Android Security

Christian Küster <c.kuester@tarent.de>

↪ tarent

LinuxTag 2009

Agenda

- **Sicherheitskonzept** der Android-Plattform (eine Bestandsanalyse)
  - Einführung - Was ist Android?
  - (Interessante) Komponenten von Android
  - Binäre Einschlüsse
  - Berechtigungskonzept und Durchsetzung
  - Zugriffsschutz und Prozess-Isolation
- **Erweiterung** um Sicherheitsfunktionen
  - Virtual Private Network
  - Dateiverschlüsselung
  - SmartCards

Allgemein

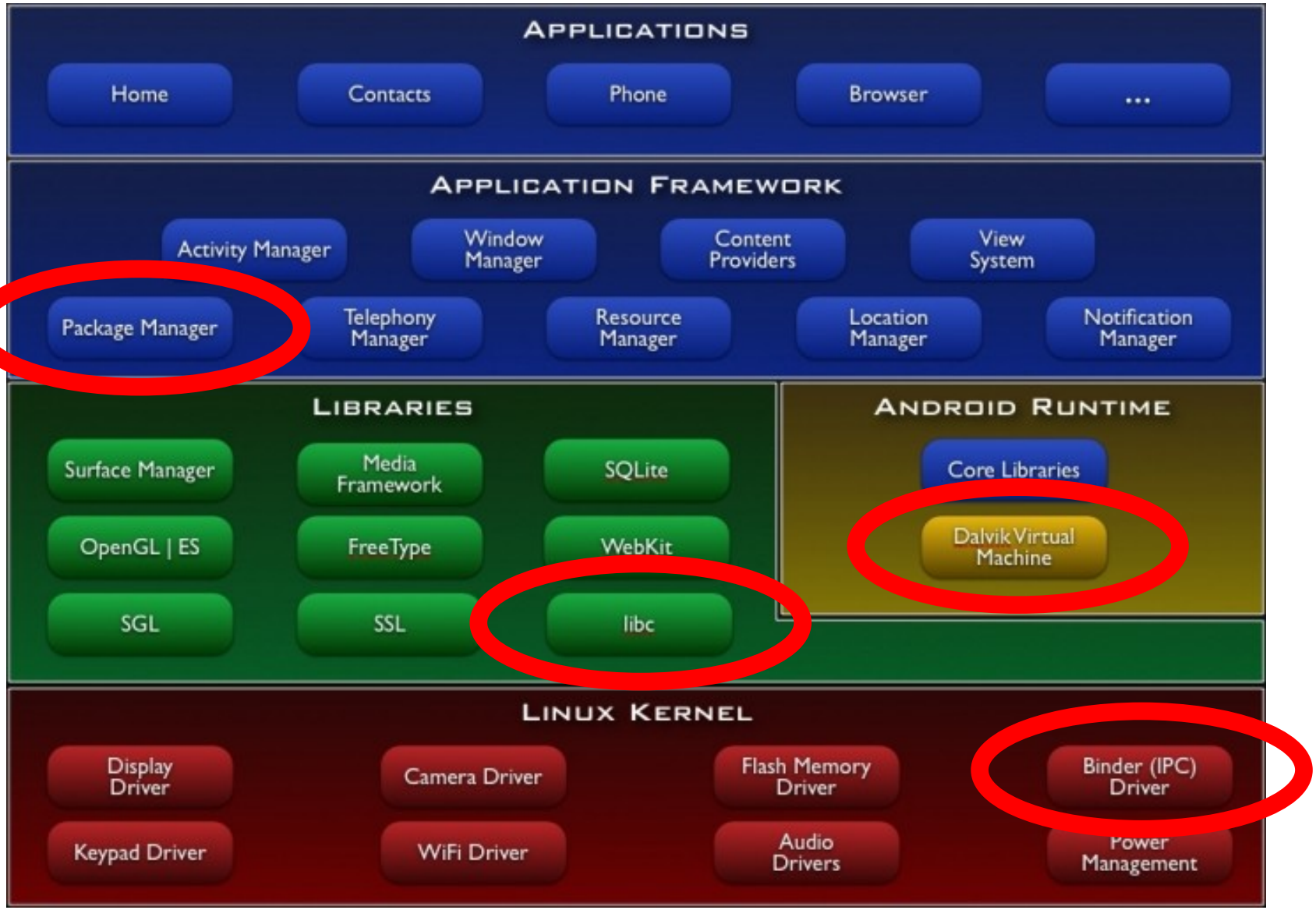
- Android ist eine (offene?) Plattform für mobile Geräte
  - PDAs, Netbooks, mobile Telefone
  - Wir haben nur ein Entwicklermodell (G1)
  - Retail-Geräte sind sehr geschlossen
- Plattform ist ein Gesamtkonzept
- Betriebssystem bis Telefon/SmartPhone Framework
  - Basiert auf Linux
  - Basisset von Telefon/SmartPhone Applikationen
- Sicherheit der Plattform offensichtlich Entwurfsziel
  - Im Sinne eines Sicherheitsmodells
  - Damit ist kein Code-Review oder “sichere Programmierung” gemeint!

Motivation – The Neverending Why

↪ tarent

- Malware ist auch für Telefone eine Gefahr
- SmartPhones enthalten eine Reihe von sensiblen Gütern
    - Kontaktdaten
    - Adressen von Remote-Peers
    - Browser-Historie
    - Historien von Anrufen/Short-Messages
    - ...
- Wussten Sie, dass Ihr iPhone Screenshots von laufenden Applikationen ablegt? (vgl. „Mobile Malware“, Dunham, 2009, p. 314)
- Android ist quelloffen und es weckt Neugier

System-Architektur



Komponenten von Android .. ? (eine Auswahl)

↪ tarent

- Kernel mit Android-Spezifischen Anpassungen
    - Ashmem
    - Logger
- OpenBinder (oder Binder) ist verwendetes IPC-Framework
    - System-V-IPC wurde unter Sicherheitsbedenken entfernt
- Bionic ist ein Subset der NetBSD-Standard-C-Bibliothek
- Framework
- Virtual Machine: Dalvik
- Ein Set von externen Bibliotheken (OpenSSL, webkit, ...)

Binäre Einschlüsse (G1)

- Entwicklertelefon G1 (HTC Dream)
- Radio Layer Userspace-Bibliothek
- Wlan Kernel-Modul
- GPS-Treiber
- Audio-Treiber
- Kamera-Treiber

Applikationen

- Android bringt einen Paketmanager mit
- Applikationspakete sind immer vom Entwickler signiert
    - Entscheidet nicht, ob Applikation überhaupt installiert werden darf
    - Vertrauensbeziehung zwischen Applikationen
- Applikationsdeployer weißt automatisch neue Linux Benutzer-Identität (UID) zu
    - UID-Sharing nur bei gleicher Entwicklersignatur
- Bionic bietet spezielle Funktionen dafür
    - kein /etc/passwd bzw. shadow
- Applikationen bestehen zur Zeit nur aus Dalvik-Byte-Code
- Dalvik-Byte-Code wird aus Java-Byte-Code transformiert

Berechtigungssystem

↪ tarent

- Um bestimmte Aktionen überhaupt durchführen zu können muss einer Applikation die Berechtigung gegeben werden
- Deny-All Ansatz für zu installierende Applikationen
- Berechtigungen werden statisch vergeben
    - AndroidManifest.xml
    - Eine vordefinierte Menge steht bereit: INTERNET_ACCESS, READ_CONTACTS, ...
    - Benutzer entscheidet bei der Installation
    - Werden während der Laufzeit geprüft
- Es können auch Berechtigungen selbst definiert werden
    - Für den Schutz eigener Services
- Jede Berechtigung hat ein Risk-Level

Durchsetzung von Berechtigungen

- Anhand von Linux-Gruppen
    - Mapping: /etc/permission/platform.xml
- OpenBinder
    - IPC ist ebenfalls für einige Durchsetzungen verantwortlich
- Java-Security-Manager?
- Kernel
    - Beispiel INTERNET_ACCESS
    - Der Kernel wurde erweitert
    - Überprüfung der Gruppenzugehörigkeit “inet”
- Eskalation der Zugriffsverletzung ins Framework (SecurityExceptions)

Durchsetzung von Berechtigungen – Beispiel Netzwerk

Aus dem Kernel: ./net/ipv4/af_inet.c

```
#ifdef CONFIG_ANDROID_PARANOID_NETWORK
static inline int current_has_network(void) {
    return (!current_euid() || in_egroup_p(AID_INET) ||
        in_egroup_p(AID_NET_RAW));
}
static inline int current_has_cap(int cap) {
   if (cap == CAP_NET_RAW && in_egroup_p(AID_NET_RAW))
       return 1;
   return capable(cap);
}
# else
static inline int current_has_network(void) {
   return 1;
}
static inline int current_has_cap(int cap) {
    return capable(cap);
}
#endif
```

Prozess-Isolation

- Android ist ein Multiprozess-System
- Linux-Prozess als Vertrauenseinheit
- Eigener Benutzer für jede Android-Applikation
    - Paketmanager weißt diese zu
- UID-Sharing möglich (vgl. Vertrauensbeziehung)
- System-V-IPC entfernt
    - Interprozesskommunikation durch Binder
- Virtualisierung durch Dalvik-VM Verteidigungslinie?
    - Vorerst ja -> JNI für Applikationen ist Entwurfsziel der Plattform

Zugriffsschutz

↪ tarent

- Basiert auf Linux Discretionary Access Control
    - -rw-r--r-- 1 christian christian 248K 20. Jun 18:42 android-security.odp
- Android-Applikationen legen Dateien ausschließlich unter ihrer Benutzer-ID ab
- Android bietet eigene Konzept zur Datenkapselung an: Content-Provider
    - Zugriff kann öffentlich gemacht werden
    - Kann mit Zugriffsberechtigung versehen werden!
    - Beispiel: Kontaktdaten sind durch Content-Provider gekapselt (READ_CONTACTS)
    - Daten sind auch abgelegt unter eigener Benutzer-ID
- WORLD_{WRITEABLE|READABLE} Berechtigung möglich
- /data/data/<paketname>/

Zugriffsschutz - Partitionsebene

↪ tarent

- /, /system, /data, /cache, ...
- Schreibbar ist grundsätzlich nur die /data Partition
- Alle Systemprogramme und Bibliotheken sind nach /system verlagert und nur lesbar
- Root (/) ist nur lesbar
- SD-Karten werden „noexec“ eingehangen
- Updates von Systemprogrammen (Beispiel: Dialer) werden durch ein Overlay in /data ermöglicht

Erweiterung um Sicherheitsfunktionen

- Geht es überhaupt?
    - Die hier vorgestellten Sicherheitsfunktionen benötigen Änderungen bzw. Erweiterungen auf Kernel und Systemebene
    - Daher nur auf einem Entwicklertelefon
    - Bedingte Flash-Möglichkeit von Retail-Geräten
    - Kernel muss meistens geflasht werden
- Was ist interessant?
    - Virtual Private Network Clients
    - Dateiverschlüsselung
    - SmartCards als Vertrauensanker

Virtual Private Network

- Sehr interessant!
    - Lösungen stehen anscheindend bereit
    - Problem: Kernel-Config Änderungen nötig
        - tun/tap für OpenVPN
        - IPsec/l2tp
            - Strongswan setzt auf Funktionen auf, die nicht in bionic vorhanden sind (thread cancellation)
- Google Clients arbeiten alle auf https Basis
    - Zum Beispiel Kontakte synchronisieren
- Keine Unterstützung für System-Policies
- OpenVPN Binary kann bei mir erfragt werden ;-)

Dateiverschlüsselung

↪ tarent

- Telefone bergen ihre ganz eigenen Tücken
- Einige OSS-Alternativen
    - Dm-crypt, Loop-AES, eCryptFS, ...
- Dm-crypt und Loop-AES setzen Container ein, wir haben jedoch YAFFS2 als Dateisystem wegen eingebauten NAND-Speichern
    - Außer DM-Crypt bisher nichts im Standard-Kernel
- EcryptFS braucht in gewissen Fällen Shared-Memory
    - Userland Werzeuge haben eine Reihe Abhängigkeiten
    - Anpassung auf binder/ashmem nötig (für bestimmte Betriebsarten)
    - Nicht im Kernel
    - Sys-Dateisystem nötig, auch nicht vorhanden

Dateiverschlüsselung (2)

- „Cold-Boot“-Angriffe
  - PCs macht man irgendwann aus
  - Mobile Telefone meistens nicht
  - Sie kommen häufig weg, wenn sie noch angeschaltet sind
  - Nützt mir dann Verschlüsselung überhaupt was?
- Initialisierung der Container
  - RNG auf Embedded Devices tricky
- SmartCard als Vertrauensanker

SmartCards unter Android

↪ tarent

- Zwei Probleme: Hardware und Software
    - Kein CardReader, Alternativen?
        - USB OTG?
        - Bluetooth?
        - MicroSD-Slot (Karte der Fa. CertGate)
            - Hat einen RNG!
            - Eigene API
    - Keine PKCS11 API wie in Sun JVM vorhanden
    - OpenSC hat Java-Bindings
- SmartCard Unterstützung braucht auch native Komponenten

Artefakte ...

↪ tarent

- SmartCard Middleware: pcsc-lite/pcscd (ungetestet)
  - Jemand hier, der eine Idee hat?
- GnuPG Version 1
  - Random Number Generator?
  - libgcrypt, libgpg-error (GnuPG 2)
- eCryptFS-utils (mit Abhängigkeit „keyutils“)
- Loop-AES (losetup)
- OpenVPN

Zusammenfassung

- Android hat ein gut durchdachtes Sicherheitsmodell
- Berechtigungssystem ist eine Kompromiss zwischen Benutzerfreundlichkeit und Sicherheitsmaßnahmen
- Erweiterungen bedingt möglich
    - Benötigt Entwicklertelefon
    - Binäre Einschlüsse die Regel
    - Hoher Portierungsaufwand für Abhängigkeiten von Software und Subset der Standard-C-Bibliothek
- The Future is bright!

Fragen ..

0x3F