Software Test Attacks to Break Mobile and Embedded Devices

Jon Duncan Hagar
Contents

Foreword by Dorothy Graham, xi
Foreword by Lisa Crispin, xiii
Preface, xv
Acknowledgments, xvii
Copyright and Trademarks Declaration Page, xix
Introduction, xxi
Author, xxxiii

CHAPTER 1 • Setting the Mobile and Embedded Framework 1
OBJECTIVES OF TESTING MOBILE AND EMBEDDED SOFTWARE SYSTEMS 1
WHAT IS EMBEDDED SOFTWARE? 2
WHAT ARE "SMART" HANDHELD AND MOBILE SYSTEMS? 3
WHY MOBILE AND EMBEDDED ATTACKS? 5
FRAMEWORK FOR ATTACKS 6
BEGINNING YOUR TEST STRATEGY 6
ATTACKS ON MOBILE AND EMBEDDED SOFTWARE 8
IF YOU ARE NEW TO TESTING 9
AN ENLIGHTENED TESTER MAKES A BETTER TESTER 10

CHAPTER 2 • Developer Attacks: Taking the Code Head On 13
ATTACK 1: STATIC CODE ANALYSIS 14
ATTACK 2: FINDING WHITE-BOX DATA COMPUTATION BUGS 21
ATTACK 3: WHITE-BOX STRUCTURAL LOGIC FLOW COVERAGE 25
TEST COVERAGE CONCEPTS FOR WHITE-BOX STRUCTURAL TESTING 28
NOTE OF CONCERN IN MOBILE AND EMBEDDED ENVIRONMENTS 29
<table>
<thead>
<tr>
<th>Chapter 3</th>
<th>Control System Attacks</th>
<th>33</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATTACK 4: FINDING HARDWARE-SYSTEM UNHANDLED USES IN SOFTWARE</td>
<td>33</td>
<td></td>
</tr>
<tr>
<td>ATTACK 5: HARDWARE-TO-SOFTWARE AND SOFTWARE-TO-HARDWARE SIGNAL INTERFACE BUGS</td>
<td>39</td>
<td></td>
</tr>
<tr>
<td>ATTACK 6: LONG-DURATION CONTROL ATTACK RUNS</td>
<td>45</td>
<td></td>
</tr>
<tr>
<td>ATTACK 7: BREAKING SOFTWARE LOGIC AND/OR CONTROL LAWS</td>
<td>49</td>
<td></td>
</tr>
<tr>
<td>ATTACK 8: FORCING THE UNUSUAL BUG CASES</td>
<td>54</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Chapter 4</th>
<th>Hardware Software Attacks</th>
<th>59</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATTACK 9: BREAKING SOFTWARE WITH HARDWARE AND SYSTEM OPERATIONS</td>
<td>59</td>
<td></td>
</tr>
<tr>
<td>SUB-ATTACK 9.1: BREAKING BATTERY POWER</td>
<td>65</td>
<td></td>
</tr>
<tr>
<td>ATTACK 10: FINDING BUGS IN HARDWARE-SOFTWARE COMMUNICATIONS</td>
<td>66</td>
<td></td>
</tr>
<tr>
<td>ATTACK 11: BREAKING SOFTWARE ERROR RECOVERY</td>
<td>69</td>
<td></td>
</tr>
<tr>
<td>ATTACK 12: INTERFACE AND INTEGRATION TESTING</td>
<td>74</td>
<td></td>
</tr>
<tr>
<td>SUB-ATTACK 12.1: CONFIGURATION INTEGRATION EVALUATION</td>
<td>80</td>
<td></td>
</tr>
<tr>
<td>ATTACK 13: FINDING PROBLEMS IN SOFTWARE-SYSTEM FAULT TOLERANCE</td>
<td>80</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Chapter 5</th>
<th>Mobile and Embedded Software Attacks</th>
<th>89</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATTACK 14: BREAKING DIGITAL SOFTWARE COMMUNICATIONS</td>
<td>89</td>
<td></td>
</tr>
<tr>
<td>ATTACK 15: FINDING BUGS IN THE DATA</td>
<td>94</td>
<td></td>
</tr>
<tr>
<td>ATTACK 16: BUGS IN SYSTEM-SOFTWARE COMPUTATION</td>
<td>97</td>
<td></td>
</tr>
<tr>
<td>ATTACK 17: USING SIMULATION AND STIMULATION TO DRIVE SOFTWARE ATTACKS</td>
<td>101</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Chapter 6</th>
<th>Time Attacks: “It’s about Time”</th>
<th>107</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATTACK 18: BUGS IN TIMING INTERRUPTS AND PRIORITY INVERSIONS</td>
<td>108</td>
<td></td>
</tr>
<tr>
<td>STATE MODELING EXAMPLE</td>
<td>114</td>
<td></td>
</tr>
<tr>
<td>ATTACK 19: FINDING TIME-RELATED BUGS</td>
<td>116</td>
<td></td>
</tr>
<tr>
<td>ATTACK 20: TIME-RELATED SCENARIOS, STORIES, AND TOURS</td>
<td>121</td>
<td></td>
</tr>
<tr>
<td>ATTACK 21: PERFORMANCE TESTING INTRODUCTION</td>
<td>125</td>
<td></td>
</tr>
<tr>
<td>SUPPORTING CONCEPTS</td>
<td>139</td>
<td></td>
</tr>
<tr>
<td>COMPLETING AND REPORTING THE PERFORMANCE ATTACK</td>
<td>140</td>
<td></td>
</tr>
<tr>
<td>WRAPPING UP</td>
<td>140</td>
<td></td>
</tr>
</tbody>
</table>
CHAPTER 7  •  Human User Interface Attacks: "The Limited (and Unlimited) User Interface" 143

HOW TO GET STARTED—THE UI 144
ATTACK 22: FINDING SUPPORTING (USER) DOCUMENTATION PROBLEMS 146
SUB-ATTACK 22.1: CONFIRMING INSTALL-ABILITY 149
ATTACK 23: FINDING MISSING OR WRONG ALARMS 149
ATTACK 24: FINDING BUGS IN HELP FILES 153

CHAPTER 8  •  Smart and/or Mobile Phone Attacks 159

GENERAL NOTES AND ATTACK CONCEPTS APPLICABLE TO MOST MOBILE-EMBEDDED DEVICES 159
ATTACK 25: FINDING BUGS IN APPS 161
ATTACK 26: TESTING MOBILE AND EMBEDDED GAMES 165
ATTACK 27: ATTACKING APP–CLOUD DEPENDENCIES 170

CHAPTER 9  •  Mobile/Embedded Security 177

THE CURRENT SITUATION 178
REUSING SECURITY ATTACKS 178
ATTACK 28: PENETRATION ATTACK TEST 180
ATTACK 28.1: PENETRATION SUB-ATTACKS: AUTHENTICATION—PASSWORD ATTACK 186
ATTACK 28.2: SUB-ATTACK FUZZ TEST 188
ATTACK 29: INFORMATION THEFT—STEALING DEVICE DATA 189
ATTACK 29.1: SUB-ATTACK—IDENTITY SOCIAL ENGINEERING 193
ATTACK 30: SPOOFING ATTACKS 194
ATTACK 30.1: LOCATION AND/OR USER PROFILE SPOOF SUB-ATTACK 199
ATTACK 30.2: GPS SPOOF SUB-ATTACK 200
ATTACK 31: ATTACKING VIRUSES ON THE RUN IN FACTORIES OR PLCS 201

CHAPTER 10  •  Generic Attacks 209

ATTACK 32: USING COMBINATORIAL TESTS 209
ATTACK 33: ATTACKING FUNCTIONAL BUGS 215

CHAPTER 11  •  Mobile and Embedded System Labs 221

INTRODUCTION TO LABS 221
TO START 222
TEST FACILITIES 223
WHY SHOULD A TESTER CARE? 224
WHAT PROBLEM DOES A TEST LAB SOLVE? 225
STAGED EVOLUTION OF A TEST LAB 227
SIMULATION ENVIRONMENTS 227
PROTOTYPE AND EARLY DEVELOPMENT LABS 228
DEVELOPMENT SUPPORT TEST LABS 228
INTEGRATION LABS 230
PRE-PRODUCT AND PRODUCT RELEASE (FULL TEST LAB) 230
FIELD LABS 230
OTHER PLACES LABS CAN BE REALIZED 232
DEVELOPING LABS: A PROJECT INSIDE OF A PROJECT 233
PLANNING LABS 233
REQUIREMENT CONSIDERATIONS FOR LABS 234
FUNCTIONAL ELEMENTS FOR A DEVELOPER SUPPORT LAB 234
FUNCTIONAL ELEMENTS FOR A SOFTWARE TEST LAB 235
TEST LAB DESIGN FACTORS 236
LAB IMPLEMENTATION 238
LAB CERTIFICATION 238
OPERATIONS AND MAINTENANCE IN THE LAB 239
LAB LESSONS LEARNED 240
AUTOMATION CONCEPTS FOR TEST LABS 241
TOOLING TO SUPPORT LAB WORK 241
TEST DATA SET-UP 243
TEST EXECUTION: FOR DEVELOPER TESTING 244
TEST EXECUTION: GENERAL 245
PRODUCT AND SECURITY ANALYSIS TOOLS 247
TOOLS FOR THE LAB TEST RESULTS RECORDING 247
PERFORMANCE ATTACK TOOLING 248
BASIC AND GENERIC TEST SUPPORT TOOLS 250
AUTOMATION: TEST ORACLES FOR THE LAB USING MODELING TOOLS 251
SIMULATION, STIMULATION, AND MODELING IN THE LAB TEST BED 253
CONTINUOUS REAL-TIME, CLOSED-LOOP SIMULATIONS TO SUPPORT LAB TEST ENVIRONMENTS 256
KEYWORD-DRIVEN TEST MODELS AND ENVIRONMENTS 259