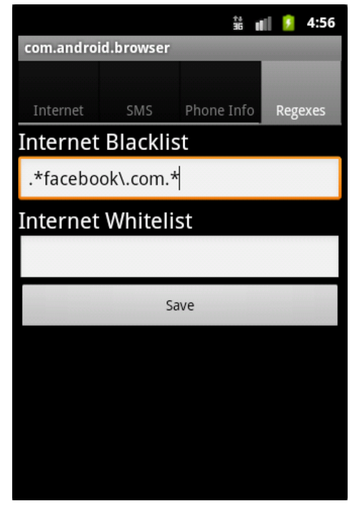
Fine-grained permission control for Android

CS 161 Spring 2012 Final Project

Team Noble 6

Edward Lu, Jason Jiang, Victor Chang, Ashwin Kamath, Josef John, Nathan Nandi

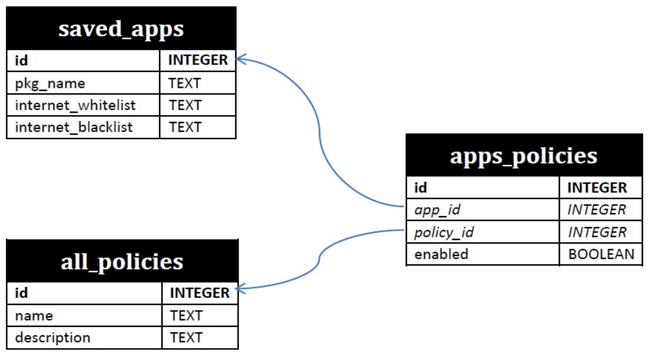


Introduction / Background

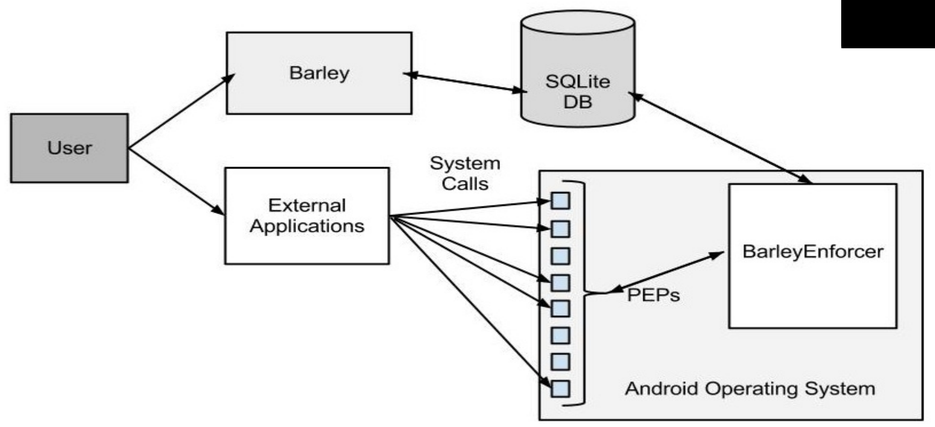
- Create a fine-grained permission control engine for the Android system.
- Current permission system is too coarse to provide users with enough security
- Allows users to define and manage their own access control policies.
- The solution? Application level-interface for users to select apps and modify their permissions

The Application

- Uses a GridView to view all the apps installed on the phone. Hooked up an event to each app icon in the view, which when clicked sends an intent to a tab view where user can modify app's policies, such as Internet (with support for regular expressions) and SMS sending.
- App stores user's policy choices in an SQLite database:



Architecture



The Operating System

- Injected code at Policy Enforcement Points (*Taming Information-Stealing Smartphone Applications*), where OS accesses Barley's database to check application permissions. Found these PEPs with the help of *Android Permissions Demystified*.
- Simplified this process by moving policy-checking mechanism into Android SDK: created a class "BarleyEnforcer", placed in an API folder.
- Permission checking became calls to BarleyEnforcer.allowed(pkg_name, permissions). For PEPs with access to Contexts, we added graceful permissin denials through use of Android Toast messages.

Conclusion

- Goal: provide users with additional layer of permission control security
- Accomplished the goal, learned much about the Android OS.
- Limitations: Certain PEPs couldn't provide application's name, so unable to implement policies. Possible solutions: stack inspection, allowing programmer to discover application identity without using API.