

Recent Work

- Generic micro-architectural solutions
- DNN-algorithm agnostic

Nonoptimal for DNN systems
Deep learning Neural Network (DNN)
DNN Accelerator Architecture (e.g., Eyeriss – MIT)
Goal

Understand error propagation in DNN accelerators through fault injection

- Quantification
- Characterization

Based on the insights, mitigate failures:

- Efficient way to detect errors
- Hardware: Selective duplication
- Software: Symptom-based detection
Fault Injection: Parameters

DNNs

<table>
<thead>
<tr>
<th>Network</th>
<th>Dataset</th>
<th>No. of Output Candidates</th>
<th>Topology</th>
</tr>
</thead>
<tbody>
<tr>
<td>ConvNet</td>
<td>CIFAR-10</td>
<td>10</td>
<td>3 CONV + 2 FC</td>
</tr>
<tr>
<td>AlexNet</td>
<td>ImageNet</td>
<td>1,000</td>
<td>5 CONV(with LRN) + 3 FC</td>
</tr>
<tr>
<td>CaffeNet</td>
<td>ImageNet</td>
<td>1,000</td>
<td>5 CONV(with LRN) + 3 FC</td>
</tr>
<tr>
<td>NiN</td>
<td>ImageNet</td>
<td>1,000</td>
<td>12 CONV</td>
</tr>
</tbody>
</table>

Data Types

- Fixed Point (FxP): 16-bit and 32-bit
- Float Point (FP): Full- and half-precision
Fault Injection Study: Setup

Fault Injection

- 3,000 random faults per each latch in each layer

Simulator

- DNN simulation in Tiny-CNN in C
- Fault injections at C line code

Fault Model

- Transient single bit-flip
- Execution Units: Latches
- Storage: buffer SRAM, scratch pad, REG

```c
... foreach layer:
  ...
  foreach weight:
    ...
    foreach input:
      ...
      R_L2.2 = inject_fault(R_L2.2)
      R_L3 = R_L2.2 + R_L3
      ...
```
Silent Data Corruption (SDC) Consequences

A single bit-flip error → misclassification of image by the DNN
Research Questions (RQs)

• RQ1: What are SDC rates in different DNNs using different data types?

• RQ2: Which bits are sensitive to SDCs in different data types?

• RQ3: How do errors affect values that result in SDCs?

• RQ4: How does an error propagate layer by layer?
SDC Types

SDC1:
Mismatch between winners from faulty and fault-free execution.

SDC5:
Winner is not in top 5 predictions in the faulty execution.

SDC10%:
The confidence of the winner drops more than 10%.

SDC20%:
The confidence of the winner drops more than 20%.
RQ1: SDC in DNNs

1. All SDCs defined have similar SDC probabilities
2. SDC probabilities are different in different DNNs
3. SDC probabilities vary a lot using different data types
RQ2: Bit Sensitivity

FP data types:

Only certain exponent bits are vulnerable to SDCs

FxP data types:

1. High-order bits are vulnerable
2. Larger dynamic value range allows more vulnerable bits
RQ3: Value Changes

AlexNet, PE Errors, Float16

If a neuron value is changed to be a large value under a fault, it likely causes SDC
RQ4: SDC in Different Layers

1. Layers 1&2 have lower SDC probabilities in AlexNet and CaffeNet
2. SDC probability increases as layer numbers increase
RQ4: Euclidean Distance of Values

1. Euclidean distance decreases by layers
2. Local Response Normalization (LRN) in Layer 1&2 re-normalizes values back towards normal range in AlexNet and CaffeNet
Mitigation: Data Type Choice

Restraining dynamic value range reduces FIT in fixed point data types

*Scaling factor = 2 by each tech. generation
All raw FIT rates are projected based on the FIT at 28nm [Neale, IEEE TNS]
Mitigation: Symptom-Based Error Detector (Software)

AlexNet, PE Faults, Float16

Recall: 92.5%
Precision: 90.21%
On selected data types
Mitigation: Selective Latch Hardening (Hardware)

Latch hardening design choices:

<table>
<thead>
<tr>
<th>Latch Type</th>
<th>Area Overhead</th>
<th>FIT Rate Reduction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>1x</td>
<td>1x</td>
</tr>
<tr>
<td>Strike Suppression (RCC)</td>
<td>1.15x</td>
<td>6.3x</td>
</tr>
<tr>
<td>Redundant Node (SEUT)</td>
<td>2x</td>
<td>37x</td>
</tr>
<tr>
<td>Triplicated (TMR)</td>
<td>3.5x</td>
<td>1,000,000x</td>
</tr>
</tbody>
</table>

~20% overhead provides 100x reduction in FIT
Summary of DNNs

1. Characterized error propagation in DNN accelerators based on data types, layers, value types and DNN topologies

2. Mitigation Methods:
   - restraining value range of data type
   - value range checker
   - selective latch hardening
This Talk

• Motivation

• Resilience of Deep Neural Networks in Self-Driving Cars from Soft Errors [SC’17 – to appear]

• Intrusion Detection Systems for Smart Embedded Devices using Dynamic Invariants [FSE’17]

• Ongoing work and conclusion
Motivation

• **Goal:** Provide low-cost security for CPS
  – Satisfying resource and real-time constraints
  – No human intervention needed
  – Is able to detect zero day attacks

**Insight:** Leverage properties of CPS for intrusion detection
  - Simplicity and timing predictability
  - Learn invariants based on dynamic execution
  - Monitor invariants at runtime for violations
Threat Model

Cyber Process (Control Algorithm)

Communication network

Physical Process

Measurements
Stuxnet[2010]

Commands

CVE-2016-1516[2016]

[A] USENIX’2015

[B] HealthCom2013

[C] [USENIX’2015]

[D] [HealthCom2013]
Intrusion Detection Systems (IDS)

- Signature-based IDSs [CSUR2014]
- Anomaly-based IDSs [Computers&Security2009]
- Specification-based IDSs [SmartGridCom2010]
  - Static analysis
  - Dynamic analysis
Dynamic Analysis Techniques

• Invariant Examples
  - Energy usage $\geq 0$
  - Current – Past $\leq$ Threshold

- Daikon [ICSE’01]
- Gk-tail [ICSE’08]
- Texada [ASE’15]
- Perfume property miner [ASE’14]
Main Idea

D: Data
E: Event
T: Time

Diagram showing relationships between data, events, and time.
Methodology

- **ARTINALI: A Real Time-specific Invariant iNference ALgorIthm**
  - 3 dimensions and 6 classes of invariants
ARTINALI Implementation

1. ARTINALI D/E MINER
   - D/E invariants

2. ARTINALI E/T MINER
   - E/T invariants

3. ARTINALI D/T MINER
   - Real-time data invariants

4. IDS PROTOTYPE
CPS Platforms for Evaluation

• Advanced metering infrastructure (AMI)
  – SEGMeter
    • http://smartenergygroups.com

• Smart Artificial Pancreas (SAP)
  – OpenAPS
    • https://openaps.org/
Experimental Setup

- CPS
  - Traces
  - Intrusion Detector
  - IDS prototype
  - Invariant Interface
    - Daikon
    - Texada
    - Perfume
    - ARTINALI

To test

CPS model (invariant set)

Attack detected!
# Targeted Attacks

<table>
<thead>
<tr>
<th>CPS Platform</th>
<th>Targeted attack</th>
<th>Attack entry point</th>
</tr>
</thead>
<tbody>
<tr>
<td>AMI (SEGMeter)</td>
<td>Meter spoofing [ACSAC2010]</td>
<td>Deception on A</td>
</tr>
<tr>
<td></td>
<td>Sync. Tampering [ACSAC2010]</td>
<td>Deception on D</td>
</tr>
<tr>
<td></td>
<td>Message dropping [CCNC2011]</td>
<td>DoS on A</td>
</tr>
<tr>
<td>SAP (OpenAPS)</td>
<td>CGM spoofing [Healthcom2011]</td>
<td>Deception on A</td>
</tr>
<tr>
<td></td>
<td>Stop basal injection [BHC2011]</td>
<td>Deception and DoS on C</td>
</tr>
<tr>
<td></td>
<td>Resume basal injection [BHC2011]</td>
<td>Deception and DoS on C</td>
</tr>
</tbody>
</table>

**Take away:**
ARTINALI detected all the targeted attacks
Arbitrary Attacks

Data mutations

Smart facial recognition system (CVE-2016-1516)

Branch flipping

CGM spoofing in SAP, [BHC2011]

Artificial delay insertion

Synchronization tampering in smart meter, [ACSAC2010]
Accuracy Metrics

- **False Negative Rate (FNR)**
  \[
  \frac{\text{Number of detected attacks}}{\text{Total number of injected attacks}} \times 100
  \]

- **False Positive Rate (FPR)**
  \[
  \frac{\text{Number of raised alarms}}{\text{Total number of attack-free tests}} \times 100
  \]

- **F-Score(\(\beta\))**
  \[
  \frac{(1+\beta^2) \times TP}{(1+\beta^2) \times TP + \beta^2 \times FN + FP}
  \]
Parameter Tuning

(a) Daikon  (b) Texada  (c) Perfume  (d) ARTINALI
False Negative (FN) Rate

- ARTINALI-based IDS reduces the ratio of FN by 89 to 95% compared with the other tools across both platforms.

- SEGMeter

![Graph showing False Negative Rate (FNR) for different tools and data mutations.](image)
False Positive (FP) Rate

- ARTINALI-based IDS reduces the ratio of FP by 20 to 48% compared with the other tools across both platforms.

- SEGMeter

FPR (%) - 95% confidence interval

(15-12)/15=20% improvement
## Performance and Memory

**SEGMeter**

<table>
<thead>
<tr>
<th></th>
<th>Performance Overhead (%)</th>
<th>Detection time (sec)</th>
<th>Memory usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Daikon</td>
<td>27.3</td>
<td>16.63</td>
<td>1.24 MB</td>
</tr>
<tr>
<td>Texada</td>
<td>23.7</td>
<td>14.45</td>
<td>3.21 MB</td>
</tr>
<tr>
<td>Pefume</td>
<td>32.08</td>
<td>19.57</td>
<td>3.94 MB</td>
</tr>
<tr>
<td>ARTINALI</td>
<td>31.6</td>
<td>19.25</td>
<td>2.96 MB</td>
</tr>
</tbody>
</table>

**Time**

- **T0**
- **T0+60**
- **T0+120**

**CPS**

- 1\textsuperscript{st} execution
- 2\textsuperscript{nd} execution
- 3\textsuperscript{rd} execution
Summary of ARTINALI

• ARTINALI: A Multi-Dimensional model for CPS
  – Captures data-event-time interplay
  – Introduces Real-time data invariants
  – Increases the coverage of IDS
  – Decreases the rate of false positives
  – Imposes comparable overheads

• Examine generalizability of ARTINALI
  – Unmanned Aerial Vehicle (UAV)

• https://github.com/karthikp-ubc/Artinali
This Talk

• Motivation

• Resilience of Deep Neural Networks in Self-Driving Cars from Soft Errors [SC’17 – to appear]

• Intrusion Detection Systems for Smart Embedded Devices using Dynamic Invariants [FSE’17]

• Ongoing work and conclusion
Ongoing Work: Formal Analysis

- Formally model the states of the CPS
- Combine with formal attacker models
- Model-check the system for security invariants
  - Identify unsafe states and paths to unsafe states
  - Automatically mount the attacks on the system
Ongoing Work: SmartJS

- **SmartJS**: Smart JavaScript-based Runtime System for programming IoT systems
  - Security and Performance constraints
  - Dynamic code migration to satisfy constraints
Ongoing Work: Resilient ML

Deriving ML algorithms resilient to perturbations
- Small changes $\rightarrow$ Similar outputs
- Convergence properties
Conclusion

CPS systems resilience and security are important challenges

Two systems for resilience and security
1. Deep Neural Network Accelerators for Self-Driving Cars
2. Invariant monitoring for embedded system security

Future work
1. Formal analysis for CPS
2. Smart runtimes for IoT
3. Resilient Machine Learning

Questions? karthikp@ece.ubc.ca