OWASP Mobile Top Ten 2014
Meet the New Addition
Agenda

• OWASP Mobile Top Ten 2014
  – Lack of Binary Protections added
  – Why is Binary Protection important?
• What Risks Need to be Mitigated?
• Where to Go For Further Guidance
What’s “Lack of Binary Protections” All About?

OWASP MOBILE TOP 2014
OWASP Mobile Top Ten 2014

• Unveiled at AppSec California 2014
  – January 2014;
  – Categories based on data collected by a number of different security vendors, consultancies;

• New Category Introduced:
  “Lack of Binary Protections”
## Mobile Top Ten 2013 -> 2014

<table>
<thead>
<tr>
<th>Category</th>
<th>2013</th>
<th>2014</th>
</tr>
</thead>
<tbody>
<tr>
<td>M1</td>
<td>Insecure Data Storage</td>
<td>2013 M2 + 2013 M10</td>
</tr>
<tr>
<td>M2</td>
<td>Weak Server Side Controls</td>
<td>2013 M1</td>
</tr>
<tr>
<td>M3</td>
<td>Insufficient Transport Layer Protection</td>
<td>2013 M3</td>
</tr>
<tr>
<td>M4</td>
<td>Client Side Injection</td>
<td>2013 M8 + 2013 M10</td>
</tr>
<tr>
<td>M5</td>
<td>Poor Authorization and Authentication</td>
<td>2013 M5</td>
</tr>
<tr>
<td>M6</td>
<td>Improper Session Handling</td>
<td>2013 M9</td>
</tr>
<tr>
<td>M7</td>
<td>Security Decisions via Untrusted Input</td>
<td>2013 M4</td>
</tr>
<tr>
<td>M8</td>
<td>Side Channel Data Leakage</td>
<td>2013 M7</td>
</tr>
<tr>
<td>M9</td>
<td>Broken Cryptography</td>
<td>2013 M6</td>
</tr>
<tr>
<td>M10</td>
<td>Sensitive Information Disclosure</td>
<td><strong>Lack of Binary Protections</strong></td>
</tr>
</tbody>
</table>
What is “Lack of Binary Protections” All About?

1. Software in untrusted environments is exposed to reverse-engineering, analysis, modification, and exploitation by attackers

2. Attackers can directly access the binary and compromise its integrity with various tools and techniques

3. Attackers may cause brand, revenue, or IP loss through reverse-engineering
What Do Binary Attacks Result In?

Compromise (disable, circumvent) of security controls, e.g., authentication, encryption, license management / checking, DRM, root / jailbreak detection

Exposure of sensitive application information, e.g., keys, certificates, credentials, metadata

Tampering with critical business logic, control flows, and program operations
What Do Binary Attacks Result In?

- Insertion of **malware or exploits** in the application and repackaging
- Exposure of **application internals** (logic, vulnerabilities) via reverse-engineering
- **IP theft** (e.g., proprietary algorithms) via reverse-engineering
- **Piracy** and unauthorized distribution
How Prevalent Are Binary Attacks?

• **HP Research Reveals Nine out of 10 Mobile Applications Vulnerable to Attack**, 18 November 2013:
  
  “86 percent of applications tested lacked binary hardening, leaving applications vulnerable to information disclosure, buffer overflows and poor performance.”

• **Arxan Research - State of Security in the App Economy, Volume 2**, November 2013:
  
  “Adversaries have hacked 78 percent of the top 100 paid Android and iOS apps in 2013.”
2013 Arxan Study

- Analyzed Top 100 Apps for Android / iPhone for serious flaws

- Binary / HTML Modification extremely common
Goals of Binary Attacks

• What were the hackers interested in doing with these cracked apps?
  – Security Control Bypass
  – Adware / Spyware Code Injection
  – Repackaging (IP Theft)
  – Stealing Information About Users
2012 Arxan Study – Android Banking Vulnerabilities

- Info leak via symbols
- Info leak via strings
- Decompilation
- Tampering

Legend:
- Not vulnerable
- Vulnerable
WHAT RISKS NEED TO BE MITIGATED?

Technical Risks and Solutions
Android / iPhone Technical Risks

- Business Risk
- Operational Risk
- Technical Risk
  - Confidentiality Risk
  - Integrity Risk
  - Reverse Engineering and Code Analysis Risk
  - Code Modification / Injection Risk
Reverse Engineering Risks

- Exposed Method Signatures
- API Monitoring
- Exposed Data Symbols
- Exposed String Tables
- Algorithm Decompilation and Analysis
- Application Decryption
Cryptographic Key Theft

Flag hardcoded keys that could be easily found by an attacker through static or dynamic analysis.
AntiDebugger Checks

Common app entrypoints should check for the unauthorized presence of a debugger.
Code Modification Risks

- Code Modification Technical Risks
  - Repackaging
  - Method Swizzle With Behavioral Change
  - Security Control Bypass
  - Automated Jailbreak / Root Detection Disable
  - Presentation Layer Modification
  - Cryptographic Key Replacement
This method will likely be swizzled and modified by an attacker.
Automated Jailbreak Bypass

```c
-(BOOL) isJailbrokenEnvironment {
    NSFileManager *filemgr = [NSFileManager defaultManager];

    BOOL jailbrokenEnvironment = [filemgr fileExistsAtPath:@"/Applications/Cydia.app"];
    return jailbrokenEnvironment;
}
```

NOTE: Methods that appear to return a simple yes/no response and appear to be doing something sensitive are excellent candidates for simple code modification.
Useful OWASP Projects

FURTHER GUIDANCE
Practical Solutions

1. Implement Adequate Algorithms for
   – Jailbreak / Root Detection (see xcon);
   – Checksum Controls;
   – Certificate Pinning Controls; and
   – Debugger Detection Controls

2. Protect these algorithms from:
   – Reverse Engineering
   – Unauthorized Code Modification
Practical Solutions

Your mobile app must be able to:

1. Prevent an adversary from reverse engineering sensitive parts of your app;
2. Detect at runtime that code modification has occurred;
3. React appropriately at runtime to integrity violations
Practical Solutions:
Follow a “Defense in Depth” Approach
Conclusions

• Binary attacks are extremely common and are much riskier than you think…

• OWSAP Mobile Top Ten 2014 Category “Lack of Binary Protections” is new and directly addresses this new threat

• To mitigate this threat, your app must strive to prevent reverse engineering and code modifications by an adversary
Useful OWASP Projects

• Check out “OWASP Mobile Top Ten 2014 Project – M10” For More Information
  https://www.owasp.org/index.php/Mobile_Top_10_2014-M10

• For more specific guidance and recommendations:
  Reverse Engineering and Code Modification Prevention OWASP Project
For more info on Arxan Technologies: http://www.arxan.com

THANK YOU!