Security Evaluation of App Runtime for Chrome

Meng Xu

Georgia Institute of Technology

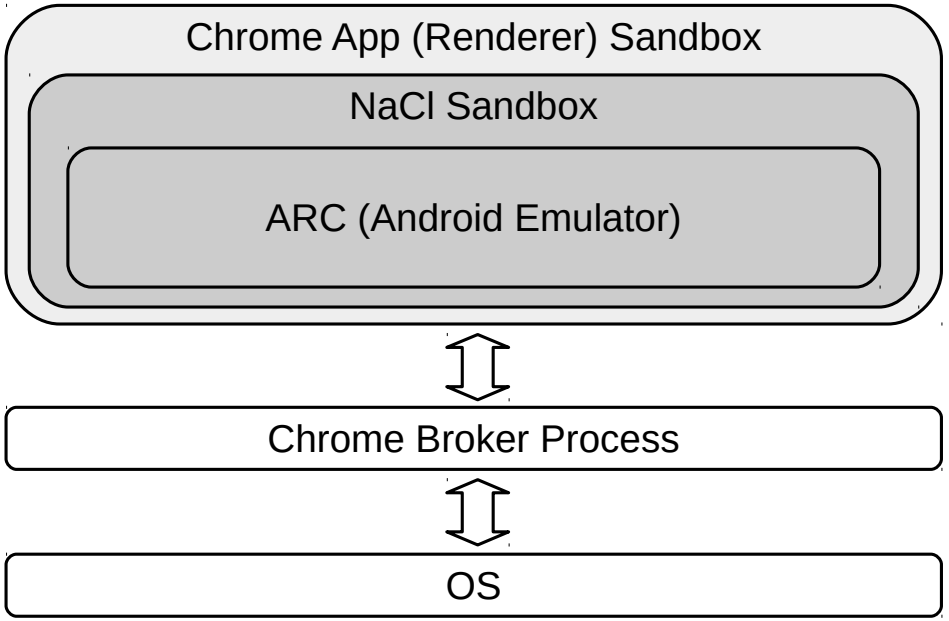
meng.xu@gatech.edu

December 1, 2014

Overview

ARC Introduction

- The goal of ARC is to build the **minimum** codebase to run a **single** Android app
- Implicitly constrained by multiple sandboxes
- Privileges operations handled by Chrome broker process (by the design of NaCl)
- Complemented by a re-packaging script



Permission Shift

- Permission model is the core of Android security
- **Problem**: Repackaged app has access to privileged operations even without declaring corresponding permissions.
- **Approach**: Instrument the repackaging script to declare correct Chrome permissions given declared Android permissions.
- Demo

Permission Shift

- Difficulties
  - Extract declarable permission list
    $\rightarrow$ Write an app to probe PackageManager on ARC dynamically
  - Map Android permissions to Chrome permissions
    $\rightarrow$ Manual process

Permission Shift

- Cause of the problem: two ways of enforcing permissions in Android
  - Assign GID to the app
    $\rightarrow$ Broken
  - Intercept API/system call to check permission
    $\rightarrow$ Still works
- Demo

Permission Shift

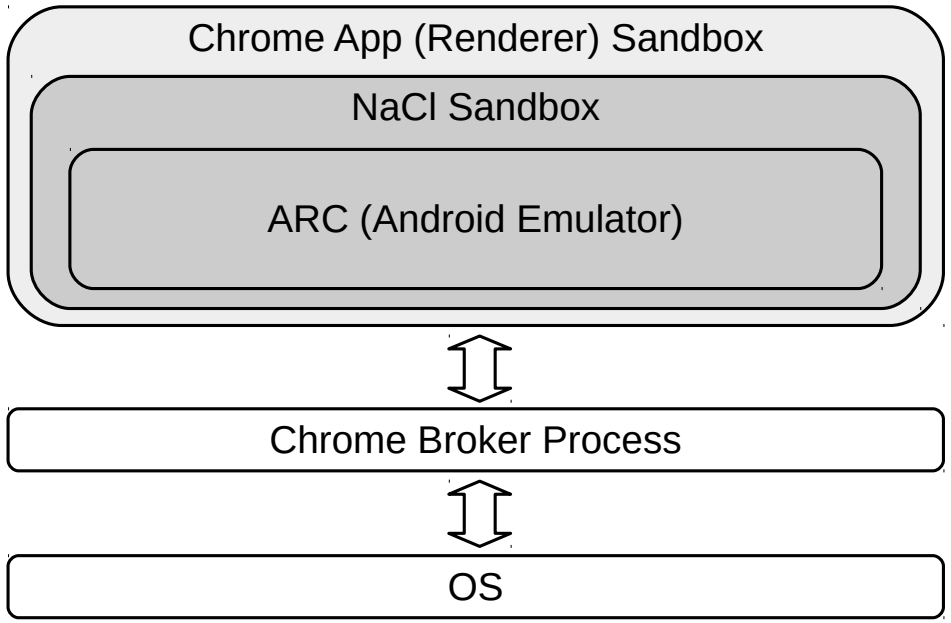
- Future work
  - Enhance the completeness of the permission shift
    $\rightarrow$ Currently support "dangerous" permissions only
    $\rightarrow$ Declarable permissions in Android and Chrome are not perfect matches
  - Bring back the GID enforcement

Inter-"Component" Communication

- Possible communications
  - App $\longleftrightarrow$ App
  - App $\longleftrightarrow$ Extension
  - App $\longleftrightarrow$ Webpage
  - App $\longleftrightarrow$ System
- General conclusion: since ARC is heavily sandboxed, there is no particular advantage gained by attacking the ARC model compared with attacking Chrome or writing an Android malware.

Inter-"Component" Communication

- App $\longleftrightarrow$ App
- App $\longleftrightarrow$ Extension
- App $\longleftrightarrow$ Webpage
- App $\longleftrightarrow$ System



Inter-"Component" Communication

- App $\longleftrightarrow$ App
  - System privilege escalation attack does not make sense
  - Component hijacking (of another app) is not possible

Inter-"Component" Communication

- App $\longleftrightarrow$ Extension
  - Chrome extension may cause a DoS on Android app (Demo)
  - Chrome extension may view the cookies generated from Android WebView (Demo)
  - Android app has no way of influencing Chrome extensions

Inter-"Component" Communication

- App $\longleftrightarrow$ Webpage
  - No interaction

Inter-"Component" Communication

- App $\longleftrightarrow$ System
  - Data stored in both "internal storage" or "external storage" are not safe (Demo)
  - Apps may have access to OS filesystem (via browser file chooser), but will not be able to modify them (Demo)

Inter-"Component" Communication

- Future work
  - Going systematical
    $\rightarrow$ Apply dynamic taint analysis or static model checking to test the interaction between ARC and extension/webpage/system
  - Side-channels / covert-channels

Questions ?

Source code available at `https://github.com/meng-xu/arc-security`.