Software Security Engineering

A Guide for Project Managers

Julia H. Allen
Sean Barnum
Robert J. Ellison
Gary McGraw
Nancy R. Mead

Addison-Wesley
Upper Saddle River, NJ • Boston • Indianapolis • San Francisco
New York • Toronto • Montreal • London • Munich • Paris • Madrid
Capetown • Sydney • Tokyo • Singapore • Mexico City

Contents

Foreword xi
Preface xiii
About the Authors xxiii

Chapter 1: Why Is Security a Software Issue? 1
1.1 Introduction 1
1.2 The Problem 2
1.2.1 System Complexity: The Context within Which Software Lives 5
1.3 Software Assurance and Software Security 6
1.3.1 The Role of Processes and Practices in Software Security 8
1.4 Threats to Software Security 9
1.5 Sources of Software Insecurity 11
1.6 The Benefits of Detecting Software Security Defects Early 13
1.6.1 Making the Business Case for Software Security: Current State 17
1.7 Managing Secure Software Development 18
1.7.1 Which Security Strategy Questions Should I Ask? 18
1.7.2 A Risk Management Framework for Software Security 20
1.7.3 Software Security Practices in the Development Life Cycle 20
1.8 Summary 23

Chapter 2: What Makes Software Secure? 25
2.1 Introduction 25
2.2 Defining Properties of Secure Software 26
2.2.1 Core Properties of Secure Software 26
2.2.2 Influential Properties of Secure Software 28

2.3 How to Influence the Security Properties of Software 36
2.3.1 The Defensive Perspective 37
2.3.2 The Attacker's Perspective 43
2.4 How to Assert and Specify Desired Security Properties 61
2.4.1 Building a Security Assurance Case 62
2.4.2 A Security Assurance Case Example 63
2.4.3 Incorporating Assurance Cases into the SDLC 67
2.4.4 Related Security Assurance and Compliance Efforts 68
2.4.5 Maintaining and Benefitting from Assurance Cases 69
2.5 Summary 71

Chapter 3: Requirements Engineering for Secure Software 73
3.1 Introduction 73
3.1.1 The Importance of Requirements Engineering 74
3.1.2 Quality Requirements 75
3.1.3 Security Requirements Engineering 76
3.2 Misuse and Abuse Cases 78
3.2.1 Security Is Not a Set of Features 79
3.2.2 Thinking About What You Can't Do 80
3.2.3 Creating Useful Misuse Cases 81
3.2.4 An Abuse Case Example 82
3.3 The SQUARE Process Model 84
3.3.1 A Brief Description of SQUARE 88
3.3.2 Tools 90
3.3.3 Expected Results 90
3.4 SQUARE Sample Outputs 91
3.4.1 Output from SQUARE Steps 92
3.4.2 SQUARE Final Results 99
3.5 Requirements Elicitation 99
3.5.1 Overview of Several Elicitation Methods 100
3.5.2 Elicitation Evaluation Criteria 103
3.6 Requirements Prioritization 106
3.6.1 Identify Candidate Prioritization Methods 106
3.6.2 Prioritization Technique Comparison 110
3.6.3 Recommendations for Requirements Prioritization 111
3.7 Summary 112

Chapter 4: Secure Software Architecture and Design 115
4.1 Introduction 115
4.1.1 The Critical Role of Architecture and Design 115
4.1.2 Issues and Challenges 117
4.2 Software Security Practices for Architecture and Design: Architectural Risk Analysis 119
4.2.1 Software Characterization 120
4.2.2 Threat Analysis 123
4.2.3 Architectural Vulnerability Assessment 126
4.2.4 Risk Likelihood Determination 130
4.2.5 Risk Impact Determination 132
4.2.6 Risk Mitigation Planning 134
4.2.7 Recapping Architectural Risk Analysis 136
4.3 Software Security Knowledge for Architecture and Design: Security Principles, Security Guidelines, and Attack Patterns 137
4.3.1 Security Principles 137
4.3.2 Security Guidelines 143
4.3.3 Attack Patterns 147
4.4 Summary 148

Chapter 5: Considerations for Secure Coding and Testing 151
5.1 Introduction 151
5.2 Code Analysis 152
5.2.1 Common Software Code Vulnerabilities 153
5.2.2 Source Code Review 156
5.3 Coding Practices 160
5.3.1 Sources of Additional Information on Secure Coding 161
5.4 Software Security Testing 163
5.4.1 Contrasting Software Testing and Software Security Testing 165
5.4.2 Functional Testing 167
5.4.3 Risk-Based Testing 169
5.5 Security Testing Considerations Throughout the SDLC 173
5.5.1 Unit Testing 174
5.5.2 Testing Libraries and Executable Files 175
5.5.3 Integration Testing 176
5.5.4 System Testing 176

5.5.5 *Sources of Additional Information on Software Security Testing* 179
5.6 Summary 180

Chapter 6: Security and Complexity: System Assembly Challenges 183
6.1 Introduction 183
6.2 Security Failures 186
6.2.1 *Categories of Errors* 187
6.2.2 *Attacker Behavior* 188
6.3 Functional and Attacker Perspectives for Security Analysis: Two Examples 189
6.3.1 *Web Services: Functional Perspective* 190
6.3.2 *Web Services: Attacker's Perspective* 192
6.3.3 *Identity Management: Functional Perspective* 196
6.3.4 *Identity Management: Attacker's Perspective* 198
6.3.5 *Identity Management and Software Development* 201
6.4 System Complexity Drivers and Security 203
6.4.1 *Wider Spectrum of Failures* 205
6.4.2 *Incremental and Evolutionary Development* 212
6.4.3 *Conflicting or Changing Goals Complexity* 213
6.5 Deep Technical Problem Complexity 215
6.6 Summary 217

Chapter 7: Governance, and Managing for More Secure Software 221
7.1 Introduction 221
7.2 Governance and Security 223
7.2.1 *Definitions of Security Governance* 223
7.2.2 *Characteristics of Effective Security Governance and Management* 224
7.3 Adopting an Enterprise Software Security Framework 226
7.3.1 *Common Pitfalls* 227
7.3.2 *Framing the Solution* 230
7.3.3 *Define a Roadmap* 235
7.4 How Much Security Is Enough? 236
7.4.1 *Defining Adequate Security* 236
7.4.2 *A Risk Management Framework for Software Security* 238

7.5 Security and Project Management 244
7.5.1 Project Scope 245
7.5.2 Project Plan 246
7.5.3 Resources 250
7.5.4 Estimating the Nature and Duration of Required Resources 251
7.5.5 Project and Product Risks 253
7.5.6 Measuring Software Security 254
7.6 Maturity of Practice 259
7.6.1 Protecting Information 259
7.6.2 Audit's Role 260
7.6.3 Operational Resilience and Convergence 261
7.6.4 A Legal View 263
7.6.5 A Software Engineering View 263
7.6.6 Exemplars 265
7.7 Summary 266

Chapter 8: Getting Started 267
8.1 Where to Begin 269
8.2 In Closing 281

Glossary 283
References 291
Build Security In Web Site References 311
Index 317