

Security for Systems Engineering – VO 06: Mobile Security

Clemens Hlauschek
Rafael Vrecar
Vanessa Hohenegger
Christian Brem

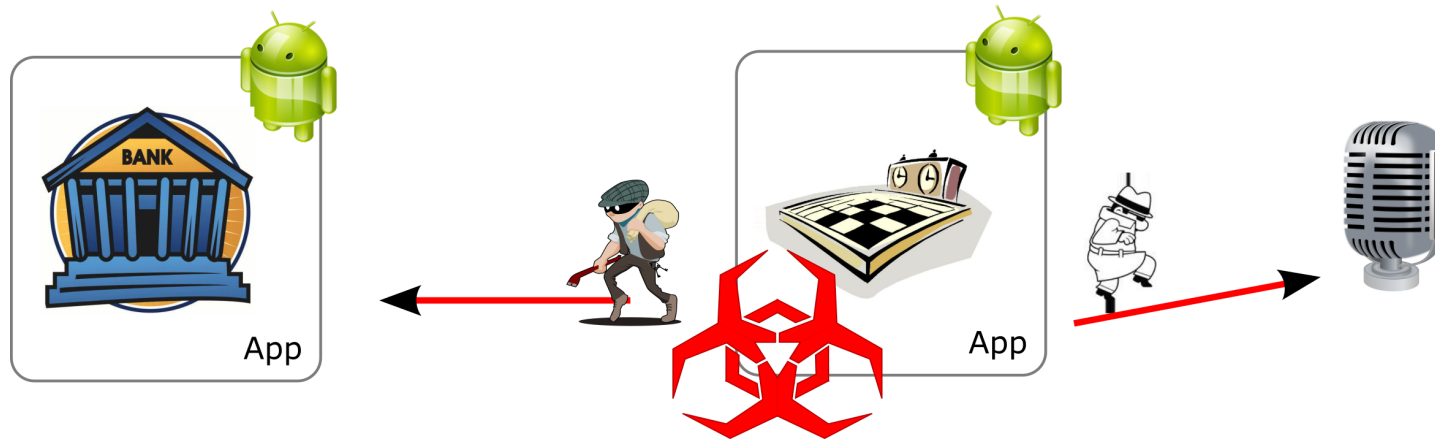
Android Threat- und Sicherheitsmodell, Vergleiche zu iOS

Angriffe Berechtigungen, Intents

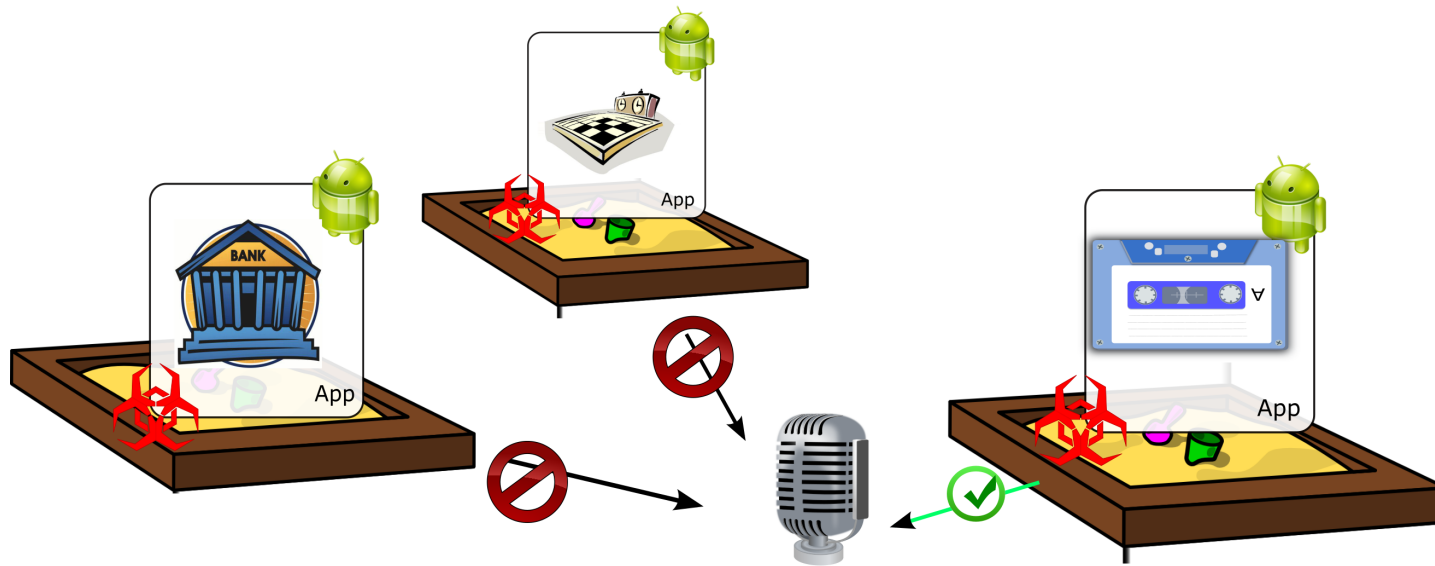
iOS: Data Protection und Keychain

Android Threat- und Sicherheitsmodell, Vergleiche zu iOS

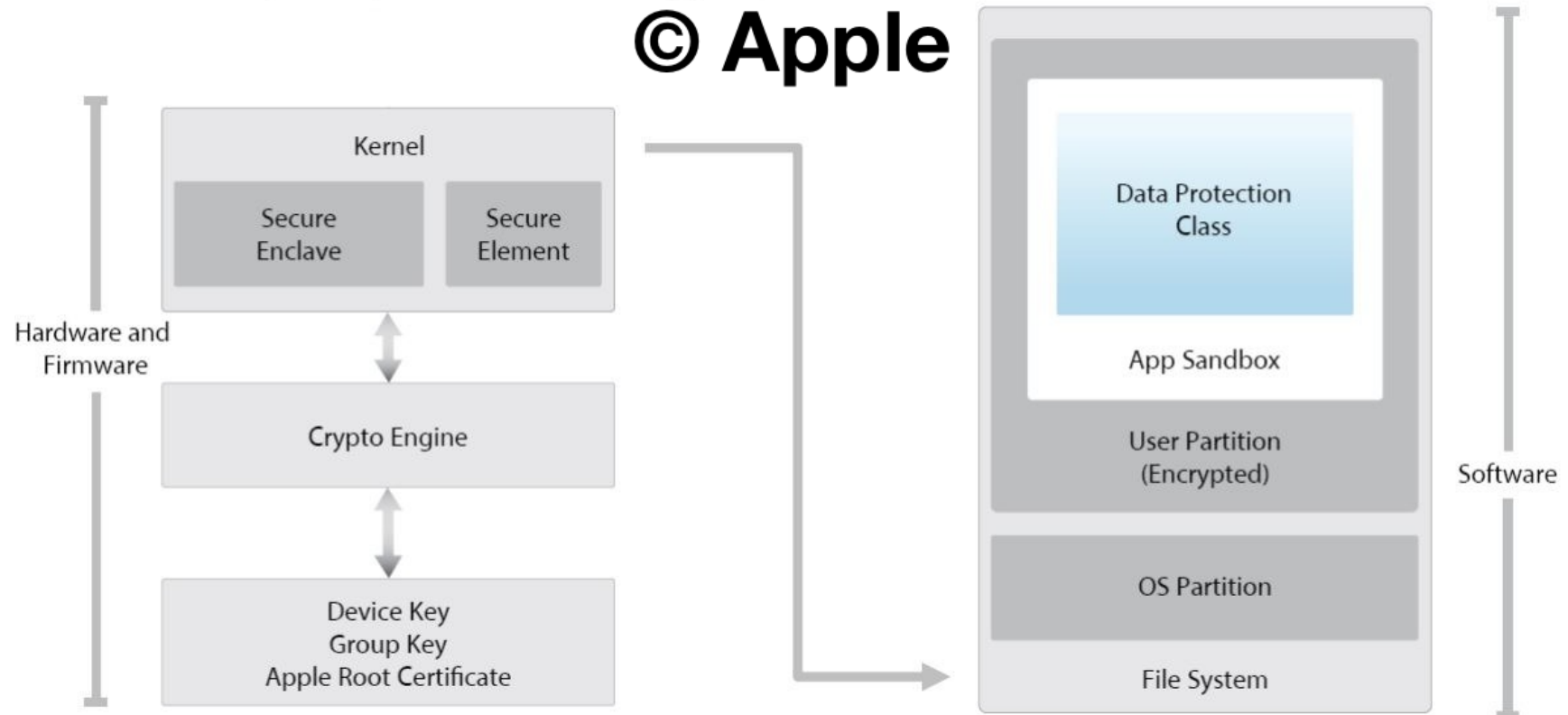
Android-App Threat Model: Motivation



- User hat unterschiedliches Vertrauen in verschiedene Apps
- Game-App soll nicht auf Bankdaten zugreifen können
- Nicht jede App soll auf alle OS-Funktionen (z.B. Mikrofon) zugreifen können)

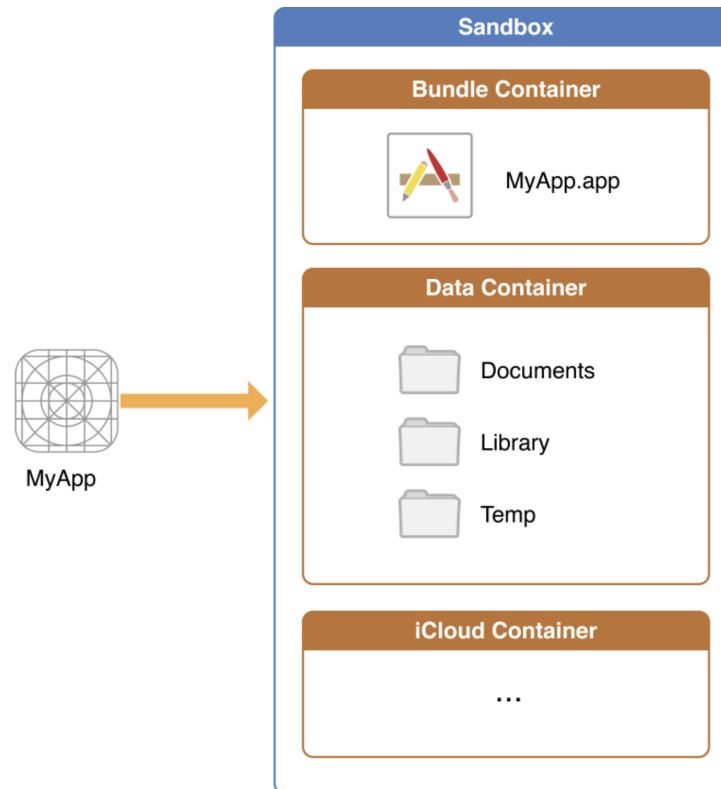


- Jede App potentiell maliziös
- Inter-App Isolation über Sandbox
- Protected System API via Berechtigungen



- Apps laufen in eigener VM isoliert durch OS
- Jede App bekommt eigenen Linux User
a<number>_a<packagenummer>
- Verzeichnis in /data/user/<username>/<packagename>
- Durch Linux-Berechtigungen geschützt
- Mandatory Access Control – SELinux (seit Android 5.0)

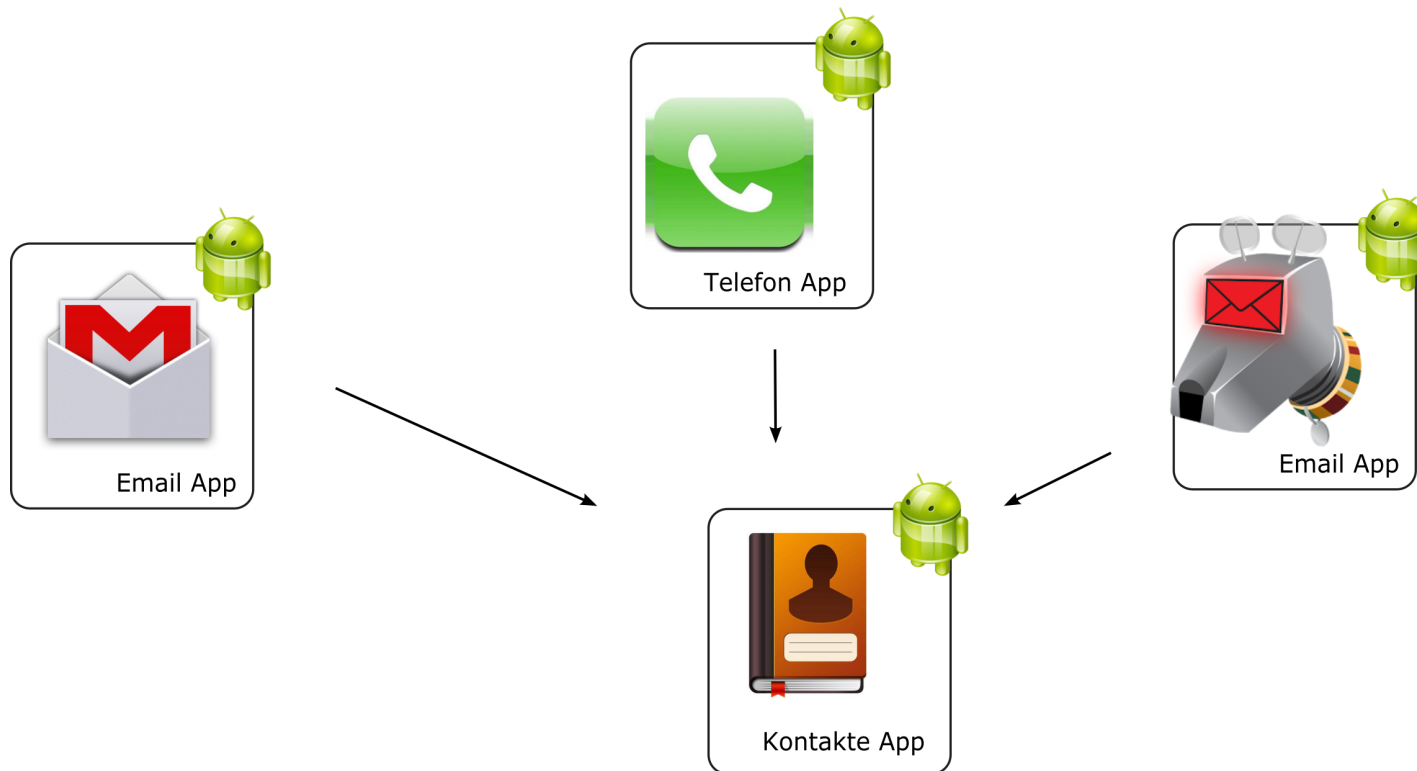
- jede App Dritter läuft in eigener Sandbox
- Apps können keine von anderen Apps gespeicherten Informationen abrufen oder verändern
- ermöglicht Trennung von anderen Apps & OS
- Systemdateien & Ressourcen werden von den User:innen-Apps abgeschirmt
- Apps Dritter sowie meiste Systemdateien & Ressourcen von iOS als nicht privilegierte:r User:in „mobile“ ausgeführt
- gesamte Betriebssystempartition $\Rightarrow$ nur Lesezugriff



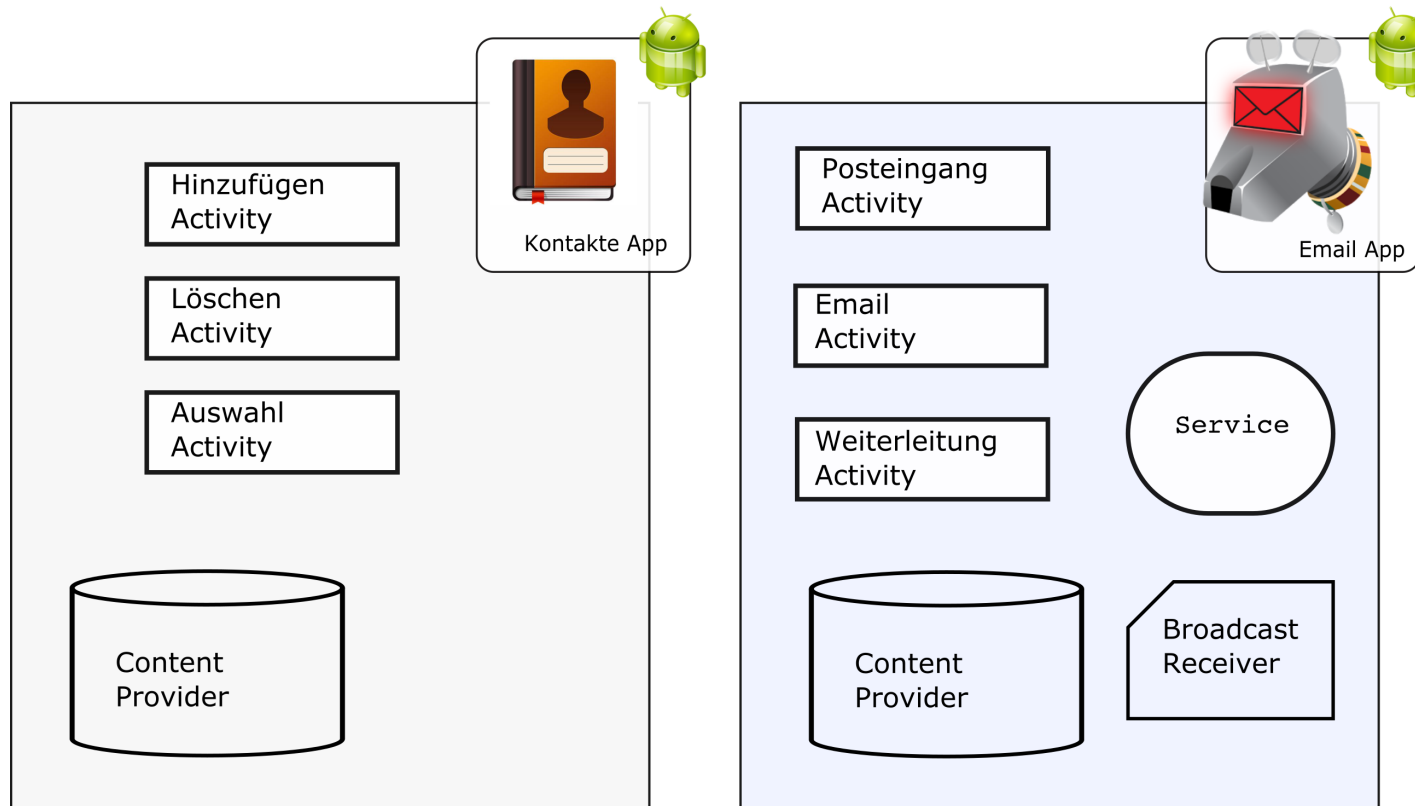
(Vergleiche <https://medium.com/techiepedia/demystifying-ios-ipados-sandbox-with-swift-code-8c9222d19380>, abgerufen: 08.04.2022)

- Berechtigungen steuern Zugriff auf Ressourcen
 - Kamera and Mikrofon
 - Kontakte and Telefonie
 - Location Services
 - ...
- Explizites Einverständnis
- Einhaltung durch OS gegeben
- Feingranularer und widerrufbar ab Android 6

⇒ *unter iOS (de facto) äquivalent, Feingranularität sogar schon länger*

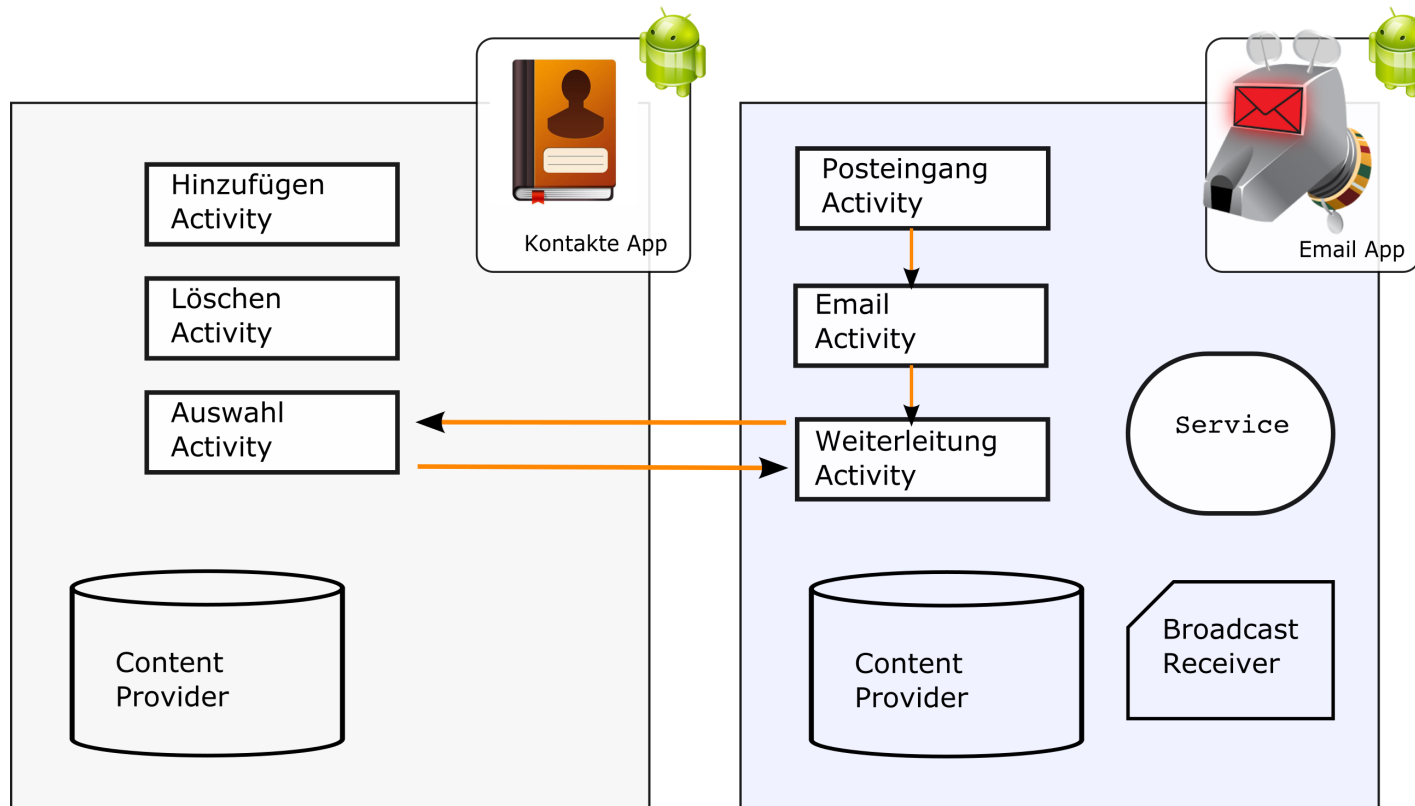


- Unterschiedliche Apps miteinander integriert: z.B. Email-Apps und Telefon-App verwenden Kontakte-App



- App besteht aus Komponenten: Activity, Service, BroadcastReceiver, ContentProvider

Inter Component Communication (ICC): Intents



- Asynchronous message passing System
- Intra- und Interapp Kommunikation

- Explicit Intent vs Implicit Intent
 - Explicit Intent: An konkrete Komponente adressiert
 - Implicit Intent: An Komponente die spezielle Action erfüllt adressiert (z.B.: `action.SEND`)
- Komponente ist public wenn
 - EXPORTED Flag in AndroidManifest gesetzt, oder
 - Mind. 1 Intent Filter deklariert

- Große Angriffsfläche für malizöse Apps
- Intents müssen als untrusted Input behandelt werden

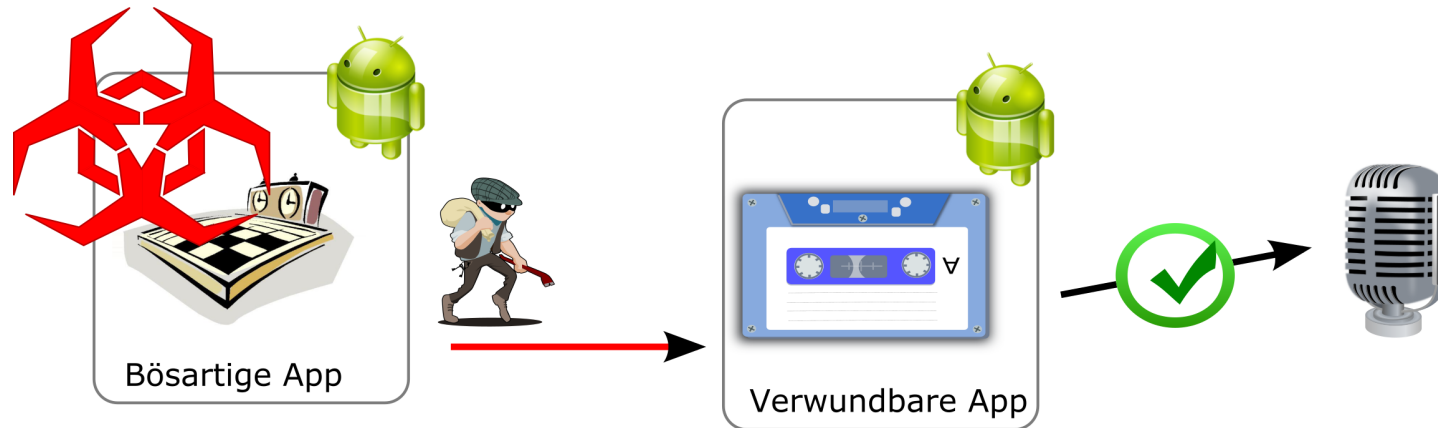
- Potenzielle Probleme
 - Informationen auslesen
 - (Arbiträre) Eingaben
 - Schlechte Filterung/Validierung

iOS – Kommunikation zwischen Apps?

- wegen Sandboxing nicht einfach möglich – ABER: Möglichkeit:
App-Gruppen
- Apps & Erweiterungen von Entwickler:innen-Accounts können Inhalte gemeinsam verwenden, wenn als Teil einer App-Gruppe konfiguriert
- Entwickler:in muss entsprechende Gruppen im Developer Portal erstellen & gewünschte Gruppe mit Apps & Erweiterungen hinzufügen
- Apps, die App-Gruppe zugewiesen wurden, haben Zugriff auf:
 - geteilten Datenträgercontainer als Speicher, der auf Gerät bleibt, solange mindestens 1 App aus Gruppe installiert
 - geteilte Einstellungen und Schlüsselbundobjekte
- Gruppen-IDs (GIDs) für Apps im gesamten Ökosystem eindeutig

Angriffe Berechtigungen, Intents

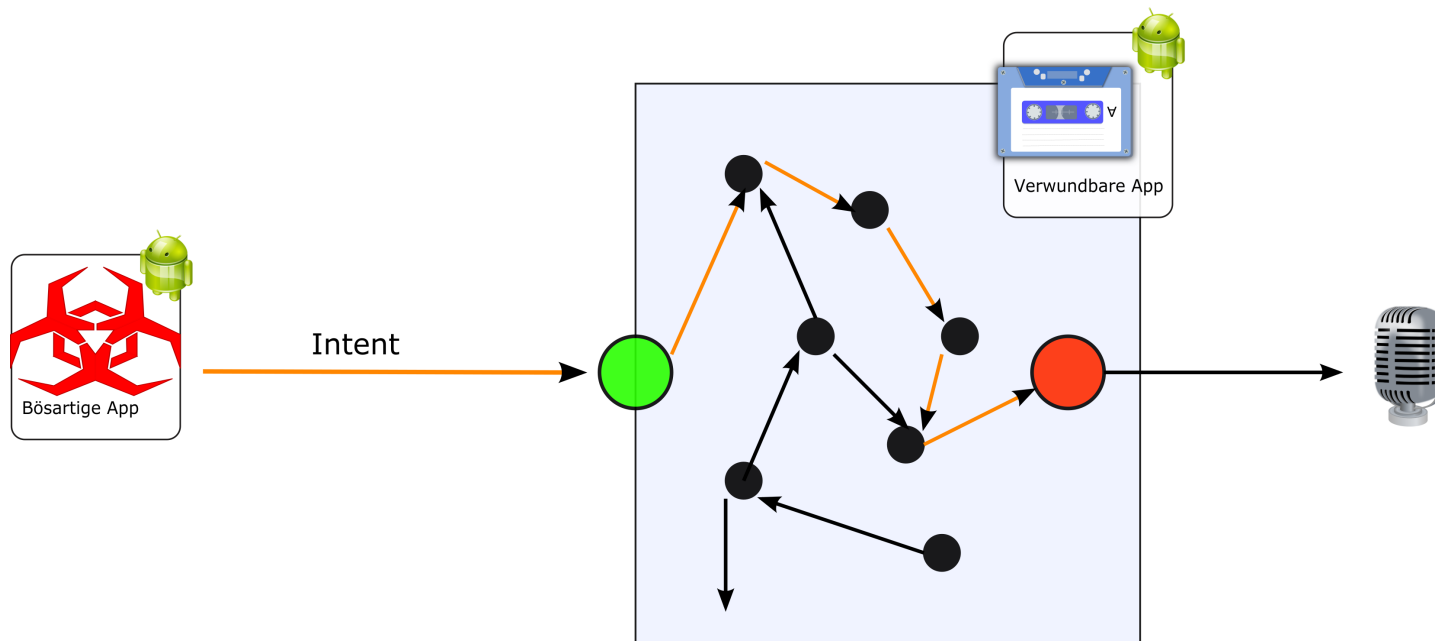
Confused Deputy Attack: Privilege Escalation



- Game-App mit Schadcode, ohne MICROPHONE permission
- Verwundbare Trusted-App (Confused Deputy), mit MICROPHONE Berechtigung
- Game-App sendet Intent an exportierte Komponente in Trusted-App, um Zugang zum Mikrofon zu erhalten.

Voraussetzungen: Confused Deputy Attack, Privilege Escalation

- Verwundbare, aber harmlose App mit mehr Permissions
- Verwundbare App exportiert Komponente
- Code path von public entry point zu Ziel-Resource

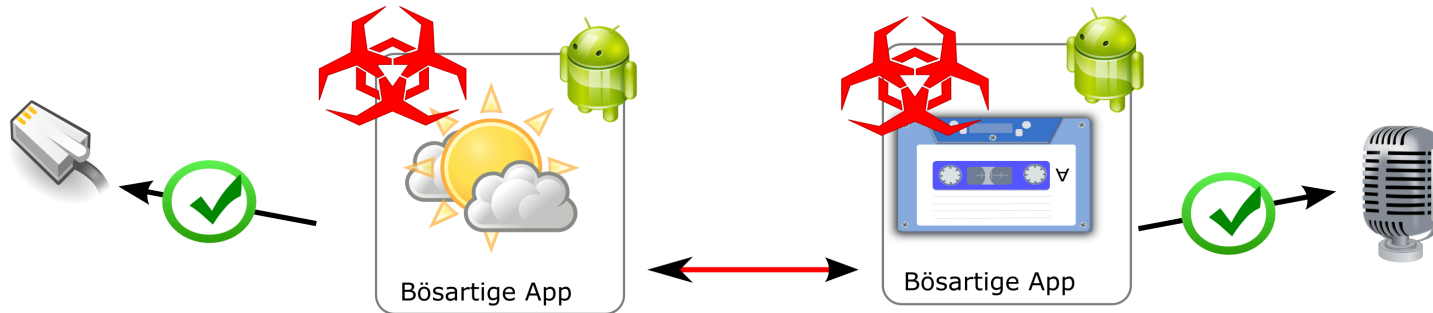


Beispiel: Facebook Information Disclosure (Takeshi Terada)

- Ziel
 - Angreifer liest private Daten aus Facebook App
- Facebook App verwundbar, da
 - Test LoginActivity wird von Facebook App exportiert
 - Intent nach LoginActivity kann weiteren Intent `continuation_intent` enthalten
 - `continuation_intent` wird von LoginActivity im Facebook-App-Kontext weitergesendet

- Android < 4.2: `ContentProvider` `EXPORTED`-Flag gesetzt per default
- Zahlreiche verwundbare Apps mit Problemen wie:
 - Auslesen von SMS Nachrichten (z.B. GoWidget), Contacts, User Credentials, Browser History, Call Log
 - Veränderung von Blacklists → Blockiere Anrufe, SMS (DoS)

- Nur Komponenten exportieren wenn absolut notwendig.
- Custom permission, Permission Level “dangerous” oder “signature”
- Wenn Komponente intern wie extern zugänglich sein muss: Aufteilen
- Saubere Programmierung



- Angreifer installiert mehrere Apps
 - Apps kommunizieren miteinander, oder
 - Sind installiert unter der selben UID (sharedUserId Attribute)
- Z.B. Recorder-App, Zugang zu Mikrofon; und Wetter-App, Zugang zum Netzwerk
- Angreifer erhält so die Vereinigung der Berechtigungen

- Implicit vs Explicit Intents: Verwende explicit Intent wenn möglich
- App-interne Kommunikation: Immer explicit Intent verwenden
- Verwende starke Berechtigungen um Implicit Intents zu schützen
- Intent filter kein Sicherheitsfeature!

- Soundcomber
- Gyroscope
- Kamera
- RAM Usage
- Vibrator, Sound Einstellung
- Ultrasonic sound

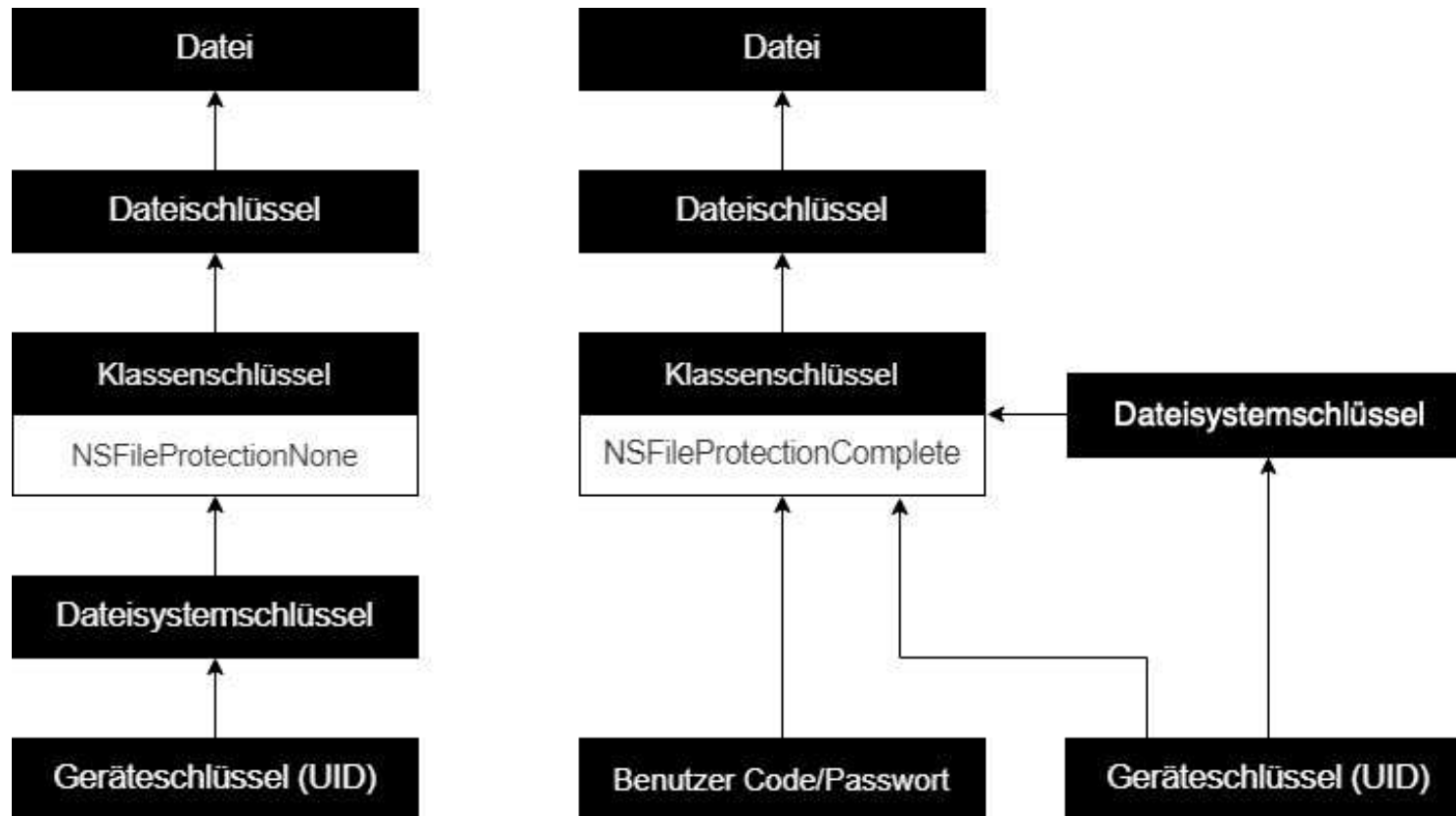
iOS: Data Protection und Keychain

- = Technologie von Apple, um im Flash-Speicher abgelegte Daten zu schützen
- erlaubt hohen Verschlüsselungsstandard für die Benutzer:innendaten
- standardmäßig aktiviert für bestimmte Systemapps (z.B. Nachrichten, Mail, Kalender, Kontakte, Fotos)
- Data Protection durch Erzeugung & Verwaltung von **Schlüsselhierarchie** implementiert
- baut auf Technologien zur **Hardwareverschlüsselung** auf

Klasse	Datensicherheitstyp
Klasse A: Vollständiger Schutz	(NSFileProtectionComplete)
Klasse B: Geschützt, außer wenn offen	(NSFileProtectionCompleteUnlessOpen)
Klasse C: Geschützt bis zur ersten Benutzerauthentifizierung	(NSFileProtectionCompleteUntilFirstUser Authentication)
<i>Hinweis:</i> macOS verwendet einen Volumeschlüssel für die Neuerstellung von FileVault-Schutzcharakteristiken.	
Klasse D: Kein Schutz	(NSFileProtectionNone)
<i>Hinweis:</i> Nicht von macOS unterstützt.	

© Apple

Verschlüsselungsprozess einer Datei



- Passwortmanager mit 2FA-Token-Speicher seit 2021
- verwaltet z.B. Zertifikate, Schlüssel, gespeicherte WLANs
- Passwörter etc. können über Apple ID mit anderen (Apple) Geräten synchronisiert werden



(macOS Icon)

Keychain (Schlüsselbund) – technische Perspektive

- sichere Methode zum Speichern von vertraulichen Datenfragmenten
- als **SQLite-Datenbank** implementiert, im Dateisystem
- Keychain Obj. werden mit 2 AES-256-GCM-Schlüsseln verschlüsselt:
 - Tabellenschlüssel (Metadaten)
 - 1 pro Zeile erzeugter Schlüssel (SecretKey)
- Schlüsselbundeinträge nur zwischen Apps *desselben* Entwicklungsteams gemeinsam nutzbar
- kann verwendet werden, um sensible Daten (z.B. Passwörter, Anmeldedaten, Zertifikate) zu speichern
- auf Datenbank kann nur über die API zugegriffen werden
- nur ein Schlüsselbund für alle Apps unter iOS

Die **nicht prüfungsrelevanten** Inhalte, die wir aus Zeitgründen kurzfristig streichen mussten, finden Sie in den Originalslides unter

<https://security.inso.tuwien.ac.at/downloads/22s/secsyseng/06-original.pdf> (bzw. 06-original-print.pdf).

Danke für Ihr großes Interesse! Die in der Vorlesung erwähnten zusätzlichen Literaturempfehlungen finden Sie im TUWEL-Forum. :)

- Android-Architektur
- Verschiedene Aspekte des Android Sicherheitsmodells
- Angriffe, um das Sicherheitsmodell zu umgehen
 - Confused Deputy Attack
 - Collusion-Angriffe
- Häufige Fehler bei der Entwicklung
 - Unsicherer Netzwerk-Traffic
 - Speichern sensibler Daten
- *sneek peek* auf Ähnlichkeiten und Unterschieden zu iOS

- Advanced Security for Systems Engineering im WS
- Projekte (PR)
- Bakkalaureatsarbeiten
- Diplomarbeiten, Dissertationen
- Mögliche Themengebiete
 - Automatisierung von Security-Tests für Android/iOS
 - Android/iOS Exploitation
 - Android Rootkits / Android Trustzone

- Japan Smartphone Security Association (2020): Android Application Secure Design/Secure Coding Guidebook
- https://www.owasp.org/index.php/OWASP_Mobile_Security_Project
- Green, et al. (2021): Data Security on Mobile Devices
- Khan, et al. (2018): Differences in Inter-App Communication Between Android and iOS Systems

- <https://developer.android.com/guide/>
- Elenkov (2014): Android Security Internals
- Drake, et al. (2014): Android Hacker's Handbook
- Zhou, et al. (2012): Dissecting Android Malware: Characterization and Evolution
- Felt, et al. (2011): A Survey of Mobile Malware in the Wild
- Felt, et al. (2011): Android Permissions Demystified

- Enk, et al. (2009): Understanding Android Security
- Bugiel, et al. (2012): Towards Taming Privilege-Escalation Attacks on Android
- Zhang, et al. (2014): Appsealer
- Chin, et al. (2011): Analyzing Inter-Application Communication in Android
- Zhou, et al. (2013): Detecting Passive Content Leaks and Pollution in Android Applications
- Jana, et al. (2012): Memento: Learning Secrets from process footprints

- Schlegel, et al. (2011): Soundcomber: A Stealthy Context Aware Sound Trojan for Smartphones
- Deshotels (2014): Inaudible Sound as a Covert Channel in Mobile Devices
- Fahl, et al. (2013): Why Eve and Mallory Love Android. An Analysis of Android SSL (In)security
- Bianchi, et al. (2015): What the App is that? Deception and Countermeasures in the Android User Interface
- <https://ibotpeaches.github.io/Apktool/>
- <https://github.com/JesusFreke/smali>
- <https://github.com/skylot/jadx>

- Apple Dokumentation & Support: <https://support.apple.com/>, abgerufen: 05.03.2022
- Apple Platform Security (May 2021): https://manuals.info.apple.com/MANUALS/1000/MA1902/en_US/apple-platform-security-guide.pdf, abgerufen: 05.03.2022
- App-Gruppen: <https://support.apple.com/de-de/guide/security/sec1a976c067/web>, abgerufen: 17.03.2022
- Data Protection: <https://support.apple.com/de-at/guide/security/secf6276da8a/web>, abgerufen: 17.03.2022
- Datensicherheit bei Apple-Geräten: <https://support.apple.com/de-at/guide/security/sece8608431d/1/web/1>, abgerufen: 23.03.2022

- Datensicherheitsklassen: <https://support.apple.com/de-at/guide/security/secb010e978a/web>, abgerufen: 23.03.2022
- David Thiel. IOS Application Security: The Definitive Guide for Hackers and Developers. San Francisco: No Starch Press, 2016.
- Item Attribute Keys and Values: https://developer.apple.com/documentation/security/keychain_services/keychain_items/item_attribute_keys_and_values, abgerufen: 30.02.2022
- Keychain: <https://support.apple.com/de-at/guide/security/secb0694df1a/web>, abgerufen: 17.03.2022

- Keychain Data Protection: <https://support.apple.com/guide/security/keychain-data-protection-secb0694df1a/1/web/1>, abgerufen: 30.02.2022
- Keychain Services: https://developer.apple.com/documentation/security/keychain_services, abgerufen: 30.02.2022
- Sandboxing 1: <https://support.apple.com/de-de/guide/security/sec15bfe098e/web>, abgerufen: 17.03.2022
- Sandboxing 2: <https://medium.com/techiepedia/demystifying-ios-ipados-sandbox-with-swift-code-8c9222d19380>, abgerufen: 08.04.2022
- Übersicht: <https://support.apple.com/de-de/guide/security/secf49cad4db/web>, abgerufen: 17.03.2022

Vielen Dank!

<https://establishing-security.at/>