Abstract

As the world becomes increasingly reliant on mobile technologies, so has the importance of research in the area of mobile security in order to protect users and their data. My research is focused on the Android permissions-based security model and enhancing the effectiveness of Android application permission analysis.

Android Permissions

Prior to installation, an Android application must explicitly request permissions before it can be installed on an Android device. A list of the permissions the application is requesting is presented to the user, and the user must choose whether to install, accepting those permissions, or cancel, declining those permissions. However, it is not necessarily transparent to the user whether an application actually needs all of the permissions (system resources) that it requests.

Underprivilege - If an application is not requesting enough permissions, then those features that require those missing permissions will not function properly and the application will generate an error which may cause the application to crash.

Overprivilege - If an application is requesting too many permissions, then the application violates the principle of least privilege and may compromise the underlying operating system which may cause the device to become vulnerable to a malicious attack.

Therefore, one of the challenges to this permission-based approach is ensuring that an application is requesting the right set of permissions. However, the user cannot modify the applications permissions; it is up to the application developer to select the appropriate set of permissions for their application.

Methodology

1. Static analysis
   
   Pros:
   - Theoretically sound (no false negatives)
   - (modulo native methods, reflection, and callbacks)
   - No need for test cases
   - No need for configuration
   - Complete (no false positives)
   - Safe
   
   Cons:
   - Unsound in practice (false negatives)
   - Conservative (false positives)

2. Dynamic analysis
   
   Pros:
   - Complete (no false positives)
   
   Cons:
   - Unsound (false negatives)
   - Need for test cases
   - Need for configuration
   - Unsafe (program under analysis can compromise the integrity or confidentiality of the system)

3. Combining Static and Dynamic Analysis
   
   Pros:
   - Complete (no false positives)
   - Theoretically sound (no false negatives)
   - (modulo native methods, reflection, and callbacks)
   - No need for test cases
   - No need for configuration
   - Safe
   
   Cons:
   - (N/A)