PRACTICAL STATIC ANALYSIS FRAMEWORK FOR INFERENCE OF SECURITY-RELATED PROGRAM PROPERTIES

By

Yin Liu

An Abstract of a Thesis Submitted to the Graduate

Faculty of Rensselaer Polytechnic Institute

in Partial Fulfillment of the

Requirements for the Degree of

DOCTOR OF PHILOSOPHY

Major Subject: COMPUTER SCIENCE

The original of the complete thesis is on file
in the Rensselaer Polytechnic Institute Library

Examining Committee:

Ana Milanova, Thesis Adviser
Stephen J Fink, Member
Mukkai Krishnamoorthy, Member
David Musser, Member
Carlos Varela, Member

Rensselaer Polytechnic Institute
Troy, New York

February 2010
(For Graduation May 2010)

ABSTRACT

For the software quality and security concerns, it is important to reason about security-related program securities. We present a static analysis framework for inference of security-related program properties. Within this framework we infer ownership, immutability and information flow for the protection of object access, data confidentiality and integrity. We propose runtime models that capture these properties. We design and implement ownership, immutability and information flow inference analyses for Java. These analyses reveal information about object access and information flow in the program, and may help uncover serious vulnerabilities.

To evaluate the framework, an empirical investigation is performed on a set of Java components, and a set of small-to-large Java programs. The results indicate that the analyses are practical and precise. Therefore, the analyses can be integrated in program comprehension tools that support effective reasoning about software security and software quality.

The usage of the inferences is illustrated by several applications of the framework. Ownership analysis is applied on reasoning about shared objects in open concurrent Java programs. Three structural patterns for object sharing are identified: the shared objects are categorized as *central*, *owned* or *distributed*. We argue that these patterns facilitate the understanding of concurrent programs. The experiments on several medium-to-large Java programs reveal the structure of sharing in real-world Java programs.

The usage of the static information flow analysis is illustrated with three applications. The first application of information flow analysis is security violation detection. We perform experiments on a set of Java web applications and the experiments show that the information flow analysis effectively detects security violations. The second application is type inference. Our experiments on the Java web applications show that our flow analysis successfully infers security types. The last application studies the effect of thread-shared variables on thread-local variables. Our experiments on a set of multi-thread programs show that most of the

thread-local variables are affected by the thread-shared variables.