# KTI - Kapsch Tablet Infrastructure

Götze Sebastian, Hammer Samuel, Kaufmann Michael,
Müller Konstanze, Steinhäuser Philip


DIPLOMARBEIT

eingereicht am
Fachhochschul-Diplomstudiengang

Abteilung Höhere Informatik

in HTBLVA Spengergasse

im Mai 2015

# Erklärung

Ich erkläre eidesstattlich, dass ich die vorliegende Arbeit selbstständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen nicht benutzt und die den benutzten Quellen entnommenen Stellen als solche gekennzeichnet habe. Die Arbeit wurde bisher in gleicher oder ähnlicher Form keiner anderen Prüfungsbehörde vorgelegt.

HTBLVA Spengergasse, am 15. Mai 2015

Götze Sebastian, Hammer Samuel, Kaufmann Michael, Müller Konstanze, Steinhäuser Philip

# Inhaltsverzeichnis

# Kurzfassung

## 0.1 Deutsche Version

### 0.1.1 Aufgabenstellung

Die Aufgabe von Kapsch an das Projektteam ist es, verschiedene Softwarelösungen zum Systemschutz von Android-Tablets zu testen. Dazu soll ein Untersuchungsbericht angefertigt werden. In diesem werden die Ergebnisse festgehalten und verglichen, um das beste System für die Kunden von Kapsch zu ermitteln. Ein Prototyp in Form eines Android-Tablets ist ebenfalls zu erstellen. Dieser soll mit dem besten, aus dem Untersuchungsbericht hervorgegangenen, Schutzsystem ausgestattet sein.

### 0.1.2 Realisierung

Die Firma Kapsch wünscht einen Untersuchungsbericht, aus dem hervorgeht, welcher Systemschutz für Android-Tablets am besten für spezifische Kunden geeignet ist. In diesem Bericht werden 2-3 verschiedene Systeme zur Absicherung eines Android-Tablets verglichen. Darin wird auf die Vor- und Nachteile des jeweiligen Systems eingegangen, sowie auf das Fehlen von Funktionen aufmerksam gemacht. Dies ermöglicht dem Auftraggeber das bestmögliche System für seine Kunden zu ermitteln.

### 0.1.3 Ergebnisse

Neben dem Untersuchungsbericht wird zusätzlich ein Tablet als Prototyp abgeliefert. Dieses Tablet ist mit dem Schutzsystem ausgestattet, welches aus dem Untersuchungsbericht als am empfehlenswertesten hervorgeht.
Am Ende des Projekts stehen 2 Endprodukte.

#### Der Untersuchungsbericht

Er vergleicht 2 bis 3 Softwarelösungen zur Absicherung von Tablets. Pro Lösung müssen ihre Vor- und Nachteile vorhanden sein. Sowie auch das Fehlen von Einstellungsmöglichkeiten.

Am Ende des Untersuchungsberichts hat ein Vergleich der Systeme zu stehen, aus dem herausgeht, welches, nach der Meinung des Projektteams, eingesetzt werden sollte. Dies muss mit Argumenten bekräftigt werden.

**Der Prototyp**

Ein Android-Tablet welches auf die Anforderungen eines Beispielunternehmens zugeschnitten ist. Dabei sollen nur bestimmte Aktionen möglich sein.

- Nutzung von 3 bestimmten Apps
- Kein Zugang zu den Einstellungen
- Keine Verbindung mit einem Computer möglich
- Kein Download von Apps, Fotos, Videos, etc.
- Kein Verlassen der gesicherten Umgebung

## 0.2 English Version

### 0.2.1 Assignment of Tasks

Our project partner Kapsch provides enterprise grade solutions for major operating systems such as Microsoft Windows 8. Due to the Consumerization trend in industry platforms like Apple iOS and Android OS are on the rise in enterprise applications and leading system integrators like Kapsch are in need to understand those platforms, their applications and limitations. Based on experience and results from existing projects Kapsch has realized that for certain applications or scenarios the Apple iOS platform has its limitations and drawbacks. In particular industrial applications have a need for a rock-solid platform which enables a system integrator like Kapsch to operate a 24/7 application and service. So Kapsch is keen to expand their knowledge and experience towards the Android platform to be able to provide solution platform for industrial applications. Aim of this cooperation is to co-work on a set of different feasibility concepts for realizing such a platform based on Android.

### 0.2.2 Realisation

The company Kapsch wants an investigation report, which shows which system protection is suitable for Android tablets best for specific customers. In this report, 2-3 different systems to secure a Android tablets are compared. It takes into account the advantages and disadvantages of each system, as well as draw attention to the lack of features. This allows the customer the best possible system for its customers to determine.

### 0.2.3 Results

In addition to the investigation report a tablet is delivered as a prototype. This tablet is equipped with the protection system, which is apparent from the investigation report as most recommendable.
At the end of the project, there are 2 final results:

#### The Investigation Report

It compares 2 to 3 software solutions for securing tablets. For each Solution, advantages and disadvantages must be present. As well as the lack of configuration options.

At the end of the investigation report, a comparison of the systems has to stand, out of the proceeds, which should be in accordance with the opinion of the project team used. This must be confirmed with arguments.

### The Prototype

An Android tablet that is tailored to the requirements of a model company. It should be possible only certain actions.

- Useof 3 specific apps
- No access to the settings
- No connection to a computer possible
- No downloading of apps, photos, videos, etc...
- Leaving the secure environment not possible

# Einleitung

## 0.3 Warum?

## 0.4 Ursprungsproblem

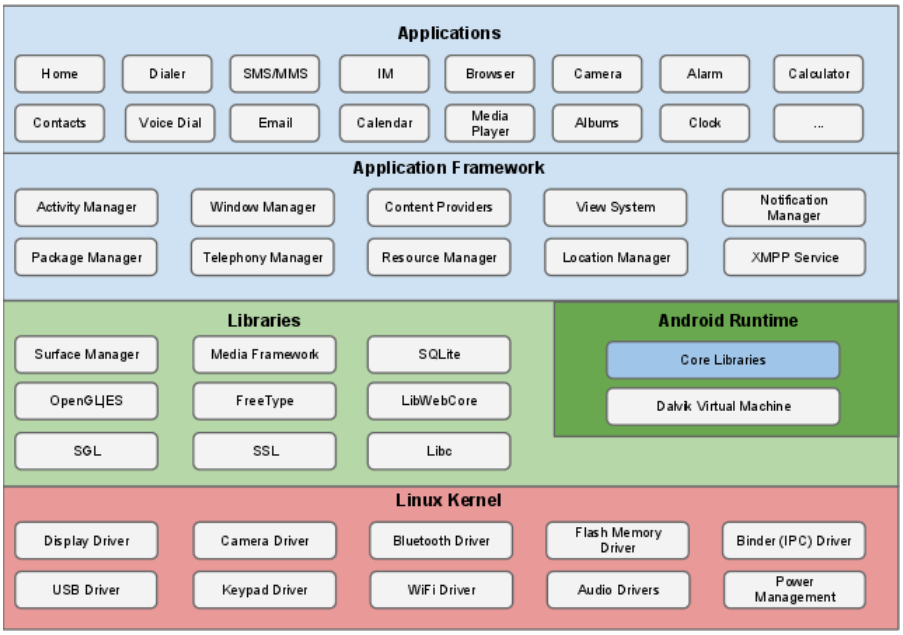
## 0.5 Vorgehensweise

# Studie

## 0.6 Android

Android ist, im Gegensatz zu anderen mobilen Betriebssystemen auf dem Markt, eine offene Plattform. Das kommt im Wesentlichen daher, dass Android auf einem open source System, dem Linux Kernel, aufbaut. Durch die freie Verfügbarkeit des Codes, gibt es im Android-Bereich eine große Developer Szene die laufend eigens modifizierte Betriebssysteme (ROMs) und Apps hervorbringt. Ein so hoher Grad an Offenheit birgt jedoch auch gewisse Sicherheitsrisiken und öffnet Angriffsvektoren. Um diese Gefahren besser zu verstehen sollte man sich eingehend mit der Android Architektur und dem darin enthaltenen Sicherheitskonzept befassen.

### 0.6.1 Architektur




Wie in Abbildung 1 zu sehen ist, bildet der Linux Kernel die unterste Schicht der Architektur. Auf ihm baut das gesamte Betriebssystem mitsamt aller Sicherheitskonzepte auf. Der Kernel stellt die Brücke zwischen Hard- und

Software dar und enthält die Treiber für diverse andere Komponenten eines Smartphones wie beispielsweise Modem, GPS Empfänger, Kamera, etc...
In der darüber liegende Ebene sind die Android Libaries sowie die Runtime, also die „Dalvik Virtual Machine“ zu finden.
Android Apps sind meist in Java Programmiert und werden jeweils in einer eigenen Virtuellen Maschine ausgeführt, der DVM [1]. Diese Systematik ermöglicht das logische, parallele, unabhängige Ausführen von verschiedenen Apps mit verschiedenen Benutzerrechten. Die Libaries steuern und kontrollieren im Wesentlichen die Funktionen des Kernels und sind für die zentralen Funktionalitäten auf niedriger Ebene zuständig.
Die nächst-höhere Ebene bildet das Application Framework. Hier befinden die Grundfunktionen von Android wie zum Beispiel Telefonie, Location Services, Window Manager, Notification Manager, etc... Auf dieser Ebene werden Entwicklern äußerst umfangreiche APIs für das Entwickeln von Benutzeranwendungen (Apps) zur Verfügung gestellt.
Die oberste Schicht in der Android-Systemarchitektur sind die Applikationen welche der Benutzer selbst installiert und auf den darunterliegenden Schichten aufbaut. Dieser gesamte „Stack“ (=Stapel) wird in einem ROM zusammengefasst und auf ein Smartphone installiert. Durch die zuvor erwähnte Offenheit des Systems können so durch Modifizieren eines ROM Paketes stark angepasste Versionen von Android entwickelt und verwendet werden.

### 0.6.2 Security

Android wird als Betriebssystem mittlerweile auf ca 84% aller Smartphones weltweit eingesetzt [2]. Ein Betriebssystem mit einem so großen Marktanteil erfordert ein durchdachtes und ausgereiftes Sicherheitskonzept. Die drei grundlegenden Sicherheitsobjekte sind:

- Schützen von Benutzerdaten
- Schützen von Systemressourcen (inklusive Netzwerk)
- Bereitstellen von Applikationsisolation

Um diese Objekte zu erreichen, stehen eine Reihe von Security-Features zur Verfügung

- Robuste Security auf der OS Ebene durch den Linux Kernel
- Erforderliche Sandbox für alle Apps
- Sichere interprozess Kommunikation
- Application signing
- Application-defined and user-grated permissons

[1] Dalvik Virtual Machine
[2] Quelle: http://www.idc.com/prodserv/ smartphone-os-market-share.jsp - Stand Q3 2014

**System- und Kernel Security**

Der Linux Kernel stellt das Fundament von Android dar. Durch dessen langjährige Wartung und Weiterentwicklung hat dieser sich zu einem äußerst sicheren und weit verbreiteten Kernel entwickelt, welcher auch in Sicherheitsempfindlichen Umgebungen eingesetzt wird.
Der Linux Kernel stellt einige der wichtigsten Sicherheitsfunktionen zur Verfügung.

- Ein Benutzer-basiertes Rechtemodell
- Prozessisolation
- Mechanismus für sichere IPC [3]
- Die Möglichkeit potentiel gefährliche Teile des Kernels zu entfernen

Im Gegensatz zum normalen Linux, wo mehrere Anwendungen mit demselben Benutzer ausgeführt werden, weist Android jeder Applikation eine eigene User ID zu und führt diese mit ebendiesem User in einem separaten Prozess aus. Diese Vorgehensweise resultiert in einer „Kernel-level Application Sandbox“. Dadurch kann eine App von Haus aus mit keiner anderen App kommunizieren sofern dies nicht explizit erwünscht ist. Standardmäßig können Apps auch nicht auf Dinge wie Standort, die Telefonie-Funktion, etc. zugreifen sondern müssen erst um die Erteilung der entsprechenden Benutzerrechte ansuchen. Die Tatsache dass diese Sandbox-Systematik auf Jahrzehnte alter UNIX Technologie aufbaut, macht das System leicht überwachbar und transparent, jedoch gleichzeitig effizient.
Speicherfehler führen in vielen Betriebssystemen zu groben Sicherheitsrisiken und sind in der Lage die Sicherheit eines Gerätes komplett zu kompromittieren. Bei Android kann ein solcher Speicherfehler durch das Sandboxing aller Applikationen auf OS-Level keinen allzu großen Schaden anrichten, da eventueller Schadcode nur im Kontext einer bestimmten App, mit eingeschränkten Benutzerrechten ausgeführt werden kann. Selbstverständlich ist auch die Applikations-Sandbox nicht zu 100% sicher, möchte man diese aber knacken, so muss man es schaffen den gesamten Linux Kernel zu kompromittieren.

### 0.6.3 Probleme

[3] Interprozesskommunikation

# Varianten

## 0.7 Linux Manipulation

## 0.8 MDM Only

## 0.9 MDM + Container

## 0.10 Samsung Knox

# Auswahl & Konzept