



SOFTWARE SECURITY SERIES



Foreword by Gary McGraw

# SECURE PROGRAMMING WITH STATIC ANALYSIS



Brian Chess

Jacob West

## Praise for *Secure Programming with Static Analysis*

“We designed Java so that it could be analyzed statically. This book shows you how to apply advanced static analysis techniques to create more secure, more reliable software.”

—Bill Joy

Co-founder of Sun Microsystems, co-inventor of the Java programming language

“If you want to learn how promising new code-scanning tools can improve the security of your software, then this is the book for you. The first of its kind, *Secure Programming with Static Analysis* is well written and tells you what you need to know without getting too bogged down in details. This book sets the standard.”

—David Wagner

Associate Professor, University of California, Berkeley

“Brian and Jacob can write about software security from the ‘been there. done that.’ perspective. Read what they’ve written - it’s chock full of good advice.”

—Marcus Ranum

Inventor of the firewall, Chief Scientist, Tenable Security

“Over the past few years, we’ve seen several books on software security hitting the bookstores, including my own. While they’ve all provided their own views of good software security practices, this book fills a void that none of the others have covered. The authors have done a magnificent job at describing in detail how to do static source code analysis using all the tools and technologies available today. Kudos for arming the developer with a clear understanding of the topic as well as a wealth of practical guidance on how to put that understanding into practice. It should be on the required reading list for anyone and everyone developing software today.”

—Kenneth R. van Wyk

President and Principal Consultant, KRvW Associates, LLC.

“Software developers are the first and best line of defense for the security of their code. This book gives them the security development knowledge and the tools they need in order to eliminate vulnerabilities before they move into the final products that can be exploited.”

—Howard A. Schmidt

Former White House Cyber Security Advisor

“Modern artifacts are built with computer assistance. You would never think to build bridges, tunnels, or airplanes without the most sophisticated, state of the art tools. And yet, for some reason, many programmers develop their software without the aid of the best static analysis tools. This is the primary reason that so many software systems are

# Secure Programming with Static Analysis

## Table of Contents

### Contents

#### Part I: Software Security and Static Analysis

##### 1 The Software Security Problem

- 1.1 Defensive Programming Is Not Enough
- 1.2 Security Features  $\neq$  Secure Features
- 1.3 The Quality Fallacy
- 1.4 Static Analysis in the Big Picture
- 1.5 Classifying Vulnerabilities
- 1.6 Summary

##### 2 Introduction to Static Analysis

- 2.1 Capabilities and Limitations of Static Analysis
- 2.2 Solving Problems with Static Analysis
- 2.3 A Little Theory, a Little Reality
- Summary

##### 3 Static Analysis as Part of the Code Review Process

- 3.1 Performing a Code Review
- 3.2 Adding Security Review to an Existing Development Process
- 3.3 Static Analysis Metrics
- Summary

##### 4 Static Analysis Internals

- 4.1 Building a Model
- 4.2 Analysis Algorithms
- 4.3 Rules
- 4.4 Reporting Results

# **Table of Contents**

Summary

## **Part II: Pervasive Problems**

### **5 Handling Input**

5.1 What to Validate

5.2 How to Validate

5.3 Preventing Metacharacter Vulnerabilities

Summary

### **6 Buffer Overflow**

6.1 Introduction to Buffer Overflow

6.2 Strings

Summary

### **7 Bride of Buffer Overflow**

7.1 Integers

7.2 Runtime Protection

Summary

### **8 Errors and Exceptions**

8.1 Handling Errors with Return Codes

8.2 Managing Exceptions

8.3 Preventing Resource Leaks

8.4 Logging and Debugging

Summary

## **Part III: Features and Flavors**

### **9 Web Applications**

9.1 Input and Output Validation for the Web

9.2 HTTP Considerations

9.3 Maintaining Session State

9.4 Using the Struts Framework for Input Validation

Summary

### **10 XML and Web Services**

# **Table of Contents**

10.1 Working with XML

10.2 Using Web Services

Summary

## **11 Privacy and Secrets**

11.1 Privacy and Regulation

11.2 Outbound Passwords

11.3 Random Numbers

11.4 Cryptography

11.5 Secrets in Memory

Summary

## **12 Privileged Programs**

12.1 Implications of Privilege

12.2 Managing Privilege

12.3 Privilege Escalation Attacks

Summary

## **Part IV: Static Analysis in Practice**

### **13 Source Code Analysis Exercises for Java**

Exercise 13.0 Installation

Exercise 13.1 Begin with the End in Mind

Exercise 13.2 Auditing Source Code Manually

Exercise 13.3 Running Fortify SCA

Exercise 13.4 Understanding Raw Analysis Results

Exercise 13.5 Analyzing a Full Application

Exercise 13.6 Tuning Results with Audit Workbench

Exercise 13.7 Auditing One Issue

Exercise 13.8 Performing a Complete Audit

Exercise 13.9 Writing Custom Rules

Answers to Questions in Exercise 13.2

### **14 Source Code Analysis Exercises for C**

# **Table of Contents**

Exercise 14.0 Installation

Exercise 14.1 Begin with the End in Mind

Exercise 14.2 Auditing Source Code Manually

Exercise 14.3 Running Fortify SCA

Exercise 14.4 Understanding Raw Analysis Results

Exercise 14.5 Analyzing a Full Application

Exercise 14.6 Tuning Results with Audit Workbench

Exercise 14.7 Auditing One Issue

Exercise 14.8 Performing a Complete Audit

Exercise 14.9 Writing Custom Rules

Answers to Questions in Exercise 14.2

Epilogue

References

Index