Taming the Robot: Efficient Sand-boxing of the Android OS

Steffen Liebergeld, March 22nd, 2011
steffen@sec.t-labs.tu-berlin.de
Outline

- Introduction
- Virtualization
  - Microkernels
  - L4Linux
- L4 Android
- Conclusion
Introduction

- Open Source
- Custom 3\textsuperscript{rd} party Apps
- Linux kernel
- New business models

- Insufficient security policies
- Software not up-to-date
- Linux kernel
  - Outdated
  - Custom drivers
  - Recent study found 88 flaws
Android Security – Press Coverage

- Apps found to “leak” private data
- “Infected” Android Apps discovered in Android Market
  - Downloaded > 50,000 times
  - Sent private information to the attacker
- Android Trojan to send (expensive) premium SMS
- Study using static code analysis found 88 critical flaws in the kernel
Security Analysis

- Android kernel at the lowest layer in software stack
  - Critical to availability, confidentiality and integrity
  - In TCB of all components
  - Insufficient access control mechanisms
    - ACLs, Users, Groups...
- Kernel contains about 14 million SLOC
  - Device drivers
  - Protocol stacks (e.g. network)
  - Filesystems
- No in-kernel isolation
  - Any vulnerability is fatal
Virtualization

- Flaws inherent with Android architecture
  - Android not suited for high-security applications
- Solution: Sand-boxing, Virtualization
  - Take Android vulnerabilities into account
  - ... but limit their effects
Virtualization

- Ability to run multiple instances of Android concurrently on one device
- Enables new opportunities for preventive security measures:
  - Out-of-band security analysis
  - Run security sensitive tasks besides Android (e.g. smartcard services, micropayment)
  - Arbitrate hardware access
  - Multiple Androids with different security clearings
Virtualization - Problems

- Virtualization layer is new attack vector
- Smart phone CPUs not virtualizable
- Performance
- Needs to be done right!
Microkernels

- Design principles
  - Implement only functionality in kernel that cannot be implemented at user level
  - Hardware enforced isolation boundaries (Address spaces)
  - Fast, explicit communication (IPC)
  - Secure access control mechanism (Object capabilities)

- Benefits:
  - Flexibility: enable per-application resource allocation strategies
  - Limit scope of faults
  - Control information flow
  - Tailored TCB for individual applications

- Added benefits
  - Execute real-time applications beside non-real-time applications
  - Supports virtual machines

- Forms a secure basis for our approach
L4Linux – Solving the Performance Problem

- Many Smart phone CPUs not natively virtualizable
  - Emulation (slow)
  - Binary translation (slow, huge effort)
  - De-privileging (good performance, but large initial porting effort)

- L4Linux:
  - Port of the Linux kernel
  - Runs in its own address space
  - Binary compatible at Linux kernel ABI
  - Applicable to non-virtualizable platforms
  - Good performance in most workloads
  - Implemented and maintained at TU-Dresden
L4 Android

- Effort to transform stock L4Linux into L4Android
  - Make L4Linux run Android userland

- Adaptions:
  - Port of Android code to current L4Linux
  - Packaging of Android userland into ramdisk
  - Lots and lots of debugging

- State of the Art:
  - L4 Android works (proof of concept)
  - Donut (1.6), Eclair (2.1) and Froyo (2.2) supported
  - Used as research vehicle

- Work in progress:
  - Virtualize mass storage, modem
  - Implement fast and stable graphics driver
  - Design secure GUI
L4android.org

- Open Source Project
- Website: l4android.org

DEMO
Conclusion

- Virtualization can help with security
  - (if implemented correctly)
- Microkernel forms a suitable basis
  - Provides isolation
  - Allows isolated high-security components (micropayment, smartcard)
- L4 Android
  - Efficient virtualized Android
  - Out-of-band security measures possible
Thank you!
References