Stony Brook University

Gloria Clivilles-Ramos <gloria.clivilles-ramos@stonybrook.edu>

N00014-15-1-2208 Annual Report Stoller_71122

Gloria Clivilles-Ramos <gloria.clivilles-ramos@stonybrook.edu> Wed, Jun 29, 2016 at 4:30 PM
To: sukarno.mertoguno@navy.mil
Cc: Scott Stoller <stoller@cs.stonybrook.edu>

Dear Dr. Sukarno,
On behalf of Dr. Scott Stoller, please find the annual report attached for N00014-15-1-2208.

regards,
Gloria

Gloria Clivilles-Ramos
Grants Administrator
Office of Sponsored Programs
W5510 Frank Melville Jr. Memorial Library
Stony Brook, NY 11794-3362
tel: 631-632-9029
fax: 631-632-6963

N000141512208 Stoller 2016 Annual Report.pdf
170K

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*
2016-06-27

2. REPORT TYPE
Annual Technical Report

3. DATES COVERED *(From - To)*
07/01/2015 - 06/30/2016

4. TITLE AND SUBTITLE
Algorithm Diversity for Resilent Systems

5a. CONTRACT NUMBER
N/A

5b. GRANT NUMBER
N000141512208

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Stoller, Scott D
Liu, Yanhong

5d. PROJECT NUMBER
N/A

5e. TASK NUMBER
N/A

5f. WORK UNIT NUMBER
N/A

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
The Research Foundation for The State University of New York

8. PERFORMING ORGANIZATION REPORT NUMBER
N/A

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
US Navy Office of Naval Research

10. SPONSOR/MONITOR'S ACRONYM(S)
ONR

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Use agency-mandated availability statements to indicate the public availability or distribution limitations of the report. See instruction 12 below.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

Diversity can increase the resilience of systems, by reducing the prevalence of shared vulnerabilities. This project explores the use of diversity to detect attacks that, directly or indirectly, cause incorrect changes to a program's state during execution. Specifically, the project aims to develop techniques to introduce algorithm-level diversity, in contrast to existing work on execution-level diversity. Algorithm-level diversity can introduce larger differences between variants than execution-level diversity and hence can provide greater resilience. Our approach to creating algorithm-level diversity is to start from a high-level executable specification and generate different algorithms that satisfy it. This approach builds on our extensive prior work on a systematic approach to generating efficient implementation from specifications, based on the fundamental principle of incremental computation. Many choices need to be made during a derivation; different choices lead to different algorithms. The generated algorithms may differ in fundamental ways involving both control structures and data structures.

15. SUBJECT TERMS
computer security, software diversity, program transformation

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Dr. Scott D. Stoller
U	U	U	UU	6	19b. TELEPHONE NUMBER *(Include area code)* 631-632-1627

Contract Number: N000141512208

Title: Algorithm Diversity for Resilient Systems

Major Goals:

Diversity can increase the resilience of systems, by reducing the prevalence of shared vulnerabilities. A promising way to use diversity to increase the resilience of a software application is to run multiple diverse versions of the application in parallel and compare their outputs. Any difference in the outputs of the variants indicates misbehavior due to an attack and triggers defensive action.

This project explores the use of diversity to detect attacks that, directly or indirectly, cause incorrect changes to a program's state during execution. Specifically, the project aims to develop techniques to introduce algorithm-level diversity, in contrast to existing work on execution-level diversity. Algorithm-level diversity can introduce larger differences between variants than execution-level diversity and hence can provide greater resilience.

Our approach to creating algorithm-level diversity is to start from a high-level executable specification and generate different algorithms that satisfy it. This approach builds on our extensive prior work on a systematic approach to generating efficient implementation from specifications, based on the fundamental principle of incremental computation. Many choices need to be made during a derivation; different choices lead to different algorithms. The generated algorithms may differ from each other in fundamental ways, both control structures and data structures, including the order in which parts of the input are accessed. In prior work, our method selected among the choices based on the time and space complexities of the resulting algorithms. This project will extend the method with additional choices and develop techniques to select among the choices based on diversity. The techniques will be implemented for and evaluated on Python programs.

Accomplishments Under Goals:

1. Literature Review. We surveyed over 20 papers related to automated software diversity and summarized each paper's diversity generation technique and evaluation methodology, especially metrics used to quantify the effectiveness of the technique. We also reviewed literature on software similarity metrics, since similarity metrics can also measure diversity.

We also wrote a report surveying languages and implementations for security protocols and secure applications.

2. Diversity Metrics. We designed initial metrics to measure software diversity for programs in high-level languages such as Python. We are designing both static diversity metrics, which are based on program structure and do not require running the software, and dynamic diversity metrics, which are based on run-time behavior of the software.

Our initial design for a static diversity metric is based on n-gram similarity with winnowing, applied to Python bytecode. n-gram similarity with winnowing has been applied at source-code level to measure similarity for software plagiarism/theft detection. We propose to apply it instead at the bytecode level, because we are using diversity to protect against flaws in the execution platform (not flaws in the application, since the diverse variants are functionally equivalent), so the diversity metrics should apply to the program representation used by the execution platform.

Our initial design for dynamic diversity metrics includes a metric based on executed bytecode instruction sequences and a metric one based on data access patterns.

3. Demand-Driven Incremental Object Queries. We continued our work on generating efficient implementations from specifications, which is the foundation for our approach to generating diverse implementations from specifications. Specifically, we worked on generation of efficient demand-driven incremental implementations of high-level queries expressed as comprehensions, quantifications, and aggregates (such as size and max) over collections such as lists and sets.

Our method supports queries that involve complex conditions on objects and sets, which can be arbitrarily nested and aliased, and it provides complexity guarantees. The objects and sets involved as well as the demand---i.e., the parameter values of interest---can change arbitrarily. The method defines invariants for not only the query results, but also all auxiliary values about the objects and sets involved, including those for propagating demand, and incrementally maintains all of them.

We implemented the method, evaluated it on programs from a variety of application areas, and confirmed the performance improvements and analyzed complexities. A paper describing this work is to appear in PPDP 2016. We released our implementation of the method on github.

We are able to generate diverse algorithms using different configurations for generating incremental implementations of object queries. An especially important configuration option is whether to make the incremental computations demand-driven.

4. Generation of Efficient Algorithms for Quantification. We developed a new, systematic method for transforming Datalog rules with general universal and existential quantification into efficient algorithms with precise complexity guarantees, and for computing the complexities from the rules. The time complexity is optimal in the sense that only useful combinations of facts for matching all hypotheses in each quantified expression are considered, and each combination is considered in constant time. It is linear in the worst case in the size of the ground rules.

There are numerous choices during the transformation that lead to diverse algorithms and different performance trade-offs.

The method has been applied successfully to problems in diverse application areas, including distributed algorithms, probabilistic inference, program analysis, and paradoxes and games.

A paper describing this work is in preparation for submission to the Computing Research Repository (CoRR) on arXiv.

5. DistAlgo. DistAlgo is a language for clear, high-level description of distributed algorithms. DistAlgo will be used for our experiments with diversity for distributed programs.

We developed a core part of a new implementation of DistAlgo to improve long-term maintainability and scalability, with more complete interface to automatic incrementalizers and more flexible network topology for communication among distributed processes. We maintained and updated the released version (https://github.com/DistAlgo/) for students doing course projects and research projects.

6. Verification of Coordination Algorithms. Algorithms are needed to coordinate the inputs and outputs of concurrently executing variants of a program. We developed the first, complete, automatically checked, formal proof of Lamport's Multi-Paxos algorithm, one of the best-known distributed agreement algorithms. Multi-Paxos and its variants are used in many open-source and commercial systems. We specified Multi-Paxos in DistAlgo and TLA+ and verified it using TLAPS, the TLA+ proof system.

Training Opportunities:

Students working on this project (listed in the Participants section) receive advanced training on program transformation and computer security. This training includes frequent one-on-one discussions and joint work on problem definition, algorithm design, implementation, papers, presentations, etc.

Jon Brandvein, a PhD student working partly on this project, graduated in May 2016 and joined Google NYC.

Results Dissemination

Results from this project are disseminated primarily through publications, software releases, and presentations at conferences.

Plans Next Reporting Period

1. Continue work on design of software diversity metrics, and study the kinds of resilience properties that software diversity techniques can provide.

2. Implement an initial set of diversity metrics, and evaluate them on benchmark programs, including the generated diverse programs.

3. Implement well-known execution-level diversity techniques and measure the diversity achieved with algorithm-level diversity alone, execution-level diversity alone, and both techniques combined.

4. Explore extensions to our algorithm derivation method to increase the diversity of the generated algorithms.

Honors and Awards

Nothing to Report

Protocol Activity Status

Distribution Statement:

Approved for public release; distribution is unlimited.

Participants

First Name: Scott **Last Name:** Stoller

Project Role: PD/PI

National Academy Member: N **Months Worked:** 2

Countries of Collaboration

First Name: Yanhong **Last Name:** Liu

Project Role: Co PD/PI

National Academy Member: N **Months Worked:** 2

Countries of Collaboration

First Name: Junao **Last Name:** Wang

Project Role: Graduate Student (research assistant)

National Academy Member: N **Months Worked:** 1

Countries of Collaboration

First Name: Jonathan **Last Name:** Brandvein

Project Role: Graduate Student (research assistant)

National Academy Member: N **Months Worked:** 2

Countries of Collaboration

First Name: Christopher **Last Name:** Kane

Project Role: Graduate Student (research assistant)

National Academy Member: N **Months Worked:** 2

Countries of Collaboration

First Name: Zhenjin **Last Name:** Wang

Project Role: Graduate Student (research assistant)

National Academy Member: N **Months Worked:** 1

Countries of Collaboration

First Name: Shubham **Last Name:** Singhal

Project Role: Graduate Student (research assistant)

National Academy Member: N **Months Worked:** 1

Countries of Collaboration

First Name: Bo **Last Name:** Lin

Project Role: Graduate Student (research assistant)

National Academy Member: N **Months Worked:** 1

Countries of Collaboration

First Name: Saksham **Last Name:** Chand

Project Role: Graduate Student (research assistant)

National Academy Member: N **Months Worked:** 1

Countries of Collaboration

This report does not include any images, figures, graphs, etc.