The New Frontier
Securing Your Android and iOS Mobile Applications
12/06/2013
Mobile Security Trends
YOUR MOBILE DEVICE IS YOUR...

Security concerns vary but are merged

Security concerns extending from device to application and data
Hackers Follow the Money

When asked why he robbed banks, Sutton reportedly replied, "Because that's where the money is."

– Willie Sutton

http://www.condenaststore.com/-sp/l-steal-from-computers-cause-that-s-where-the-money-is-Cartoon-Prints_6639625_.htm
Mobile Malware Growing Exponentially

INCREASE IN MALWARE

472%

Source: Juniper Mobile Threat Report, 2/12

Spyware and SMS Trojans Top Two
Native Mobile Application Security Risks

- Data leakage
- Confidentiality leaks
  - Private conversations leaked to public
  - Private contact information leaked to public
  - Location leaked to public
- Integrity violations
  - Corruption of local databases
  - Fraudulent use of application
- Abuse of privileges
  - Sending text messages
  - Placing calls
  - Surveillance of device’s user

Which QR code is evil?

- QR Code contained a URL to download malware
- The malware sent SMS messages to a premium rate number (US $6 per message)

No One Is Spared (…it’s not just Android)

Mobile Security Can Mean Different Things

Our focus today

- Application Security
- Buffer Overflows (Jailbreak)
- Malware/Anti-Virus
- Physical Device Security
## Android vs. iOS

<table>
<thead>
<tr>
<th></th>
<th>Android</th>
<th>iOS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ecosystem</td>
<td>Linux / Java</td>
<td>Unix (Darwin) / Objective C</td>
</tr>
<tr>
<td>Vetting Process</td>
<td>Sorta</td>
<td>●</td>
</tr>
<tr>
<td>App Sandbox</td>
<td>●</td>
<td>●</td>
</tr>
<tr>
<td>Protection against memory corruption attacks</td>
<td>●</td>
<td>●</td>
</tr>
<tr>
<td>Data Encryption</td>
<td>●</td>
<td>●</td>
</tr>
<tr>
<td>Inter-app Communication</td>
<td>Very common</td>
<td>Uncommon (&quot;Intents&quot;, explicit vs. implicit)</td>
</tr>
<tr>
<td>Common Problems</td>
<td>Malware</td>
<td>Jailbreak</td>
</tr>
</tbody>
</table>

…but what about the other guys? Windows Phone 7/8, Blackberry 10? Apache Cordova (JS, HTML, CSS), IBM Worklight
Mobile Application Types

Mobile Web Apps

Mobile Native Apps

Mobile Hybrid Apps

Mobile Browser

Native App

Native Shell

<HTML>...
...
</HTML>

01101001001
10010010101
01010010010
10010100001

OS APIs

<HTML>

Browser

OS APIs
Mobile App Vulnerabilities

**Client-side**
- **mobile web app**
  - Same as “regular” web app client-side issues:
    - XSS, HTML5 issues, etc.

- **mobile native/hybrid app**
  - Platform-specific:
    - XAS (hybrid), Insecure Local Storage, Unencrypted Comm., Client-side SQLi, Poor Auth., Improper Session Handling, Data Leakage, Information Disclosure, etc.

**Server-side**
- Same as “regular” web apps:
  - SQL Injection, Path Traversal, Response Splitting, File Inclusion, OS Commanding, etc.
User vs. Enterprise Mobile Application Security Risk

**User**
- Threat from Malware (Trojans & Spyware)
- Fake Android marketplace
  - Malware bundled with valid app
- Phishing
- Unauthorized Use of:
  - Contact DB
  - Email
  - SMS (text messages)
  - Phone (placing calls)
  - GPS (public location)

**Enterprise**
- BYOD
  - Hackers looking for weaknesses
    - Easy access to applications
    - Reverse engineering
  - **Data leakage**
    - Attack from malware
    - Account info on mobile device

**OWASP Mobile Security Project: Top 10 Mobile Risks**
1. **Insecure Data Storage**
2. Weak Server Side Controls
3. Insufficient Transport Layer Protection
4. Client Side Injection
5. Poor Authorization and Authentication
6. Improper Session Handling
7. Security Decisions Via Untrusted Inputs
8. Side Channel Data Leakage
9. Broken Cryptography
10. Sensitive Information Disclosure
Permissions Madness

- Users don’t understand
- Permissions vary by OS & release
  - Path Address Book Upload
- Developers over permission
- Tyranny of the default
IBM MobileFirst Security

For Clients That Need To:

- Protect devices and data
- Defend the network
- Ensure secure access
- Safeguard mobile apps
- Preserve user experience without compromising security

IBM MobileFirst Security Offers:

- Context aware risk-based access control
- Mobile threat protection
- Strong session management & Single Sign-on
- Vulnerability analysis for mobile apps
- Visibility and analysis of security events from the device, network, user and app behavior

IBM Security Access Manager for Mobile and Cloud
IBM AppScan

- State-of-the-art high-quality vulnerability analysis for mobile apps
- Native support extended for iOS to accelerate enterprise usage
- Enhanced support for JavaScript analysis in hybrid mobile apps
- Out-of-the-box support for IBM Worklight built apps to incorporate context aware risk-based access
The most comprehensive mobile application security!

- Support for **Native Android** and **iOS** applications
- Security SDK research & risk assessment of over 40,000 APIs
- **Full** call and data flow analysis
  - Java
  - JavaScript
  - Objective-C
- **Identify** where sensitive **data** is being leaked
- Ensures applications are not susceptible to malware
Apple iOS Scanning

- Hot off the presses! Available in AppScan 8.7 March 2013
- Comprehensive Objective-C and JavaScript coverage
- Zero Config! (…sorta)
  - Drag and Drop Xcode import
- Native Mac OSX Support (10.7, 10.8)
- Automated Scanning (build integration)
### 100% coverage of OWASP Mobile Top Ten

<table>
<thead>
<tr>
<th>OWASP Mobile TOP 10</th>
<th>IBM Security AppScan Coverage</th>
</tr>
</thead>
<tbody>
<tr>
<td>✓ 1. Insecure Data Storage</td>
<td>Trace routes of sensitive data</td>
</tr>
<tr>
<td>✓ 2. Weak Server Side Controls</td>
<td>Security scanning of server side code</td>
</tr>
<tr>
<td>✓ 3. Insufficient Transport Layer Protection</td>
<td>Check for use of SSL/TLS</td>
</tr>
<tr>
<td>✓ 4. Client Side Injection</td>
<td>Checks for common injection flaws including SQLi, HTMLi, and XSS</td>
</tr>
<tr>
<td>✓ 5. Poor Authentication and Authorization</td>
<td>Track where IDs and Passwords enter/exit the system</td>
</tr>
<tr>
<td>✓ 6. Improper Session Handling</td>
<td>Verify UUID is not used for session management</td>
</tr>
<tr>
<td>✓ 7. Security Decisions via Untrusted Inputs</td>
<td>Track where data originates and how it is used</td>
</tr>
<tr>
<td>✓ 8. Side Channel Data Leakage</td>
<td>Test for data leakage to log files, pasteboard, property lists, etc</td>
</tr>
<tr>
<td>✓ 9. Broken Cryptography</td>
<td>Identify proper usage of cryptographic usage</td>
</tr>
<tr>
<td>✓ 10. Sensitive Information Disclosure</td>
<td>Test for data leakage to peripherals, network, sockets, etc.</td>
</tr>
</tbody>
</table>
Common Problems That AppScan Can Detect in Mobile Applications

- **Inter-Application Messaging**
  - Not validating data before using it
  - Not removing confidential data before sending message

- **Databases**
  - Creating and using unsafe SQL
  - Using unsafe DB API’s

- **Data leakage**
  - Sensitive data leaving the boundaries of the application
  - Sensitive data stored on the device in plain text

- **Command Injection**
  - Not validating data before executing it
#10 OWASP Mobile Top 10
Sensitive Information Disclosure

- Mobile app is collecting GPS location
- GPS data being sent via http request
- If hacker snoops the traffic coming into the web server, they would gain access to customer’s location
- AppScan identifies sensitive information disclosures
Mobile Security Case Study
02/22/2013: HTC America has agreed to settle Federal Trade Commission charges that the company failed to take reasonable steps to secure the software it developed for its smartphones and tablet computers, introducing security flaws that placed sensitive information about millions of consumers at risk.
HTC America has customized the software on these devices in order to differentiate itself from competitors and to comply with the requirements of mobile network operators. These customizations have introduced security vulnerabilities.

**FTC Charges: HTC America failed to…**

1. Employ reasonable and appropriate security practices in the design and customization of the software on its mobile devices.
2. Provide its engineering staff with adequate security training
3. Review or test the software on its mobile devices for potential security vulnerabilities
4. Follow well-known and commonly accepted secure coding practices
5. Establish a process for receiving and addressing vulnerability reports from third parties.
Case Study: HTC America settles FTC charges

Consequences of Failures:

- Insecure implementation of two logging applications on HTC devices - Carrier IQ and HTC Loggers -> theft and leakage of private consumer data
- Introduction of programming flaws that allow third-party applications to bypass Android’s permission-based security model.

Mobile Privacy Concerns Prompt Class Action Lawsuit Over Carrier IQ:

HTC settles with FTC over leaving Carrier IQ and other logging tools open to hackers:
http://www.theverge.com/2013/2/22/4017746/htc-settles-with-ftc-over-insecure-logging-software
Consequences of Failures:

- Millions of HTC devices compromised sensitive device functionality, permitting malicious applications to:
  - Read and Send SMS text messages
  - Record audio
  - Install additional malware onto a consumer’s device
  - Record and transmit data entered into or stored on the device:
    - Financial account numbers and related access codes or medical information such as text messages received from healthcare providers and calendar entries concerning doctor’s appointments.
  - Gain access to users geo-location data
  - Read and Write private contact data on devices
FTC Requirements:

- Develop and release software patches to fix vulnerabilities found in millions of HTC devices.
- Establish a comprehensive security program designed to address security risks during the development of HTC devices.
- Undergo independent security assessments every other year for the next 20 years.
- HTC America prohibited from making any false or misleading statements about the security and privacy of consumers’ data on HTC devices.
Case Study: HTC America settles FTC charges

- FTC introduced *Mobile App Developers: Start with Security*
  A new business guide that encourages app developers to aim for reasonable data security. Set of high-level secure coding best practices and guidelines.

IBM AppScan Source Edition can enforce these guidelines and more

Enforcing FTC “Start with Security” Guidelines

- Use transit encryption for usernames, passwords, and other important data. Deploy SSL/TLS in the form of HTTPS
  - AppScan Source can identify code where SSL/TLS is not enforced
- Use due diligence on libraries and other third-party code
  - AppScan Source can scan third-party code and libraries for vulnerabilities
- Protecting data you store on a user’s device.
  - AppScan Source can identify when no encryption or weak encryption is used
- Don’t store passwords in plaintext
  - AppScan Source can identify issues with hard coded passwords and unprotected credentials
- Maintain and Protect server communication
  - AppScan Source can identify cross-site scripting, injection issues, etc.
- Application Permissions
  - AppScan Source can identify what permissions an application has been provided
Example: SQL Injection from implicit intent

Sends malicious search as an implicit intent

Content Provider returns all records stored in SQLite DB

Confidential data exposed
Example: Malicious intent reaches query method of content provider
SQLi (try finding this with a manual code review)