Detect & Adapt: A Resiliency Enhancement Mechanism for Space Computing Platforms

Shafkat Islam, Nagender Aneja, Ruy de Oliveira, Sandhya Aneja, Bharat Bhargava, Jason Hamlet, Chris Jenkins

#SAND2023-11822A
**Motivation**

- Applications of space systems:
  - Navigation
  - Communication
  - Weather forecast
  - Remote sensing

- Heterogeneous Compute Platforms (HCP):
  - CPU
  - GPU
  - FPGA
  - DSP

- HCP lacks in built-in security features

Introduction

- Applications of HCP outside of space mission: critical infrastructure, autonomous vehicles, edge AI [LLH+22].
- HCP integrates different processing units into single chip.
- Provides efficient computing in terms of performance and power consumption.
- HCP executes computation without the supervision of any central entity.
- Gap exists in the literature regarding the security features of HCP platform in space systems [SMM+21, Cas10, VHJ19, MBH+21, ZLZT21].
- There is a pressing need to explore ways to develop security solutions for the HCP-based computing environment.
Proposed HCP orchestration & Monitoring Mechanism

- Orchestrator assigns:
  - Data, algorithm, and task.
- Compute units:
  - Execute task.
  - Send the result to the monitor.
- Monitor:
  - Verifies results from each compute unit.
  - Outputs the final compute result.
  - Evaluates performance of each compute unit.
  - Sends the evaluation summary to the orchestrator.
Heterogeneous Computing: Threat Model

- Trustworthy:
  - Orchestrator.
  - Monitor.

- Attacks:
  - Manipulation of results.
  - Degradation of performance.

- Manipulation of results:
  - Data manipulation.
  - Output alteration.
  - Packet drop.

- Degradation of performance:
  - Algorithmic trojan (exploits worst case complexity of an algorithm).
  - Memory leakage.

Proposed HCP orchestration & Monitoring Mechanism.
Heterogeneous Computing: Computing Strategy

- **Computation Partition:**
  - Partitioning a computation task into multiple sub-tasks.
  - Monitor combines the result of each subtask.

- **Triple Modular Redundancy:**
  - Three compute units perform an identical task/sub-task, using the same algorithm or execution process.

- **Triple Modular Diversity:**
  - Three compute units performing identical tasks/sub-tasks but with different task execution processes/algorithms.

- **Hybrid Strategy:**
  - Both the algorithm and compute unit can change over time.
  - Can be performed consistently with the three other methods stated above.
Heterogeneous Computing: Hierarchical Detection

- **Hierarchical Detection:**
  - Sequence of multiple detection strategies.

- **Brute Force Approach:**
  - Monitor checks each small segment of results from compute units.

- **Probabilistic Approach:**
  - Monitor checks certain portions (consecutive or discrete) of output result from compute units.

- **Fingerprinting:**
  - Monitor accumulates execution statistics, such as memory usage, computation time consumed, etc.

- **Hashing based Approach:**
  - Matches hash value of correct output (ideally partial).

- **Attribute-based Checksum Approach:**
  - Monitor matches attributes of output results.
Heterogeneous Computing: Orchestration

- **Machine Learning based Adaptive Orchestration:**
  - Data driven model.
  - Supervised or reinforcement learning.

- **Rule based Adaptation:**
  - Follows certain rules to change orchestration process based on performance data provided by the monitor.

- **Random Orchestration (Baseline):**
  - Orchestrator randomly assigns task to compute units.
  - Unresponsive to any adversarial incident.

- **Round-robin Orchestration (Baseline):**
  - Assigns tasks in round robin manner.
  - Unresponsive to any adversarial incident.
Conclusion & Future Works

• **Observation:**
  - We conducted preliminary experiments using a **sorting application**, and it shows that the platform can **defend** against abnormalities satisfactorily even if approximately **two-thirds** of the compute units are **under attack**. We have not considered colluding attacks in the experiments yet. We used **baseline** orchestration mechanisms in the experiments.

• **Targeted applications:**
  - Sorting.
  - Compression.
  - Gradient descent calculation.
  - Machine learning algorithm.
References


