PermissionFlow: Detecting permission-protected information leaks in Android applications

Dragoș Sbîrlea, Michael Burke, Salvatore Guarnieri+, Marco Pistoia+, Vivek Sarkar
Rice University, IBM T.J. Watson Research Center

Android permissions
- Android maintains privacy through permissions
- Each sensitive API has associated a permission
- Calls to sensitive API without owning the corresponding permission fail.
- Permissions are accepted at application install time

Permission Security Limitations
- There is no way of knowing what the application does
- Applications are allowed to invoke each other, freely exchanging information through Intents.
- There is no way of knowing what the application does
- Applications are allowed to invoke each other, freely exchanging information through Intents.

Solution: PermissionFlow
- PermissionFlow identifies when private information can leak to other applications or to the network.
- This can be either because of misconfiguration or because the application is malevolent.

System Design

Why static analysis?
- Can find all permissions enforced in Java (sound)
- Fully automatic
- Alternative: automatic testing

Android Intents API
- Starting a child Activity:
  Intent i = new Intent();
  i.setClassName("this", "package. Activity”);
  startActivity(i);
- Returning information to parent:
  Intent intent = new Intent();
  intent.putExtra("key", "my value");
  this.setResult(Result_OK, intent);
  finish();
- Reading the returned information:
  void onActivityResult(..., Intent data) {
    intent.getStringExtra("key");
  }

Streaming CnC: Efficient unification of streaming and task parallelism

Dragoș Sbîrlea, Jun Shirako, Ryan Newton*, Vivek Sarkar
Rice University, *Indiana University

The problem
- Streaming parallelism offers excellent performance, but programmers must learn new languages or libraries
- Task parallelism offers flexibility, but lacks performance for streaming applications

Concurrent Collections
- The dataflow Concurrent Collections (CnC) model has no streaming optimizations:
  - Memory management (garbage collection)
  - Task management overhead
  - Data synchronization
  - Streaming languages
  - Lack the expressivity of dataflow models
    - No dynamic parallelism

Streaming CnC (SCnC)
SCnC = Subset of CnC that can be run with a streaming runtime as opposed of a task based one.
- Built using the synchronization primitives of the Habanero Java language.

Dynamic parallelism
- System tested on Top 500 Free Android applications.
- Three vulnerabilities in widely used applications:
  - Adobe Photoshop Express
  - Sygic GPS
  - Soundtracking

Results

Folding tagged single assignment values for memory efficient parallelism

Dragoș Sbîrlea, Kath Knobe*, Vivek Sarkar
Rice University, *Intel Corporation

The problem
- It is difficult to automatically identify when items are dead
  - Cause: item keys can be recomputed, unlike general references
  - User intervention may simplify the task and offer the opportunity for further optimization

Solution: Folding
- Transform the item keys with a folding function, which maps logical (dynamic single assignment) keys to fewer folded (multiple assignment) keys.

Results
- More results in our Europar 2012 paper!