A-SIT Technology
E-Government Innovationszentrum

16. August 2024 Überblick · Mobile Sicherheit · Privatsphäre & Krypto · 17 Min. Lesezeit

# Untersuchung des Berechtigungssystems unter Android

Gerald Palfinger



| | |
|---|---|
| **Autorinnen und Autoren** | Gerald Palfinger |
| **Stand** | 2024-08-14 |
| **Quelle** | https://technology2.a-sit.at/veroeffentlichungen/untersuchung-des-berechtigungssystems-unter-android/ |

## Kernaussagen

- Android 6.0 führte Laufzeitberechtigungen statt pauschaler Vergabe bei der Installation ein; Android 10 und 11 ergänzten nicht-binäre und temporäre.
- Gerätekennungen wurden eingeschränkt: ANDROID_ID gilt seit Android 8.0 pro Signierschlüssel, die IMEI ist seit Android 10 gesperrt.
- Android-Berechtigungen gliedern sich in fünf Klassen: normale, Laufzeit-, besondere, Signature- und Systemberechtigungen.
- Die Sandbox beruht auf eigener UID/GID je App, Dateisystem-Berechtigungen, Namespaces und SELinux; Scoped Storage kam mit Android 10.

- Studien zeigen, dass Apps mehr Berechtigungen anfordern als nötig und dass die Zahl der Berechtigungen über die Versionen steigt.

Das Berechtigungssystem unter Android ist ein zentrales Sicherheitsmerkmal, das dazu dient, den Zugriff von Applikationen auf bestimmte Ressourcen und Funktionen des Geräts zu regeln. Android bietet eine breite Palette von Berechtigungen, die von Applikationen angefordert werden können, darunter der Zugriff auf Kamera, Standort, Kontakte, Speicher und mehr. Benutzerinnen und Benutzer haben die Kontrolle über diese Berechtigungen und können entscheiden, ob sie einer Applikation die Erlaubnis erteilen, auf bestimmte Ressourcen zuzugreifen oder nicht. Das Berechtigungssystem hat sich über die Jahre verändert um auf neue Bedrohungen einzugehen und den Anforderungen an mehr Privatsphäre gerecht zu werden. Im Rahmen des Berichts wird gezeigt, wie sich das Berechtigungssystem über verschieden Android-Versionen hinweg verändert hat und wie das Berechtigungssystem technisch aufgebaut ist. Ebenso wird erläutert, welche Studien es in der Literatur gibt, die Sicherheitsaspekte des Berechtigungssystem evaluieren.

# 1. Einleitung

Das Berechtigungssystem von Android spielt eine zentrale Rolle bei der Sicherheit und dem Schutz der Privatsphäre von Nutzerinnen und Nutzern mobiler Anwendungen. Seit der Einführung von Android im Jahr 2008 hat sich das Berechtigungsmodell kontinuierlich weiterentwickelt, um den wachsenden Anforderungen an Sicherheit und Datenschutz gerecht zu werden. Frühe Android-Versionen basierten auf einem starren Berechtigungsmodell, bei dem Nutzerinnen und Nutzer bei der Installation einer Applikation alle angeforderten Berechtigungen pauschal erteilen mussten. Mit zunehmender Komplexität mobiler Anwendungen und wachsendem Sicherheitsbewusstsein der Nutzerinnen und Nutzer wurde dieses System jedoch zunehmend als unzureichend angesehen. Im Laufe der Jahre hat Google das Berechtigungsmanagement in Android schrittweise verfeinert und verbessert, um eine feinere Kontrolle durch die Nutzerinnen zu ermöglichen. Insbesondere ab Android 6.0 (Marshmallow) wurde das Berechtigungssystem grundlegend überarbeitet, um Berechtigungen kontextbezogen vergeben zu können.

Dieser Bericht analysiert die Entwicklung des Android-Berechtigungssystems von seinen Anfängen bis zu den aktuellen Versionen. Dabei werden die wesentlichen Veränderungen und Neuerungen in Bezug auf Sicherheitsmechanismen, Benutzerfreundlichkeit und Datenschutz herausgearbeitet. Ziel ist es, ein umfassendes Verständnis für die Evolution dieses zentralen Aspekts des Android-Ökosystems zu vermitteln und die Auswirkungen zu beleuchten.

# 2. Hintergrund

## 2.1. Historie des Berechtigungssystems unter Android

Seit der ersten Veröffentlichung von Android im Jahr 2008 wurden zahlreiche Datenschutzmaßnahmen eingeführt und in den verschiedenen Versionen verbessert. Eine der wichtigsten Änderungen am Berechtigungssystem von Android wurde in Version 6.0 vorgenommen. In früheren Versionen mussten die Berechtigungen für Anwendungen zum Zeitpunkt der Installation erteilt werden. Diesem Modell fehlte es an Granularität [1] und es ermöglichte den Benutzerinnen und Benutzern nicht, den Zugriff auf einzelne

Berechtigungen zu beschränken. Daher mussten die Benutzerinnen entweder alle von einer Anwendung geforderten Berechtigungen gewähren oder die Installation abbrechen, wodurch sie die Anwendung nicht nutzen konnten. Dieses Problem wurde darüber hinaus dadurch verschärft, dass Anwendungen mehr Berechtigungen als nötig anforderten, was schlussendlich zu einer übermäßigen Vergabe von Berechtigungen an Anwendungen führte [2]. Abbildung 1 zeigt einen Informationsbildschirm, der dem Nutzer beziehungsweise der Nutzerin angezeigt wird, wenn eine Applikation installiert wird, die für eine Android-Version vor 6.0 entwickelt wurde.

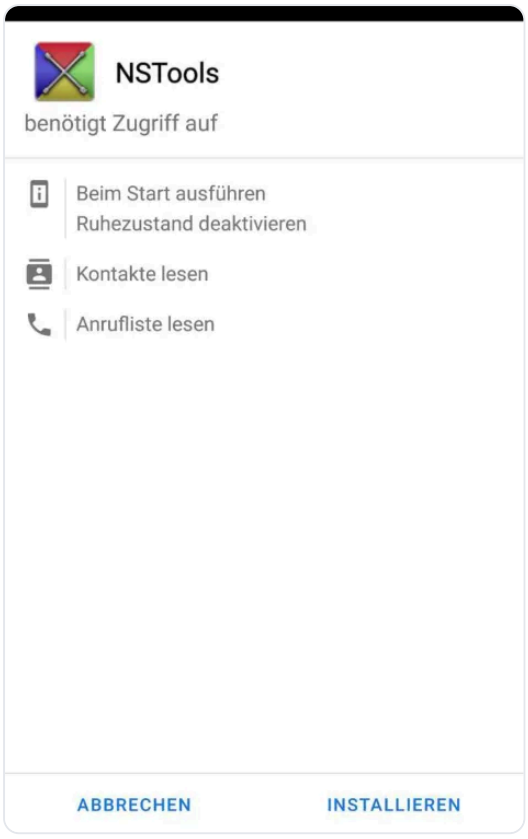



Abbildung 1: Informationsbildschirm der bei der Installation einer Anwendung angezeigt wird, welche noch nicht für Laufzeitberechtigungen optimiert wurde. Es gibt nur die Möglichkeit, der Gewährung aller angeforderten Berechtigungen zuzustimmen oder die Installation abzubrechen.

Um dieses Problem zu lösen und den Nutzerinnen und Nutzern mehr Kontrolle zu geben, wurde mit Android 6.0 ein Laufzeit-Berechtigungsmodell eingeführt. Das neue Berechtigungsmodell erfordert, dass Anwendungen während der Laufzeit bestimmte, insbesondere als gefährlich eingestufte, Berechtigungen anfordert. Dabei gibt es zwei wichtige Berechtigungstypen, die von jeder Drittanbieteranwendung angefordert werden können. Die so genannten normalen (oder „audit-only“) Berechtigungen werden nach wie vor zur Installationszeit erteilt. Diese stellen laut der Dokumentation für Entwickler „ein sehr geringes Risiko für die Privatsphäre des Benutzers und den Betrieb anderer Anwendungen dar“ [3]. Im Gegensatz dazu erlauben Laufzeit- oder als “dangerous” eingestufte Berechtigungen den Zugriff auf potenziell datenschutzrelevante Informationen, wenn sie gewährt werden. Neuere Versionen von Android haben dieses Modell verfeinert. Beispielsweise wurde mit Version 10 nicht-binären Berechtigungen eingeführt, also Berechtigungen, die mehr als nur die beiden Entscheidungen „Erlauben“ oder „Verweigern“ ermöglichen. So kann beispielsweise die Berechtigung zum Abrufen des Gerätestandorts einer Applikation nur dann erlaubt werden, wenn diese vom Nutzer bzw. von der Nutzerin aktiv im Vordergrund verwendet wird. Mit Android 11

wurde dieses System weiter verfeinert. So wurden temporäre Berechtigungen hinzufügt, die nur für eine einmalige Verwendung gewährt werden. Die dazugehörige Benutzeroberfläche wird in Abbildung 2 gezeigt. Ebenso werden unter Android 11 die gewährten Berechtigungen für nicht verwendete Anwendungen zurückgesetzt. Weiters wurden sogenannte besondere Berechtigungen eingeführt, welche eine noch höhere Schutzklasse als Laufzeitberechtigungen aufweisen und nur über die Einstellungsapplikation gewährt werden können. Mehr zu den verschiedenen Berechtigungsarten sind in Abschnitt 3.2 zu finden.




Abbildung 2: Benutzeroberfläche zum Erteilen des Zugriffs auf den Gerätestandort. Der Zugriff kann verhindert werden, einmalig erteilt werden oder dauerhaft bei Benutzung der Applikation erlaubt werden.

Der dauerhafte Zugriff auf den Gerätestandort im Hintergrund wird besonders geschützt und kann daher nur in der Einstellungsapplikation erlaubt werden.

Ebenso wurde über die Versionen hinweg der Zugriff auf Identifizierungsmerkmale erschwert beziehungsweise verhindert. So wurde mit Android 6.0 der Zugriff auf einige der eindeutigen Gerätekennungen, wie die IMEI und die MAC-Adressen, die mit Bluetooth- und Wi-Fi-Hardware verbunden sind, hinter eine gefährliche Berechtigung verschoben. Daher benötigten Anwendungen, die auf Android 6.0 oder höher abzielten, die Zustimmung des Benutzers beziehungsweise der Benutzerin, um darauf zuzugreifen. Mit Android 8.0 wurde das Verhalten der ANDROID_ID geändert [4]. In früheren Versionen wurde der ANDROID_ID erstellt, wenn ein Benutzerkonto auf einem Gerät eingerichtet wurde. So konnte sie von Applikationen als eindeutige Kennung verwendet werden, um eine Benutzerin beziehungsweise einen Benutzer über verschiedene Anwendungen von verschiedenen Entwicklern hinweg zu verfolgen. Ab Android 8.0 wurde der ANDROID_ID so geändert, dass pro Signierschlüssel ein eigener ANDROID_ID generiert wird, wodurch es unmöglich wurde, Benutzerinnen und Benutzer über Anwendungen verschiedener Entwickler hinweg zu verfolgen. Mit Android 10 [5] wurden zusätzliche Einschränkungen für

eindeutige Identifikatoren eingeführt. Insbesondere wurde der Zugriff auf die IMEI und andere nicht zurücksetzbare Gerätekennungen auf privilegierte Berechtigungen, wie die Berechtigung READ_PRIVILEGED_PHONE_STATE, umgestellt. Diese privilegierten Berechtigungen können von Drittanbieteranwendungen nicht angefordert werden. Daher haben Anwendungen von Drittanbietern, die auf Android 10 oder neuer ausgerichtet sind, keinen Zugriff auf diese Identifikatoren.

Als Alternative zu nicht zurücksetzbaren Geräteidentifikatoren hat Google den Google Advertising Identifier eingeführt. Beim Google Advertising Identifier handelt es sich um eine eindeutige, vom Nutzer bzw. Nutzerin zurücksetzbare sowie löschbare Kennung für Werbeanzeigen, der zu den Google-Play-Diensten [6] gehört. Im Gegensatz zu unveränderlichen Gerätekennungen kann sie in den Einstellungen des Geräts zurückgesetzt werden. Die Nutzer bzw. Nutzerinnen eines Geräts können in den Einstellungen außerdem angeben, dass sie nicht mit dem Google Advertising Identifier getrackt werden wollen. In früheren Versionen der Google Play Services bedeutete dies für eine Anwendung lediglich, dass die Werbekennung nicht verwendet werden sollte. Seit 2021 kann der Google Advertising Identifier jedoch vollständig deaktiviert werden, was dazu führt, dass Anwendungen bei der Abfrage der Werbekennung auf einem Gerät, auf dem sie deaktiviert wurde, nur eine Kette von Nullen erhalten. Außerdem muss eine Anwendung, die auf Android 13 und höher abzielt, zusätzlich die Berechtigung com.google.android.gms.permission.AD_ID anfordern. Da es sich dabei jedoch um eine normale Berechtigung handelt, kann eine Benutzerin bzw. ein Benutzer den Zugriff auf diese Berechtigung nicht auf der Basis einzelner Anwendungen verweigern. Zum Vergleich wird im folgenden Abschnitt auf das Berechtigungssystem von iOS näher eingegangen.

## 2.2. Historie des Berechtigungssystems unter iOS

Das Betriebssystem iOS verfügt bereits seit frühen Versionen über ein Berechtigungssystem, welches durch den Benutzer bzw. die Benutzerin applikationsspezifische Anpassungen ermöglicht. Wenn eine Anwendung Zugriff auf datenschutzrelevante Informationen anfordert, fordert das Betriebssystem zunächst die Zustimmung der Benutzerin beziehungsweise des Benutzers an, bevor es den Zugriff auf die Berechtigung gewährt. Wie bei Android wurde das System im Laufe der Zeit verfeinert. So erlaubt die Version 8.0 des Betriebssystems einen ausgefeilteren Grad an Kontrolle über den Zugriff auf Standortdaten. Im Gegensatz zu früheren Versionen können Benutzerinnen und Benutzer den Zugriff auf Standortdaten so einschränken, dass die Anwendung nur Zugriff auf diesen hat, wenn die Anwendung in Gebrauch ist, wodurch ein durch den Benutzer beziehungsweise die Benutzerin unbemerkter Zugriff im Hintergrund verhindert wird.

Wie unter Android wurde auch der Zugriff auf eindeutige Identifizierungsmerkmale entfernt. So wurde mit iOS 7.0 der Zugriff auf eindeutige Bezeichner wie MAC-Adressen der Netzwerkschnittstellen [7] oder der eindeutige Identifikator des Geräts, der über die Eigenschaft -[UIDevice uniqueIdentifier] [8] verfügbar war, für Drittanbieteranwendungen entfernt. Ähnlich wie bei Android gibt es auch bei iOS eine Alternative zu nicht zurücksetzbaren Gerätekennungen, die Identifier for Advertisers genannt wird. Der Zugriff auf diese Kennung kann für Anwendungen von Drittanbietern in der Einstellungsanwendung deaktiviert werden. Darüber hinaus bietet iOS eine zweite Kennung, die für einen Applikationsentwickler eindeutig ist, ähnlich wie der ANDROID_ID . Diese Eigenschaft wird unter iOS als Identifier for Vendors bezeichnet.

Mit iOS 14 wurden zusätzliche Beschränkungen dafür eingeführt, wie auf eindeutige Identifikatoren wie den Identifier for Advertisers zugegriffen werden kann und wie sie verwendet werden können, um ein Gerät zu identifizieren. Im Kern wurde dafür das Apple Tracking Transparency-Framework eingeführt. Insbesondere seit iOS 14.5 müssen Anwendungen die Zustimmung der Nutzerin beziehungsweise des Nutzers mit Hilfe des Apple Tracking Transparency-Frameworks einholen, bevor sie Nutzerinnen und Nutzer anwendungsübergreifend verfolgen können. Verweigert der Benutzer bzw. die Benutzerin die Aufforderung Tracking zu erlauben, wird der Anwendung der Zugriff auf den Identifier for Advertisers untersagt.

# 3. Technische Umsetzung des Berechtigungssystems unter Android

Um die Berechtigungen wirkungsvoll umzusetzen, muss die Applikation von den zu schützenden Informationen und Gerätefunktionen technisch abgeschottet werden. Dazu dient unter Android die sogenannte Applikationssandbox. In dieser werden Android-Applikationen ausgeführt und sind so von anderen Applikationen und dem System abgeschottet. Diese ist dabei ein wesentlicher Teil des Sicherheitskonzepts von Android und baut auf verschiedenen Technologien auf, die Teil des Betriebssystemkernels und des Betriebssystems selbst sind. Im folgenden Abschnitt wird näher auf die technische Umsetzung eingegangen.

## 3.1. Aufbau der Android Sandbox

Die Android Sandbox ist ein wesentlicher Bestandteil des Sicherheitsmodells von Android und basiert auf Funktionen des Linux-Kernels. Sie sorgt dafür, dass jede Anwendung in einer isolierten Umgebung ausgeführt wird, um den Zugriff auf Systemressourcen und Daten anderer Applikationen zu verhindern. Grundlage der Sandbox ist der Linux-Kernel, der verschiedene Mechanismen zur Trennung und Kontrolle von Prozessen und Ressourcen bietet. Eine der wichtigsten Funktionen ist die Verwendung von Benutzerund Gruppenrechten (User/Group IDs). Jede installierte Applikation erhält eine eigene Benutzer-ID (UID) und Gruppen-ID (GID), sowie einen applikationsspezifischen Ordner, der der Applikation zugeordnet ist. Durch diese Trennung hat eine Applikation keine direkten Zugriffsrechte auf die Daten und Ressourcen anderer Applikationen. Dies ist der Kern der Android-Sandbox-Architektur. Hierbei unterscheidet sich die Herangehensweise stark von anderen Linux- beziehungsweise Unixsystemen, bei denen Applikationen unter der Benutzer-ID des Nutzers beziehungsweise der Nutzerin laufen. Die Vergabe einer eigenen Benutzer-ID für jede Applikation ermöglicht es hierbei, Applikationen besser voneinander abzuschotten. Insbesondere müssen dadurch bei der Vergabe von Berechtigungen nicht temporäre Identifikatoren wie Prozess-IDs (PIDs) verwendet werden, die gegebenenfalls wiederverwendet werden und deshalb anfällig für Race Conditions sind [9].

Zusätzlich nutzt Android die klassischen Linux-Dateisystem-Berechtigungen. Jede Applikation hat standardmäßig nur Zugriff auf ihre eigenen Dateien, die in einem privaten Speicherbereich abgelegt sind. Andere Verzeichnisse und Dateien auf dem System bleiben ohne explizite Berechtigungen für die Applikation unzugänglich. Über FUSE (Filesystem in Userspace) beziehungsweise später einem im Kernel integrierten Dateisystem wurde das Konzept der Linux-Dateisystem-Berechtigungen auch auf externe Speichermedien, welche kein Linux-Dateisystem nutzen, erweitert. Mit Android 10 wurde zusätzlich Scoped Storage eingeführt, der den Zugriff auf den externen Speicher weiter einschränkt, indem Applikationen nur auf ihren eigenen externen Verzeichnispfad und die von ihnen selbst erstellten Mediendateien im gemeinsamen Medienspeicher zugreifen können. Dies verhindert, dass eine Applikation ohne Weiteres auf sensible Daten zugreift, die zu anderen Anwendungen oder dem System gehören.

Ein weiteres wichtiges Element der Sandbox ist die Verwendung von Kernel-Level-Namespace-Isolation. Durch die Verwendung von Namespaces werden Ressourcen wie Prozess-IDs (PIDs) und Netzwerkschnittstellen isoliert. Dies sorgt dafür, dass Applikationen nur auf ihre eigenen Prozesse zugreifen können und keine Informationen über andere laufende Prozesse erhalten. Ebenso wurden über die Versionen hinweg Schnittstellen für Drittanbieteranwendungen abgeschafft, welche den Zugriff auf Informationen über andere laufende Anwendungen ermöglichten.

Seit Android 4.3 ist zudem Security-Enhanced Linux (SELinux) integriert, um die Sicherheitsmechanismen weiter zu verstärken. Dabei handelt es sich um eine Implementation von Mandatory Access Control (MAC) für den Linux-Kernel. SELinux arbeitet mit feingranularen Richtlinien, die festlegen, welche Ressourcen eine Applikation unter welchen Bedingungen nutzen darf. Diese Richtlinien ermöglichen eine präzise Kontrolle und verhindern oder erschweren beispielsweise Angriffe, bei denen eine Applikation versucht, höhere Berechtigungen zu erlangen. Besonders ins System integrierte beziehungsweise vorinstallierte Anwendungen, welche höhere Betriebssystemberechtigungen benötigen, profitieren von SELinux. Da diese oft als Administrator („root“) ausgeführt werden, waren diese unter dem Discretionary Access Control mit Benutzer-IDs von der Sandbox ausgenommen. Ebenso waren Prozesse, die unter der System-UID liefen, von der Android-Berechtigungsprüfung ausgenommen und durften so viele privilegierte Operationen durchführen. Durch die feingranulareren Optionen von SELinux können diese besser abgeschottet werden. Doch auch „normale“ Applikationen profitieren von SELinux. So wird beispielsweise seit Android 9.0 jede Applikation in einer eigenen SELinux Sandbox ausgeführt.

Das Android-Berechtigungssystem ergänzt die Isolation der Sandbox auf dem Level der Android API, indem es den Nutzer beziehungsweise die Nutzerin befähigt, spezifische Berechtigungen für sensible Systemressourcen, die zu großen Teilen über die Android API zugänglich sind, zu erteilen. Im folgenden Abschnitt wird auf die verschiedenen Arten an Berechtigungen, die es unter Android gibt, näher eingegangen.

## 3.2. Berechtigungsarten

Die Berechtigungen unter Android werden grob in fünf verschiedene Klassen eingeteilt. Drei davon können von Drittanbieteranwendungen direkt vom System angefordert werden, eine ist privilegierten Applikationen vorbehalten, während die letzte Klasse Berechtigungen beschreibt, welche ein Applikationsentwickler seinen eigenen Applikationen zur Verfügung stellen kann. Alle Berechtigungen haben gemein, dass Applikationen diese Berechtigungen in der Manifest-Datei anfordern müssen. Im Folgenden werden die fünf Klassen genauer beschrieben.

- Normale Berechtigungen (Audit-Only Berechtigungen): Diese Berechtigungen betreffen Funktionen, die keinen direkten Zugriff auf sensible Benutzerdaten oder wichtige Gerätesysteme gewähren. Android gewährt diese Berechtigungen automatisch bei der Installation, ohne dass die Benutzerin bzw. der Benutzer aktiv zustimmen muss (oder kann). Darunter fallen beispielsweise die Berechtigung um Internetzugriff zu erhalten ( INTERNET ) oder das Vibrationsmodul zu verwenden ( VIBRATION ).
- Laufzeitberechtigungen (Dangerous Permissions): Diese Berechtigungen gewähren den Zugriff auf sensible Informationen des Benutzers beziehungsweise der Benutzerin oder kritische Systemressourcen. Vor dem Zugriff auf diese Berechtigungen muss die Benutzerin oder der

Benutzer explizit zustimmen. Darunter fallen beispielsweise die Berechtigung zum Zugriff auf Kontakte ( READ_CONTACTS ), den Standort ( ACCESS_FINE_LOCATION) oder die Kamera ( CAMERA ).

- Besondere Berechtigungen (Special Permissions): Berechtigungen dieser Gruppe schützen den Zugriff auf Ressourcen, die als risikoreicher eingestuft werden als diejenigen, die durch Laufzeitberechtigungen geschützt sind. Um diese zu gewähren, wird vom Nutzer beziehungsweise der Nutzerin mehr Aufwand abverlangt, als dies bei den Laufzeitberechtigungen der Fall ist. Damit eine Benutzerin oder ein Benutzer einer Anwendung die Verwendung einer speziellen Berechtigung erlauben kann, muss der Benutzer beziehungsweise die Benutzerin in die Einstellungsapplikation wechseln und der Anwendung die Berechtigung manuell erteilen. Darunter fallen zum Beispiel die Möglichkeit einer Applikation Geräteadministratorfähigkeiten zu erteilen, der uneingeschränkte Zugriff auf alle Dateien im Verzeichnis des Benutzers beziehungsweise der Benutzerin ( MANAGE_EXTERNAL_STORAGE ) oder der Zugriff auf alle Benachrichtigungen (also auch solche von anderen Applikationen) ( BIND_NOTIFICATION_LISTENER_SERVICE ). Ein Beispiel der Benutzeroberfläche zur Erteilung einer solchen Berechtigung ist in Abbildung 3 dargestellt.
- Signature-Berechtigungen: Diese Berechtigungen werden nur dann gewährt, wenn die Applikation, die die Berechtigung anfordert, mit demselben Zertifikat signiert ist wie die Applikation, die die Berechtigung definiert. Dies ermöglicht einen sicheren Datenaustausch und die Zusammenarbeit zwischen Applikationen desselben Entwicklers oder Herstellers.
- Systemberechtigungen (Privileged Permissions): Diese Berechtigungen werden vom Betriebssystem oder von Systemanwendungen verwendet und sind für normale Applikationen nicht zugänglich. Sie bieten Zugriff auf Systemressourcen und Funktionen und sind für sicherheitskritische Operationen gedacht. Dazu zählt zum Beispiel die Fähigkeit, Systemeinstellungen zu ändern ( WRITE_SETTINGS ) oder der Erhalt von Zugriffsrechten auf eindeutige Identifikationsmerkmale ( READ_PRIVILEGED_PHONE_STATE ).

Die Art einer Berechtigung wird durch das gesetzte protectionLevel in der Definition der Berechtigung bestimmt [10].




Abbildung 3: Benutzeroberfläche zur Erteilung der besonders geschützten Berechtigung zum Zugriff auf alle Nutzerdaten.

## 4. Literatur

In [11] wird das frühe Berechtigungssystem von Android in der Version 2.2 untersucht, bei dem die Berechtigungen noch zum Installationszeitpunkt erteilt werden mussten. Im Rahmen der Studie wurden über 900 Applikationen auf deren Verwendung von Berechtigungen untersucht. Dabei stellte sich heraus, dass etwa ein Drittel der Applikationen mehr Berechtigungen als nötig anforderten. Die Autoren führten dies vor allem auf fehlende beziehungsweise unpräzise Entwicklerdokumentation zurück. In [12] wird die Veränderung des Berechtigungssystems von Android über mehrere Jahre hinweg verfolgt. Insbesondere sollte dabei die Frage beantwortet werden, ob das Berechtigungssystem so verändert wird, dass es die Sicherheit des Systems stärkt. Im betrachteten Zeitraum hat sich laut der Studie die Anzahl der Berechtigungen erhöht. Die Erhöhung war aber nicht auf eine feinere Aufteilung der Berechtigungen zurückzuführen, sondern der Einführung neuer Funktionen, insbesondere neuen Hardwarefunktionen, geschuldet. Dadurch hat sich besonders die Anzahl der als „gefährlich“ eingestuften Berechtigungen erhöht. Ebenso wurden in der Studie Applikationen auf ihre Nutzung von Berechtigungen untersucht. Ähnlich wie in der vorherigen Studie wurde festgestellt, dass Applikationen mehr Berechtigungen anfordern, als für deren Ausführung notwendig ist. Ebenso wurde gezeigt, dass Applikationen im Zuge von Aktualisierungen tendenziell immer mehr Berechtigungen benötigen. Die Studie bezieht sich auf den Zeitraum bis 2012, also auf einen Zeitraum vor der Einführung des Laufzeitberechtigungssystems mit Android 6.0.

In einer jüngeren Untersuchung von Zhauniarovich und Gadyatskaya [13] wurden die Änderungen durch das Laufzeitberechtigungssystem untersucht. Dabei wurde festgestellt, dass im Zuge der Umstellung viele der als früher „gefährlich“ eingestuften Berechtigungen nun als normal eingestuft wurden und so keine

explizite Zustimmung durch die Nutzerin beziehungsweise den Nutzer benötigen. Ebenso sind seit der Umstellung manche Signature-Berechtigungen nun auch für Drittanbieteranwendungen verfügbar. Um die Anzahl der Aufforderungen zum Erteilen von Berechtigungen an die Nutzerin beziehungsweise den Nutzer gering zu halten, wurden die verbleibenden als „gefährlich“ eingestuften Berechtigungen in Berechtigungsgruppen zusammengefasst. Dadurch haben erfahrene Benutzerinnen und Benutzer jedoch nicht die Möglichkeit, die Berechtigungen auf einem feineren Level zu vergeben. In [14] wird die Sicherheit des Laufzeitberechtigungssystems evaluiert. Dabei wurde festgestellt, dass durch die gezielte Nachahmung von vertrauenswürdigen Applikationen der Nutzer beziehungsweise die Nutzerin zur Vergabe von Berechtigungen getäuscht werden könnte. Ebenso können durch die gezielte Ausnutzung von berechtigten Applikationen über Intents manche Berechtigungen umgangen werden. Ebenso argumentiert die Studie, dass die Aufteilung der Berechtigungen in Gruppen nicht fein genug für eine umfassende Kontrolle ist. In einer Studie aus dem Jahr 2020 stellten Almomani und Khayer [15] fest, dass sich die Anzahl der Berechtigungen weiterhin erhöht hat – in manchen Berechtigungskategorien gibt es so um bis zu siebenmal mehr Berechtigungen als in früheren Versionen. Ebenso wurden Applikationen über mehrere Versionen hinweg untersucht. Auch hier stellte sich heraus, dass eine Vielzahl an Applikationen über die Versionen hinweg mehr Berechtigungen anforderten.

## 5. Fazit

In diesem Bericht wurde das Berechtigungssystem von Android untersucht. Dieses wurde im Laufe der Jahre kontinuierlich weiterentwickelt. Während in den frühen Versionen von Android ein statisches Modell verwendet wurde, ermöglicht das seit Android 6.0 eingeführte System der Laufzeitberechtigungen dem Nutzer bzw. der Nutzerin viele Berechtigungen individuell an einzelne Applikationen zu vergeben bzw. die Vergabe zu verweigern. Darüber hinaus ermöglichen temporäre Berechtigungen die Vergabe von Berechtigungen für einzelne Nutzungsvorgänge, wodurch eine größere Kontrolle durch den Benutzer bzw. die Benutzerin ermöglicht wird. Da das Berechtigungssystem als zentrales Element in der Sicherheitsarchitektur von Android gilt, gibt es auch umfangreiche Studien zum Berechtigungssystem in der Literatur.

## Referenzen

Z. Fang, W. Han und Y. Li, „Permission based Android security: Issues and countermeasures,“ Computers & Security, pp. 205-218, 2014.

A. P. Felt, E. Chin, S. Hanna, D. Song und D. A. Wagner, „Android permissions demystified,“ Proceedings of the 18th ACM Conference on Computer and Communications, pp. 627-638, 2011.

Android Developers, „Permissions on Android - Types of permissions,“ Google Inc, 2024. [Online]. Available: https://developer.android.com/guide/topics/permissions/overview#normal-dangerous. [Zugriff am 14 08 2024].

Android Developers, „Settings.Secure - ANDROID_ID,“ Google Inc, 2024. [Online]. Available: https://developer.android.com/reference/android/provider/Settings.Secure.html#ANDROID_ID/. [Zugriff am 19 07 2024].

Android Developers, „Privacy Changes in Android 10 - Restriction on non-resettable device identifiers,“ Google Inc, 2019. [Online]. Available: https://developer.android.com/about/versions/10/privacy/changes#non-resettable-device-ids/. [Zugriff am 07 02 2024].

Google Inc, „Play Console Help - Advertising ID,“ 2024. [Online]. Available: https://support.google.com/googleplay/android-developer/answer/6048248. [Zugriff am 23 07 2024].

Apple Developer, „iOS 7.0 Release Notes (Documentation Archive),“ Apple Inc, 2013. [Online]. Available: https://developer.apple.com/library/archive/releasenotes/General/RN-iOSSDK-7.0/index.html. [Zugriff am 26 07 2024].

Apple Developer, „Deprecated UIDevice Methods (Archived Version),“ Apple Inc, 2014. [Online]. Available: https://web.archive.org/web/20140703160701/https://developer.apple.com/library/ios/documentation/UIKit/Reference/UIDevice_Class/DeprecationAppendix/AppendixADeprecatedAPI.html. [Zugriff am 19 07 2024].

R. Mayrhofer, J. V. Stoep, C. Brubaker, D. Hackborn, B. Bonné, G. S. Tuncay, R. P. Jover und M. Specter, „The Android Platform Security Model (2023),“ in Google Inc, https://research.google/pubs/theandroid-platform-security-model-2023/, 2023.

Android Developers, „<permission> - android:protectionLevel,“ Google Inc, 03 10 2023. [Online]. Available: https://developer.android.com/guide/topics/manifest/permission-element. [Zugriff am 14 08 2024].

A. P. Felt, E. Chin, S. Hanna, D. Song und D. Wagner, „Android permissions demystified,“ Proceedings of the 18th ACM conference on Computer and communications security (CCS), pp. 627-638, 2011.

X. Wei, L. Gomez, I. Neamtiu und M. Faloutsos, „Permission evolution in the Android ecosystem,“ Proceedings of the 28th Annual Computer Security Applications Conference (ACSAC), pp. 31-40, 2012.

„Small Changes, Big Changes: An Updated View on the Android Permission System,“ Research in Attacks, Intrusions, and Defenses, pp. 346-367, 07 09 2016.

„Unravelling Security Issues of Runtime Permissions in Android,“ Journal of Hardware and Systems Security, pp. 45-63, 25 10 2018.

I. M. Almomani und A. A. Khayer, „A Comprehensive Analysis of the Android Permissions System,“ IEEE Access, pp. 216671-216688, 30 11 2020.