Vulnerability Testing of Cassandra Queries
A GRA Proposal Submitted by
Dr. Hossain Shahriar

Background
Apache Cassandra [1, 2] is an open source, distributed, decentralized, elastically scalable, highly available, fault-tolerant, tuneably consistent, column-oriented database that bases its distribution design on Amazon’s Dynamo and its data model on Google’s Bigtable. Created at Facebook, it is now used at some of the most popular sites on the Web. Cassandra data model consists of columns, rows, column families, and keyspace. Cassandra Query Language (CQL) [3] is based on SQL (Structured Query Language), the standard for relational database manipulation. Although CQL has many similarities to SQL, there are fundamental differences between CQL and SQL. For example, CQL is adapted to the Cassandra data model and architecture so there is still no option for a number of SQL-like operations such as JOINs or range queries over rows on clusters. Astyanax is a popular Java Cassandra client library [11] that allows developers implementing CQL queries based on abstract connection manager and a set of APIs.

Research challenge
Testing of database queries is an important aspect to understand the quality of implemented queries by developers. In particular, mistakes or faults in implemented can be found well ahead of time during testing process. Mutation-based program testing is a well-known testing approach to reveal code-level vulnerabilities [10]. Mutation testing has been applied for other query languages to test vulnerabilities such as SQL [4, 7, 8, 9] and EPL [6]. Unfortunately, there is no research work that applies mutation-based testing of Cassandra queries.

Research objective
This research project is intended to apply the concept of mutation-based testing technique for applications implementing Cassandra queries. In particular, we intend to answer the following key questions:

- What type of common faults can occur during the implementation of CQL and which of these faults lead to security breaches?
- What are the possible choices of mutation operators and mutant killing criteria to perform mutation testing?
- Given that we have a set of mutation operators and killing criteria, how do we reduce the list of operators to generate less number of mutants and perform testing effectively?

Other details
This project is intended for 20 weeks of duration and a graduate student is required to spend 20 hours/week for performing research and development with close collaboration with Dr. Hossain Shahriar. The outcome of the project is intended to submit as a conference paper during Fall
Mentorship will be provided to the student for writing research articles. Students interested for this project are suggested to visit the website http://cs.kennesaw.edu/hshahria to learn more on other related publications.

References