Fail through the Cracks: Cross-System Interaction Failures in Modern Cloud Systems

Lilia Tang*, Chaitanya Bhandari*, Yongle Zhang, Anna Karanika, Shuyang Ji, Indranil Gupta, Tianyin Xu

What are Cross-System Interaction Failures?

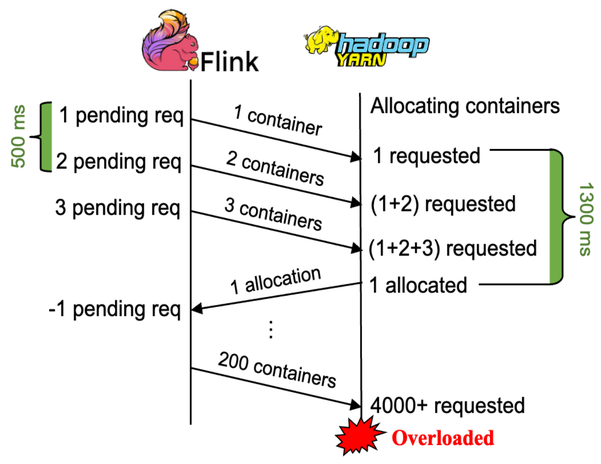
- Modern cloud systems are orchestrations of interacting systems specializing in important services
  - Further perpetuated by sky computing, microservices, etc.
- **CSI Failure:** An emerging failure model that manifests via interactions of independent and interacting systems
  - Root cause not contained within one system
  - Each system inspected in isolation behaves correctly
  - Cannot be detected by unit/integration level testing

Contributions

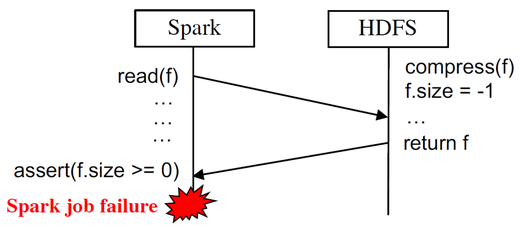
- **A call to attention for the emergence of CSI failures**
- **Study of 120 CSI failures across 7 systems**
  - **Prevalence:** 20% of cloud incidents are caused by CSI failures
  - **Failure location:** Data- and management-planes are the dominant contributors (83%) to CSI failures
  - **Symptoms:** Most are manifested through crashing behavior
  - **Root causes**
    - Most data-plane issues are caused by metadata discrepancies
    - Most config issues about coherently configuring systems
  - **Fix strategies:** Common fixes do not fix the interactions
- **A case study on cross-system testing**

Examples of Cross-System Interaction Failures

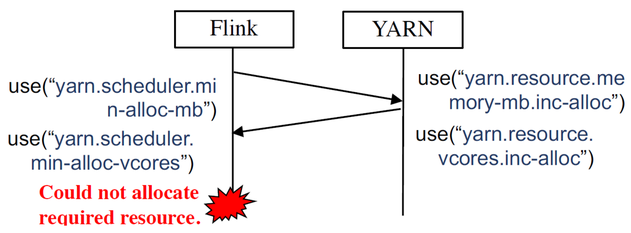
Control plane



Data plane



Management plane



Open Problems

- Existing testing practices don't cover cross-system interactions
- Many dimensions of complexity (e.g., versions, configurations)
- **Cross system testing, verification, model checking:** target connector modules, feedback-based fuzzing
- **Unification and standardization:** layers to abstract away system-specific details, e.g. serialization, language-specific
- **Rethinking data/API specifications**
- **Serialization library for complicated data abstraction**
- **Change analysis for CSI:** software evolution-targeted
- **CSI fault tolerance:** redundancy of interfaces

Selected Findings and Implications

General characteristics
- 83% manifest at data and management planes
- CSI are often single points of failure
- Not covered by existing redundancy and recovery techniques

Root causes: Data Plane
- Discrepancies in metadata, e.g., addressing and data schemas are prevalent (82%)
- Table schema interoperability between systems is a challenge
- Data serialization is error prone due to custom serializers

Root causes: Management Plane
- Primarily about not coherently configuring systems
- E.g., config lost when propagating or merging values
- Monitoring issues come from incorrect/missing metrics and conflicting policies

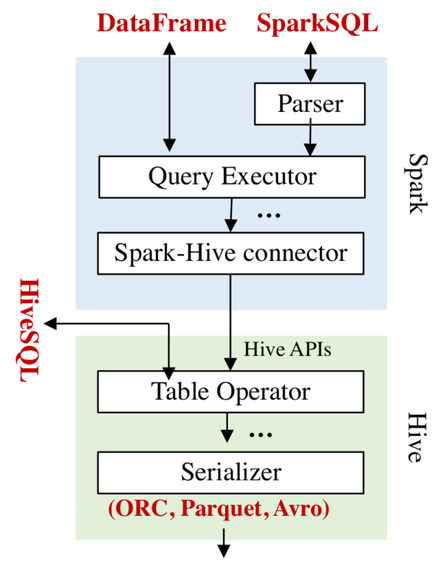
Root causes: Control Plane
- Primarily related to implicit properties

Fixes
- 40% of fixes improve condition checking and error handling
- Majority focus on improving failed interactions but not all fundamentally resolving CSI issue
- Majority of fixes occur in connector modules

Cross-System Testing Case Study: Spark-Hive Data Plane

- Cross-system testing of Spark-Hive data plane interfaces
- **Key idea**: interactions and interfaces should be consistent
- Found **15** discrepancies over 422 inputs over all data types
- **3** interfaces, **3** data formats

Test Oracles
- Write-Read
- Error Handling
- Differential



15 discrepancies found, **9** acknowledged, **2** confirmed
- Using DF, Avro BYTE/SHORT converted to INT but missing case to convert back
- Configurations are specific to serializer (spark.sql.hive.caseSensitiveInferenceMode)

Dataset and code:
https://github.com/xlab-uiuc/csi-ae