Security Evaluation of App Runtime for Chrome

Meng Xu

Georgia Institute of Technology

meng.xu@gatech.edu

October 20, 2014

Overview

1. ARC Introduction

2. Motivation

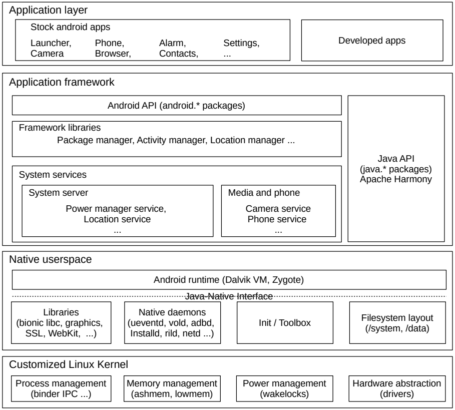
3. Schedule

4. Evaluation

ARC Introduction

- Allows Android apps to run in Chrome, without porting
- Officially designed for Chrome OS
- ARChon Custom Runtime allows every major OS with Chrome browser to run Android apps
- Accompanied by a re-packaging script

ARC Introduction



Application layer

Stock android apps

Launcher, Camera

Phone, Browser,

Alarm, Contacts,

Settings, ...

Developed apps

Application framework

Android API (android.* packages)

Framework libraries

Package manager, Activity manager, Location manager ...

System services

System server

Power manager service, Location service ...

Media and phone

Camera service Phone service ...

Java API (java.* packages) Apache Harmony

Native userspace

Android runtime (Dalvik VM, Zygote)

Java-Native Interface

Libraries (bionic libc, graphics, SSL, WebKit, ...)

Native daemons (ueventd, vold, adbd, Installd, rild, netd ...)

Init / Toolbox

Filesystem layout (/system, /data)

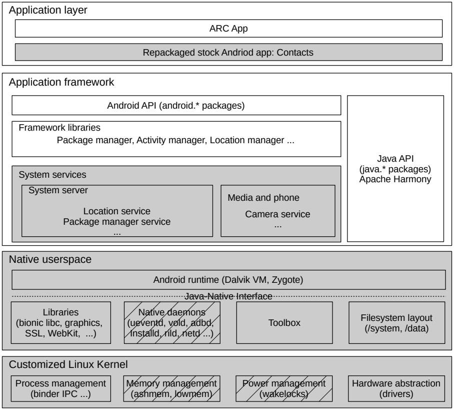
Customized Linux Kernel

Process management (binder IPC ...)

Memory management (ashmem, lowmem)

Power management (wakelocks)

Hardware abstraction (drivers)



Application layer

ARC App

Repackaged stock Andriod app: Contacts

Application framework

Android API (android.* packages)

Framework libraries

Package manager, Activity manager, Location manager ...

System services

System server

Location service Package manager service ...

Media and phone

Camera service ...

Java API (java.* packages) Apache Harmony

Native userspace

Android runtime (Dalvik VM, Zygote)

Java-Native Interface

Libraries (bionic libc, graphics, SSL, WebKit, ...)

Native daemons (ueventd, vold, adbd, Installd, rild, netd ...)

Toolbox

Filesystem layout (/system, /data)

Customized Linux Kernel

Process management (binder IPC ...)

Memory management (ashmem, lowmem)

Power management (wakelocks)

Hardware abstraction (drivers)

Motivation

- Brand new combination, still in proof-of-concept stage which means a lot of security-related issues unaddressed
- Turning Android into a universal runtime that works securely on any computing device with Chrome installed

Problem & Approach

- Yet fully open sourced, very limited resources available
- Past research on Android and NaCl, but not the combination of them
- Time constraints

Problem & Approach

Problem definition

Focus on security issues related to permission and inter-"component" communication

- Shift from Android permission model to Chrome Extensions permission model
- Interaction between web apps (Javascript), Chrome Extensions and Android apps

Approaches

Hack, Try-and-error, Reverse engineering

Problem & Approach

Permission model shift

Current implementation of ARC does not respect Android permission model, instead, it relegates the permission checking to Chrome Runtime.

- Map between Android permission model to Chrome Extensions permission model
- Patch the packaging script to automatically do the permission shift

Inter-"component" communication

Current implementation of ARC is based on Chrome Extension architecture, which is possible to be accessed by other components in the Chrome runtime, e.g., web apps, other extensions etc.

- chrome://inspect/

Schedule

- Permission model - 2 weeks
- Inter-”component” communication - 3 weeks
- Summary and presentation - 1 week

Evaluation

Permission model shift

- Completeness and effectiveness of permission map
- Proposals to leverage Chrome Extensions' finer-grained permission model to enhance Android app security

Inter-"component" communication

- Possibility of launching attacks from outside of ARC
- Possibility of mitigating these attacks

Questions ?