AGENDA

Definitions and Concepts

Problem and Context

Attack based testing to find security issues
- Specific samples
- Historic attacks
- And yet more attacks

Impact to Engineering Domains

Summary
DEFINITIONS

Embedded Software Systems:

- Interact with unique hardware/systems to solve specialized problems in the “real world”
- IT software runs with largely “generic” hardware
- User is only dimly aware the device uses or has software
- Usually have significant hardware interface issues and concerns
- Initialization, noise, power up/down, timers, sensors, etc.
- Often resource constrained: RAM, ROM, stack, power, speed, time, etc.
- Typically has a restricted or no Human User Interface (HCI) but this is evolving rapidly
- Often no way (or only a risky way) to update and/or change the software
- Involves risks, hazards, safety, and/or some specialized domain knowledge and logic/algorithms usually controlling hardware
MOBILE, SMART, AND HANDHELD

As the names imply, these are small, hand held devices that are often connected to communication networks, including:

- Cell and smart phones – apps
- Tablets
- Medical devices

Typically these devices have:

- Many of the features of classic “embedded” systems (and problems)
- Many of the power and capabilities of PCs/IT (and problems)
- More/different user interfaces (UI) than classic embedded systems
- Fast updates
- Are getting more power, memory, and features (software, e.g., apps)

These are “hot” areas of computers and software

Testing rules are “evolving”
A LOOK AT MOBILE AND EMBEDDED SYSTEMS

Examples

- Avionics systems: planes, cars, rockets, military,.....
- Telecom: switch, routers, phones, cell devices,....
- Transportation: traffic control, railroad, trucking, ....
- Industrial control: lighting, machines, HVAC, nuclear/power,.
- Medical: pacemaker, dispensers, .......
- Home and office systems: control, entertainment (TV box), ...

- And the list goes on

• Now starting to include things like PDA’s and other items that “blur” the lines
DEFINITION: FUNDAMENTAL SOFTWARE CAPABILITIES

Dr. James Whittaker lists these 4:
- Software accepts inputs from its environment
- Software produces output and transmits it to its environment
- Software stores data internally in one or more data structures
- Software performs computations using input or stored data

Embedded software can be refined with:
- Functions constrained in/with Time
- Use/control of “unique” hardware
OTHER DEFINITIONS (FOR THIS PRESENTATION)

**Taxonomy** - *the practice and science of classification.*

**Test** – the act of conducting experiments on something to determine the quality and provide information

**Test case** – One set of inputs, environmental set up, and results (expected and un)

**Attack** – to set up, forcefully, and attempt to “damage” the system or software, using tools, and techniques, may use one or more test cases or procedures

**Bug (error)** – Results that depart from the expected (from requirements, design, standards, user, etc.)

**Lifecycle** – From beginning-to-end, the steps, stages, and activities to create (birth-to-death)

**Procedure** – a particular way of accomplishing tests, usually written (one or more test cases)

**Scenario** – a sequence of events with a test plot or story

**Script** – see procedure, normally uses automation

**Users** – someone/something that interacts with the system/software (can be human or machine, or?)

**Quality** – Value to someone that they will pay for
THE PROBLEM AND CONTEXT
THE CURRENT SITUATION

Mobile and embedded systems are highly integrated hardware–software system solutions which:

- Must be highly trustworthy since they handle sensitive data
- Often perform critical tasks

Security holes and problems abound

- **Coverity Scan 2010 Open Source Integrity Report** - Android
  - static analysis test attack found 0.47 defects per 1000 SLOC
  - 359 defects in total, 88 of which were considered “high risk” in the security domain

**OS hole Andriod with Angry Birds** (researchers Jon Oberheide and Zach Lanier)

Robots and Drones rumored to be attacked

Cars and medical devices being hacked
**OTHER DATA POINTS** *(SCARY)*

**Davis-Besse nuclear plant** - Attack

- Night Dragon
- Shamoon

**Oil and Gas industry impacts**

- Harrisburg water plant attack
- Texas waste treatment plant hack

Even some reports of criminal “black mail”
WORLD CHANGE: MOBILE/EMBEDDED SECURITY

Security remains focused on PC/IT networks (web), and “Traditional” software

This is changing with . . .

- Stuxnet virus loose in factory control systems
- Mobile Usage (by soldiers in the field on their own)
- Networked embedded devices and system are open to hacks
  - e.g. GPS spoofing

Design, hacking and security concerns will only increase
DESIGN & CODE ERRORS PRODUCE SOFTWARE CYBER VULNERABILITIES

Features/capabilities are known
- Some might say all features are known but there can be “undocumented” features

In perfect software, we would not need to be concerned with security vulnerabilities because we could just “build it” secure
- But many vulnerabilities come from errors or are “accidently” introduced by new use situations
EMBEDDED/MOBILE SECURITY CONCERNS

Fraud – Identity
Worms, virus, etc
- Fault injection

Processing on the run

Hacks impact
- Power
- Memory
- CPU usage

Eavesdropping – yes everyone can hear you
- Hijacking
- Click-jacking
- Voice/Screen

Physical Hacks
- File snooping
- Lost phone
Handheld/Embedded software has similar defects to traditional software
- Requirements & Design
- Logic & Math
- Control Flow
- Data
- Initialization & Mode changes
- Interfaces
- Security
- Game interfaces
- etc.

• **Embedded adds defects/issues**
  - Software and hardware development cycles done in parallel, where aspects of the hardware may be unknown to the software development effort
  - Hardware problems which are often fixed with software late in the project
  - Small amounts of dense complex functions often in the control theory or safety/hazard domains
  - (a BIG one) Very tight real-time performance issues (often in milli- or microsecond ranges)
SECURITY AND VULNERABILITY ACTIONS FOR MOBILE AND EMBEDDED DEVICES

Prepare for theft or loss of devices (encryption, IT controls, memory wipe programs, etc.)

Establish physical control (locked door and limited access to facilities - historic)

IT operations (VPN, network control, access monitors, registry logon)

Prevent development and test processes such as, developers leaving back doors in the code, testers doing something they shouldn’t when they shouldn’t

Software bugs we need to test for (this presentation)

Work third part operating systems and COTS bugs (secure OS, encrypted files, authenticated files, trusted software, etc.);

Promote regulatory and legal constraints

Attack test data/file input and output (this presentation)

Attack tests Impersonation (this presentation)
SECURITY TESTING ATTACKS
PART OF A COMPLETE DEFENSE
WHAT IS AN ATTACK

Attacking your software system – In part, the process of attempting to demonstrate that a system (hardware, software, and operations) does not meet requirements or functional and non-functional objectives.

Embedded/handheld software testing must include “the system” (hardware, software, servers, operations, users, etc.)

Attacks go after common modes of failure and bugs, attempting to demonstrate that “does not meet” exists.
AN ATTACK IS.

Based on a common mode of failure seen over and over
- Maybe seen as a negative, when it is really a positive
- Goes after the “bugs”
- Based on or using classic test techniques and test concepts

Testers learn these after years and form a mental model (most good testers attack)

I offer a few embedded attacks
- Based on literature research of published bugs
- Be suspicious
ATTACK: IDENTITY SECURITY FRAUD

Apply when the device is mobile/embedded and has
- Account numbers
- User ids and passwords
- Location tags
- Restricted data

Current authentication approaches in use on embedded/mobile devices
- Server based
  - Registry (user/password)
- Location-device based
- Profile based
Sub-attack: Identity Fraud Spoofing
- who am I and where am I, or not

- In this sub-attack, the tester is trying to fool or spoof the device/app on identity and/or location

- The tester should see if the identity can be “hijacked”
  - Hagerman (Unpublished PhD work) reports how to do this using Wireshark tool to sniff and decode data being broadcast.
NEXT APPROACH: ATTACK LOCATION

Location used as part identity?

- Check how the location is used – Is authorization temporary or permanent?
  - If temporary, the attack should check for remnant data files
    - Use development tools and/or the OS to poke around in the file system
    - Warning, the file may be encrypted, in which case you may need a file encryption cracker for that type of file/encryption, e.g. pkcrack
  - If file is not temporary, the tester next needs to determine if any of the permanent information can be accessed, abused, or corrupted
    - In many devices and apps, this data should be encrypted, and here again apply the cracking encryption tools
    - How hard or easy is it to read the file (text – bad -> encrypted better)?
SPOOFING LOCATION AND USER

Once you have the location-identity file information, ask yourself “can I spoof the location either inside of the device or what is broadcasting?”

- Each system/app will be structured differently

Closely related to location-identify spoofing is the user profile spoof, if used

- Here the tester attempts to take over an identity by understanding how user profile checks work (or don’t)

- This will require understanding the internal data points of what your system is checking

- Use factors to look for are: location, time, where transactions are occurring, types of transactions, money amounts in transactions, provider/store, product, signal location/type, and biometric data

- Input them to the system; determine if the server gets confused and gives or uses “the wrong/sensitive” data
YET MORE ATTACKS (OUTLINE)

- Attack: App configuration update
- Attack: Embedded phishing
- Attack: Virus/malware embedded in a hijacked apps
- Attack: OS and other (NOT) “trusted” COTS software

Ref. Whittaker and Hagerman
Software Test Attacks to Break Mobile and Embedded Devices

Attack 28 Penetration Attack Test – mobile and embedded

Attack 30.1 Identity Social Engineering

Attack 30.2 GPS Spoof Sub-Attack

Attack 31: Attacking Viruses on the Run in Factories or PLCs
<table>
<thead>
<tr>
<th>Number</th>
<th>Attack name</th>
<th>Applicable to mobile-embedded</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Block access to libraries and/or OS internals</td>
<td>yes</td>
</tr>
<tr>
<td>2</td>
<td>Manipulate the application's registry values</td>
<td>yes</td>
</tr>
<tr>
<td>3</td>
<td>Force the application to use corrupt files</td>
<td>yes</td>
</tr>
<tr>
<td>4</td>
<td>Manipulate and replace files that the application creates, reads from, writes to, or executes</td>
<td>yes</td>
</tr>
<tr>
<td>5</td>
<td>Force the application to operate in low memory, disk-device, and network availability</td>
<td>yes</td>
</tr>
<tr>
<td>6</td>
<td>Overflow input buffers</td>
<td>yes</td>
</tr>
<tr>
<td>7</td>
<td>Examine all common switches and options</td>
<td>yes</td>
</tr>
<tr>
<td>8</td>
<td>Explore escape characters, character sets, and commands</td>
<td>yes</td>
</tr>
<tr>
<td>9</td>
<td>Try common default and test account names and passwords</td>
<td>yes</td>
</tr>
<tr>
<td>10</td>
<td>Use a tool to expose unprotected APIs</td>
<td>yes</td>
</tr>
<tr>
<td>11</td>
<td>Connect to all ports</td>
<td>yes</td>
</tr>
<tr>
<td>12</td>
<td>Fake the source of data</td>
<td>yes</td>
</tr>
<tr>
<td>13</td>
<td>Create loop conditions in any app that interprets script, code, or other user-supplied logic</td>
<td>yes</td>
</tr>
<tr>
<td>14</td>
<td>Use alternate routes (in the app) to accomplish the same task</td>
<td>yes</td>
</tr>
<tr>
<td>15</td>
<td>Force the system to reset values</td>
<td>yes</td>
</tr>
<tr>
<td>16</td>
<td>Create files with the same name as files protected with a higher classification</td>
<td>yes</td>
</tr>
<tr>
<td>17</td>
<td>Force all error messages</td>
<td>yes</td>
</tr>
<tr>
<td>18</td>
<td>Use a tool to look for temporary files and screen their contents for sensitive info</td>
<td>yes</td>
</tr>
</tbody>
</table>
MANY MORE ATTACKS EXIST

Simply doing Verification testing is not enough
Some say “Wait, let the bad guys find the holes”
- Many mobile-embedded systems this is not a good idea

Progressive organization put forth a good offense as well as defense
- Attack testing before the bad guys
SYSTEMS ENGINEERING

Security Requirements and Requirements Engineering

Good performance and non-functional requirements

Model-based system development

Better analysis

Secure architectures

Careful

- Trade studies
- User/customer “understanding”
SOFTWARE ENGINEERING

Ditto to the Systems Engineering impacts

Secure Design
- Agile, incremental, and iterative design

Secure practices
- Defensive coding
- Modular coding

Good Software Engineering
- Static code analysis (SCA)
- Unit testing
HARDWARE/ELECTRONICS

Ditto to the Systems Engineering and Software impacts

Sneak circuit analysis

Right COTS hardware

Secure practices
  - Isolation
  - Compartmentalization
  - Fault tolerance cross checks

Good Hardware Engineering
  - Threat analysis
  - FMECA
  - Physical security
  - Redundancy
TEST ENGINEERING

Test needs to do the “standard” efforts, and needs to play the “bad guys”

- Hacking attacks
- Vulnerabilities
- Test from day one, so other teams can plug the holes based on attack information
- At the end, get really nasty
  - Some tester are really good at that
WARNINGS

- Security attacks must be done with the knowledge and approval of owners of the system and software
- Severe legal implications exist in this area
- Many of these attacks must be done in a lab (sandbox)
- In these attacks I tell you conceptually how to “drive a car very fast (150 miles an hour) but there are places to do this with a car legally (a race track) and places where you will get a ticket (most public streets)”
- Be forewarned - Do not go attack your favorite app on your phone or connected server without the right permissions due to the legal implications
WRAP UP

These attacks are just starting points

Embedded/mobile device use, features, and connections will grow meaning that security threats will increase

Be careful—there are impacts in all domains

- Systems
- Software
- Hardware
- Support
- But security must be “attacked” within IT&E
THANKS (IDEAS USED FROM)

- James Whittaker (attacks)
- Elisabeth Hendrickson (sims)
- Lee Copeland (techniques)
- Brian Merrick (testing)
- James Bach (tours)
- Cem Kaner (test thinking)

- Many teachers
- Generations past and future
- Books, references, etc.
BOOK LIST (FAVORITES THAT I USE)

- Software Test Attacks to Break Mobile and Embedded Devices, Jon D. Hagar 2013
- How to Break Software Security, Whittaker & Thompson
  - And Whittaker’s other “How To Break...” books

Honorable mentions:
- “Embedded System and Software Validation” Roychoudhury 2009
- “Systems Testing with an Attitude” 2005
- “Software System Testing and Quality Assurance” Beizer 1987
- “Testing Computer Software” Kaner et. al. 1988
- “Systematic Software Testing” Craig & Jaskiel, 2001
- “Managing the Testing Process” Black 2002
- “Hacking Exposed” McClure, Scambray, Kurtz
RESOURCES

• www.stickyminds.com
• www.embedded.com
• www.breakingembeddedsoftware.com

• Association of Software Testing
  • http://www.associationforsoftwaretesting.org/
    Offers Free Classes on Testing