## Project Executive Summary

**Project Name:** Lineage-driven fault injection  
**Summary as of Date:** Nov 2015  
**Project Type (Proposed, New or Continuing):** Proposed

<table>
<thead>
<tr>
<th>Project Center (UCSC), Center Director: Ethan L. Miller</th>
</tr>
</thead>
</table>

| **Project Investigators:** Peter Alvaro  
**Affiliates/Collaborators:** Wang-Chiew Tan, Kamala Ramasubramanian (UCSC) |

<table>
<thead>
<tr>
<th><strong>Project Description:</strong></th>
</tr>
</thead>
</table>

| **Proposed Budget:** $45,000 per year (plus additional and student if funding is available) |

<table>
<thead>
<tr>
<th><strong>Estimated (or Actual) Start Date:</strong> N/A (Proposal)</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th><strong>Progress to Date:</strong></th>
</tr>
</thead>
</table>

Initial LDFI results published in the proceedings of ACM SIGMOD 2015.  
Worked with Netflix, Inc. Summer 2015 to integrate LDFI into their production infrastructure.

<table>
<thead>
<tr>
<th><strong>Publications (past 6 months)</strong></th>
</tr>
</thead>
</table>

Milestones and Deliverables for Current Year

Months 1–6

- Identify one or more additional student researchers.
- Survey existing provenance collection frameworks
- Survey existing large-scale service-oriented architectures, towards identifying lineage collection points.

Months 7–12

- Devise a locality-aware measure of redundancy of support for system outcomes, visible in lineage.
- Experiment using state-of-the-art mixed integer programming solvers to prioritize the search through failure scenarios.
- Investigate new algorithms, data structures and methodologies for collection, factoring, compression, storage and search of lineage information.
- Publish a preliminary result in a workshop or a poster session at a leading conference

Completion Criteria (if known)

- Eliminate dependence on a specification language and on simulation: generalize LDFI analysis to traces collected from service-oriented architectures and use existing fault-injection frameworks.
- Evaluate LDFI against best-in-breed approaches, including fault injection and formal methods.
- Identify a general measure of fault-tolerance for distributed services that identifies likely future interruptions of service.

Related Work within CRSS:

Related Work Elsewhere:

Fault injection is a popular strategy for improving the ‘resilience’ of large-scale distributed systems, which are difficult to formally verify.
### How Our Proposal Is Different:

While fault injection approaches similar to Netflix’s Simian Army are becoming the rule at major internet companies, based on the authors’ experience none of these approaches address the problem of how to search the astronomically large space of possible executions under fault. All existing approaches either explore the space randomly or rely on the intuition of engineers to select specific failures. PI Alvaro’s work (SIGMOD’15) converts data lineage into a SAT problem and leverages fast solvers to choose “interesting” combinations of faults to inject. While this results in a significant pruning of the space, some systems have a large number of models and there is no obvious way to order them. Worse, this work assumes that systems are specified and their executions simulated using the Dedalus language, which rules out applying LDFI to existing systems.

This proposal involves two major extensions to the existing work on LDFI:

1) A systems effort, to devise a generalized and reusable lineage collection infrastructure for existing systems.
2) A theoretical effort, to convert the decision problem (is there a set of faults that prevents a good outcome?) to an optimization problem (what is the “cheapest” collection of faults that prevents a good outcome?) This will allow us to prioritize the search through faults, as well as to quantify the level of resiliency of a given system.

### Potential Benefits to CRSS Members:

Collection and processing of lineage metadata poses significant challenges as well as opportunities for storage technologies. To minimize overhead of collection requires high sustained write throughput; offline reads can tolerate high latency and variability so long as they do not interfere with writes.

LDFI – which equates fault tolerance with data redundancy by *reifying computation as data* -- is inspired in part by storage solutions such as RAID.