HAW HAMBURG

BACHELOR THESIS
Finn-Lukas Armbruster

Datenschutzorientierte Sicherheitsanalyse von Androidsystemen

FAKULTÄT TECHNIK UND INFORMATIK
Department Informatik

Faculty of Engineering and Computer Science
Department Computer Science

HOCHSCHULE FÜR ANGEWANDTE WISSENSCHAFTEN HAMBURG
Hamburg University of Applied Sciences

Finn-Lukas Armbruster

Datenschutzorientierte Sicherheitsanalyse von Androidsystemen

Bachelorarbeit eingereicht im Rahmen der Bachelorprüfung
im Studiengang *Bachelor of Science Angewandte Informatik*
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr.-Ing. Martin Hübner
Zweitgutachter: Prof. Dr.-Ing. Birgit Wendholt

Eingereicht am: 8. Februar 2024

Finn-Lukas Armbruster

Thema der Arbeit

Datenschutzorientierte Sicherheitsanalyse von Androidsystemen

Stichworte

Sicherheit, Forensik, Analyse, Android, Datenschutz, DSGVO

Kurzzusammenfassung

Android Smartphones stellen heutzutage eine essenzielle Rolle im privaten und geschäftlichen Alltag dar. Sie können deshalb sowohl private als auch geschäftliche Daten enthalten, an dessen Sicherheit die Unternehmen als auch die Smartphonebesitzer interessiert sind. Um den Schutz dieser Daten zu gewährleisten, benötigt es Sicherheitsanalysen, welche die Smartphones auf schadhafte Inhalte überprüfen. Für die Umsetzung einer Sicherheitsanalyse wird ein uneingeschränkter Zugriff auf die Smartphone Daten benötigt. Da sich auf dem Smartphone aber personenbezogene Daten aufhalten, muss dabei auf geltende Datenschutzgesetze wie der DSGVO geachtet werden. Ziel dieser Arbeit ist es, ein System zu entwickeln, welches die rechtswidrige Datenverarbeitung von personenbezogenen Daten verhindert sowie den Großteil der Sicherheitsanalyse automatisiert, um den Durchsatz zu maximieren. Als Lösung dieses Problems wurde ein Konzept für ein System entworfen und als Prototyp implementiert, welches die datenschutzrechtlichen Rahmenbedingungen einhält und dabei trotzdem alle sicherheitsrelevanten Dateien in die Analyse mit einbezieht.

Finn-Lukas Armbruster

Title of Thesis

Data protection oriented security analysis of Android systems

Keywords

Security, Forensic, Analysis, Android, Data protection, GDPR

Abstract

Nowadays, Android smartphones represent an essential role in both private and business everyday lives. They can therefore contain both private and business data, whose security is in the interest of both companies and smartphone owners. To ensure the protection of this data, security analyses are required to check the smartphones for malicious content. To realize the security analysis, unrestricted access to the smartphone's data is required. Because smartphones contain personal data, applicable data protection laws such as the GDPR must be followed. The goal of this thesis is to develop a system which prevents unlawful processing of personal data as well as automate most parts of the security analysis to maximize the throughput. To solve this problem, a concept of a system was designed and implemented as a prototype that complies with the data protection regulations while still including all security relevant files in the analysis.

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einleitung

1.1 Motivation

Smartphones sind ein wichtiger Bestandteil unseres privaten als auch geschäftlichen Alltages. Von trivialen Tätigkeiten zum Prokrastinieren, Grundbedürfnissen wie der Kommunikation und der kreativen Verwirklichung bis hin zu Rechtsgeschäften wie Bezahlvorgängen, Vertragsabschlüssen usw.

Durch die rapide Adaption von Smartphones und dem stetigen Zuwachs an Funktionalität werden sowohl den Geräten als auch der Weiterentwicklung dieser immer mehr Wert und daraus folgend Gelder zugeschrieben. Die daraus resultierende hohe Adaptionsquote und Innovationsgeschwindigkeit sorgt dafür, dass immer mehr Tätigkeiten im Geschäftssektor heutzutage über Smartphones abgewickelt werden.

Dies hat zur Folge, dass durch die Smartphones ein Sicherheitsrisiko für dessen Daten, Benutzer und oder Besitzer entsteht, da durch eine Kompromittierung von Ihnen finanzielle Schäden als auch rechtliche Konsequenzen entstehen können.

Insbesondere in den letzten Jahren hat die Schadenshöhe durch Cyber-Angriffe drastisch zugenommen. So stieg der wirtschaftliche Schaden in 2021 um über 117 % im Vergleich zu 2019 (von 102,9 Mrd. Euro auf 223,5 Mrd. Euro) [3].

Ein beliebtes Mittel zur Risikominimierung, um Manipulationen oder Spionage von Smartphones zu erschweren, sind geschäftlich separierte Smartphones, welche vom Arbeitgeber zur Verfügung gestellt werden. Diese werden mit entsprechender Verwaltungs- und Sicherheitssoftware ausgestattet. Zusätzlich werden vertragliche Vereinbarungen zwischen Arbeitgeber und Arbeitnehmer abgeschlossen, welche die Nutzung für private oder nicht arbeitsbezogene / dienstliche Zwecke ausschließen.

Während eine vollständige und dauerhafte Überwachung nach einer validen Lösung klingt, so ist dies nur mit rechtlichen Hürden umsetzbar, als auch sicherheitstechnisch nicht unfehlbar.

Ein andere Vorgehensweise ist hingegen der Bring Your Own Device (BYOD) Ansatz, in welchem private Smartphones von Arbeitnehmern nicht nur für private Zwecke verwendet werden, sondern auch für Arbeits- / Dientsttätigkeiten [4].

Bei dem BYOD Ansatz ergibt sich das Problem, dass auch bei einem sicheren Umgang mit Unternehmensdaten seitens der Arbeitnehmer, sicheren Firmenanwendungen sowie geschützter Netzwerkkommunikation, es durch private Anwendungen dazu kommen kann, dass die Sicherheit der Unternehmensdaten kompromittiert wird.

Des Weiteren können Benutzer / Arbeitnehmer aufgrund der übergreifenden Kontrolle in ihrer Privatsphäre eingeschränkt werden. Durch die Unternehmensanwendungen wird in der Theorie ein Zugriff auf die privaten Daten ermöglicht. Im Allgemeinen kann ein wechselseitiger Zugriff von privaten und Unternehmensdaten nicht in Gänze ausgeschlossen werden [23].

Neben dem bereits erwähnten BYOD Ansatz und reinen Geschäftsgeräten gibt es noch als weitere Möglichkeit die private Nutzung von Firmengeräten. Dies ist oft bei öffentlichen Institutionen der Fall und wird durch Arbeits- oder Dienstvereinbarungen festgelegt. Diese setzen jedoch voraus, dass durch die private Nutzung dienstliche Belange nicht beeinträchtigt werden. Hier bestehen jedoch im Kern die gleichen Problematiken wie beim BYOD Ansatz.

1.2 Problemstellung

Während diese Ansätze sich für kleine und mittlere Unternehmen (KMU) als wirksame Mittel erweisen können, so stoßen diese bei größeren Konzernen oder Personen mit großem Einfluss (politischen oder wirtschaftlichen Ursprunges) und daraus resultierendem Risiko als auch strikteren Rechtsverpflichtungen auf Probleme.

Bei einer ferngesteuerten Überwachung ergibt sich als direkter Aspekt, dass sensible Daten, darunter auch die privaten Daten, zur Analyse übertragen werden müssen oder können. Insbesondere bei der Industriespionage oder bei einer späteren Kompromittierung, kann es durch vorhandene Logs auch im Nachhinein noch zu Problemen kommen.

Dabei können externe (remote) Überwachungssoftwares von Angreifern für Living-off-the-Land-Angriffe (LotL) verwendet werden. Dadurch wird ein mögliches Eintrittstor auf die Smartphones durch die Überwachungssoftware erschaffen. Auch der Ursprung von fremder Software kann problematisch sein. Diese kann es den Herstellern oder den dahinter stehenden Regierungen ermöglichen, Anwender auszuspionieren oder zu kompromittieren. Davon sind Betriebssysteme wie z. B. Android oder IOS auch nicht ausgenommen [18].

Zudem empfinden Arbeitnehmer oft Unwohlsein, potenziell alle ihren privaten Daten dem Arbeitgeber in Echtzeit verfügbar zu machen und auch eine mögliche Leistungsüberwachung zu schaffen.

Hinzu kommt, dass durch eine rein automatische Überwachung und Sicherheitskonfigurationen wie z. B. einer sog. SSideloading Blockade", welche das Installieren und ausführen von nicht freigegebenen Anwendungen verhindert, eine Kompromittierung nicht ausgeschlossen werden kann. Es bleibt ein Restrisiko, da durch sog. Zero-Day-Angriffe, bei welchem Angreifer noch nicht bekannte Sicherheitslücken ausnutzen, es auch bei gewarteten Smartphones zur Ausführung von Fremd- bzw. Schadcode kommen kann.

Es ergibt sich daher als potenzielle Lösung, die Smartphones sicher zu konfigurieren und diese mit physischem Zugriff unter Einhaltung von Privatsphäre und Datenschutz, periodisch oder auf Verdacht zu kontrollieren. Verdächtig wären z. B. unbekannte Dateien, hoher mobiler Datenverbrauch, langsames Ansprechverhalten oder hohe Akkunutzung / Temperaturentwicklung im Ruhestand.

1.3 Zielsetzung

Ziel dieser Arbeit ist es, ein Konzept und Entwurf sowie eine Teilrealisierung als 'Proof of Concept' (POC) eines Systems hervorzubringen, mit welchem eine effektive und halb automatische Auswertung von kompletten Android Smartphones durchgeführt werden kann.

Kernaspekt des Systems soll dabei darauf liegen, dass der Datenschutz der Smartphone Benutzer / Besitzer eingehalten wird. Das System soll dabei zum Teil automatisch arbeiten, um den Kontakt von der prüfenden Person mit den privaten Daten des Besitzers zu minimieren. Zudem soll es modular sein und unterschiedliche Erkennungsmethoden von

z. B. Schadsoftware erlauben, um dem Prüfer sowohl Arbeit abzunehmen als auch um die Fehlerquote von False-Negatives zu reduzieren.

Bei nicht eindeutig klassifizierten Fällen oder bei Verdacht muss es jedoch die Möglichkeit geben, dass die prüfende Person manuell das Smartphone oder die betroffenen Daten untersuchen und daraufhin klassifizieren kann.

1.4 Gliederung

Das erste Kapitel der Arbeit besteht aus der Motivation und der Zielsetzung. Im zweiten Kapitel werden technischen Grundlagen als auch Eigenschaften des Android Betriebssystems, welche für diese Arbeit wichtig sind, erklärt und beschrieben. Darauf erfolgt im dritten Kapitel ein Einstieg in das Thema Datenschutz, um Konsens zu schaffen, welche Rechte und Gesetze bei einer solchen Analyse zu beachten sind. Anschließend werden im vierten Kapitel die Anforderungen an das zu entwickelnden Systems ausgearbeitet. Im fünften Kapitel folgt die Architektur des Systems mit einem dazugehörigem technischen Entwurf. Anschließend im sechsten Kapitel erfolgt eine Prototypenimplementierung und ein Testablauf als POC. Das letzte Kapitel fasst die Ergebnisse der Arbeit zusammen und bietet einen Ausblick auf mögliche Weiterentwicklungen des Systems.

2 Technische Grundlagen

Das folgende Kapitel bietet einen Einstieg über die technischen Themen, welche für das Verständnis dieser Arbeit benötigt werden. Im Unterkapitel 2.1 werden dabei zuerst die wichtigsten Konzepte und Mechanismen für IAM-Systeme erklärt. Das zweite Unterkapitel 2.2 befasst sich mit den wesentlichen Eigenschaften von Android Smartphones und Ansätzen für die Datenextraktion, welche für die hier behandelte Sicherheitsanalyse relevant sind.

2.1 IAM - Identity and Access Management

Ein IAM-System ist, eine Aggregation an Sicherheits- und Administrationsframeworks. Das Ziel eines solchen Systems ist es, ein Trust Framework aufzubauen. Ziel eines Trust Frameworks ist es, Dienste als auch Benutzer und dessen jeweiligen Rechte strukturiert zu verwalten und dabei Vertrauensstellungen zu erschaffen [21].

Der Kernaspekt liegt dabei auf der Benutzerverwaltung. Dabei erlaubt es das System, Benutzer Accounts autonom zu verwalten und über Schnittstellen wie LDAP andere Verzeichnisdienste zu koppeln (User Provisioning). Zudem bietet es unterschiedliche Authentifizierungsmechanismen für die Identitätskontrolle als auch Autorisationsmechanismen für eine präzise Rechtesteuerung.

Dabei werden alle Zugriffe protokolliert und übersichtlich dargestellt, wodurch sowohl eine Nachverfolgung als auch Sicherstellung, dass alle Benutzer die korrekten Rechte haben, ermöglicht wird. Dies hat zur Folge, dass Risiken in Bezug auf Zugriffsrechtsverletzungen reduziert werden und die Nachvollziehbarkeit für Datenzugriffe im Sinne des Datenschutzes nachvollzogen und verifiziert werden können.

Bezüglich der Nachvollziehbarkeit bieten IAM-Systeme den weiteren Vorteil, dass nicht nur Zugriffsanfragen, sondern auch Rechtemodifikationen protokolliert werden. Dadurch

ergibt sich, dass bei einer Betriebs- bzw. Datenschutzprüfung (Audit) als auch bei einer Nachverfolgung von Zugriffen nach einem Incident, die Einhaltung von Zugriffseinschränkungen überprüft werden kann.

2.1.1 MFA

Eine Multi-Faktor-Authentifizierung (Multi-Factor Authentication - MFA) erlaubt es zusätzlich zu einem normalen Passwort einen weiteren unabhängigen Identitätsnachweis wie Eigentumsnachweise, Wissensnachweise oder Eigenschaftsnachweise zu verwenden / einzufordern. Ein beliebtes Beispiel hierfür sind sog. Authenticator Apps, welche Zeitbasiert Passwörter generieren oder One Time Codes, welche per SMS versendet werden. Die zusätzlichen Identitätsnachweise sollten, sofern möglich, getrennt von dem System auf dem die Passworteingabe erfolgt, ausgelagert werden.

Generell gibt es auch bereits gesetzliche Vorschriften, so müssen Banken seit 2021 durch die PSD2 Vorschrift (Payment Services Directive 2) bei Onlinebanking-Vorgängen eine MFA erzwingen. Alle gängigen IAM-Systeme bieten dabei MFA Unterstützung.

2.1.2 RBAC

Eine rollenbasierte Zugriffskontrolle (Role-Based Access Control - RBAC) erlaubt es unabhängig von einzelnen Nutzern, jeweiligen Gruppen Rechte zuzuordnen. Diese Rollen definieren dabei die jeweiligen Zugriffs- und Ausführungsrechte. IAM-Systeme bieten dabei durch die Kombination von User Provisioning und RBAC eine einheitliche und strukturierte Rechteverwaltung. Dabei werden die Rechteverteilungen den organisatorischen Gegebenheiten, wie Job Positionen oder Aufgabentätigkeiten nachmodelliert.

Mittels RBAC lässt sich zudem auch das sog. "Principle of Least Privilege" (PoLP) umsetzten. Hierfür werden jedem Nutzer spezifische Gruppen mit den minimal benötigten Rechten, um ihre Aufgaben auszuführen, zugewiesen. Dies hat zur Folge, dass bei einer Kompromittierung des Accounts oder böswilliger Absichten seitens des Benutzers die potenziellen Schäden minimiert werden und neben den bereits erwähnten Protokollen eine einfachere Nachvollziehbarkeit über mögliche Datenzugriffe ermöglicht wird.

2.1.3 SSO, Token und Scopes

Um mehrmaliges Anmelden von Benutzern zu vermeiden, verwenden viele Systeme heutzutage sog. Single-Sign-On (SSO) Verfahren. Ziel dieser ist es, dass nach dem Anmelden der Benutzer ein Passwort unabhängiger Nachweis (sog. Token / Bearer) über dessen Identität und Rechten erstell wird, mit welchem er sich bei unterschiedlichen Diensten authentifizieren, als auch autorisieren kann. Dieser Nachweis ist dabei zeitlich gebunden und kann von dem Benutzer erneuert werden, wodurch ein neuer Nachweis generiert wird.

Das Besondere bei dem Token ist, dass dieser kein reiner Identitätsnachweis ist, sondern auch auf einen Verwender (Anwendung / Dienst) gebunden werden kann, als dass auch Rechte für den Verwender ausgewiesen werden können (sog. Scopes). Diese Scopes beschreiben dabei, wozu der jeweilige Token den Verwender berechtigt. So kann ein Benutzer einen Token für den Dienst X mit den Rechten ausstatten, dass nur X ihn akzeptieren kann und dabei seinen vollen Namen aber hingegen nicht seine E-Mail vom IAM-System abrufen kann.

Durch die Tokens und Scopes wird es dem Benutzer daher ermöglicht, sich mit einem Account bei beliebig vielen Diensten mittels SSO anzumelden und nur die vorher festgelegten eignen Informationen offen zu legen. Dabei können durch die Scopes, die Rechte der Nutzer gegenüber den Diensten erzwungen werden, als auch durch RBAC die Rechte von den Benutzern auf den Diensten.

2.2 Android

2.2.1 Zertifikate

Zertifikate bilden einen Grundbaustein in der IT-Sicherheit und dienen primär zum Herstellen von Vertrauensketten, Verschlüsselung, Authentifizierung als auch der Signierung.

Die wichtigsten Zertifikate stellen dabei die Wurzelzertifikate (Root Certificates) von den Zertifizierungsstellen (Certificate Authorities - CA) dar. Nur durch diese kann eine PKI (Public Key Infrastructure) erstellt werden um z. B. sicher SSL / TLS Zertifikate durch Vertrauensketten zu validieren und für eine sichere Kommunikation zu verwenden.

Eine Kompromittierung eines CA Zertifikates ermöglicht es Angreifern sich beliebig viele gültige und glaubwürdige Zertifikate auszustellen, um damit z. B. die Identität eines anderen anzunehmen (Spoofing) oder um Man-in-the-Middle-Angriffe (MitM) durchzuführen.

Bei Android werden diese CA Zertifikate mit dem Betriebssystem ausgeliefert, ohne direkte Möglichkeit diese zu überprüfen [10]. Es gibt dabei keinen Prozess diese Zertifikate zu validieren oder zu prüfen, ob diese z. B. aufgrund einer Kompromittierung zurückgezogen worden sind mittels Zertifikatsperrliste (Certificate revocation list) [9].

2.2.2 Eindeutige Smartphone Erkennung

Um Android Smartphones zu verfolgen und zu identifizieren, wird eine Kennung benötigt. Dabei bieten sich bei Android Smartphones 3 Eigenschaften an, die Vendor (Hersteller) ID, Android ID und die IMEI.

Die IMEI (International Mobile Equipment Identity) wird für jedes GSM (Global System for Mobile Communications) fähige Gerät weltweit eindeutig vergeben. Das bedeutet auch, dass bei Dual-SIM fähigen Smartphones zwei IMEI-Nummern vorliegen.

Die Vendor ID ist eine eindeutige Hardware-Seriennummer, welche vom jeweiligen Hersteller vergeben wird. Sie wird vom Hersteller für eine eindeutige Identifizierung verwendet, um z. B. Garantien und Support zu abzuwickeln.

Die letzte Kennung, die Android ID, wird aus der Hardware (Zufallszahlengenerator) und dem angelegten Hauptbenutzer generiert. Die Art der Generierung hat daher auch zur Folge, dass die Kennung sich bei jeder Neuinstallation des Gerätes ändert und daher Benutzer statt Gerätespezifisch ist. Sie wird primär von den Google Diensten verwendet und kann von Apps zur Identifizierung des Gerätes bzw. der aktuellen Installation verwendet werden.

2.2.3 Dateisystem und Startvorgang

Android Smartphones besitzen mehreren Partitionen mit dementsprechenden Dateisystemen, meistens ext4. Dabei gibt es systemrelevante Partitionen wie *Boot*, *System* und *Recovery* als auch die Benutzerpartition *Data*, welche alle vom Benutzer installierten

Anwendungen wie auch dessen eigene Daten beinhaltet. Hinzu kommen Hilfspartitionen wie *Cache* [11].

Beim Starten des Smartphones wird dabei immer zuerst der Bootloader von der *Boot*-Partition geladen. Der Bootloader wird daraufhin entweder das normale Android Betriebssystem (Android OS) von der *System*-Partition starten oder falls aufgefordert das Wiederherstellungsbetriebssystem (Recovery OS) von der *Recovery* Partition. Das Recovery OS wird geladen, wenn eine herstellerspezifische Tastenkombination gedrückt wird oder eine Flag, welche über USB oder vor einem Neustart gesetzt wird aktiv ist (Siehe Abbildung 2.1).

Das Wiederherstellungsbetriebssystem ist dabei meistens eine abgespeckte Version des Hauptbetriebssystems, um von diesem aus Einstellungen oder Anwendungen zu ändern oder deinstallieren, welche das normale Betriebssystem beim Start oder der Bedienung einschränkt oder verhindert.

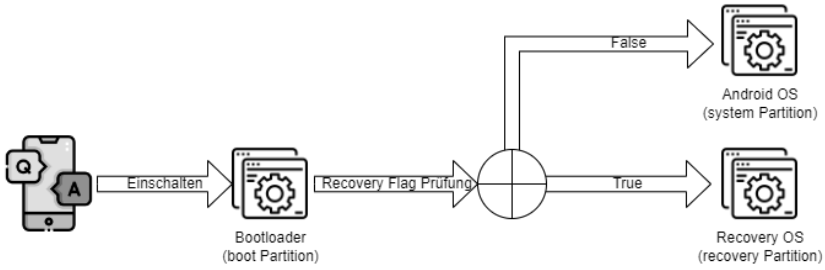

Abbildung 2.1: Android Bootvorgang

2.2.4 Dateisystemakquisition

Für eine forensische Untersuchung (Sicherheitsanalyse) müssen die Daten zuvor von dem Gerät extrahiert werden. Android Smartphones stellen dabei aber lediglich Medien über MTP (Media Transfer Protocol) zur Verfügung.

Ein Ansatz, um an alle Daten zu kommen, ist es das Smartphone zu "rooten". Dabei werden dem Benutzer Root Rechte erteilt, wodurch dieser alle System Dateien einsehen, bearbeiten und extrahieren kann. Dieser Ansatz sollte nicht in der Praxis verwendet werden, da auch die Anwendungen mit Root Rechten ausgeführt werden, daher ohne Einschränkungen schadhafte Systemänderungen vornehmen können.

Ein weiterer Ansatz ist es, das Recovery OS zu verwenden, welches unabhängig vom eigentlichen Android OS zur Verfügung steht (siehe 2.1). Der Vorteil dieses Ansatzes liegt vor allem darin, dass das zu untersuchende System nicht ausgeführt wird, wodurch schadhafter Code nicht zur Ausführung kommt und sich vor der Datenextraktion verstecken könnte. Außerdem erlaubt es ein physisches Abbild der Dateisysteme / der Partitionen zu erstellen, wodurch alle Eigenschaften des Datei- bzw. Betriebssystems wie z. B. die EAs (Extended Attributes) erhalten bleiben.

Das Recovery OS kann dabei in den meisten Fällen standardmäßig nur eingeschränkt auf die Partitionen zugreifen, da es den Benutzer nicht mit Root Rechten ausstattet. Abhilfe schafft hierbei ein eigenes Wiederherstellungsbetriebssystem / Recovery OS (Custom Recovery - CR). Diese können beliebig angepasst werden, um alle auf alle Partitionen und dazugehörigen Dateisysteme uneingeschränkt zugreifen zu können. Beim Startvorgang könnten sie dabei auch mit einem Passwort geschützt werden. Dabei werden dem Besitzer auf dem normalen Betriebssystem keine Root Rechte erteilt, indem nur im Recovery OS temporär Root Rechte erlangt werden [25].

Für die Installation eines solchen CR-Systems muss jedoch vorher der Bootloader entsichert werden. Von seitens der Hersteller werden diese oft Passwortgeschützt gesichert. Viele der Android Smartphone Hersteller bieten es jedoch an, auf Anfrage das Passwort für die Entriegelung des Bootloaders anzufordern. Jedoch bieten dies nicht alle an, so gibt es auch Hersteller wie Samsung, welche ihre Smartphones zusätzlich mit weiteren Sicherheitsmechanismen wie Secure Boot / Trust Boot ausstatten. Diese Prüfen beim Starten eine kryptografische Signatur der jeweiligen Systeme (Bootloader und Android / Recover OS), wodurch ein CR-Ansatz ohne die Herstellergarantie zu verlieren, nicht funktioniert [1].

Es existieren Unternehmen, welche sich darauf spezialisiert haben explizit solche Sicherheitslücken (Zero Days oder Hardware Sicherheitslücken, welche nicht gepatcht werden können) für die forensische Datenakquisition anzubieten und diese nicht beim Hersteller einzureichen / zu melden. Ein beliebtes Beispiel hierfür wäre GrayKey von Magnetic Forensics, welches bei allen populären Smartphones, darunter auch IPhones, die Datenextraktion ermöglicht [15]. Die meisten Hersteller solcher Lösungen, darunter auch Magnetic Forensics, verkaufen diese jedoch nur an Strafverfolgungsbehörden unter hohen Anforderungen, um sowohl eine bösartige Ausnutzung als auch die Weiterverbreitung der Sicherheitslücken zu unterbinden.

3 Datenschutz Grundlagen

Heutzutage verarbeitet jedes Unternehmen Daten für eigene Zwecke oder für andere in Form von Dienstleistungen. Egal ob es dabei um Rechnungen, Finanzdienstleistungen, Zeiterfassung oder anderern Tätigkeiten wie der Sicherheitsanalyse von Androidsystemen geht.

Diese Daten sind oft personenbezogen (identifizierend oder zuordbar zu natürlichen Personen) und unterliegen daher seit dem 25 Mai 2018 der durch die Europäische Union in Kraft getretenen DSGVO (Datenschutz-Grundverodnung). Im Englischen General Data Protection Regulation (GDPR)). Diese ist daher für alle EU-Mitgliedsstaaten als geltendes Recht zu betrachten, anders als bei EU-Richtlinien. Aus der DSGVO ergeben sich daher unmittelbar Datenschutzanforderungen und Verarbeitungsvorschriften, welche für alle Unternehmen in Deutschland gelten.

Die DSGVO ist jedoch nur ein Gesetzeswerk, welches von der EU vorgegeben wird. Je nach Land (auch Bundesland) Art der Daten sowie dem Grund der Verarbeitungstätigkeit können jeweils weitere Datenschutzanforderungen durch weitere Gesetzestexte hinzukommen. Beispiele hierfür wären das Bundesdatenschutzgesetz (BDSG), Landesdatenschutzgesetze oder Gesetze wie dem Telekommunikation Telemedien Datenschutz Gesetz (TTDSG). Dabei zu beachten ist, dass die DSGVO als geltendes EU-Recht über dem deutschen Recht steht und weitere Gesetzestexte nur ergänzen oder, falls vorgegeben, präzisieren dürfen.

3.1 Personenbezogene Daten

Es gilt zuerst einzuordnen, um was für Daten es sich handelt, die sogenannten personenbezogenen Daten. Definiert werden personenbezogene Daten im Artikel 4 Absatz 1 der DSGVO (Begriffsbestimmungen):

> „personenbezogene Daten“ alle Informationen, die sich auf eine identifizierte oder identifizierbare natürliche Person (im Folgenden „betroffene Person“) beziehen [Art. 4 I DSGVO]

Beispiele hierfür sind neben offensichtlichen wie dem Namen, der Anschrift, biometrische Daten wie Fingerabdrücken auch z. B. Standortdaten, Einkommen, Krankheiten oder Zeugnisnoten.

3.2 Verarbeitung

Sobald Daten in irgendeiner Weise sowohl manuell als auch automatisch verwendet werden, egal ob sie gesammelt / erhoben, gelesen, bearbeitet oder nur gespeichert werden, wird von Verarbeitung gesprochen (Artikel 4 Absatz 2 DSGVO).

Für die Verarbeitung von personenbezogenen Daten gibt es Grundsätze, wie Daten zu verarbeiten sind. Diese werden durch Artikel 5 DSGVO (Grundsätze für die Verarbeitung personenbezogener Daten) vorgegeben:

- Abs. 1 a): Personenbezogene Daten müssen transparent und nachvollziehbar verarbeitet werden. Die Person, dessen Daten verarbeitet werden, muss daher aufgeklärt werden, welche Daten von ihnen durch wen wie verarbeitet werden und wofür sie konkret verwendet werden. Es reicht z. B. daher nicht zu sagen, dass die Smartphones einfach nur kontrolliert bzw. einer Sicherheitsanalyse unterzogen werden. Die betroffene Person muss genau aufgeklärt werden, dass alle ihre Daten auf dem Smartphone für eine Sicherheitskontrolle durch das System und auch durch Menschen verarbeitet werden. Sollte das System von einem externen Dienstleister betrieben werden, so muss dies auch kommuniziert werden.
- Abs. 1 b): Die Daten dürfen nur wie vorher festgelegt und vereinbart verarbeitet werden (Daten sind zweckgebunden). Bei der Sicherheitsanalyse würde dies bedeuten, dass die Daten nur für die Sicherheit ausgewertet werden dürfen. Nicht hingegen, um z. B. eine Leistungskontrolle durchzuführen.
- Abs. 1 c): Es dürfen nur die Daten erhoben und verwendet werden, welche unbedingt für die Erreichung des Ziels benötigt werden (Datenminimierung). Bei der Analyse würde dies bedeuten, dass z. B. Nachrichtenverläufe zwischen Mitarbeitern

nicht einbezogen werden dürfen, da diese mit dem vorher festgelegten Zweck nichts konkret zu tun haben.

- Abs. 1 d): Nur richtige Daten dürfen verwendet werden. Sind Daten nicht mehr richtig, so müssen neue Daten von der betroffenen Person eingefordert oder verworfen werden. Eine Verarbeitung mit einem alten Datenbestand ist daher nicht rechtens. Sicherheitsanalysen auf einem alten Abbild des Systems durchzuführen, wäre somit rechtswidrig.

- Abs. 1 e): Personenbezogene Daten dürfen nur so lange gespeichert werden, wie sie auch benötigt werden. Sollten sie für die ursprüngliche Zweckverbindung nicht mehr benötigt werden, so müssen diese gelöscht oder anonymisiert werden. Bei der Anonymisierung muss ausgeschlossen werden, dass die Daten eine Person identifizieren oder sie mit einer Person in Verbindung zu setzen sind. Für die Analyse bedeutet dies, das sollte aus einem Grund Daten wie z. B. eine kompromittierte Anwendung gespeichert werden, für eine späteren Wiedererkennung, so darf es nicht möglich sein, diese auf das ursprüngliche Smartphone und dadurch auf den Besitzer zurückzuführen. Auch darf eine mögliche Dokumentation der Sicherheitsanalyse keine personenbezogenen Daten über die Dauer der Kontrolle hinweg beinhalten.

- Abs. 1 f): Personenbezogene Daten müssen sicher gehandhabt werden nach bestmöglichem Stand vor Manipulation, Exfiltration oder Abhandenkommen geschützt werden (Integrität und Vertraulichkeit der Daten).

Diese Pflichten müssen nicht nur eingehalten werden, sondern müssen bei einer Prüfung z. B. mittels eines Sicherheitskonzeptes und eines Löschungskonzeptes nachweisbar sein (Artikel 5 Absatz 2 DSGVO).

Unter einem Löschungskonzept versteht man dabei ein Dokument, welches den genauen Ablauf / Prozess in Unternehmen beschreibt, wie welche Daten von wem unter welcher Verantwortung wie gelöscht werden.

Die Verarbeitung wird dabei durch Artikel 9 DSGVO (Verarbeitung besonderer Kategorien personenbezogener Daten) weiter eingeschränkt. Dieser untersagt die Verarbeitung von sensiblen personenbezogenen Daten, welche z. B. zur Diskriminierung verwendet werden könnten, solange kein berechtigtes Interesse und oder eine explizite Einwilligung oder eigenständige Veröffentlichung durch die Person vorliegt. Betroffene Daten hiervon sind:

> personenbezogener Daten, aus denen die rassische und ethnische Herkunft, politische Meinungen, religiöse oder weltanschauliche Überzeugungen oder die Gewerkschaftszugehörigkeit hervorgehen, sowie ... genetischen Daten, biometrischen Daten zur eindeutigen Identifizierung einer natürlichen Person, Gesundheitsdaten oder Daten zum Sexualleben oder der sexuellen Orientierung [Art. 9 I DSGVO]

Für die Sicherheitsanalyse bedeutet dies, dass jegliche Einstellungen oder Anwendungsdaten, welche diese Informationen enthalten könnten, nicht untersucht werden sollten oder nur unter geeigneten Umständen und einer vorhandenen Einverständniserklärung. Ein Beispiel hierfür wäre die Erkennung von bösartigen Zertifikaten. Das Durchsuchen von Nachrichtenverläufen nach Auffälligkeiten jedoch nicht. Entsteht bei einer Verarbeitung von Daten ein hohes Risiko für die Rechte und Freiheiten einer natürlichen Person, wie z. B. bei der Verarbeitung von personenbezogenen Daten besonderer Kategorien, so muss derjenige, der vorhat, die Daten zu verarbeiten, eine sogenannte Datenschutz-Folgenabschätzung (DSFA) anfertigen (Artikel 35 DSGVO Datenschutz- Folgenabschätzung). Diese muss anschließend an die zuständigen Behörden gemeldet werden. In der DSFA wird präzisiert, um welche Daten es sich handelt, wie diese verarbeitet werden und unter welchem Umständen und was ein Fehler in der Verarbeitung für Schäden bei der betroffenen Person verursachen kann (Nutzen-Risiko-Abwägung).

In Deutschland gibt es hierfür offizielle Vorlagen (Muster mit Hinweisen für die Durchführung einer Risiko- und Datenschutzfolgenabschätzung (DSFA)) durch die BfDI (Bundesbeauftragter für den Datenschutz und die Informationsfreiheit).

Eine solche Folgenabschätzung muss bei dem produktiven Einsatz, wie bei der hier vorgeschlagenen Analyse auch stattfinden. Diese wird benötigt, da es zu einer systematischen Überwachung / Sicherheitsanalyse der Smartphones kommt, in welcher die Verarbeitung von personenbezogenen Daten besonderer Kategorien nicht auszuschließen ist. Zudem kann es bei bösartigen Funden auf den Smartphones, welche auf die Grobfahrlässigkeit der Besitzer zurückzuführen ist, zu rechtlichen Konsequenzen kommen [14].

3.2.1 Datenhandhabung

Auch wie die Daten zu handhaben sind, wird in der DSGVO durch Artikel 25 (Datenschutz durch Technikgestaltung und durch datenschutzfreundliche Voreinstellungen) und Artikel 32 (Sicherheit der Verarbeitung) vorgeschrieben:

> Unter Berücksichtigung des Stands der Technik, der Implementierungskosten und der Art, des Umfangs, der Umstände und der Zwecke der Verarbeitung sowie der unterschiedlichen Eintrittswahrscheinlichkeit und Schwere des Risikos für die Rechte und Freiheiten natürlicher Personen treffen der Verantwortliche und der Auftragsverarbeiter geeignete technische und organisatorische Maßnahmen, um ein dem Risiko angemessenes Schutzniveau zu gewährleisten; [Art. 32 I DSGVO]

An die Datenhaltung und Übertragung gibt Abs. 1 a) dabei vor, dass personenbezogene Daten pseudoanonymisiert (Separierung von identifizierenden Daten vom Rest der Daten, Artikel 4 Absatz 5 DSGVO) und verschlüsselt werden. Zur Veranschaulichung für die Pseudoanonymisierung von Daten dient die Abbildung 3.1.

Des weiteren fordern Abs. 1 a) und b), dass die Systeme und Dienste das sog. C.I.A. IT Sicherheitsprinzip, Vertraulichkeit (Confidentiality), Integrität (Integrity) und Verfügbarkeit (Availability) erfüllen. Dies ist auch unter hoher Belastung der Systeme und bei Fehlern bei der Software und oder Hardware sicherzustellen beziehungsweise zeitnah wiederherzustellen.

Zusätzlich reicht es nicht, dies einmalig sicherzustellen, da Abs. 1 d) den Verarbeiter bindet, technische und organisatorische Maßnahmen (TOM) zu ergreifen, um die Sicherheit der Daten zu gewährleisten. Die TOM müssen dafür kontinuierlich neu re-evaluiert und bei Verbesserungsmöglichkeiten anpasst werden. Zudem müssen nach Absatz 3 die TOM mittels z. B. anerkannter Zertifikate (Art. 42 DSGVO Zertifizierung) nachgewiesen werden.

Artikel 25 greift dabei im Kern die gleichen Ziele auf, jedoch wird in diesem verpflichtet, dass die Weitergabe von personenbezogenen Daten an Dritte nicht erfolgen darf, ohne dass dies die betroffene Person explizit einwilligt (sog. Voreinstellungen wie z. B. die Cookies in Webbrowsern). In diesem Zusammenhang werden auch oft die Bezeichnungen "privacy by default" und "privacy by design" in der Softwareentwicklung als Indikatoren für die Einhaltung und Gestaltung der Software nach diesen Richtlinien genannt.

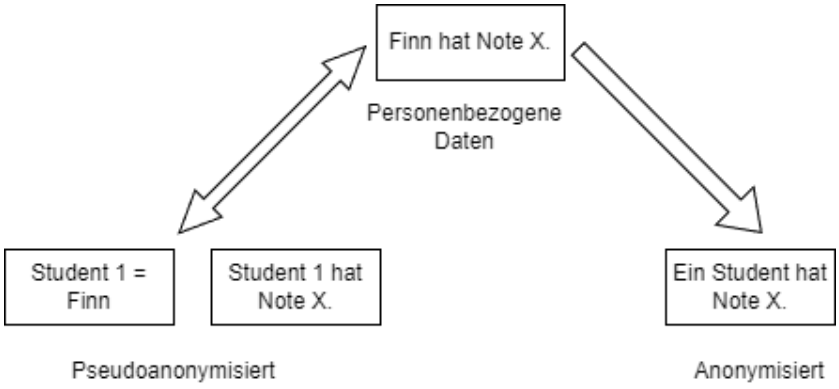

Abbildung 3.1: Unterschied Pseudoanonymisierung und Anonymisierung

3.2.2 Meldepflicht

Kommt es bei der Verarbeitung von personenbezogenen Daten durch Fehlern (Artikel 4 Absatz 12 DSGVO) zu einem Datenschutzvorfall, so sind Unternehmen verpflichtet, dies zu melden. Ein Fehler wäre z. B. ein Datendiebstahl, eine falsche Verwendung (Verletzung der Zweckbindung), die Weitergabe an nicht Berechtigte oder dem Verlust der Daten.

Es handelt sich hierbei nicht um einen Canossagang in die Öffentlichkeit, um das Ansehen zu bewahren, sondern um die Verpflichtungen durch Artikel 33 DSGVO (Meldung von Verletzungen des Schutzes personenbezogener Daten an die Aufsichtsbehörde). Diese verlangt die Meldung eines Datenschutzvorfalls innerhalb von 72 Stunden bei dem Datenschutzbeauftragen und der zuständigen Aufsichtsbehörden. Zudem müssen auch alle Personen, welche von dem Datenschutzvorfall potenziell betroffen sein könnten, ebenfalls darüber direkt oder über eine allgemeine Bekanntmachung informiert werden (Artikel 34 Benachrichtigung der von einer Verletzung des Schutzes personenbezogener Daten betroffenen Person).

Die personenbezogene Meldung kann unter bestimmten Umständen entfallen, wenn ausgeschlossen werden kann, dass die Daten nicht den Personen zugeordnet werden können. Beispielsweise wenn die Daten beim Abhandenkommen noch irreversible verschlüsselt oder nicht direkt personenbezogen waren, wie bei der Pseudoanonymisierung, sofern nicht auch die dazugehörigen identifizierenden Daten mit betroffenen sind, da ansonsten eine Zuordnung wieder möglich ist.

Dies gilt bei der Sicherheitsanalyse von Smartphones besonders, da es hierbei sowohl bei der eigentlichen Analyse zu Datenschutzvorfällen kommen kann, als das auch während einer Analyse ein solcher Vorfall festgestellt werden kann.

Trotz geeigneter TOM ist es möglich, dass ein Datenschutzvorfall auftritt. Dies kann z. B. an direktem menschlichen Versagen liegen (Murphys Gesetz). Gerade die Verheimlichung solcher Vorfälle führt nicht oft zum sog. Streisand-Effekt, wodurch das Ganze zu noch mehr negativem Ansehen führt als auch zu einer noch höheren Strafe.

Insgesamt richten sich Strafen bei Verletzungen der DSGVO und daher auch bei Datenschutzvorfällen nicht nur danach, wie mit dem Datenschutzvorfall umgegangen wurde, wie viel Schaden die betroffenen Personen erlitten haben, sondern auch danach, wie viel das betroffene Unternehmen in Prävention gesetzt hat (Qualität der TOM) oder ob es sogar fahrlässig gehandelt hat (Artikel 83 DSGVO Allgemeine Bedingungen für die Verhängung von Geldbußen).

3.2.3 Auftragsverarbeiter

Doch nicht immer erfolgt die Verarbeitung personenbezogener Daten in dem Unternehmen, welches sie ursprünglich erhoben hat. Dies ist die Norm, da die wenigsten Unternehmen alle Dienstleistungen selber durchführen und dabei die Daten zu keinem Zeitpunkt in die Hände von unbefugten Dritten gelangen dürfen. Ist dies der Fall, so wird in den meisten Fällen ein Auftragsverarbeitungsvertrag (AVV) zwischen dem Unternehmen, welches die personenbezogenen Daten erfasst hat (Auftragsgeber / Verantwortlicher) und dem Auftragsverarbeiter erstellt (Artikel 28 DSGVO Auftragsverarbeiter). Somit ist der Auftragsverarbeiter nicht mehr als unbefugter Dritter anzusehen.

Ein AVV kann nur dann entfallen, wenn ein Dritter sie nur für eine konkrete Ausführung braucht und diese Daten nicht weiter verarbeitet. Daher braucht es z. B. keinen AVV mit der Post. Ein Auftragsverarbeiter darf dabei keine weiteren AVV ohne Einwilligung seines Auftraggebers (Verantwortlichen) abschließen (Absatz 2).

Ein Paradebeispiel aus der IT, welches nicht alle Unternehmen beachten, sind Rechenzentren. Werden dort personenbezogene Daten verarbeitet (und sei es nur gespeichert) wie z. B. Mitarbeiterdaten für die Analyse, so muss ein AVV abgeschlossen werden.

Ein wichtiges Augenmerk sollte dabei auf die Verantwortlichkeit für die Daten gelegt werden. Nicht nur der Auftragsverarbeiter hat gemäß DSGVO TOM zu ergreifen und

ist bei einem Datenschutzvorfall haftbar, sondern auch der ursprüngliche Auftraggeber, da dieser verpflichtet ist, sicherzustellen, dass alle Datenschutzvorschriften eingehalten werden, auch aufseiten der Auftragsverarbeiter (Absatz 1). Daher bleibt derjenige, der die Daten erhoben hat, der Verantwortliche und ist auch verpflichtet, den Auftragsverarbeiter zu prüfen.

3.3 Personenrechte

Neben den Datenschutzanforderungen, wie Daten gehandhabt und verarbeitet werden, sowie die Pflichten der Verantwortlichen und oder Verarbeiter, so haben auch die Personen, dessen Daten verarbeitet werden, Rechte. Das wichtigste der Personenrechte ist dabei die Rechtmäßigkeit der Verarbeitung (Artikel 6 DSGVO Rechtmäßigkeit der Verarbeitung). Dieses fordert immer eine konkrete Einwilligung, sofern die Verarbeitung von personenbezogenen Daten nicht durch Lebensnotwendigkeit, Erfüllung rechtlicher Pflichten oder Verletzung von rechtlichen Vorgaben wie bei der Strafverfolgung, zu begründen ist.

Da es sich bei der Analyse um keine der Ausnahmen handelt, ist eine Einwilligung vor jeder Analyse einzuholen und zugleich muss auch ein Widerspruch akzeptiert werden (Artikel 21 DSGVO Widerspruchsrecht). Wie ein Unternehmen damit umzugehen hat, muss durch interne Richtlinien geklärt werden, so könnte es z. B. sein, dass diese Geräte dann nicht mehr verwendet werden dürfen.

Ein weiter Sonderfall kann dabei vorliegen, wenn ein Beschäftigungsverhältnis vorliegt (Artikel 88 DSGVO Beschäftigungskontext), da in diesem Fall länderspezifische Gesetzes Regelungen gelten können. In den meisten Fällen greift dabei das Bundesdatenschutzgesetz (BDSG) mit dem Paragrafen 26 (Datenverarbeitung für Zwecke des Beschäftigungsverhältnisses). In diesem werden Ausnahmen, in denen keine Einwilligung seitens der Arbeitnehmer benötigt werden, aufgeführt. Da es sich dabei bei Smartphones nicht direkt oder exklusiv um die Ausübung von Arbeitstätigkeiten handelt und es auch nicht konkret um eine Strafverfolgung geht, wird in den meisten Fällen wie bei der Analyse trotzdem eine Einverständniserklärung benötigt.

Neben der benötigten Einwilligung und dem Widerspruchsrecht hat jede Person durch Artikel 13 der DSGVO (Informationspflicht bei Erhebung von personenbezogenen Daten

bei der betroffenen Person) und Artikel 15 (Auskunftsrecht der betroffenen Person) Anrecht darauf, darüber informiert zu werden, welche personenbezogenen Daten für welchen Zweck wie erhoben, verarbeitet, gespeichert und auch gelöscht werden.

Da bei der Sicherheitsanalyse eine ungewollte Verarbeitung von personenbezogenen Daten nicht auszuschließen ist, bedeutet dies konkret, dass alle Datenverarbeitungen zu protokollieren sind. Dieses Protokoll ist der betroffenen Person auf Anfrage auszuhändigen. Das Protokoll stellt dabei auch wieder personenbezogene Daten dar und unterliegt somit auch der DSGVO.

Diese Daten sind entweder sofort, sofern die Aushändigung durch die betroffene Person nicht eingefordert wurde oder spätestens auf Nachfrage, da jede Person dieses Recht nach Artikel 17 DSGVO (Recht auf Löschung / Recht auf Vergessenwerden) hat, nach der Analyse zu löschen.

Bei der Löschung müssen dabei die betroffenen Personen benachrichtigt werden, sodass diese die Löschung und daher das Wissen, welche Daten ein Unternehmen noch über sie hat, nachvollziehen können (Artikel 19 DSGVO Mitteilungspflicht im Zusammenhang mit der Berichtigung oder Löschung personenbezogener Daten oder der Einschränkung der Verarbeitung).

Gerade hinsichtlich Schadensersatzes sollten Unternehmen sehr penibel darauf achten, dass es zu keinem Datenschutzvorfall kommt. Hierbei hat jede betroffene Person das Recht auf Schadenersatz bei Missbrauch der eigenen Daten bei dem verantwortlichen Unternehmen geltend zu machen (Artikel 82 DSGVO Haftung und Recht auf Schadensersatz).

Insbesondere der EuGH (Europäischer Gerichtshof) hat erst kürzlich entschieden, dass es nicht zu einem konkreten Schaden kommen muss, sondern dass das bloße Abhandenkommen in die Hände unbefugter Dritter bereits einen Schadensersatz nach sich zieht:

> Allein der Umstand, dass eine betroffene Person infolge eines Verstoßes gegen die DSGVO befürchtet, dass ihre personenbezogenen Daten durch Dritte missbräuchlich verwendet werden könnten, kann einen „immateriellen Schaden“ darstellen. [7]

Dieses Urteil ist besonders wegweisend, da anders als nach dem Grundrecht der Europäischen Union, nicht die betroffene Person die Schuld beim Unternehmen beweisen

muss (Artikel 48 Unschuldsvermutung und Verteidigungsrechte), sondern das Unternehmen, welchem die Daten abhandengekommen sind, nachweisen muss, dass sie unschuldig sind.

Der reine Kontrollverlust über die Daten reicht dabei für die Beantwortung der Schuldfrage. Eine konkrete Ausnutzung wie z. B. das nach einem Datenabfluss Waren durch Identitätsdiebstahl auf falschen Namen gekauft wurden und der betroffenen Person einen finanziellen Schaden zugefügt worden ist, muss daher nicht eingetreten sein. Ihre Unschuld können Unternehmen daher nur beweisen, indem sie nachweisen, dass sie korrekt nach der DSGVO gehandelt haben und alle möglichen TOM getroffen haben. Konkret bedeutet dies aber, das Unternehmen außer im Falle von Zero-Day-Angriffen in den meisten Fällen als schuldig anzusehen sind.

3.4 Daten auf Androidsystemen

Auf Androidsystemen existieren viele personenbezogene Daten, welcher dem Datenschutz unterliegen. Die geläufigsten sind dabei:

- Kontakte
- Bilder und Videos
- Profildaten wie dem Namen, Geschlecht usw.
- Nachrichten in Form von SMS
- Anwendungsdaten wie von Messenger, Browser usw.

Für die Sicherheitsanalyse sind dabei primär die Anwendungen, Bibliotheken, Systemdateien oder Systemkonfigurationen wie Wurzelzertifikate wichtig. Personenbezogene Daten wie SMS könnten allerdings auch darunter fallen, um z. B. Phishingversuche zu erkennen.

Es bietet sich daher an, die benutzerbezogenen Dateien, welche in dem "User" Benutzerverzeichnis auf der *Data* Partition liegen (2.2.3) von der Analyse auszuschließen, sofern keine berechtigte Annahme besteht, dass sich in diesem sicherheitsrelevante Dateien aufhalten könnten.

Insbesondere Anwendungen können sich trotz der Tatsache, dass sie populär sind und offiziell über den Playstore bezogen werden können, im Nachhinein als Schadsoftware entpuppen (Trojaner). Gerade kostenlose VPN Anbieter fielen dabei häufiger auf an DDoS (Distributed Denial of Service) Angriffen beteiligt zu sein [13].

Auch bei Anwendungsuntersuchungen kann es dabei zu personenbezogenen Daten kommen. Fertigt man z. B. eine Sammlung oder Protokollierung von dessen Existenz an, wie z. B. für einen Softwarekatalog, um dessen Schwachstellen über die Analyse und Androidsysteme hinweg zu verfolgen, so kann es durch die Anwendungen zu einem eindeutigen Personenprofil kommen (Profling), wodurch diese Daten personenbezogen werden.

4 Anforderungsanalyse

4.1 Einstieg

Im Rahmen der Bachelorarbeit soll ein System entwickelt werden, welches jedoch zuerst konzipiert werden muss. Während in der Softwareentwicklung heutzutage agile Modelle existieren und iteratives / inkrementelles Entwickeln von Softwaresystem die Norm ist, so wird im Folgenden das Wasserfallmodell als Vorgehensmodell verwendet.

Das Wasserfallmodell bietet sich hierfür an, da es ein System und dessen kompletten Softwareentwicklungslebenszyklus (SDLC, aus dem englischen Software Development Life Cycle) klassisch in einem Durchgang wie bei einem Pflichten- und Lastenheft abwickelt. Dabei werden die Anforderungen erfasst, ein Entwurf / eine Architektur erstellt bis hin zur Implementierung, Inbetriebnahme und der anschließenden Wartung ohne Zwischenabnahmen vom Kunden [19].

Die Nachteile des Modells liegen vor allem darin, dass Fehler in den frühen Phasen, insbesondere der Anforderungsanalyse und dem Entwurf im späteren Projektverlauf für exponentiell steigende Kosten sorgen [5].

Im Folgenden werden Anwendungsfälle (Use Case) des zu entwickelnden Produktes erarbeitet, in dem detailliert durch simuliert wird, wie das Produkt später verwendet werden soll und wie es welche Aufgaben unter welchen Kriterien und Sicherheitsaspekten umzusetzen hat.

Der Fokus liegt dabei darauf, echte Anwendungsszenarien der Endnutzer für dessen spätere Produktverwendung zu erfassen. Unnütze oder sinnlose Funktionen sollen dabei eliminiert werden und potenzielle Hürden, wie z. B. die erwartete Reaktionen oder das Verhalten des Systems in Arbeitsabläufen festgehalten werden.

Am Ende der Anforderungsanalyse ergibt sich dadurch ein Anforderungsportfolio. Dieses soll sicherstellen, dass die gewollte Funktionalität unter Berücksichtigung der Anwender

Bedürfnisse und unter den gegebenen Projektvoraussetzungen beziehungsweise Vorgaben (dem Datenschutz) erfüllt werden.

4.1.1 Prozessentwicklung und Standardisierung

Unter der Annahme, dass das neu entstehende System in Unternehmen eingesetzt wird, bedeutet dies, dass die Abläufe des zu entstehenden Systems optimiert und sofern möglich automatisiert werden sollten.

Dies folgt aus der Tatsache, dass die Masse von Smartphones, welche periodisch oder auf Verdacht geprüft werden müssen, ansonsten für einen enormen Zeit- und Kostenaufwand sorgen.

Für die Benutzer der Smartphones, daher den potenziellen Mitarbeitern, bedeutet dies konkret, dass je länger die Untersuchung dauert, umso länger müssen sie auf ihre Geräte verzichten. Für den potenziellen Arbeitgeber bedeutet dies wiederum, dass er nicht nur mehr Forensiker benötigt, sondern auch mit längeren und häufigeren Ausfällen zu rechnen hat, da die Mitarbeiter womöglich ohne ihre Geräte arbeitsunfähig sind.

Durch den potenziell hohen Kostenaufwand entsteht nicht nur eine Abneigung von Unternehmen für die Nutzung eines solchen Systems, da eine Kosten-Nutzen-Analyse für die Entscheidungsträger das Vorhaben als unwirtschaftlich wirken lassen könnten, sondern dass es auch bei der Anwendung (Sicherheitsanalyse) zu Zeitdruck und dadurch zu einem höheren Fehlerpotenzial kommen könnte.

Um das Fehlerpotenzial zu reduzieren, welches zu Datenschutzvorfällen führen kann, bietet es sich an, Prozesse als auch SOPs (Standard Operating Procedures) einzuführen, da durch diese TOM (bzgl. 3.2.1) ein sichererer und effizienterer Ablauf gewährleistet werden kann.

Während des operativen Einsatzes sollte daraufhin die Einhaltung als auch Verbesserungsmöglichkeiten der Prozesse und SOPs (TOM) überwacht werden. Dies könnte in Form eines PDCA-Zyklus (Plan, Do, Check, Act zu Deutsch Planen, Umsetzen, Überprüfen, Handeln) umgesetzt werden.

4.1.2 Technische Gegebenheiten

Da das entstehende System später in dem Firmennetz des kooperierenden Unternehmens (der Dataport AöR) mit integriert werden soll, müssen die benötigten Schnittstellen und Technologien als Rahmenbedingung festgehalten werden. Hintergrund dessen ist, dass später bei der Integration und der Wartung durch Mitarbeiter, welche nicht an der ursprünglichen Entwicklung beteiligt waren (in dem Sinne jeder aufgrund der Bachelorarbeit Charakteristik), keine Probleme auftreten.

Folgende Anforderungen wurden dabei festgehalten:

- Die Nutzeranbindung soll dabei über LDAP an das Unternehmens Active Directory (AD) realisierbar sein oder über ein IAM-System wie Keycloak oder Authelia direkt oder in einer Reverse Proxy Architektur (4.1).
- Sollten Datenbank Systeme verwendet werden, so müssen diese Industriestandards sein wie MySQL von Oracle für relationale Datenbanksysteme oder MongoDB für dokumentenbasierte Datenbanksysteme.
- Zu verwendende Programmiersprachen sind C# oder Java sowie geläufige Skriptsprachen (Javascript, Python, Bash, Powershell).
- Daten / Objekte, welche über Schnittstellen ausgetauscht werden, müssen über JSON oder XML realisiert werden. Keine proprietären Objekt De-/Serialisierungen.
- Die serverseitigen Komponenten müssen alle mittels Containerisierung in einer skalierbaren Kubernetes oder Portainer Umgebung (Docker) eingebunden werden können.

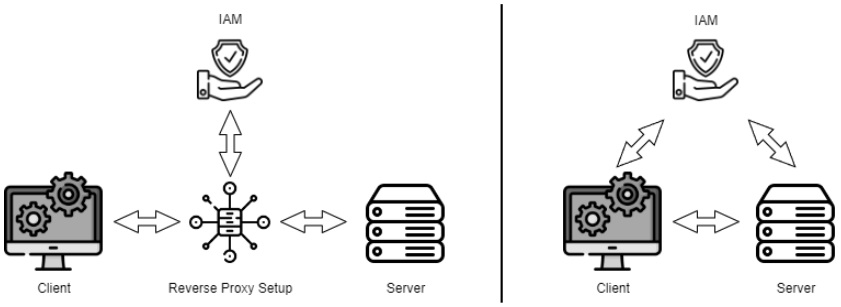

Abbildung 4.1: IAM in einer Reverse-Proxy- und in einer direkten Client-Server-Kommunikationsarchitektur

4.1.3 Nomenklatur

Bevor die Anwendungsfälle konkretisiert werden können, muss zunächst festgestellt werden, welche Personengruppen oder andere Systeme mit dem zu entwickelnden System interagieren werden. Die sogenannten Akteure:

- **System:** Das zu entwickelnde System selbst.
- **Externes System:** Andere bestehende System, welche mittels interner oder externer Schnittstellen mit dem neu entstehenden System kommunizieren.
- **Besitzer:** Personen, welche die Smartphones besitzen und verwenden. Diese müssen jedoch nicht die Eigentümer sein.
- **Forensiker:** Personen, welche die hauptsächlichen Endnutzer des entstehenden Systems sind und die Sicherheitsanalyse durchführen.
- **Sachbearbeiter:** Personen, welche die Smartphones verwalten und für die Sicherheitsanalysen vorbereiten.
- **Benutzer:** Tätigkeiten / Anwendungsfälle, welche sowohl durch Forensiker als auch Sachbearbeiter durchgeführt werden können.

Authentifizierung und Autorisierung

In den Anwendungsfällen werden die ausführenden oder auslösenden Akteure angegeben. Sollte es sich bei diesen um Benutzer oder externe Systeme, welche über externe Schnittstellen mit dem System kommunizieren, handeln, so ist vor der Ausführung ihrer Aktionen von Ihnen eine Authentifizierung und Autorisierung durch das entstehende System vorzunehmen.

Anwendungsfall 1: Ein Benutzer muss sich am System anmelden können.

Die Anmeldung soll dabei einmalig zu Beginn der Arbeitstätigkeit erfolgen (SSO). Außerdem sollte es nicht möglich sein, ohne Anmeldung mit dem System zu kommunizieren, als auch Anwendungsdaten des Systems, welche auf dem Benutzer PC gespeichert werden, zu exfiltrieren, da diese verschlüsselt vorliegen müssen. Diese Maßnahme dient zur Prävention vor Daten-Exfiltration bei Diebstahl des Benutzer PCs (bzgl. 3.2.1, 3.2).

Alle nachfolgenden Anwendungsfälle, welche sich auf Benutzer beziehen, setzen voraus, dass der Benutzer angemeldet ist.

4.2 Inventar

Für die logistische Verwaltung der Smartphones dienen die folgenden Anwendungsfälle. Bezüglich der Terminologie wird unterschieden zwischen unbekannten, registrierten und eingepflegten Geräten (Smartphones).

Unbekannte Geräte sind Smartphones, welche beschafft wurden oder bereits existieren und nun vom System erfasst werden sollen. Registrierte Geräte sind jene, welche bereits vom System erfasst wurden, jedoch noch nicht einem Besitzer zugeordnet wurden. Die eingepflegten Geräte hingegen sind Smartphones, welche im normalen Systemlauf teilnehmen und daher periodisch untersucht werden sollen. Der Inventarzyklus (Lebenszyklus) von Smartphones wird dabei in Abbildung 4.2 dargestellt.

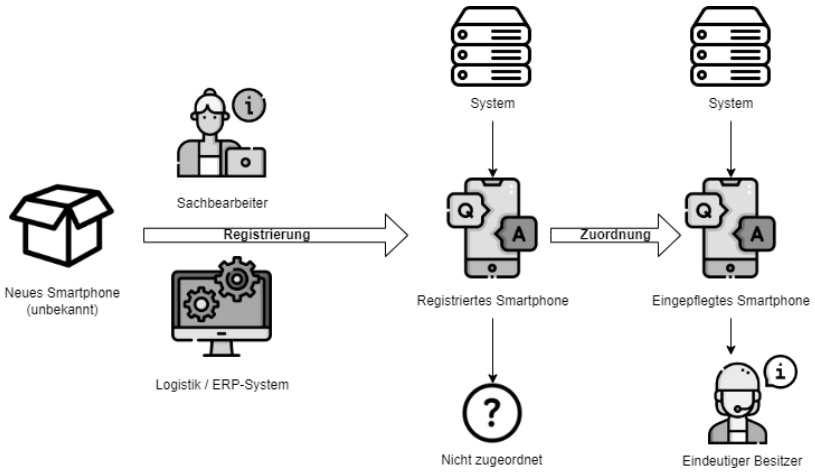

Abbildung 4.2: Inventarzyklus von Smartphones

Anwendungsfall 2: Ein Sachbearbeiter soll registrierte Geräte verwalten (einsehen und hinzufügen) können.

Neu angeschaffte Smartphones sollen dabei erfasst werden. Dies dient primär dazu, dass eine Übersicht über die Geräte erstellt wird, welche noch nicht am Systemlauf teilnehmen und noch eingepflegt werden müssen. Nicht eingepflegte Geräte könnten ansonsten potenziell in die Hände von Mitarbeitern gelangen und nicht untersucht werden, wodurch sie ein Sicherheitsrisiko darstellen. Diese unsicheren Smartphones können daher über die Übersicht nachverfolgt werden. Für die Identifizierung der Smartphones sollte die Vendor ID (Hardware-Seriennummer) verwendet werden. Die Android ID als auch die IMEI

werden von anderen Unternehmen wie Anwendungsentwicklern oder Mobilfunkanbietern usw. verwendet. Sie sollten daher nicht verwendet werden, da sie personenbezogen sind und dafür verwendet werden könnten, Sicherheitsanalysen unabhängige Daten mit den Daten von der Analyse in Verbindung zu stellen oder dem Besitzer zuzuordnen.

Anwendungsfall 3: Ein externes System soll registrierte Geräte hinzufügen können.

In Unternehmen werden Neuanschaffungen wie Smartphones nicht nur händisch, sondern häufig automatisiert über ERP (Enterprise Resource Planning) Systeme wie z. B. SAP abgewickelt. Dabei führen die Sachbearbeiter oft nur einen Anschaffungsantrag aus, welcher daraufhin über Freigabeprozesse genehmigt wird. Anschließend wird der restliche Ablauf (Bezahlung, Versand und Empfang) automatisch oder von anderen Mitarbeitern des Unternehmens übernommen. Dabei bieten diese Systeme oft Empfangs / Erfassungsbenachrichtigungen (Wareneingangsbuchung) an konfigurierbare Schnittstellen an, wenn neue Ressourcen wie die Smartphones erfasst wurden und daher nun dem Unternehmen physisch zur Verfügung stehen. Die automatische Erfassung bei solchen Anschaffungen dient auch zur Nachverfolgung, dass die Smartphones nicht aus dem Systemlauf fallen und dadurch nicht untersucht werden.

Anwendungsfall 4: Ein Sachbearbeiter soll registrierte Geräte zuordnen können.

Bei der Zuordnung werden Smartphones ihrem Besitzer zugewiesen. Dabei werden sowohl identifizierende Daten (z. B. Name, Mitarbeiternummer usw.) als auch Kontaktdaten (z. B. die E-Mail) des Besitzers erfasst. Es kommt daher zu personenbezogene Daten. Zudem wird ein Passwort durch den Besitzer vergeben ($KBesitzer$) als auch ein zufälliges generiert ($KSystem$), womit das Smartphone abgesichert wird. Bei der Absicherung wird sich dabei auf die Sicherung des Bootloaders bezogen (2.2.3), wobei die Bootloader Sperre dabei die Konkatenation der beiden Passwörter ist:

$$BootloaderPasswort = KSystem + KBesitzer$$

Im System wird dabei nur das zufällig generierte Passwort $KSystem$ gespeichert.

Durch die Nicht-Speicherung des Besitzerpasswortes wird ein sog. "Poka-Yoke" Mechanismus erschaffen [20], welche eine fehlerhafte Benutzung des Systems unmöglich macht. Ziel dabei ist es, dass der Besitzer bei einer späteren Untersuchung seine Einwilligung erteilen muss, wodurch sein Passwort als zusätzliche Einwilligungssicherung fungiert. Tut

er dies nicht oder macht Gebrauch von seinem Widerspruchsrecht, so kann die Untersuchung technisch nicht stattfinden, wodurch es zu keiner Rechtsverletzung diesbezüglich kommen kann. Ein zusätzlicher Effekt durch die Verwendung von 2 Passwörtern ist, dass der Besitzer selbst auch nicht auf den Bootloader Bereich zugreifen kann, um sich z. B. Root Rechte zu verschaffen.

Anwendungsfall 5: Ein Sachbearbeiter soll eingepflegte Geräte verwalten (einsehen, bearbeiten und löschen) können.

Sinn dieser Anforderung ist die Einsicht in die personenbezogenen Daten (sollte der Besitzer von seinem Recht auf Auskunft oder Vergessenwerden Gebrauch machen), als auch die Richtigkeit der Daten durch Bearbeitungen sicherstellen zu können.

Zudem muss es auch möglich sein, ein Gerät von seinem Besitzer zu löschen / die Zuordnung zu entfernen, sollte dieses nicht mehr existieren oder dem Besitzer zur Verfügung stehen. Die Daten über den Besitzer werden daher nicht mehr benötigt und müssen gelöscht werden.

4.3 Fallverwaltung

Die Orchestrierung der Untersuchungen werden durch die folgenden Anwendungsfälle abgedeckt. Jede Sicherheitsanalyse stellt dabei einen Fall (Fallakte) dar. Dabei wird sich an dem S-A-P-Modell (Secure Analyse Present Model)[8] orientiert, welches vom BSI als bevorzugtes Vorgehensmodell für die IT-Forensik vorgeschlagen wird [6].

Der Sinn des S-A-P-Modells liegt dabei in der systematischen Vorgehensweise bei forensischen Untersuchungen. Dabei werden in der ersten Phase (Secure) alle Beweise, in dem Fall alle benötigten Daten bzw. das Smartphone (*System* und *Data* Partition) gesichert, um diese daraufhin in der zweiten Phase (Analyse) auswerten zu können. Dabei muss es möglich sein, bei Bedarf auf vorher nicht berücksichtigte Artefakte zugreifen zu können, wodurch ein großer Wert auf die Genauigkeit der ersten Phase gelegt wird. Als Artefakt wird dabei jegliches Beweismittel in einer forensischen Untersuchung verstanden. Konkret für diese Arbeit bedeutet dies, jegliche Datei von dem zu untersuchenden System wie z. B. Anwendungen (APKs), Zertifikate, Systembibliotheken, Systemdateien usw. Die letzte Phase (Present) dient hierbei der Aufbereitung der Untersuchungsergebnisse und der daraus anschließenden Klassifikation, ob und im welchem Ausmaße ein Sicherheits- und oder Datenschutzvorfall festgestellt wurde. Des weiteren kann reflektiert werden,

wie die Sicherheitsanalyse abgelaufen ist bzw. was daraus gelernt werden kann (Lessons Learned). Die gesammelten Erfahrungen können daraufhin für eine Verbesserung der Analyse verwendet werden (PDCA-Zyklus). Der Ablauf wurde mittels Abbildung 4.3 dargestellt.

Abbildung 4.3: S-A-P Ablauf mit dem Zusatzpunkt Trigger (Auslöser)

Anwendungsfall 6: Ein Forensiker kann für ein zugeordnetes Smartphone einen Fall eröffnen.

Anwendungsfall 7: Das System eröffnet nach einer periodischen Zeit für jedes eingepflegtes Smartphone einen Fall.

Sinn dieser Anwendungsfälle ist, dass bei verdächtigem Verhalten, welches vom Benutzer festgestellt wurde oder nach dem Ablauf einer festgelegten Zeit ein Untersuchungsfall eröffnet wird (4.3 Trigger Phase). Dabei soll der Besitzer benachrichtigt werden, dasss eine Untersuchung eingefordert wird. Anwendungsfall 6 setzt voraus, das der Forensiker für die Fall-Eröffnung eine Liste aller zugeordneten Geräte zur Auswahl erhält.

Anwendungsfall 8: Ein Forensiker kann sich eine Übersicht über alle Fälle anzeigen lassen.

Anwendungsfall 9: Ein Forensiker kann sich einen Fall zuweisen.

Diese Anforderungen dienen der Verwaltung und der Arbeitsverteilung unter den Forensikern.

Anwendungsfall 10: Ein Forensiker kann einen zugewiesenen Fall mit dem betroffenen Gerät bearbeiten.

Dieser Anwendungsfall bezieht sich auf die eigentliche Durchführung des S-A-P-Modells (bzgl. 4.3). Hierfür muss das Smartphone als auch die Einverständniserklärung des Besitzers (das Benutzerpasswort $KBesitzer$) vorliegen, um das Smartphone zu entsichern (bzgl. 4.2 Anwendungsfall 4).

Sollte die Untersuchung beginnen, so müssen zuerst die Daten des Smartphones gesichert werden (Abbildung 4.3 Secure Phase). Hierfür kann der Recovery OS Ansatz aus 2.2.3 verwendet werden. Dieses sollte mit dem gleichen Passwort, welches auch für den Bootloader verwendet wird, gesichert werden. Es bietet sich jedoch an, nicht alle Dateisysteme / Partitionen zu sichern, sondern nur gezielt Artefakte zu extrahieren und zu untersuchen. Dies hat den Vorteil, dass die Datenverarbeitung von personenbezogenen Daten minimiert wird, indem z. B. das Benutzerverzeichnis nicht gesichert und untersucht wird (bzgl. 3.4).

Für die Untersuchung wird zudem ein Protokoll automatisch vom System angefertigt, welches die genaue Verarbeitung jeglicher Daten vom Smartphone als auch die Zugriffe auf dieses festhält, um den Benutzer daraufhin korrekt über das Ausmaß der Verarbeitung seiner Daten zu informieren. Das Protokoll sollte dabei mit dem Benutzerschlüssel $KBesitzer$ verschlüsselt werden. Dies ermöglicht es, das Protokoll zu speichern, ohne das dieses ein Dritter entschlüsseln / lesen kann. Die forensischen Tätigkeiten und Aspekte des Systems werden im nachfolgenden Unterkapitel behandelt (4.4).

Anwendungsfall 11: Ein Forensiker kann einen Datenschutzvorfall melden.

Datenschutzvorfälle müssen unabhängig von der Phase, sobald diese festgestellt werden, gemeldet werden. Auch wenn die 72 Stunden, welche rechtlich vorgeschrieben sind, noch nicht erreicht wurden oder noch nicht alle Informationen vorliegen, da diese später nachgereicht werden können (bzgl. 3.2.2).

Anwendungsfall 12: Ein Forensiker muss einen Fall schließen können.

Nachdem eine Untersuchung abgeschlossen ist (Abbildung 4.3 Present Phase), müssen die extrahierten Daten von dem Smartphone gelöscht werden (Löschkonzept). Unabhängig vom Ergebnis muss das angefertigte Protokoll dem Besitzer ausgehändigt werden, damit dieser über eine potenzielle Verarbeitung seiner personenbezogenen Daten informiert

wird. Zudem kann eine Post-Mortem-Analyse durchgeführt werden und eine Reevaluation des Vorgehens stattfinden (Abbildung 4.3 Present Phase).

4.4 Forensik

Das folgende Unterkapitel handelt über die eigentliche forensische Untersuchung der Smartphones (Abbildung 4.3 Secure und Analyse Phase, Anwendungsfall 10). Zuerst sollte jedoch untersucht werden, ob und wenn ja, welche Arbeiten es bereits in diesem Themengebiet existieren, um den Status quo zu etablieren. Bezüglich der Terminologie wird unter dem Wort Scanner eine Anwendung, Dienst oder externes System verstanden, welches ein Artefakt klassifiziert. Eine Klassifikation setzt dabei fest, ob ein Artefakt gut- oder bösartig ist.

4.4.1 Existierende Ansätze

Klassische Anti-Viren-Scanner Ansätze

Auch für Android existieren von zahlreichen Anbietern wie Avira, Avast oder Kaspersky bereits Viren / Artefakte Scanner. Diese basieren dabei hauptsächlich auf jeweils vorher festgelegten Datensätzen von bekannten bösartigen Artefakten und untersuchen daraufhin mit diesem Wissen neue Artefakte / Anwendungen. Die Qualität der Klassifikationen und dem Vertrauen, was die Forensiker diesen Klassifikationen zuschreiben, basiert daher auf blindem Vertrauen auf die Aussagen des jeweiligen Anbieters, über dessen Genauigkeit.

Es ist daher bereits gängige Praxis, mehrere solcher Lösungen anzuwenden und daraufhin eine finale Klassifikation aus den jeweiligen Klassifikationen zu bilden. Ein populäres Beispiel wäre hierfür der Onlinedienst Virustotal von Google, welcher es erlaubt, beliebige Artefakte von etlichen Scannern untersuchen zu lassen.

Auch für Android Artefakte gibt es bereits Forschung über solche Ansätze, welche aufzeigen, dass die Einigkeit von vielen Scanner Lösungen nicht vorhanden ist und daher eine gleiche Gewichtung von unterschiedlichen Anbietern vermieden werden sollte [12].

Auch seitens der künstlichen Intelligenz, gibt es bereits Forschungen über die Erkennung von bösartigen Artefakten mittels einer sog. Principal Component Analysis (PCA, KI

gesteuerte Mustererkennung mit Gruppierung / Clustern) und anschließender Outlier Erkennung (Erkennung nicht bekannter / häufige Muster). Mit diesem Vorgehen können bereits hohe Trefferquoten / korrekte Klassifikationen bereits mit kleinen Datensätzen erreicht werden [26].

Es entsteht durch die aufgeführten Argumente der Konsens, dass sich nicht alleine auf eine vollkommen automatisierte Klassifikation vertraut werden kann, sondern eine manuelle Nachforschung und abschließende Beurteilung erforderlich ist. Zudem sammeln gängige Anti-Viren-Scanner auch Daten bezüglich ihrer Erkennungen, was zu möglichen Übertragung von personenbezogenen Daten führen kann.

Sandbox Ansätze

Ein weiterer Ansatz neben den klassischen Anti-Viren-Scannern ist es, Android Anwendungen nicht nur mittels Mustererkennung bzw. Heuristiken zu als bösartig zu erkennen, sondern diese in einem virtualisierten Umfeld auszuführen und dessen Verhalten zu untersuchen [2]. Dieser Ansatz ist gerade hinsichtlich neuartiger Schadsoftware interessant, da diese ihren Schadcode oft verschlüsseln und oder erst zur Laufzeit entpacken oder nachladen (sog. Payloads).

Auf die Genauigkeit solcher automatisierten Untersuchungen sollte sich jedoch nicht verlassen werden, da die Schadsoftware auch über sog. Sandbox Awareness verfügen könnte, wodurch die Ausführung des Schadcodes bei Erkennung einer solchen Umgebung unterdrückt wird und es dadurch zu keiner korrekten Klassifizierung kommt.

Auch kann es wie bei den bereits erwähnten problematischen VPN Anwendungen (bzgl. 3.4) dazu kommen, dass die reine Ausführung nicht direkt schadhaftes Verhalten auslöst, sondern erst bei einer spezifischen Nutzung auftritt (VPN Verbindungsaufbau).

Cloud Ansätze

Auch für die hier behandelte "Massen Forensik" (bezogen auf die Anzahl zu untersuchender Smartphones) gab es bereits in der Vergangenheit Forschung bezüglich cloudbasierter Sicherheitsanalysen. In diesen werden die Daten (Artefakte) von den Smartphones auf einen Server hochgeladen und dort analysiert werden [24].

Wie in der Einleitung bereits erwähnt (bzgl. 1.2) entstehen hierbei aber potenziell LotL-Angriffsmöglichkeiten, als das auch das Risiko für Datenschutzvorfälle drastisch erhöht wird.

Insbesondere hinsichtlich des Datenschutzes kommt es hierbei zu Problemen, da mit den verwendeten externen Diensten AVVs abgeschlossen werden müssen. Durch diese kommt es nicht nur zu einer potenziell unsichereren Verarbeitung, sondern es kommt auch zu einer weiteren Fehlerquelle mit möglicher Haftung für denjenigen, der die Untersuchung durchführt.

4.4.2 Anwendungsfälle

Durch die Forschungserkenntnisse, welche im vorherigen Abschnitt aufgeführt wurden, ergibt sich, dass es datenschutztechnisch am sichersten ist, mehrere unterschiedliche lokale Scanner (On Premise) mit unterschiedlichen Methoden und einer abschließender manuellen Endklassifizierungs Möglichkeit zu verwenden.

Ziel dabei sollte es sein, die Androidforensik nicht neu zu erfinden, sondern bereits existierende Teilproblemlösungen systematisch und modular mit zu integrieren.

Für die Forensik entsteht dabei der folgende Prozess:

1. **Extraction Phase:** In dieser Phase werden gezielt Artefakte extrahiert.
2. **Analysis Phase:** Anschließend werden in dieser Phase die extrahierten Artefakte durch unterschiedliche Scanner untersucht und klassifiziert.
3. **Forensic Phase:** In dieser Phase werden die klassifizierten Artefakte noch einmal überprüft und auf Verdacht genauer untersucht und wenn nötig manuell endklassifiziert.

Aus den ersten beiden Prozessphasen (Extraction und Analysis) ergeben sich die folgenden Anwendungsfälle:

Anwendungsfall 13: Das System kann Artefakte von dem zu untersuchenden Smartphone automatisch extrahieren.

Hierbei wird automatisch durch den Pfad, in welchem sich ein Artefakt aufhält oder dem Typ (z. B. Anwendungen) entschieden, welche Artefakte extrahiert werden.

Anwendungsfall 14: Das System kann Artefakte automatisch klassifizieren.

Dabei soll die Klassifizierung durch eigene Implementierungen realisiert werden können als auch durch selbstverwaltete Scanner oder Untersuchungslösungen wie einem Sandbox-Ansatz.

Anwendungsfall 15: Das System kann Artefakte verfolgen.

Ziel dieser Anforderung ist es, dass Artefakte über Sicherheitsanalysen und Smartphones hinweg verfolgt werden können. Die Verfolgung der Artefakte bedeutet dabei, dass ihre Präsenz auf einem Smartphone bekannt ist. Dies ermöglicht es, dass bei einer bösartigen Klassifizierung eines Artefaktes durch z. B. eine Meldung vom BSI überprüft werden kann, ob die Smartphones, welche vom System verwaltet werden, betroffen sind und falls ja, sodass eine Sicherheitsanalyse angefordert werden muss. Eine direkte Zuordnung / Verfolgung zu einem gezielten Smartphone darf dabei nicht erfolgen, da diese Daten ansonsten personenbezogen sind bzw. zu Profiling führen könnten (bzgl. 3.4). Neben der Präsenz können auch vorherige Klassifizierungen gespeichert werden, wodurch ein Cache entsteht.

Aus der letzten Prozessphasen (Forensic) ergeben sich die folgenden Anwendungsfälle:

Anwendungsfall 16: Ein Forensiker kann die Ergebnisse der automatischen Klassifikation einsehen.

Unter Ergebnis wird dabei die vergebene Klassifikation als auch einer Begründung, wie diese zustande gekommen ist, gemeint.

Anwendungsfall 17: Ein Forensiker durch manuelle Aufforderung Artefakte extern überprüfen lassen.

Die Formulierung "extern" bezieht sich dabei auf einen Scanner von Dritten (nicht On Premise) wie z. B. Virustotal, welche nicht selbst verwaltet und gehostet wird. Untersuchungen von automatisch extrahierten Artefakten durch Dritte sollten nur manuell angesteuert werden, um das Risiko von Datenschutzvorfällen zu reduzieren.

Anwendungsfall 18: Ein Forensiker kann Artefakte manuell untersuchen.

Diese Anforderung dient der gezielten Analyse von Artefakten durch die Forensiker. Hierbei werden manuelle Untersuchungsmöglichkeiten gemeint. Ein Beispiel hierfür wäre eine Sandbox Untersuchung für eine extrahierte Anwendung (Artefakt), in welcher ein Forensiker diese z. B. dekompiliert, um den Quellcode oder dessen Verhalten zu untersuchen.

Anwendungsfall 19: Ein Forensiker kann Artefakte manuell klassifizieren und die automatische Klassifikation damit überschreiben.

Anwendungsfall 20: Ein Forensiker kann Kommandozeilenbefehle auf dem zu untersuchenden Gerät manuell ausführen.

Sinn dieser Anforderung ist es, dem Forensiker zu ermöglichen, selbstständig nicht automatisch extrahierte Artefakte oder das Smartphone anderweitig zu untersuchen.

Anwendungsfall 21: Ein Forensiker kann Einträge bzw. ergänzende Kommentare in das Protokoll hinzufügen.

Anwendungsfall 22: Ein Forensiker kann unabhängig, ob ein Fall bearbeitet wird oder bereits geschlossen wurde, das Fall Protokoll exportieren.

4.5 Zusammenfassung

In der Anforderungsanalyse wurde ein Portfolio an Anforderungen erarbeitet, welches die Sicherheitsanalyse so weit wie möglich automatisiert, um sowohl die Arbeit der Forensiker zu erleichtern als auch Fehler zu reduzieren, ohne ihm dabei tiefere Eingriffs- als auch Untersuchungsmöglichkeiten vorzuenthalten (bzgl. 1.3). Auch bezüglich der Verwaltung und Inventarisierung der Smartphones wurden alle relevanten Anwendungsfälle, welchen für einen realen operativen Betrieb benötigt werden, erarbeitet.

Die Anforderungen wurden dabei so gestellt und optimiert, dass der Datenschutz oberste Priorität hat und die Forensiker durch das entstehende Protokoll überwacht werden. Personenbezogene Daten werden dabei, so weit es möglich ist, vom System ignoriert (Benutzerverzeichnis) und von der eigentliche Sicherheitsanalyse / Forensik systematisch ausgeschlossen.

Ein direkter Eingriff in die Daten der Smartphones wurde somit, so weit es geht vermieden, wodurch die Anzahl an möglichen Berührungspunkten mit potenziell personenbezogenen Daten reduziert wurde. Beachtet werden muss jedoch, dass die Forensiker trotzdem freie Hand haben, insbesondere durch den Anwendungsfall 20, welcher eine uneingeschränkte Kommandozeileneingabe ermöglicht. Diese wird jedoch protokolliert und den manuell getätigten Befehlen sollte daher großen Wert bei der Betrachtung des Protokolls

gegeben werden. Bei diesen besteht das höchste Risiko bezüglich Kontakt und Verarbeitung von personenbezogenen Daten durch die Forensiker. Das Risiko, Daten ungewollt an Dritte zu übermitteln, wurde durch die manuelle Aufforderung entgegengewirkt.

5 Entwurf/Architektur

Nachdem die Anforderungsanalyse abgeschlossen wurde, erfolgt nun der Entwurf / die Architektur des zu entstehenden Systems. Orientiert wird sich dabei an dem Softwarearchitekturentwicklungsmodell arc42. Das Kapitel wird dabei in 3 Unterkapitel unterteilt. In 5.1 wird eine Kontextsicht für das zu entstehende System entwickelt. Daraufhin erfolgt in 5.2 die Konstruktion und Separierung der einzelnen Bausteine und dessen Komponenten. Unter dem Begriff Baustein wird eine Gruppierung von Komponenten gemeint (Aufgabengruppierung nach Aufgabengebiet). Jede Komponente in einem Baustein übernimmt dabei einen Teil des Aufgabengebiets. Im letzten Unterkapitel (5.3) erfolgt daraufhin eine Zusammenfassung des Entwurfs.

5.1 Kontextsicht

Vor dem Entwickeln einer Baustein- oder Komponentensicht muss zuerst erfasst werden, welche Akteure und Systeme miteinander interagieren, um eine High-Level-Übersicht zu erschaffen und um benötigte Schnittstellen zu identifizieren.

Dabei fallen zuerst die Hauptnutzer des Systems an die Forensiker, Sachbearbeiter und Besitzer (bzgl. 4.1.3). Zudem soll die Authentifizierung und Autorisation mittels eines IAM-Systems abgewickelt werden (bzgl. 4.1.2). Außerdem soll es möglich sein, dass ERP-Systeme neue Smartphones in dem System registrieren können (bzgl. 4.2). Bei einem Datenschutzvorfall oder einer anstehenden Sicherheitsanalyse soll das System Benachrichtigungen in Form von E-Mails versenden können (bzgl. 4.3). Zur Analyse muss das System mit den Smartphones interagieren können (bzgl. 4.2, 4.3 & 4.4.2). Zudem müssen die bei der Untersuchung extrahierten Artefakte sowohl durch lokale als auch externe Scanner untersucht werden können (bzgl. 4.4.2). Die daraus resultierende Kontextsicht wurde in der Abbildung 5.1 dargestellt.

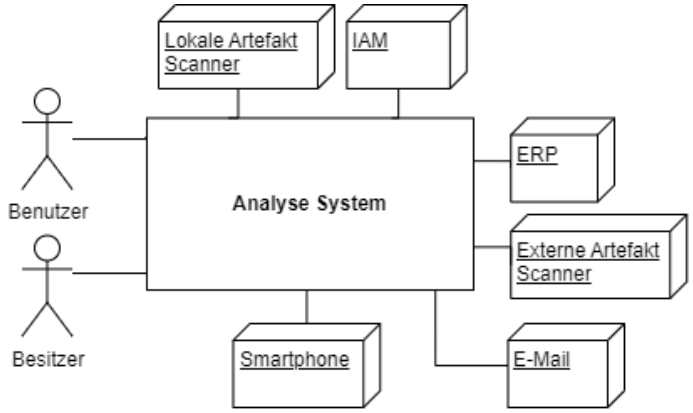

Abbildung 5.1: Kontextsicht

5.2 Baustein- und Komponentensicht

Nachdem in der Kontextsicht die beteiligten Systeme und ihre benötigten Schnittstellen als auch die Akteure erfasst wurden, wird im Folgenden eine Bausteinsicht angefertigt. Anschließend erfolgt eine Komponentensicht für die jeweiligen Bausteine. Dabei wird unterschieden zwischen Bausteinen, welche auf dem Forensiker und Sachbearbeiter (Benutzer) PC realisiert werden (FrontEnd) und Bausteinen, welche auf Server ausgelagert werden (Backend). Es entsteht dabei ein Client(FrontEnd)-Server(BackEnd)-Modell.

Bei der Unterteilung bzw. Separierung zwischen FrontEnd und BackEnd in die jeweiligen Bausteine und Komponenten wird dabei Wert auf den Datenschutz gelegt. Daher sollte auf die Datenhaltung und potenziell entstehenden Datensilos geachtet werden als auch auf die Datenzugriffe und mögliche Datenabfluss Möglichkeiten.

Von sog. Datensilos wird dabei gesprochen, wenn Daten potenziell separiert und redundant vorhanden sind und dadurch anderen Personen oder Systeme nicht auf diese zugreifen können. Ein Vergleich dafür wäre die Normalisierung bei Datenbanktabellen. Ziel dieser liegt dabei darauf, eine redundante Speicherung von Daten zu vermeiden. In Unternehmen ergibt es Sinn aus wirtschaftlicher Sicht Datensilos zu vermeiden, um einen effizienteren Arbeitsworkflow zu erschaffen, Dateninkonsistenzen zu vermeiden, als auch bessere Auswertungsmöglichkeiten für z. B. die Marktforschung zu schaffen [17].

Durch die Einführung der DSGVO wurde diesem Trend gesetzlich entgegengewirkt. Daten müssen seitdem zweckgebunden und systematisch isoliert gespeichert und verarbeitet werden.

Bezogen auf die Datenzugriffe und Abfluss Möglichkeiten wird bei der Bildung der Bausteine auf die Zugriffsrichtung bzw. Datenflussrichtung als auch den jeweils zu verarbeitenden oder zu haltenden Daten geachtet. Um dies korrekt bewerten zu können, müssen die anfallenden Daten als auch mögliche Bedrohungen betrachtet werden.

5.2.1 Bedrohungsszenarien

Um festzustellen, welche Daten wo und wie gespeichert oder verarbeitet werden, werden im folgenden Abschnitt unterschiedliche Kompromittierungsszenarien durch simuliert. Dies dient primär dazu festzustellen, welche Maßnahmen bezogen auf die Gestaltung des Entwurfs getroffen werden können, um die entstehenden Bausteine und Komponenten sinnvoll auf den Datenschutz bezogen als auch auf ihre Verantwortlichkeiten / Aufgabengebiete zu trennen.

Kompromittierung des FrontEnds

In diesem Szenario wird angenommen, dass es zu einer Kompromittierung eines Benutzer-PCs (von Forensikern oder Sachbearbeitern) kommt.

Ein mögliches Szenario wäre dabei die Kompromittierung oder das Mitschneiden, beispielsweise bei einem MitM-Angriff, der Kommunikation zwischen dem FrontEnd und dem BackEnd. Betroffenen sind daher die Kommunikationsdaten. Es sollte daher darauf geachtet werden, dass so wenig wie möglich Untersuchungsdaten (Dateien / Artefakte, potenzielle personenbezogene Daten) ausgetauscht werden. Die eigentliche Sicherheitsanalyse sollte daher auf dem Forensiker Geräten / PCs stattfinden und lediglich in Ausnahmen sollten Teile der Daten / Artefakte übertragen werden.

Ein weiteres Szenario ist dabei die Kompromittierung des eigentlichen PCs durch Schadsoftware. Es wird dabei von einer vollkommenen Kontrolle des Benutzer-PCs ausgegangen. In diesem Szenario fallen die gleichen Risiken wie in dem oben genannten Szenario an, da die Kommunikation auch kompromittiert werden kann. Hinzu kommt jedoch, dass auch alle lokalen Daten, darunter auch die Untersuchungsdaten, abgegriffen werden können. Es sollte daher darauf geachtet werden, dass alle anfallenden Daten fallspezifisch verschlüsselt werden und der Schlüssel dafür zugriffsgeschützt auf einem Server / einer

ausgelagerten Komponente liegt. Der fallspezifische Schlüssel $KFall$ (zufallsgeneriert) sollte nur auf dem BackEnd gespeichert werden und bei einer aktiven Durchführung einer Sicherheitsanalyse für den jeweiligen Fall abgefragt werden. Zudem sollte der Fall-Schlüssel mit dem Besitzer Schlüssel konkateniert werden, da dies die Entschlüsselung der Fall-Daten bei einer Kompromittierung der Server verhindert und der Forensiker nur den Fall mit der Einverständniserklärung wieder entschlüsseln / öffnen kann. Dies sorgt dafür, dass nur eine aktuell ausgeführte Untersuchung und dessen Daten abfließen können.

Kompromittierung des Android Smartphones

Dieses Szenario nimmt an, dass ein zu untersuchendes Smartphone mit bösartiger Software infiziert wurde. Da die Analyse auf einem nicht Android basierten Betriebssystem wie Windows, Linux oder MacOS stattfindet, welches nicht in der Lage ist, nativ Android Anwendungen auszuführen, geht ein niedriges Risiko von nativen Android Anwendungen aus. In der Theorie ist es möglich, dass Android Anwendungen mit viel Adaptionen auf Linux und auch umgekehrt lauffähig sind, da Android auf dem Linux Kernel basiert. Bei einer dynamischen Untersuchung sollte trotzdem sichergestellt werden, dass diese in einer isolierten bzw. virtualisierten Umgebung stattfindet, ohne Hostzugriffe (bzgl. 4.4.1 Sandbox Ansätze).

Bei anderen Artefakten wie z. B. bösartigen Dokumenten mit Makrounterstützung besteht trotzdem ein direktes Risiko, da diese auch auf dem untersuchenden System geöffnet werden und Schaden anrichten können. Auch hier sollte vor einer direkten Ausführung bzw. direkten Inspektion durch nicht forensische Programme wie z. B. Word auf dem Untersuchungssystem abgesehen werden. Sollte der Schadcode aber auf die Betriebssystemebene vorgedrungen sein und dadurch direkten Hardwarezugriff haben, besteht auch bei der Verbindung mit den untersuchendem System / PC eine Gefahr.

Das kompromittierte Gerät könnte dabei z. B. sog. Rubber Ducky Angriffe ausführen, indem es sich als HDI (Human Device Interface) ausgibt und Tasten- oder Mauseingaben simuliert. Ein solcher Angriff könnte auch über ein mitgeliefertes USB-Kabel ausgeführt werden, da es bereits jetzt frei verfügbare Varianten von unterschiedlichen Herstellern gibt, welche diese Angriffe durch die Kabel an sich ermöglichen.

Bei den aufgeführten Angriffen kann es zu einer Kompromittierung des Besitzer PCs / FrontEnds und daher auch zu den gleichen Folgen kommen. Es sollte daher verzichtet werden, mitgelieferte Kabel zu verwenden und generell neue USB Geräte auf dem PC der

Benutzer zu unterbinden. Mit dem eigentlichen Smartphone sollte nur über eine abstrakte Schnittstelle kommuniziert werden, welche ungewollte Nebeneffekte verhindert.

Kompromittierung des BackEnds

In diesem Szenario wird eine Kompromittierung eines oder mehrerer BackEnd Dienste / Server angenommen. Hierbei wird unterschieden zwischen den jeweiligen Tätigkeiten / Aufgaben, welche die jeweiligen Dienste übernehmen sollen.

Ein Aufgabengebiet bildet dabei die logistische Verwaltung der Smartphones und der eingehenden Kommunikation mit den ERP-Systemen (bzgl. 4.2). Sollte es bei diesem Teilsystem zu einer Kompromittierung z. B. mittels der angebotenen Schnittstelle kommen, so besteht die Möglichkeit, die logistischen Daten zu manipulieren. Es sollte daher aufgrund der eingehenden Kommunikation von externen Systemen keine sensiblen Daten verarbeiten oder persistieren, sondern lediglich die logistischen Smartphone Daten verarbeiten. Die Zuordnung und die dabei entstehenden Daten (personenbezogene Daten über den Besitzer) sollten daher ausgelagert werden. Bei einem Abfluss der organisatorischen Daten würde es zu keinem Datenschutzvorfall kommen.

Ein weiteres Aufgabengebiet bildet die forensische Untersuchung und Verfolgung von Artefakten. Diese erfolgt zum Großteil auf den Forensiker-PCs. Auf Aufforderung können jedoch externe Artefakte Scanner für die Untersuchung verwendet werden (bzgl. 4.4.2). Aufgrund der ausgehenden Kommunikation und der möglichen Verarbeitung von personenbezogenen Daten sollten keine identifizierenden Daten zusätzlich auf dem Teilsystem / Dienst verarbeitet oder persistiert werden (bzgl. 3.2.1 Pseudoanonymisierung).

Ein weiteres Szenario ist die Kompromittierung des IAM-Systems. Durch die Kompromittierung dieses Systems entsteht ein Sicherheitsrisiko für alle beteiligten Systeme. Es würde Angreifern ermöglichen, Forensiker und Sachbearbeiter von dem System auszuschließen, als sich auch gegenüber den Diensten zu authentifizieren und autorisieren und Daten extrahieren oder manipulieren. Ein möglicher Mechanismus, um sich gegen solch ein Vorgehen zu wehren, wäre die Nutzung von Maschinen-, Anwendung- oder Benutzerzertifikaten. Durch diesen zusätzlichen Identitätsnachweis, welcher unabhängig vom IAM-System ist, können die Dienste Angreifer, welche lediglich das IAM-System kompromittiert haben, ausschließen.

Die restlichen Aufgaben, wie der Verwaltung der Benutzer, Zuordnungen als auch der Fallverwaltung sollten dementsprechend in einen eigenen Dienst (Server) ausgelagert werden. Durch diese Aufteilung wird eine getrennte Datenspeicherung für die ausgelagerten

personenbezogenen Daten erschaffen. Eine Kompromittierung dieses Systems würde einen Datenschutzvorfall mit sich ziehen.

5.2.2 Entwurf

Der entstandene Entwurf, welcher die aufgeführten Punkte aus der Anforderungsanalyse, den Bedrohungsszenarien sowie den Datenschutz berücksichtigt, wird in der Bausteinsicht in Abbildung 5.2 dargestellt. Dabei wurde darauf geachtet, dass die Abhängigkeiten durch Entwurfsmuster reduziert werden und zyklische Abhängigkeiten vermieden werden, wodurch die Wartbarkeit als auch die Nachvollziehbarkeit der Datenflüsse gesteigert wird. Das Bindeglied für die Client-Server Kommunikation daher zwischen dem FrontEnd und dem BackEnd, bildet dabei der *Audit* Baustein.

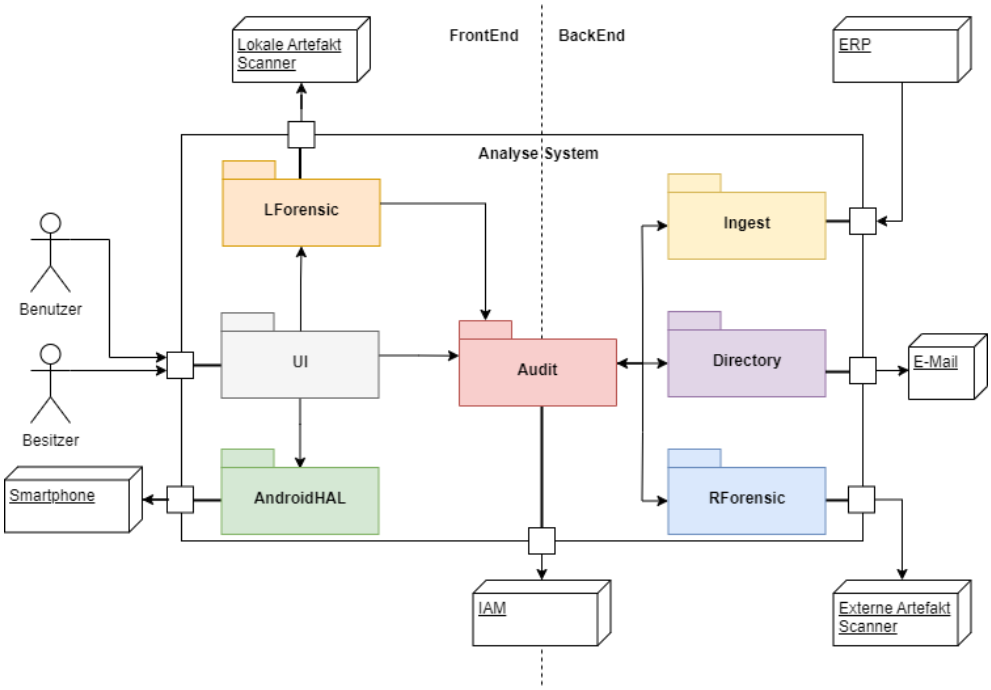

Abbildung 5.2: Bausteinsicht

In Abbildung 5.3 wurden dabei die Bausteine und externen Systeme abgebildet, mit welchem die Sachbearbeiter-Tätigkeiten abgedeckt werden. Die Bausteinsicht für Forensiker, welche ihre Tätigkeiten abdeckt als auch die dafür benötigten Bausteine, Scanner und externe Systeme wurde in Abbildung 5.4 dargestellt.

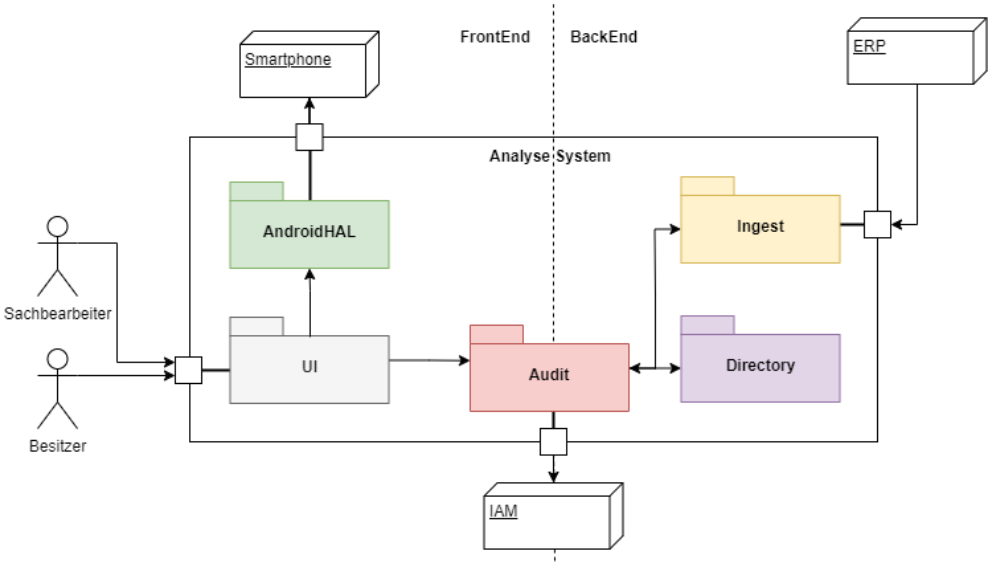

Abbildung 5.3: Bausteinsicht - Sachbearbeiter Sicht

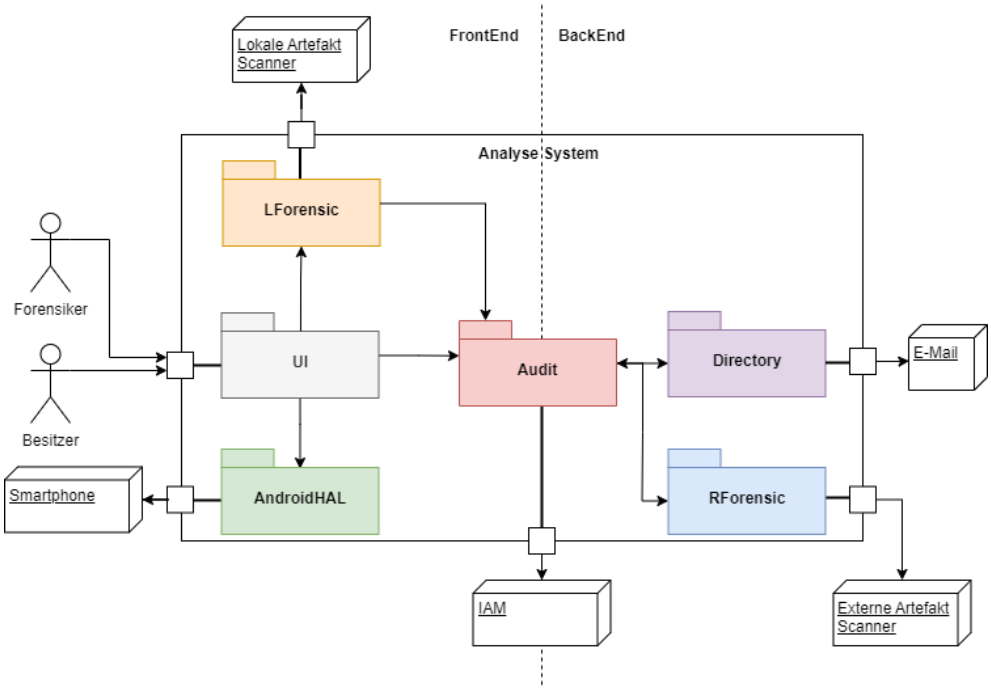

Abbildung 5.4: Bausteinsicht - Forensiker Sicht

5.2.3 Client - FrontEnd

Im Folgenden werden die FrontEnd Bausteine präzisiert und ihre Komponenten herausgearbeitet. Für ein besseres Verständnis befindet sich der ausgearbeitete Entwurf als Gesamtbild im Anhang (A.1).

AndroidHAL Baustein

Der AndroidHAL Baustein bildet dabei eine Abstraktionsebene (HAL - Hardware Abstraction Layer) für die Kommunikation mit den Android Smartphones und wird in Abbildung 5.5 dargestellt. Die Aufgabe des Bausteins besteht daher, im wesentlichen Schnittstellen für die benötigten Funktionen der Sicherheitsanalyse (z. B. der Datenextrahierung) anzubieten. Die eigentlichen Hardware-Interaktionen mit dem Smartphone und die damit verbundene Komplexität und dem möglichen Schadenspotenzial wird durch den Baustein isoliert und vom Rest des FrontEnd-Systems ferngehalten.

Diese Gestaltung bietet den weiteren Vorteil, dass die Schnittstellen des Bausteins für mögliche Tests durch Mock-Objekte realisiert werden können, wodurch kein physisches Smartphone für diese benötigt wird.

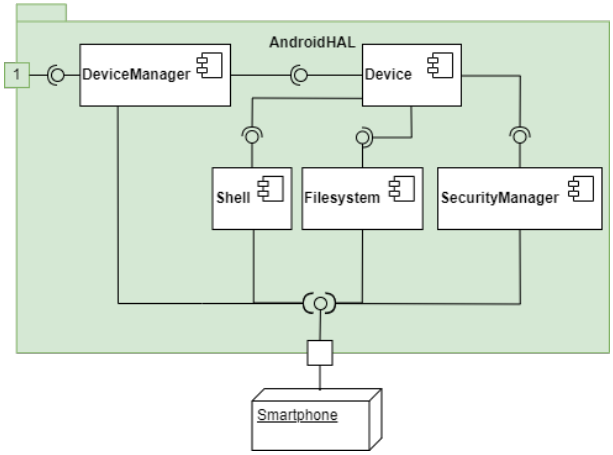

Abbildung 5.5: Komponentensicht AndroidHAL Baustein

Die *DeviceManager* Komponente stellt dabei eine zentrale Zugriffsmöglichkeit an alle angeschlossenen Smartphones, welche als *Device* Komponente repräsentiert werden, dar. Die für die Sicherheitsanalysen benötigten Funktionalitäten werden dabei durch die 3 Unterkomponenten von *Device* abgebildet.

Die *Shell* Komponente bietet dabei eine Schnittstelle für Kommandozeilenbefehle an (Anwendungsfall 20). Das aggregierte Dateisystem (System und Data Partition) des zu untersuchenden Smartphones wird durch die *FileSystem* Komponente zugreifbar (Iteration und Extrahierung) gemacht. Sicherheitsrelevante Anforderungen wie dem Sperren und Entsperren des Bootloaders und CR-Betriebssystems als auch dem Aufspielen von dem CR-Betriebssystem übernimmt dabei die *SecurityManager* Komponente (Anwendungsfall 4 und 10). Diese Komponenten nehmen die eigentliche Hardware- und Softwarekommunikation mit dem Smartphone und den jeweiligen Smartphone-Systemen (Bootloader und CR-Betriebssystem) vor.

LForensic Baustein

Der LForensic (Local Forensic) Baustein bildet dabei die forensischen Funktionen der Sicherheitsanalyse auf dem FrontEnd ab und wird in Abbildung 5.6 dargestellt.

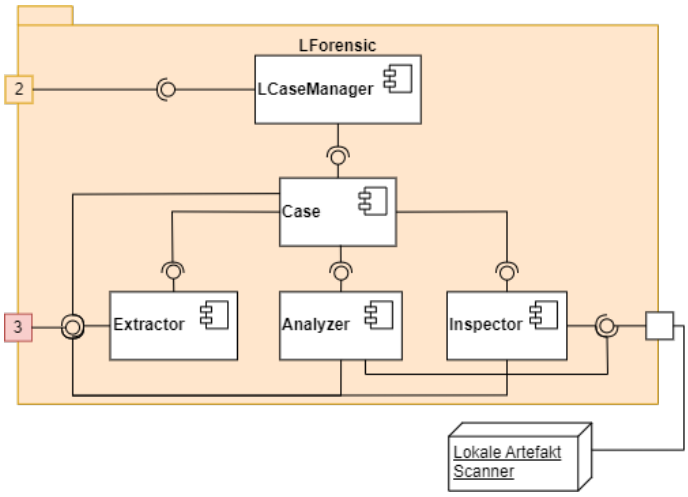

Abbildung 5.6: Komponentensicht LForensic Baustein

Die *LCaseManager* Komponente wird mittels Singleton Pattern realisiert und verwaltet dabei die Fälle, welche lokal auf dem Gerät liegen und referenziert den aktuell bearbeiteten Fall. Die *Case* Komponente stellt dabei den jeweiligen Fall dar und ist für dessen lokale Datenhaltung und Verschlüsselung zuständig. Die lokalen Daten belaufen sich dabei auf die extrahierten Artefakte und dessen Klassifizierungen. Der Fall-Schlüssel und die Daten, welche den Fall beschreiben (Metadaten wie ID, Besitzer, Smartphone-Kennung, Protokoll usw.), werden dabei nicht lokal gespeichert. Diese Daten werden ausgelagert

und müssen für die Fallbearbeitung über den Directory Baustein (5.2.5) hinzugezogen werden.

Die unterliegenden 3 Komponenten *Extractor*, *Analyzer* und *Inspector* stellen die 3 Phasen aus dem in 4.4.2 entwickelten Prozess dar. Diese werden dabei wie in einer Pipeline sequenziell über die *Case* Komponente durchlaufen / aufgerufen. Die Komponenten weisen dabei grundlegend nur Dienst-Schnittstellen auf, wodurch beliebig viele Implementierungen der jeweiligen Dienste umgesetzt werden können. Während der Laufzeit können dann für die jeweiligen Komponenten / Dienst-Art die Implementierungen mittels Dependency-Injection verwendet werden. Alle haben dabei über die Adapter Schnittstelle aus dem Audit Baustein (5.2.4) Zugriff auf das zu untersuchende Smartphone.

1. **Extractor (Extraction Phase):** Ziel dieser Dienste ist es, von dem Smartphone Artefakte automatisch zu extrahieren (Anwendungsfall 13). Für diese wird das Producer-Consumer Pattern verwendet. Der Extractor-Dienst (Producer) nutzt dabei die Smartphoneadapter Instanz und extrahiert daraufhin die durch den jeweiligen Dienst aufzuspürenden Artefakte. Die extrahierten Artefakte werden daraufhin mit dem Artefakte-Typ (zum Beispiel APK für Android Anwendungen) zusammen zurück an die *Case* Komponente (Consumer) gegeben. Die *Case* Komponente sammelt dabei die Artefakte und ordnet sie in ihrer jeweiligen Artefakte-Gruppe anhand des Artefakte-Typs zu. Die Verwaltung der extrahierten Artefakte und dessen Speicherung bleibt daher bei der *Case* Komponente.

2. **Analyzer (Analysis Phase):** Ein Analyzer-Dienst bedient sich auch dem Producer -Consumer Pattern. Dem Dienst (Producer) werden dabei von der *Case* Komponente die jeweils für sie untersuchbaren Artefakten-Gruppen übergeben. Die jeweiligen Artefakte werden daraufhin automatisch untersucht (Anwendungsfall 14) und klassifiziert. Die eigentlichen Untersuchungen der Artefakte müssen dabei nicht explizit durch den Dienst erfolgen, sondern können auch an lokale Scanner delegiert werden, wodurch der Dienst lediglich eine delegierende Rolle einnimmt. Die dabei vergebene Klassifizierung stellt dabei eine Wahrscheinlichkeit dar, wie sicher das Artefakt bösartig / unsicher ist. Der Wertebereich liegt dabei von 0 (harmlos / sicher) bis 100 (Garantiert bösartiges / unsicheres Artefakt). Zusätzlich zu der Klassifizierung wird für jede Untersuchung / je Artefakt ein Bericht (Report) abgegeben, wie die Klassifizierung zustande gekommen ist. Die *Case* Komponente (Consumer) verbindet daraufhin intern die Resultate (Klassifikation und Report) an das jeweilige

Artefakt. Dabei sind mehrere Resultate durch unterschiedliche Analyzer Dienste für das gleiche Artefakt möglich.

Die jeweiligen Analyzer-Dienste müssen dabei nicht bei jedem Aufruf / Artefakt die Analyse durchführen. Sie können die Artefakte durch den RForensic Baustein verfolgen (Anwendungsfall 15) (5.2.5) und ein vorheriges Artefakt Resultat (Klassifizierung und Report) wiederverwenden. Eine Wiederverwendung sollte dabei jedoch nur bei dem gleichen Dienst stattfinden, welcher dieses ausgestellt hat und auch nur bei der gleichen Version des Dienstes. Hintergrund hierfür ist, dass der Dienst neuen / bessere Erkennungsmechanismen verwenden könnte, wodurch die vorherigen Resultate nicht mehr gültig sein könnten.

3. **Inspector (Forensic Phase):** Die Inspector-Dienste können von den Forensikern auf die Artefakte aufgerufen werden. Die manuellen Aufrufe der Inspector-Dienste werden dabei durch das Command Pattern realisiert. Durch lokale Implementierungen wird der Anwendungsfalle 17 abdeckt. Der Anwendungsfall 18 (externe Überprüfung) lässt sich durch Aufrufe von externen Scannern über den RForensic Baustein (delegiert über den Audit Baustein / Smartphoneadapter) abdecken.

UI Baustein

Der UI (User Interface) Baustein dient als zentraler Punkt für Benutzerinteraktionen mit dem System. Er ermöglicht dabei sowohl den Zugriff auf die Client (FrontEnd) Bausteine als auch den Server seitigen (BackEnd) Bausteinen. Da das UI in der Softwareentwicklung sich an den Anwendungsfällen orientiert, wird eine eigene Bausteinsicht für die Anwendungsfälle der Forensiker und eine Bausteinsicht für die Sacharbeiter erstellt.

Beide Sichten enthalten dabei eine GUI (Graphical User Interface) Komponente, welche beliebig nach Präferenzen und nach UX (User Experience) Design Tests / Workshops entworfen werden kann. Für die eigentliche Umsetzung und Funkitonalität des Systems ist dies nicht von Relevanz. Der in beiden Bausteinsichten enthaltene GUI Komponente realisiert den Anwendungsfall 1 (Anmeldung des Benutzers) über die $SessionManager$ Komponente aus dem Audit Baustein.

Die eigentliche Logik wird dabei von der GUI Komponente entkoppelt über ein Fassaden Pattern. Die GUI Komponente ruft daher lediglich die jeweiligen Anwendungsfälle über die jeweiligen Komponenten auf und enthält selbst keine Logik außer der für die Darstellung benötigten.

UI - Sachbearbeiter

Die Anwendungsfälle, die durch Sachbearbeiter ausgeführt werden, belaufen sich dabei auf die in dem Inventar aufgeführten Anforderungen aus dem Unterkapitel 4.2.

Die *DeviceIngest* Komponente dient dabei der Verwaltung von registrierten Geräten (Anwendungsfall 2). Hingegen dient die *DeviceManagement* Komponente der Zuordnung von Geräten und dessen Verwaltung (Anwendungsfall 4 und 5).

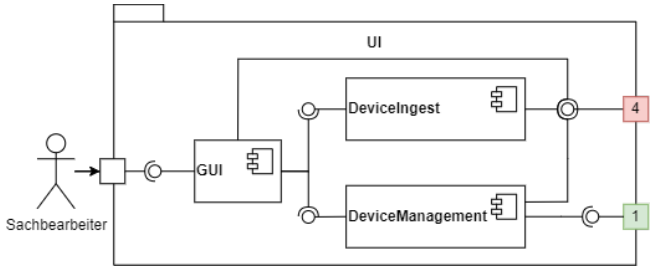

Abbildung 5.7: Komponentensicht UI Baustein - Sachbearbeiter

UI - Forensiker

Tätigkeiten / Anwendungsfälle, welche durch die Forensiker ausgeführt werden, belaufen sich auf die Anwendungsfälle aus dem Unterkapitel der Fallverwaltung (4.3) und dem der Forensik (4.4.2).

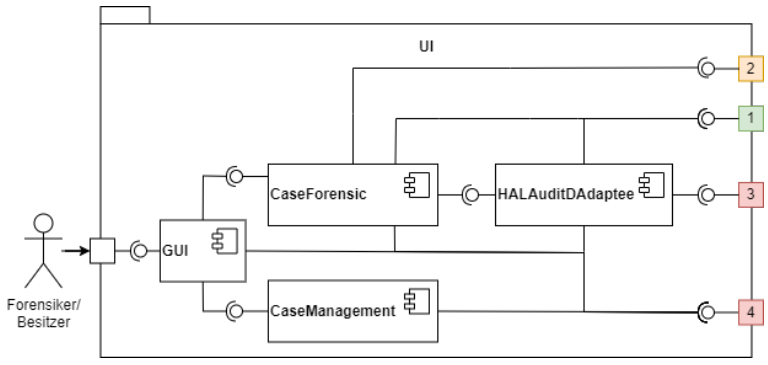

Abbildung 5.8: Komponentensicht UI Baustein - Forensiker

Die *CaseManagement* Komponente dient dabei für die Fallverwaltung (Anwendungsfälle 6, 8, 9 & 22). Dabei wird eine Verbindung zwischen der clientseitigen Fallverwaltung

(LForensic Baustein) und der serverseitigen Fallverwaltung (Directory Baustein) hergestellt.

Die eigentlichen Anforderungen für die Bearbeitung der Fälle (Anwendungsfall 10 - 12 und 16 - 20) wird durch die *CaseForensic* Komponente übernommen. Zudem werden die manuellen Untersuchungsanwendungsfälle 17 und 18 durch die *Inspector* Dienste über den LForensic Baustein angesteuert. Diese müssen dabei dynamisch angesteuert werden / in der GUI abgebildet werden. Dies hat den Grund, dass die *Inspector* Dienste beliebig erweitert werden können und erst zur Laufzeit bekannt ist, welche Möglichkeiten / Dienste zur Verfügung stehen (Dependency Injection).

Die *HALAuditDAdaptee* Komponente dient dabei als Adapter Implementierung (Adaptee) der *AuditDeviceAdapter* Komponente, welche von dem LForensic Baustein und dessen Komponenten verwendet wird. Die Komponente repräsentiert dabei im Kern einen Art Proxy und delegiert die Aufrufe der forensischen Untersuchung aus dem LForensic Baustein an den AndroidHAL Baustein und dessen Komponenten. Sinn dieser Komponente ist es, dass die Smartphone und RForensic Zugriffe von dem LForensik Baustein automatisch protokolliert werden (Abbildung 5.9). Die Protokolleinträge werden dabei an den Directory Baustein weitergeleitet und mit dem Smartphone Besitzer Schlüssel verschlüsselt. Dies stellt sicher, dass ein Forensiker das Protokoll nicht modifizieren kann oder es abhandenkommen kommt. Zudem steht es noch nach der Fall-Schließung zur Verfügung (Anwendungsfall 12). Die Kommandozeilenbefehle (Anwendungsfall 20) der *CaseForensic* Komponente werden dabei über die *HALAuditDAdaptee* Komponente realisiert. Der direkte Zugriff von der *CaseForensic* Komponente auf die Smartphones über den AndroidHAL Baustein wird lediglich für das Entsichern des Smartphones (die *SecurityManager* Komponente) verwendet.

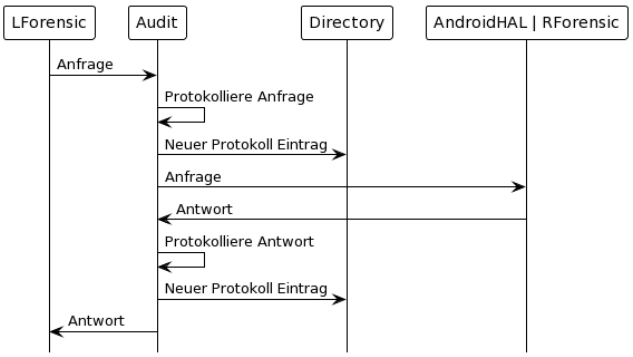

Abbildung 5.9: Zugriff / Protokollierung Konzept

5.2.4 Client-Server Kommunikation

Die Verbindung des Clients (FrontEnds) mit dem Server (BackEnd) wird durch den Audit Baustein realisiert.

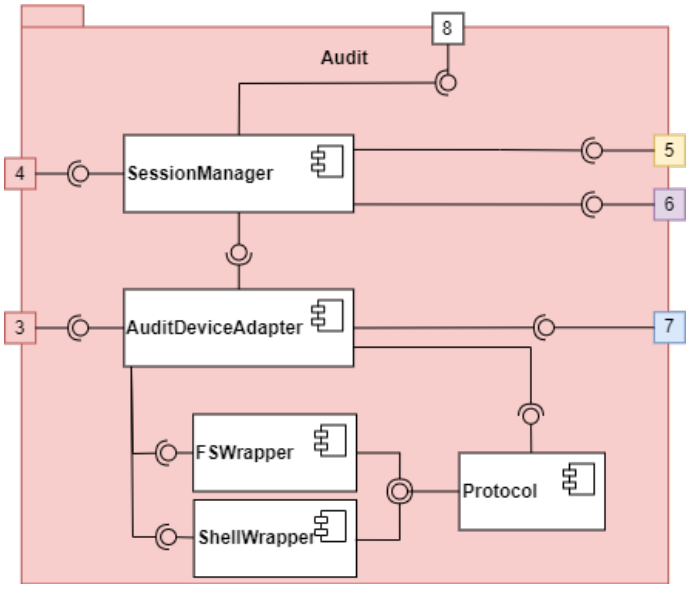

Abbildung 5.10: Komponentensicht Audit Baustein

Die *SessionManager* Komponente, welche als Singleton realisiert wird, übernimmt dabei die Sitzungsverwaltung des Benutzers mit dem IAM-System (Anwendungsfall 1). Alle Anfragen an das BackEnd nutzen diese Komponente, um sich gegenüber dem Backend zu authentifizieren und zu autorisieren. Zudem können über diese zusätzliche Wissensnachweise mitgesendet werden, um z. B. gegen die in der Bedrohungsanalyse erwähnte IAM-System Kompromittierung vorzugehen.

Die Ausnahme liegt dabei bei Anfragen an den RForensic Baustein. Für diese ist die *AutidDeviceAdapter* Komponente zuständig. Diese agiert als Bindeglied zwischen den AndroidHAL und RForensic Bausteinen und dem LForensic Baustein (bzgl. Abbildung 5.9). Die LForensic Komponenten (mit Ausnahme der *LCaseManager* Komponente) können zusätzlich Protokoll / Log Einträge über die Komponente hinzufügen, diese werden daraufhin an die *Protocol* Komponente weitergeleitet. Die *Protocol* Komponente dient dabei als Sammelpunkt und in diesem können mittels Adapter Pattern Ausgaben an beliebige Punkte wie z. B. der GUI oder dem Directory Baustein weitergeleitet werden. Die *SecurityManager* Komponente aus dem AndroidHAL Baustein wurde bewusst

nicht mit in die *AuditDeviceAdapter* Komponente mit integriert, da die Zugriffe auf diese aus Sicherheitsgründen nicht protokolliert werden sollten. Die *FSWrapper* Komponente sorgt dabei zusätzlich dafür, dass das Benutzerverzeichnis nicht untersucht / Daten extrahiert werden können. Hingegen dient die *ShellWrapper* Komponente als reine Kapselung der Kommandozeileneingaben und dessen Protokollierung.

5.2.5 Server - BackEnd

Im Folgenden werden die BackEnd Bausteine präzisiert und dessen Komponenten erläutert. Die Bausteine wurden bezüglich ihrer zu haltenden / verarbeitenden Daten gebildet. Jeder dieser Bausteine sollte als eigener Server umgesetzt werden. Die Schnittstelle, welche mit 8 annotiert wurde, bildet dabei die Schnittstelle mit dem IAM-System ab. Eine Kommunikation mit diesen Bausteinen / Komponenten ist nur für authentifizierte als auch autorisierte Benutzer möglich (Anforderung 1).

Ingest Baustein

Der Ingest Baustein übernimmt die Verwaltung von neuen / nicht registrierten Smartphones (Anwendungsfall 2) und dessen Meldungen von externen ERP-Systemen (Anwendungsfall 3). Der Zugriff auf diesen Baustein und dessen Komponenten erfolgt nur durch Sachbearbeiter und sollte daher mittels RBAC mit der Sachbearbeiter Rolle geschützt werden.

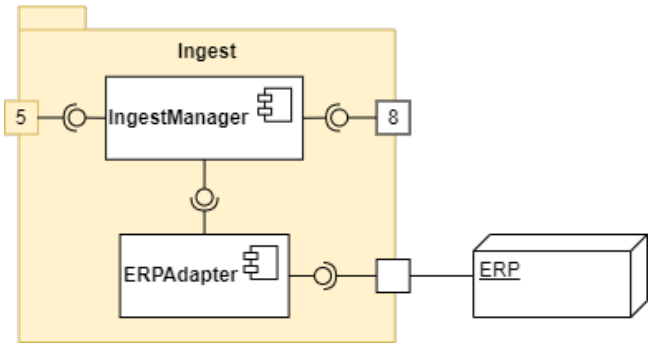

Abbildung 5.11: Komponentensicht Ingest Baustein

Die Annahme der ERP-System Anfragen als auch die anschließende Speicherung und Weitergabe für die Registrierung der Smartphones durch die Sachbearbeiter übernimmt

die *IngestManager* Komponente. Die einzelnen ERP-System-Schnittstellen Implementierungen werden dabei durch das Adapter Pattern (*ERPAdapter*) abstrahiert. Dies ermöglicht es, eine einheitliche Smartphoneannahme Schnittstelle für die einzelnen ERP-Systeme zu erschaffen, während die einzelnen Details und Konventionen dieser durch die Adapter abstrahiert werden.

Directory Baustein

Der Directory Baustein übernimmt dabei eine Verzeichnisfunktion. Der Zugriff auf diesen Baustein und dessen Komponenten erfolgt sowohl durch Forensiker als auch Sachbearbeiter.

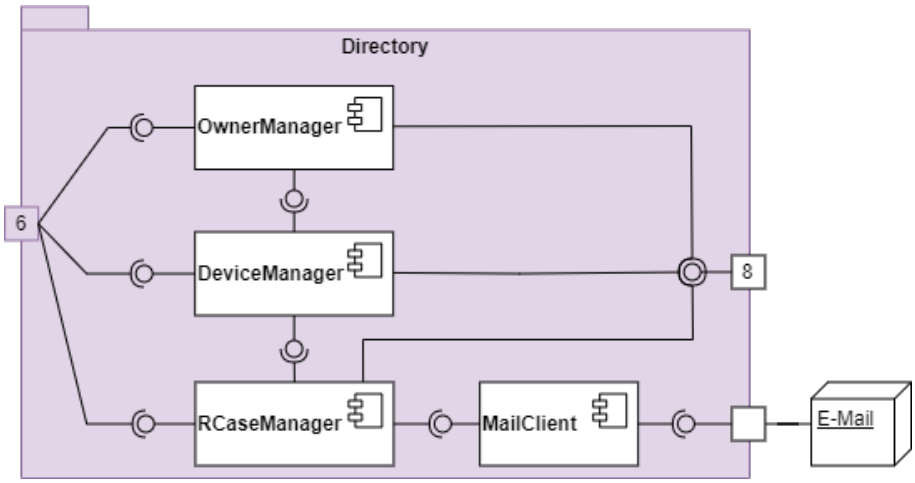

Abbildung 5.12: Komponentensicht Directory Baustein

Die Sachbearbeiter greifen dabei auf die *OwnerManager* und *DeviceManager* Komponenten zu und diese sollten daher mit der RBAC Sachbearbeiter Rolle geschützt werden. Die Forensiker greifen dabei auf die *DeviceManager* und *RCaseManager* Komponenten zu und diese sollte durch die RBAC Forensiker Rolle geschützt werden. Auf die *DeviceManager* Komponente greifen daher 2 Rollen zu.

Die *OwnerManager* Komponente übernimmt dabei die Verwaltung der Besitzer und dessen Daten (Kontaktdaten). Die *DeviceManager* Komponente verwaltet die Smartphones. Bei reiner Betrachtung dieser Komponente sind nur registrierte Smartphones zu sehen, die eigentliche Zuordnung von Besitzern wird durch Verweise auf die *OwnerManager* Komponente erschaffen (Anwendungsfall 4 und 5).

Die *RCaseManager* Komponente übernimmt die BackEnd Fallverwaltung (Anwendungsfälle 6 - 9). In ihr werden die Fälle verwaltet und orchestriert als auch die Falldaten gespeichert. Zu den Falldaten gehört das Fall-Passwort, mit dem der jeweilige Fall auf dem Forensiker Client verschlüsselt wird, das Protokoll als auch weiterer Metadaten. Der E-Mail Versand für die Benachrichtigung von Datenschutzbeauftragten bei einem Datenschutzvorfall (Anwendungsfall 11) und der Benachrichtigung des Besitzers für eine Untersuchung (Anwendungsfall 6 und 7) wird durch die *MailClient* Komponente realisiert. Diese sendet die E-Mails über einen Mail-Server ab.

Die *OwnerManager*, *DeviceManager* und *RCaseManager* Komponenten sollten in einem produktiven Einsatz jeweils auf einen eigenen Server verteilt werden und mit Referenzen auf die jeweiligen Daten arbeiten. Durch die Verwendung von Referenzen zwischen den Komponenten wird eine Pseudoanonymisierung erreicht (bzgl. 3.2.1).

RForensic Baustein

Der RForensic (Remote Forensic) Baustein übernimmt dabei die Ansteuerung von externen Scannern (Anwendungsfall 17) und der Artefakte Verfolgung (Anwendungsfall 15). Der Zugriff auf diesen Baustein und dessen Komponenten erfolgt nur durch Forensiker und sollte daher mittels RBAC mit der Forensiker Rolle geschützt werden.

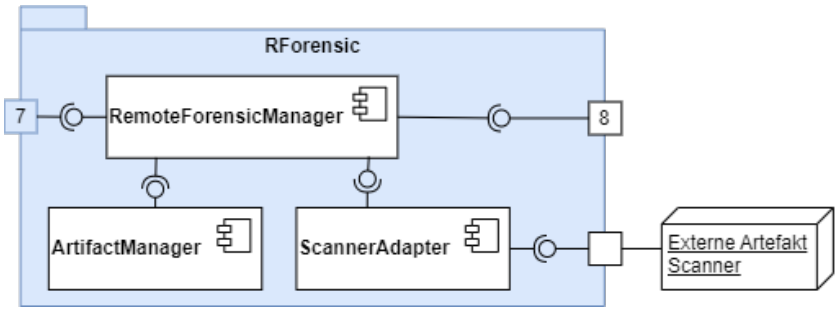

Abbildung 5.13: Komponentensicht RForensic Baustein

Die Orchestrierung der jeweiligen externen Scanner und der Anfragen an diese übernimmt dabei die *RemoteForensicManager* Komponente. Die einzelnen Scanner Schnittstellen Implementierungen werden dabei durch das Adapter Pattern (*ScannerAdapter*) abstrahiert. Dies ermöglicht es, eine einheitliche Schnittstelle für die einzelnen Scanner zu erschaffen, während die einzelnen Details und Konventionen von Ihnen Schnittstellen unabhängig sind. Da es bei der Scanner-Ansteuerung zur Übertragung von personenbezogenen Daten und dessen Verarbeitung kommen kann, benötigt es mit jedem Anbieter

eines externen Scanners einen AVV. Unter der Annahme, dass die Server im eigenen Rechenzentrum liegen, bedeutet dies, dass ein zentraler Ort für die Verarbeitung von personenbezogenen Daten durch Dritte entsteht. Andernfalls müsste auch mit dem Rechenzentrumsbetreiber ein AVV abgeschlossen werden (bzgl. 3.2.3).

Die benötigte Artefakte Verfolgung wird dabei durch die $ArtifactManager$ Komponente realisiert. Die $RemoteForensicsManager$ Komponente delegiert die Anfragen dabei an die $ArtifactManager$ Komponente weiter, wodurch eine zentrale Schnittstelle mit dem Baustein erschaffen wird. Ziel dabei ist es, eine Art Artefakte-Katalog bzw. Artefakte Cache anzufertigen, wodurch Artefakte unabhängig und untersuchungsübergreifend verfolgt und Klassifizierungen wieder verwendet werden können. Da ein solcher Artefakte-Katalog aber zu Profiling führen könnte (bzgl. 3.4) sollte ein "Last Seen" Ansatz verwendet werden.

Dies ermöglicht es, einen globalen anonymen Artefakte-Katalog anzufertigen, in dem ein anonymes identifizierendes Artefakte Merkmal wie z. B. einer Prüfsumme mit einem Datum gespeichert wird. Das eigentliche Artefakt sollte nicht gespeichert werden, da es personenbezogene Daten enthalten könnte oder direkt darstellen könnte. Es ermöglicht trotzdem, die Präsenz eines Artefaktes auf irgendeinem der verwalteten Smartphones nachzuweisen. Dies funktioniert durch den Fakt, dass, wenn das "Last Seen" Datum von einem Artefakt älter ist als das des am längsten nicht untersuchten Smartphones + der periodischen Zeit, so kann davon ausgegangen werden, das dieses nicht mehr vorhanden ist. Ist dies nicht der Fall, sprich das Artefakt befindet sich noch potenziell auf einem Smartphone, so kann bei einer bösartigen Klassifizierung durch z. B. eine Meldung vom BSI überprüft werden, ob die eigenen Smartphones betroffen sein könnten.

5.3 Zusammenfassung

Die in diesem Kapitel entwickelte Architektur ermöglicht es Sicherheitsanalysen für Android Smartphones sicher und datenschutzkonform durchzuführen. Die Architektur beschreibt dabei ein komplettes System, welches den kompletten Smartphonezyklus von der Anschaffung, Zuordnung und Analyse abdeckt. Es wurde darauf geachtet, dass durch Entwurfsmuster die Möglichkeiten für die Sicherheitsanalyse modular gestaltet wurden. Dabei können die einzelnen Untersuchungs-Dienste durch beliebig viele Implementierungen / Scanner erweitert werden.

Die externe Datenverarbeitung wurde dabei zentralisiert, sodass die Verarbeitung durch Dritte über eine Stelle (dem RForensic Baustein) abgewickelt wird, wodurch bei einem Audit (einer Prüfung) diesbezüglich nur auf diesen Baustein geachtet werden muss.

Durch die komplette Abdeckung der Smartphone-Lebenszyklen und der für das Vorhaben anfallenden Daten entsteht ein gewolltes Datensilo. Das System ist daher von keinen weiteren externen Datenquellen abhängig. Das hier angewendete Schlüsselkonzept ($KBesitzer$, $KSystem$ & $KFall$) sorgt dabei für die sichere Datenhaltung auf den jeweiligen Systemen (Smartphone, Forensiker PC & Directory Baustein / Server).

Durch die übersichtliche Datenhaltung und Verarbeitung wird ein nachweisbares Löschkonzept durchführbar. Die personenbezogenen Daten belaufen sich dabei auf die Falldaten, welche auf dem Forensiker PCs vorliegen (LForensic Baustein) als auch den Daten in dem Directory Baustein (Server) und den darin enthaltenen Komponenten / Persistenzen. Die Falldaten werden dabei nach der Fallschließung gelöscht und in den Directory Baustein Fall-Metaden vermerkt. Ein weiteres Personenrecht, welches durch die DSGVO für enorme Aufwände gesorgt hat, ist das Recht auf Auskunft. Dieses lässt sich analog durch die bereits erwähnten Aspekten für das Löschkonzept und den geführten Fall-Protokollen abwickeln.

Auch bezüglich der Datenverarbeitung und der Datenzugriffe wurde darauf geachtet, dass die jeweiligen Benutzer (Forensiker und Sachbearbeiter) nur auf die für sie relevanten Daten Zugriff haben. Die Umsetzung der Autorisation ist daher durch wenige Rollen mittels RBAC möglich und durch den Einsatz eines IAM-Systems nachweisbar im Falle einer Prüfung / eines Audits.

6 Implementierung

Im folgenden Kapitel wird auf die Implementierung eines Proof of Concept (POC) und der Umsetzung des Entwurfs eingegangen. Für das POC wird die eigentliche Hauptarbeit des Systems, die des Forensikers, implementiert (Abbildung 5.4). Die dabei zu verwendenden Programmier- und Scriptsprachen sowie der Technologien wurden dabei gemäß der nicht funktionalen Anforderungen (4.1.2) ausgewählt. Zudem wird auf die Entwicklung eines eigenen CR-Betriebssystems verzichtet. Es wird hierfür das CR TWRP (Team Win Recovery Project [22]) verwendet.

Als IAM-System wird Keycloak verwendet. Dabei wird ein Realm ("androidforensic") für die Anwendungsdomäne (die Thematik dieser Bachelorarbeit) inklusive einer Rolle für die Forensiker angelegt ("android-analyst"). Für die Authentifikation und Autorisation zwischen dem Client und den Servern wurden 2 Keycloak-Clients angelegt ("androidforensic-directory-client" und "androidforensic-rforensic-client"). Ein Keycloak-Client bietet dabei die Möglichkeit, Tokens zu generieren, mittels einer Anmeldung oder einen Token zu validieren / dessen Attribute wie z. B. Rollen abzurufen. Für die Kommunikation mit den Keycloak-Clients wird die Angabe eines Client-Secrets (Wissensnachweis) benötigt. Beide setzen dabei standardmäßig die Scopes "roles" und "email", wodurch die Server die Rollen der Benutzer einsehen und dementsprechend validieren können als auch die E-Mail für benötigte Funktionen verwenden können. Die standardmäßige Scope Setzung bedeutet, dass die Clients keine zusätzlichen Scopes bei der Anmeldung angegeben müssen. Als Testnutzer wurde dabei der Account mit der E-Mail "test@dev.dev" mit dem Passwort "rootroot" und der Forensiker Rolle "android-analyst" angelegt.

6.1 FrontEnd Entwicklung

Das FrontEnd wird dabei als eine Client Anwendung realisiert, welche in Python programmiert wird. Der Modul Mechanismus von Python erlaubt es dabei, die Bausteine

aus dem Entwurf (UI, LForensic, AndroidHAL und Audit) in die Projektstruktur 1:1 zu übertragen.

6.1.1 Audit

Der Audit Baustein wurde nach dem Entwurf implementiert und zusätzlich wurde für die Schnittstellen an die Directory und RForensic Bausteine (Server) API Hilfsklassen, welche die Ansteuerung übernehmen, implementiert. Die SessionManager Komponente und die in ihr enthaltene gleichnamige Klasse verwaltet dabei die Kommunikation und dementsprechend auch die Anmeldung / Token Akquirierung beim Keycloak IAM-Server, für die beiden im BackEnd realisierten Server. Hinzu kommt, dass eine asynchron laufende Methode (in einem eigenen Thread) in einem Takt von 60 Sekunden die beiden Tokens (Dicrectory und RForensic) aktualisiert. Für die Synchronisation des Zugriffs auf die Tokens wurde eine Kombination aus einem Mutex für das Setzen und Lesen und ein Signalisierungsmechanismus für die Beendigung des Threads verwendet.

Für das Protokoll wurde eine Implementierung der repräsentativen Klasse *Protocol* angefertigt, welche die eigentlichen Protokoll / Log Nachrichten annimmt. Zusätzlich wurde noch ein Adapter Pattern implementiert, mit welchem beliebige Ausgaben bei erzeugten Protokoll Einträgen erzeugt werden können. Für die Ausgabe in die Konsole (stdout) wurde eine Adapter Implementierung in der Komponente angelegt.

6.1.2 AndroidHAL

Der AndroidHAL Baustein wurde zuerst grundlegend über Interfaces abgebildet. Die *DeviceManager* Komponente dient dabei als Zugriffsschnittstelle für die anderen Komponenten / Interfaces und wurde mittels des AbstractFactory Patterns implementiert. Es wurde sowohl eine Variante für die Kommunikation mit physisch angeschlossenen Smartphones implementiert, als auch eine Mock-Variante, welche ein Gerät simuliert und das Speicherabbild von einem auf dem Client liegenden Verzeichnis abgreift. Für die Kommunikation mit physischen angeschlossenen Smartphones werden die von Google verfügbaren Plattform-Tools verwendet, welche über CLI (Command Line Interfaces) die Kommunikation mit den angeschlossenen Smartphones ermöglichen.

6.1.3 LForensic

Von dem LForensic Baustein wurde die *LCaseManager* Komponente lediglich als delegierende Singleton Klasse implementiert, welche den aktuell bearbeiteten Fall referenziert. Die *Case* Komponente wurde dabei durch die *Case* Klasse umgesetzt, welche Aufrufschnittstellen für die Implementierungen der Extractor, Analyzer und Inspector Dienste mittels Dependency Injection verfügbar macht. Die entfernten (remote) Scanner des RForensic Servers werden dabei über die gleiche Schnittstelle angeboten (*Case*), da diese über den *AuditDeviceAdapter* und der dadurch verfügbaren RForensic-API enumerierbar und ansprechbar sind. Die lokalen Fall-Daten, sprich die Artefakte als auch dessen Klassifizierungen werden dabei durch die *CaseFileManager* Klasse abgedeckt, welche als interne Klasse von der *Case* Klasse verwendet wird.

Im Folgenden wird auf die Implementierungen der jeweiligen Phasen-Dienste eingegangen:

Extractor

Um Anwendungen (APKs) von den Smartphones zu extrahieren, wurde der APKExtractorService implementiert, welcher das gesamte Smartphone nach APK Dateien durchsucht und extrahiert. Als weiterer Extractor Dienst wurde CertExtractorService implementiert. Dieser extrahiert alle CA Zertifikate aus dem /system/etc/security Verzeichnis. Die extrahierten Zertifikate werden in der Artefakte-Gruppe *cert* gespeichert und die Anwendungen in der *apk* Artefakte-Gruppe.

Analyzer

Um die extrahierten Zertifikate zu analysieren, wurde der CerStoreAnalyzerService implementiert. Dieser prüft zuerst, ob das Zertifikat mit einem Zertifikat aus dem lokalen Zertifikateverzeichnis übereinstimmt. Das lokale Zertifikateverzeichnis ist dabei eine manuell angefertigte Kopie der aktuellen, von Google veröffentlichten CA Zertifikate (bzgl. 2.2.1). Bei Bedarf können zusätzlich eigene Zertifikate, welche auf den verwalteten Smartphones mit installiert werden, dem lokalen Zertifikateverzeichnis hinzugefügt werden, wodurch diese automatisch als bekannte / gutartige Zertifikate klassifiziert werden. Wurde keine Übereinstimmung mit einem bekannten Zertifikat gefunden, werden die Zeitdaten des Zertifikates betrachtet. Ist ein Zertifikat abgelaufen, so geht von diesem generell keine Gefahr mehr aus, da es nicht mehr verwendet werden kann. Trotzdem wird ihnen eine

Klassifizierung von 50 gegeben, da diese durch Zeit Angriffe potenziell wieder vom Smartphone als gültig angesehen werden könnten. Bei Zertifikaten, dessen Gültigkeit erst in der Zukunft liegt, wird eine höhere Klassifizierung von 80 gegeben, da diese von Angreifern hinterlegt und potenziell in der Zukunft verwendet werden können. Tritt keiner dieser Fälle ein, so ist das Zertifikat gültig und unbekannt. Ihm wird daher eine Klassifizierung von 90 zugewiesen und sollte daher untersucht werden.

Für die zu analysierenden Anwendungen (APKs) wurde der MobSFAPKAnalyzerService implementiert. Dieser analysiert die APKs nicht selber, sondern delegiert die Analyse an den lokal installierten MobSF Dienst (Mobile Security Framework) [16]. Installiert werden kann der Dienst über ein Docker Image und ist daraufhin über eine REST Schnittstelle ansprechbar. Diese verwendet einen API-Key als Authentifikation. Dieser ist in dem hier verwendeten Szenario trivial, da der Dienst zusammen auf dem Forensiker Gerät nur lokal betrieben wird und von anderen nicht ansprechbar ist. Das Artefakt / die APK werden dafür auf dem Dienst (MobSF) hochgeladen, daraufhin wird die APK automatisch untersucht und einen "Reportünd eine Klassifikation generiert, welche anschließend heruntergeladen und dem Artefakt zugeordnet wird. Der MobSF Dienst verwendet dabei einen umgekehrten Score (0 bösartig - 100 gutartig), weshalb dieser adaptiert (umgekehrt) wird auf den hier verwendeten Wertebereich (0 gutartig - 100 bösartig). Anschließend wird das Artefakt und die bei der Analyse angefallenen Daten wieder vom Dienst gelöscht. Sollte es bei dem Ablauf zu einem Fehler kommen, z. B. wenn der Dienst die Anwendung nicht analysieren / entpacken kann, so wird das Artefakt mit der maximalen Klassifizierung ausgestattet, um eine Ursachenforschung zu provozieren.

Beide Analyzer Dienste verwenden dabei die Artefakte Verfolgung und können daher Klassifizierungen und Resultate wiederverwenden. Dies ermöglicht zudem die automatische Wiederverwendung der manuellen Klassifizierungen durch Forensiker. Auf Cache Poisoning wird in diesem POC nicht eingegangen, sollte aber in einer für den produktiven Betrieb gestalteten Umsetzung beachtet werden. Angreifer könnten dabei bösartige Artefakte über den Cache als gutartig klassifizieren, wodurch Forensiker diesen Artefakten weniger Aufmerksamkeit schenken würden.

Inspector

Als Inspector Dienste, welche über das Command Pattern realisiert werden, wurde eine lokale laufende als auch eine remote laufende (BackEnd / Server) Nutzungsmöglichkeit realisiert. Beide beziehen sich dabei auf die APK Artefakte. Für die Zertifikate aus dem CertExtractorService wurden keine Inspector Dienste implementiert.

Der lokale Inspector Dienst ("MobSFVMInspectorService") leitet dabei das Artefakt an eine MobSF Instanz. Diese wird vorher so konfiguriert, dass der Inspector Dienst den Zugriff über die MobSF Instanz auf eine lokal aufgesetzte Android VM Instanz ermöglicht. Die Anwendung (APK), welche genauer untersucht werden soll, wird dafür über MobSF automatisch auf die VM installiert. Daraufhin kann die APK (Anwendung) interaktiv untersucht werden und mittels Frida Script beliebig manuell oder automatisiert über Skriptvorlagen interaktiv analysiert werden. Der Inspector Dienst richtet selbst nur die Umgebung ein und liefert daraufhin eine URL zurück, welche im Browser die dynamische Analyseumgebung über MobSF öffnet.

Der entfernte Inspector Dienst ("VirustotalScanner") wird über die *Case* Schnittstelle aufgerufen und leitet dabei das Artefakt an die namensgebende RForensic Adapter Implementierung weiter. Dieser ruft daraufhin die Virustotal API für eine Analyse auf und liefert das Ergebnis zurück. Eine Implementierung im FrontEnd wird nicht benötigt, da das FrontEnd sich eine Liste von allen Scanner Adaptern als repräsentative Inspector Dienste vom RForensic Server anfragen kann.

6.1.4 UI

Für die Umsetzung des UI Bausteins wurden die *CaseManager* und *CaseForensic* Komponenten als Singleton Implementierungen realisiert. Diese bieten die Anwendungsfälle, welche durch Forensiker durchgeführt werden, als Fassade an, welche wiederum die jeweils benötigten Klassen der Bausteine / Komponenten aufrufen. Die Umsetzung der graphischen Anwendung / GUI Komponente wurde durch das PySimpleGUI Paket in Verbindung mit dem Chrome-Embedding-Framework (CEF) Paket realisiert. Der Ablauf wurde dabei in einem Dialog geführten Verfahren rekursiv gestaltet. Angefangen wird dabei mit dem Login Dialog, welcher über die *SessionManager* Komponente die Anmeldung des Benutzers realisiert. Das CEF Paket wird dabei für die Interaktion mit dem MobSFVMInspector Dienst bzw. MobSF verwendet. Die entstandenen GUIs wurden in den Abbildungen A.2 - A.16 dargestellt.

6.2 BackEnd Entwicklung

Die BackEnd Dienste wurden in C# unter Verwendung des ASP.NET SDK implementiert. ASP.NET liefert dabei alle benötigten Funktionen für die Entwicklung einer REST Schnittstelle inklusive dem plattformunabhängigen integrierten Webserver Kestrel.

Für die Persistenz wird das Entity Framework von Microsoft in Kombination mit einer ORM (Object-Relational Mapping) Bibliothek an eine MySQL Datenbank verwendet. Dies erlaubt es Entitäten in Form von Klassen im Quellcode abzubilden, welche daraufhin über simple Methodenaufrufe in die Datenbank hinzugefügt, gelöscht oder modifiziert werden. Die SQL-Anfragen werden dabei dynamisch zur Laufzeit vom ORM generiert. Die Anbindung an die Keycloak Instanz wird über eine Drittanbieter Bibliothek realisiert, welche über Kestrel Schnittstellen automatisch unter Angabe der benötigten Rollen die Authentifikation als auch die Autorisation übernimmt.

Beide Implementierungen teilen sich dabei den grundlegenden Aufbau als auch die Abhängigkeiten von extern verwendeten Paketen / Bibliotheken. Beide Dienste implementieren dabei eine RBAC über die Rollen Konfiguration "IsAnalyst", welche überprüft, ob der Anfragen Steller die Rolle "android-analyst" besitzt. Durch diese Konfigurationen werden automatisch alle API Zugriffe vor der Ausführung der eigen implementierten Logik validiert, sodass der Anfragen Steller nur wenn er authentifiziert und autorisiert ist, diese aufrufen / ausführen kann.

6.2.1 Directory

Der Directory Baustein wurde dabei als einzelner Server implementiert, ohne die einzelnen Komponenten aufzuteilen. Dies ermöglicht es, die Verbindung zwischen den einzelnen Entitäten technisch über Datenbankverweise und dadurch bequem über das ORM zu realisieren. In einem produktiven Betrieb sollten zwecks der Pseudoanonymisierung die Komponenten auf eigene Server verteilt werden. Diese müssten im Betrieb die benötigten Referenzen untereinander auflösen oder sofern es sicherheitstechnisch unbedenklich ist, dies dem Client überlassen. Der Smartphonebesitzer wurde dabei über die *DeviceOwner* Entität abgebildet und die Smartphones über die *Device* Entität. Die Fälle wurden über die *Case* Entität abgebildet. Das realisierte ERM (Entity-Relationship-Modell) befindet sich in der Abbildung 6.1.

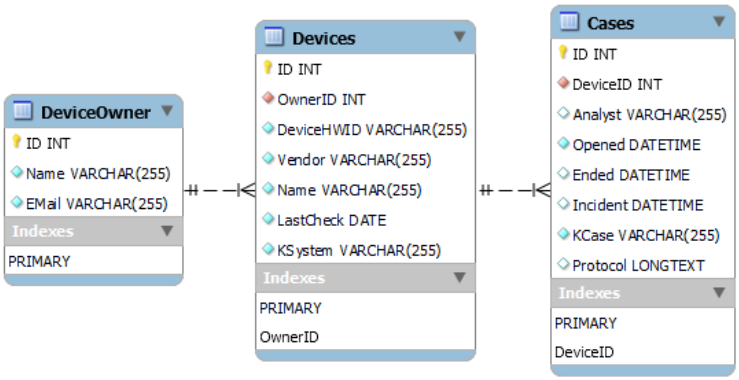

Abbildung 6.1: Directory Entity-Relationship-Modell

6.2.2 RForensic

Der RForensic Server implementiert die RForensic Komponenten aus dem Entwurf. Als eine Scanner Adapter Implementierung wurde eine Anbindung für Virustotal realisiert (VirustotalAdaptee), über welche APK-Dateien hochgeladen und automatisch analysiert werden können. Die Adapter Implementierungen werden dabei während der Laufzeit mittels Depdency Injection ermittelt und enumerierbar / verfügbar gemacht. Für die Verwendung von Virustotal muss ein Zugang beantragt werden und der daraufhin verfügbare API Schlüssel der Instanz beim Start mit übergeben werden. Die Artefakte Verfolgung wird dabei über die *Artifact* Entität persistiert, wodurch die *ArtifactManager* Komponente die gewollte Cache Funktionalität übernehmen kann.

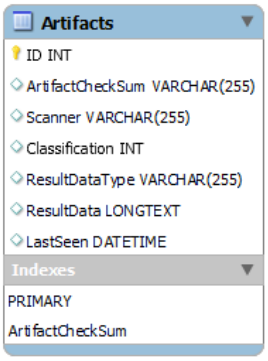

Abbildung 6.2: RForensic Entity-Relationship-Modell

6.3 Verteilung und Auslieferung

Die Verteilung der Dienste wird mittels Containerisierung realisiert (bzgl. 4.1.2). Dafür werden aus den jeweiligen Server Projekten (RForensic und Directory) jeweils ein Container-Image erstellt. Für ihre jeweilige Persistenzen werden MySQL Datenbank Container hinzugezogen. Die Konfiguration der Datenbank (Tabllenstruktur) und die Erstellung von Testdaten wird dabei durch ein mitgeliefertes SQL-Skript realisiert. Die Realm Konfiguration, welche am Anfang des Kapitels erwähnt wurde, wird über ein mitgelieferte JSON-Dokument importiert.

Umgesetzt wurde die Verteilung des BackEnds mittels Docker Compose. Dabei wurde jeweils ein Netzwerk für die Server und ihre Datenbanken definiert (directory-system und rforensic-system) als auch ein Gemeinsames (backend), in welchem sie sich mit dem IAM / Keycloak befinden. Durch die Netzstruktur können nur die jeweiligen Server auf ihre eigene Datenbank Instanz zugreifen. Das gemeinsame backend Netzwerk ermöglicht es jedoch trotzdem, dass die Server auf die Keycloak Instanz zugreifen können, um Tokens zu validieren, als auch von Clients angesprochen werden zu können.

In einem späteren produktiven Einsatz ermöglicht dieser Ansatz es, die APIs einfach in ein Firmennetz zu integrieren, als auch die Skalierbarkeit zu gewährleisten. Für eine Skalierung könnte dabei ein sog. load balancer eingesetzt werden wie NGINX oder in einem Cluster wie Kubernetes, der interne load balancer, welcher daraufhin die Last auf mehrere Server Instanzen verteilt.

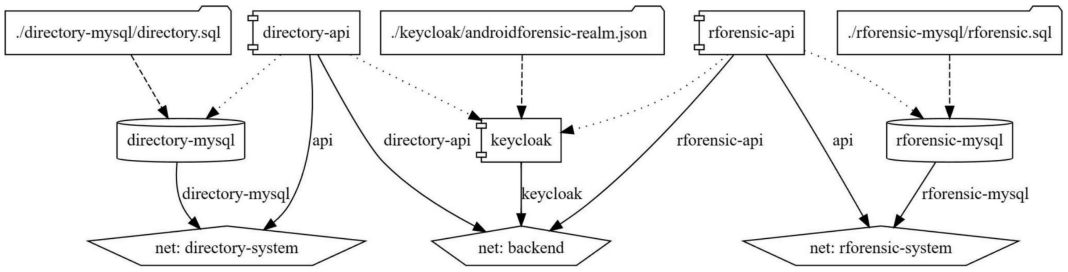

Abbildung 6.3: BackEnd Netzwerk- und Konfigurationssicht

Für das FrontEnd wird für die eigentliche Clientanwendung kein Container benötigt. Benötigte Scanner hingegen sollten dabei aber für ein einfaches Rollout mittels Containern ausgeliefert / auf dem Client mit installiert werden. In der hier entwickelten Anwendung

(6.1) bedeutet dies konkret, dass eine MobSF Instanz über einen Container als auch eine Android VM bereitgestellt wird.

MobSF bietet dabei ein konfigurierbares Docker Container Image an, welches als Parameter / Umgebungsvariablen lediglich den Endpunkt der Android VM als auch den zu verwendenden API-Schlüssel zwecks Authentifikation / Autorisation benötigt. Als Android VM können Container basierte, die offiziell von Google bereitgestellte, in dem Android SDK / Android Studio als auch kommerziellen Angeboten wie Genymotion verwendet werden.

Die Verteilung des FrontEnds wurde dabei in Abbildung 6.3 dargestellt. Die Dienste sollten dabei lokal auf dem Forensiker PC betrieben und erreichbar sein.

Bei dem Betrieb der hier vorgestellten Konfiguration muss darauf geachtet werden, dass die Keycloak / IAM Instanz mit dem gleichen Hostnamen / der gleichen Domain angesprochen wird. Hintergrund hierfür ist das Keycloak prüft, ob der Token auf dem gleichen Wege / über den gleichen Hostnamen angefragt als auch validiert wird. Dies ist sicherheitstechnisch gewollt, da dadurch sichergestellt wird, dass der Token Ersteller explizit für diesen IAM-Endpunkt (Keycloak-Client) ein Token angefordert hat und dieser nicht über ungewollte Umleitungen oder böswillige Angriffe für andere Dienste verwendet wird. Durch die Verwendung von getrennten Keycloak-Clients und dadurch jeweils ausgestellten Tokens für den Directory Server und den RForensic Server wird im Zusammenhang durch das Keycloak-Client Passwort (Client-Secret) auch ein Token Missbrauch ausgeschlossen. Die Keycloak-Clients werden dafür als nicht öffentliche Endpunkte generiert, wodurch diese nur unter Angabe des Client-Secrets über eine REST-API ansprechbar sind und die von ihnen generierten Tokens an den ausstellenden Keycloak-Client zu binden. Die Tokens sind daher nur von dem Keycloak-Client validierbar, von welchem diese ausgestellt wurde. Wird als Beispiel der RForensic Server kompromittiert, so kann ein Angreifer die Tokens der Forensiker abfangen, jedoch kann nicht nutzen, um sich gegenüber dem Directory Dienst zu authentifizieren und autorisieren und Daten zu exfiltrieren. Auch ist es nicht möglich, dass ein Angreifer sich durch blosses Phishing mit gestolenen Anmeldedaten authentifiziert, da Keycloak das jeweilige Client-Secret (Wissensnachweis) erfordert, welches nur auf dem Client und Server vorliegt. Für die Kommunikation zwischen den Client und den jeweiligen Server (Directory und RForensic) wurde jeweils noch ein Client-Server-Secret eingeführt, welches bei jeder Anfrage in den HTTP-Headern mitgesendet wurde. Dies verhindert ein Angreifer durch die Kompromittierung des Keycloaks sich selber Tokens für die Server ausstellen kann, da er zusätzlich das jeweilige Client-

Server-Secret benötigt. In einem produktiven Betrieb sollte dies statt eines einheitlichen Client-Server-Secrets über Benutzerzertifikate und einer PKI implementiert werden (bzgl. 5.2.1).

Es ist daher notwendig, dass ein Hostnamen-Eintrag in dem Betriebssystem vorgenommen wird, wenn der Keycloak Instanz nicht über einen externen bekannten und benannten Endpunkt erreichbar ist. In dem Fall, dass die Keycloak Instanz über eine öffentliche Domain angesprochen wird, müssen die Einträge diesbezüglich sowohl aufseiten des Clients als auch aufseiten der Server ergänzt werden. Die erstellte Docker Compose Datei konfiguriert dies dabei bereits für die Server, sodass diese die Keycloak Instanz unter den Hostnamen 'keycloak' ansprechen können. Auf dem PC muss in diesem Fall (Client und Server auf einem PC) zusätzlich noch ein Eintrag mit dem Hostnamen 'keycloak' erstellt werden, welcher auf die im Container laufende Keycloak Instanz verweist. Bei einer kompletten lokalen Konfiguration muss dementsprechend ein Verweis auf das Loopback Interface erfolgen (127.0.0.1 keycloak).

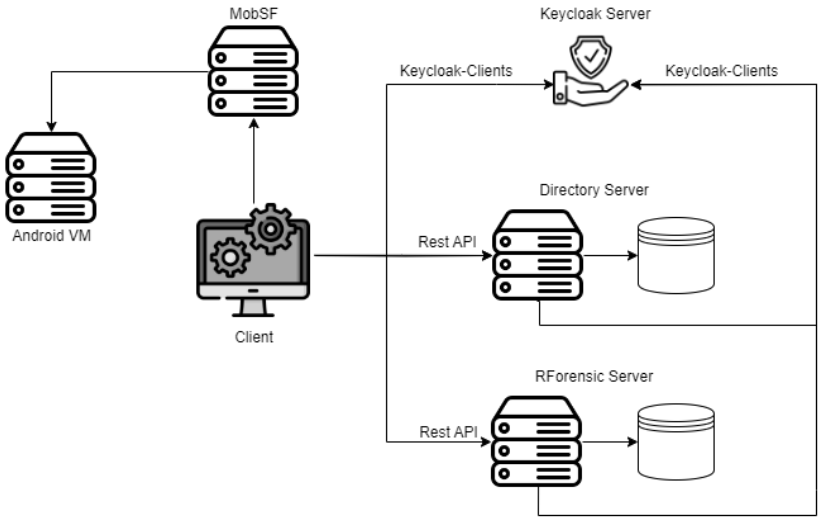

Abbildung 6.4: Client Server Kommunikationsflüsse

6.4 Tests

Für die Tests wurde ein HTC One M7 Android Smartphone mit dem TWRP CR präpariert. Von diesem Smartphone wurde zudem ein Speicherabbild erstellt, welches mit der Mock Implementierung des AndroidHAL Bausteins verwendet werden kann. Für die Umstellung wird aufgrund der Designentscheidung lediglich ein Austausch der verwendeten *DeviceManager* Klasse / Implementierung beim Programmstart benötigt.

Der folgende Test betrachtet dabei die Zeit, welche für die Ausführung der Extractor und Analyzer Dienste benötigt wird. Dabei wird unterschieden, ob das präpariert Smartphone (in den Testdaten mittels HTC notiert) verwendet wird oder der Mock mit dem Speicherabbild (notiert mittels Mock-HTC). Des weiteren wird untersucht, wie effektiv die Artefakte Verfolgung ist und welchen Unterschied es bei dem Durchsatz des Systems macht. Bei dem Durchlauf wurde zusätzlich getestet, ob beide Testkandidaten unabhängig von der Verwendung der Artefakte Verfolgung / Caches die gleichen Ergebnisse liefern. In allen Varianten wurden dabei jeweils 413 Artefakte extrahiert. Von diesen waren 257 Anwendungen (APK) und 156 Zertifikate. Die Anwendungen hatten dabei eine Gesamtgröße von 824 MB und die Zertifikate 794 KB. Die Zeit Ergebnisse der Tests wurden in der Tabelle 6.1 dargestellt.

Geräte Type	**Artefakte Cache**	**Zeit**
HTC	Nein	2:09H
HTC	Ja	0:46H
Mock-HTC	Nein	1:34H
Mock-HTC	Ja	0:10H

Tabelle 6.1: Zeit Messungen unterschiedlicher Konfigurationen

Die Testergebnisse weisen bei der Extrahierung der Daten einen Zeitunterschied von ca. 30 Minuten auf, wenn ein physisches Smartphone verwendet wird im Vergleich zu dem implementierten Mock. Die Zeitdifferenz lässt sich dabei auf den eingebauten USB 2.0 Anschluss des Smartphones und der Art der Artefakte Extrahierung zurückführen. Die Geschwindigkeit eines USB 2.0 Anschlusses liegt dabei bei 480 megabits/s (60 megabytes/s). Es ergibt sich daher eine theoretische Übertragungszeit von 13,75 Sekunden (825 / 60) für die reine Übertragungszeit der Artefakte von dem Smartphone auf den Forensiker PC / Client. Die 13,75 Sekunden beziehen sich dabei auf das bestmögliche Szenario, wenn alle Bauteile die Spezifikation einhalten. Von den gemessenen 30 Minuten ist dieser

Wert jedoch weit entfernt. Auch bei einer angenommenen Effizienzrate von 10% dürfte die Extrahierung nur 2,5 Minuten benötigen.

Nach genauerem Debugging konnte die Zeitdifferenz auf die Art des Hardwarezugriffs und der Vorgehensweise bei der Auffindung und Extrahierung von Artefakten zugeordnet werden. Die rekursive Suche nach Anwendungen (APK) von den Wurzelpfaden der gemounteten *data* und *system* Partitionen durch den dafür vorgesehenen Extractor Dienst nimmt viel Zeit in Anspruch. Bei jeder Betrachtung der Verzeichnisse wird dabei ein *ls* Befehl mit dem Verzeichnispfad an das Smartphone geschickt. Hierfür wird für jeden Befehl die ADB-Binary (Android Debug Bridge) ausgeführt, da die Kommunikation zwischen dem PC und dem Smartphone nicht sitzungsbasiert ist, wodurch für jede Anfrage ein Verbindungsaufbau durchgeführt werden muss. Dies könnte optimiert werden, indem diese nur auf die Pfade von installierten Anwendungen eingeschränkt werden. Mögliche Anwendungen, welche noch nicht auf dem System installiert sind, jedoch vorliegen, würden dadurch nicht betrachtet werden.

Hinzu kommt, dass Artefakte über die Befehlszeile (ADB Ausgabe) entgegengenommen werden müssen. Hierfür müssen diese vorher in eine für die Befehlszeile / Ausgaben konforme Art (Text) konvertiert werden. Dies wird über die Ausgabe des Artefaktes als hexadezimaler Text erreicht, welche daraufhin von dem Client wieder in die binäre Form zurück konvertiert wird. Eine weitere Verbesserungsmöglichkeit wäre es, die Extractor Dienste auf die Smartphones auszulagern. Dies könnte über einen Gegendienst auf dem CR oder über den Kommandozeilen Zugriff in Kombination mit einem Skript, welches durch den jeweiligen Extractor Dienst auf dem Smartphone / CR ausgeführt wird, realisiert werden.

Bei der Untersuchung der Artefakte durch die Analyse Dienste wurde wie zu erwarten eine nahe zu identische Zeit gemessen. Dies liegt an der Tatsache, dass die Artefakte bei der Analyse bereits in dem Client auf dem Forensiker vorliegen, unabhängig von der Test-Variante. Die Artefakte Verfolgung erweist sich dabei durch ihre Cache-Funktionalität als immense Zeitersparnis.

6.5 Zusammenfassung

Die Arbeitsabläufe der Forensiker wurden in dieser POC-Implementierung abgedeckt. Nebensächliche Anforderungen wie der automatischen periodischen Fall-Eröffnung, dem

eigentlichen Mail Versand usw. wurden hingegen nicht implementiert, da diese für den Nachweis des Konzepts nicht von Nöten sind. Auch wurden größere Datensätze wie die Protokolle und Analyseresultate mit in die MySQL Datenbank integriert. In einer für den Betrieb orientierten Umsetzung sollte ein Datenbanksystem verwendet werden, welches besser mit größeren Text-Daten umgehen kann. Ein Beispiel hierfür wäre MongoDB als Dokumenten basiertes Datenbanksystem. Anschließend wurde der Ablauf einer Untersuchung in Tests untersucht und eine mögliche Optimierungsmöglichkeit festgestellt, welche jedoch als Trade-Off Ausführungszeit und Genauigkeit gegenüber Entwicklungszeit und Komplexität stellt.

7 Fazit

Ziel dieser Arbeit war es, ein Konzept und Entwurf so wie eine Teilimplementierung als POC umzusetzen, mit welchem eine datenschutzkonforme Sicherheitsanalyse von Android Smartphones möglich ist. Die Analyse sollte, so weit es sinnvoll ist, automatisiert werden, um den Durchsatz zu maximieren und Fehler der Prüfer (Forensiker) zu minimieren. Das in dieser Arbeit entstandene System, bestehend aus einem Client-Server-Modell setzt diese Rahmenbedingungen mit genügend restlicher Flexibilität und Eingriffstiefe für die Arbeit der Prüfenden um.

Als fachlichen Einstieg wurden im zweiten Kapitel die essenziellen Konzepte und Mechanismen von IAM-Systemen und Android Smartphones erläutert, welche für das Verständnis dieser Arbeit nötig sind. Zudem wurden bereits existierende Ansätze für mögliche Teilprobleme wie der Dateisystemakquisition untersucht und bewertet. Das dritte Kapitel bietet dabei einen Einstieg in den Datenschutz. Hierfür wurden die relevanten Gesetzte und dessen Bedeutung für diese Arbeit erläutert. Zudem wurde deutlich gemacht, welche Folgen auf Unternehmen zukommen, wenn sie nicht die nötigen Mittel investieren, um die Daten korrekt zu verarbeiten und zu schützen. Im vierten Kapitel wurden daraufhin systematisch Anwendungsfälle aus den jeweiligen Aufgabengebieten, welche das entstehende System umzusetzen hat, abgeleitet. Hierfür wurde sich bei der Ausführung der eigentlichen Analyse an dem vom BSI empfohlenen S-A-P-Prozess orientiert und bestehende Ansätze für die Untersuchung der Daten untersucht. Aus den Phasen und den damit verbundenen Tätigkeiten wurden daraufhin Anwendungsfälle abgeleitet, welche die aus der Problemstellung enthaltenen Kriterien berücksichtigen. Das fünfte Kapitel beschäftigte sich daraufhin mit der Entwicklung eines Entwurfes / einer Architektur, welche die aufgestellten Anforderungen abdeckt und Datenschutz relevanten Aspekte berücksichtigt. Hierfür wurde mit einer oberflächlichen Betrachtung der beteiligten Systeme und Akteure angefangen, um daraufhin schrittweise den Entwurf zu verfeinern. Das Endresultat besteht dabei aus einem Client-Server-Modell, welches intern präzise Komponenten mit eindeutig zugewiesenen Aufgaben enthält. Im sechsten Kapitel

wurde eine Prototypenimplementierung aus dem Entwurf und der Architektur aus Kapitel 5 umgesetzt. Die Implementierung deckt dabei die Anwendungsfälle der eigentlichen Sicherheitsanalyse ab. Dabei wurde erläutert, wie die einzelnen Komponenten umgesetzt wurden, um die Komplexität zu reduzieren und die Wartbarkeit und Erweiterbarkeit zu gewährleisten. Abgeschlossen wurde das Kapitel durch Laufzeittests, welche die Funktionalität und das Zeitverhalten untersucht haben. Zeitintensive Teile des Systems wurden dabei identifiziert und Verbesserungsmöglichkeiten aufgezeigt.

Bei der Erstellung des Entwurfs wurde auf die Verwendung von Entwurfsmustern geachtet, welche Abhängigkeiten reduzieren als auch flexible Erweiterungsmöglichkeiten erlauben. Es können daher ohne erhebliche Aufwände beliebig viele Extrahierungs-, Analyse- und Untersuchungsdienste realisiert / erweitert werden. Dies ermöglicht es in der Zukunft verfügbare Lösungen zu implementieren oder einzubinden, ohne signifikante Änderungen / Restrukturierungen am Quellcode vornehmen zu müssen.

Der in den Tests festgestellte Ansatz, über das CR-Betriebssystem Teile der Sicherheitsanalyse auszulagern, würde eine noch effektivere Sicherheitsanalyse ermöglichen. Zusätzlich könnten aus dem erwähnten Vorgehen weniger Zugriffe auf das Smartphone, geringere Zeitaufwände und noch weniger Kontaktmöglichkeiten von den Prüfern mit den Daten resultieren. Durch den Shell Zugriff über die *AuditDevice* Komponente könnten somit auch Analyzer Dienste auf dem Smartphone selber realisiert werden, welche während der Analyse dorthin installiert, ausgeführt und anschließend nur deren Ergebnis extrahiert werden.

Der hier entstandene Ansatz mit dem dazugehörigen System könnte im ersten Moment einen zu hohen Aufwand für Unternehmen darstellen. Beachtet werden muss jedoch, dass die Datenschutzregelungen von Jahr zu Jahr zunehmen und komplexer werden, wodurch Unternehmen gezwungen sind, aufwendigere Lösungen zu entwickeln, welche diesen Anforderungen nachkommen. Unternehmen sollten daher bereits jetzt großen Wert darauf legen, ihre Prozesse, Systeme und Arbeitsmodelle an dem Datenschutz auszurichten, da diese Thematik in Zukunft nur noch komplexer wird.

Literaturverzeichnis

[1] Alendal, Gunnar ; Dyrkolbotn, Geir O. ; Axelsson, Stefan: Forensics acquisition — Analysis and circumvention of samsung secure boot enforced common criteria mode. In: *DFRWS 2018 EU - Proceedings of the 5th Annual DFRWS Europe* (2018), S. S60–S67. – ISSN 17422876

[2] Anglano, Cosimo ; Canonico, Massimo ; Guazzone, Marco: The Android Forensics Automator (AnForA): A tool for the Automated Forensic Analysis of Android Applications. In: *Computers & Security* 88 (2020), 1, S. 101650. – ISSN 0167-4048

[3] Bitkom ; Berg, Achim ; Selen, Sinan: *Wirtschaftsschutz 2021.* 2021. – URL https://www.bitkom.org/sites/main/files/2021-08/bitkom-slides-wirtschaftsschutz-cybercrime-05-08-2021.pdf. – Zugriffsdatum: 02.11.2023

[4] Bitkom ; Dehmel, Susanne ; Stanek, Daniela: Bring Your Own Device. (2013). – URL https://www.bitkom.org/sites/main/files/file/import/130304-LF-BYOD.pdf. – Zugriffsdatum: 02.11.2023

[5] Boehm, Barry W.: Verifying and Validating Software Requirements and Design Specifications. In: *IEEE Software* 1 (1989), S. 75–88

[6] Bundesamt für Sicherheit in der Informationstechnik: BSI Leitfaden IT-Forensik. (2011). – URL https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Cyber-Sicherheit/Themen/Leitfaden_IT-Forensik.pdf. – Zugriffsdatum: 05.12.2023

[7] Gerichtshof der eurpäischen Union (EuGH): Cyberkriminalität: Die Befürchtung eines möglichen Missbrauchs personenbezogener Daten kann für sich genommen einen immateriellen Schaden darstellen. In: *Pressemittelung Nr. 191/23* (2023). – URL https://curia.europa.eu/jcms/upload/docs/application/pdf/2023-12/cp230191de.pdf. – Zugriffsdatum: 15.12.2023

[8] Geschonneck, Alexander: Computer Forensik : Systemeinbrüche erkennen, ermitteln, aufklären. (2008). ISBN 3-89864-253-4

[9] Google: Android platform does not check for SSL certificate revocation. (2014). – URL https://issuetracker.google.com/issues/36993981. – Zugriffsdatum: 07.11.2023

[10] Google: Android CA Zertifikate. (2023). – URL https://android.googlesource.com/platform/system/ca-certificates/+/master/files/. – Zugriffsdatum: 07.11.2023

[11] Google: Android Documentation - Partitions Overview. (2023). – URL https://source.android.com/docs/core/architecture/partitions. – Zugriffsdatum: 13.11.2023

[12] Hurier, Médéric ; Allix, Kevin ; Bissyandé, Tegawendé F. ; Klein, Jacques ; Traon, Yves L.: On the lack of consensus in anti-virus decisions: Metrics and insights on building ground truths of android malware. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 9721 (2016), S. 142–162. – ISBN 9783319406664

[13] Ikram, Muhammad ; Vallina-Rodriguez, Narseo ; Seneviratne, Suranga ; Kaafar, Mohamed A. ; Paxson, Vern: An Analysis of the Privacy and Security Risks of Android VPN Permission-enabled Apps. In: *Proceedings of the 2016 Internet Measurement Conference* (2016). ISBN 9781450345262

[14] Intersoft consulting services AG: *DSGVO Datenschutz-Folgenabschätzung.* – URL https://dsgvo-gesetz.de/themen/datenschutz-folgenabschaetzung/. – Zugriffsdatum: 19.11.2023

[15] Magnet Forensics: *MAGNET GRAYKEY.* – URL https://go.magnetforensics.com/2023_GRAYKEY_Product_Overview_Brief_EN

[16] OpenSecurity: Mobile Security Framework (MobSF). . – URL https://github.com/MobSF/Mobile-Security-Framework-MobSF. – Zugriffsdatum: 29.11.2023

[17] Rashedi, Jonas: Was ist ein Datensilo. In: *Springer Professional* (2020). – URL https://www.springerprofessional.de/datenmanagement/crm/was-ist-ein-datensilo-/18510004

[18] Reuters: *Governments spying on Apple, Google users through push notifications.* – URL https://www.reuters.com/technology/cybersecurity/governments-spying-apple-google-users-through-push-notifications-us-senator-2023-12-06. – Zugriffsdatum: 08.12.2023

[19] Royce, W. W.: Managing the Development of Large Software Systems: Concepts and Techniques. In: *Proceedings of the 9th International Conference on Software Engineering*. Washington, DC, USA : IEEE Computer Society Press, 1987 (ICSE '87), S. 328–338. – ISBN 0897912160

[20] Sebestyèn, Otto G. ; Eschenbach, Rolf (Hrsg.): *Management-Geheimnis Kaizen.* Vienna, Austria : Ueberreuter, 1994

[21] Stallings, William ; Brown, Lawrie: *Computer Security: Principles and Practice.* 3rd. USA : Prentice Hall Press, 2014. – ISBN 9780133773927

[22] TeamWin: Team Win Recovery Project (TWRP). . – URL https://github.com/Teamwin/android_bootable_recovery. – Zugriffsdatum: 29.11.2023

[23] Wulf, RA Dr. Hans M.: *ByoD - Bring your own Device | Fachanwalt informiert.* – URL http://www.it-rechtsinfo.de/byod-rechtsanwalt/. – Zugriffsdatum: 02.11.2023

[24] Yang, Chung-Huang ; Lai, Yen-Ting: Design and Implementation of Forensic Systems for Android Devices based on Cloud Computing. In: *Appl. Math. Inf. Sci* 6 (2012), S. 243–247

[25] Yudha, Fietyata ; Ramadhani, Erika ; Sudyana, Didik ; Hamzah, Waldi N.: A custom recovery approach for physical forensic imaging of android device. In: *AIP Conference Proceedings* 2508 (2023), 3. – ISBN 9780735443198

[26] Zhang, Jason: Machine Learning With Feature Selection Using Principal Component Analysis for Malware Detection: A Case Study. In: *ArXiv* abs/1902.03639 (2019)

A Anhang

A.1 Elektronischer Anhang

Am Ende dieser Arbeit befindet sich eine CD mit dem elektronischen Anhang.

Auf dieser CD befinden sich:

- Thesis als PDF-Dokument "Thesis.pdf"
- Gesamtübersicht des Entwurfs als Bild Datei "Entwurf.png"
- FrontEnd Quellcode im Ordner "FrontEnd"
- BackEnd Quellcode, Docker Compose Datei und die benötigten Konfigurationen im Ordner "BackEnd"

A.2 Gesamtübersicht des Entwurfs

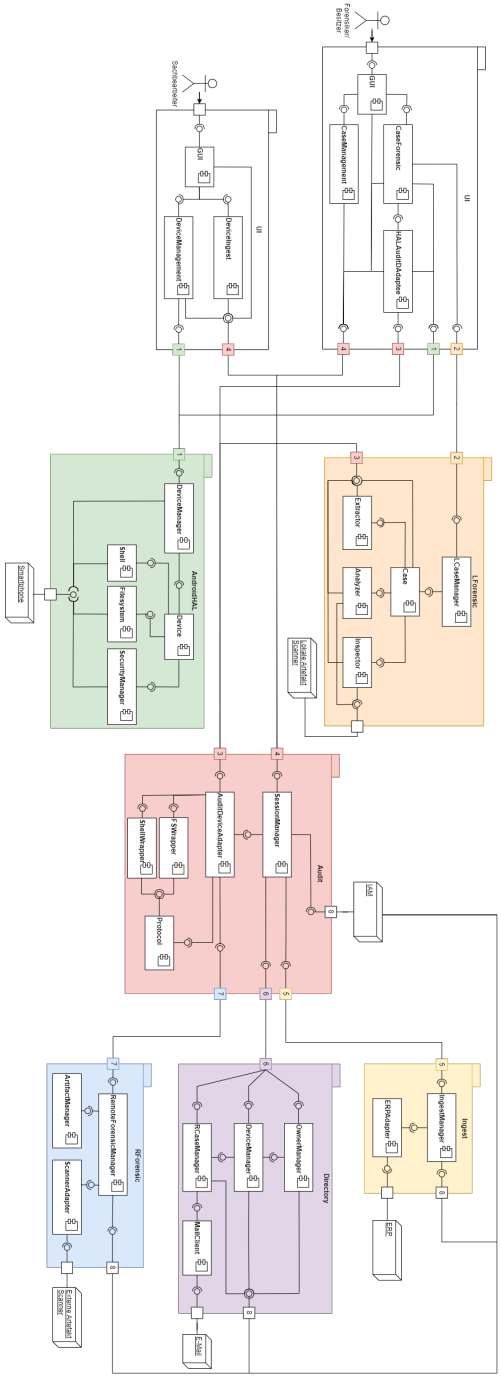

Abbildung A.1: Gesamtübersicht des Entwurfs

A.3 GUI

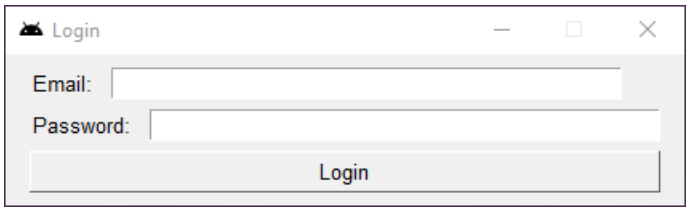

Abbildung A.2: Login GUI

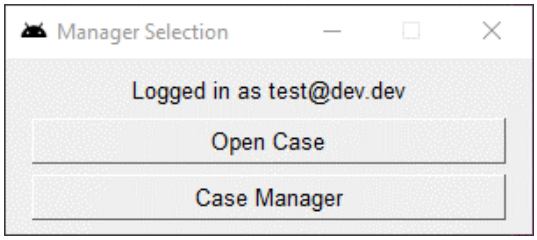

Abbildung A.3: Tätigkeitsauswahl GUI

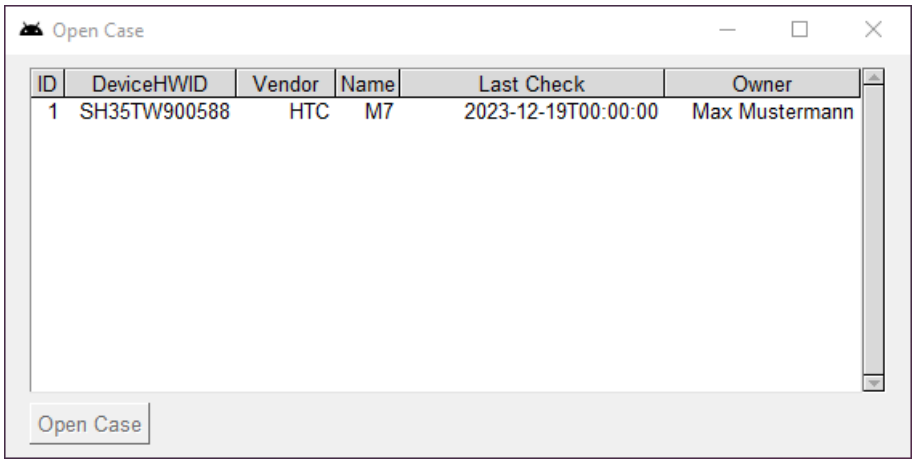

Abbildung A.4: Fall Eröffnung GUI

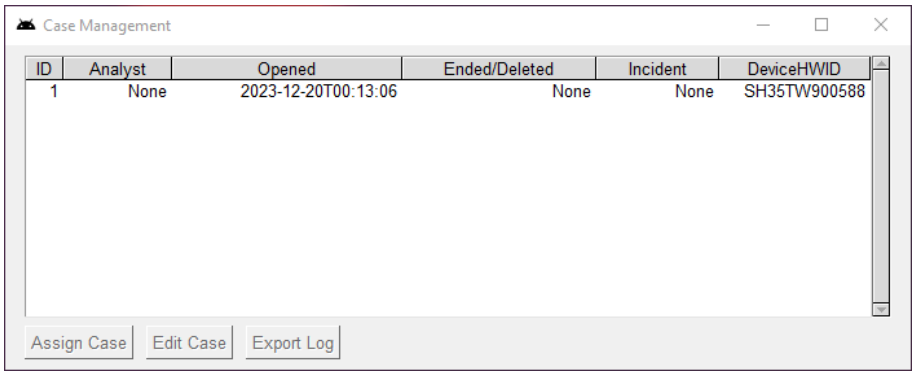

Abbildung A.5: Fall Verwaltung GUI

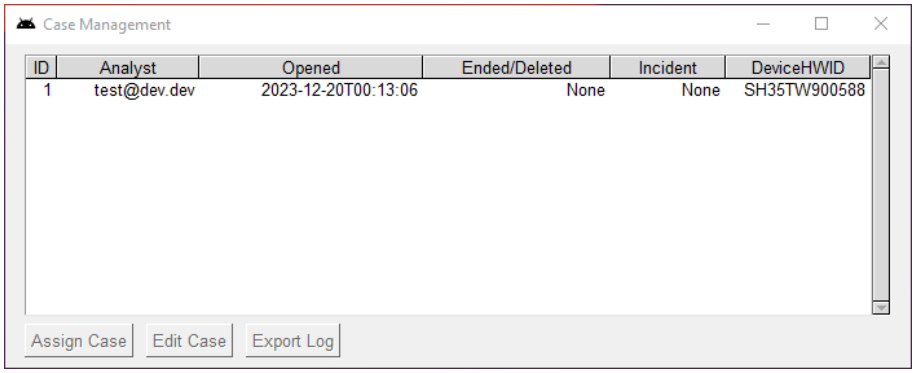

Abbildung A.6: Fall Zuweisung GUI

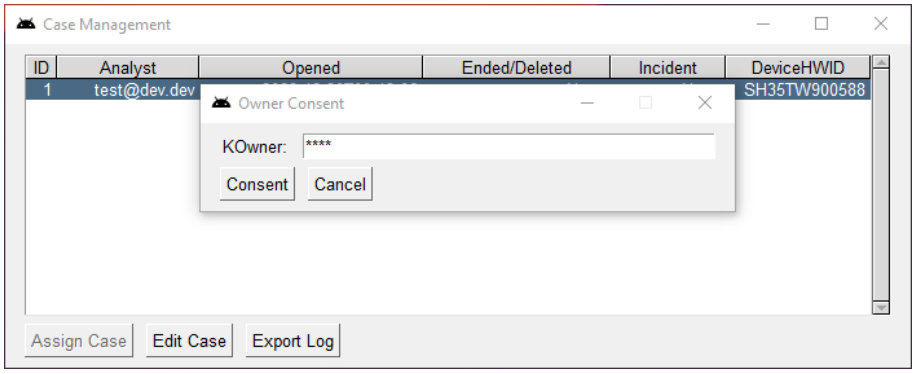

Abbildung A.7: Einverständniserklärung GUI

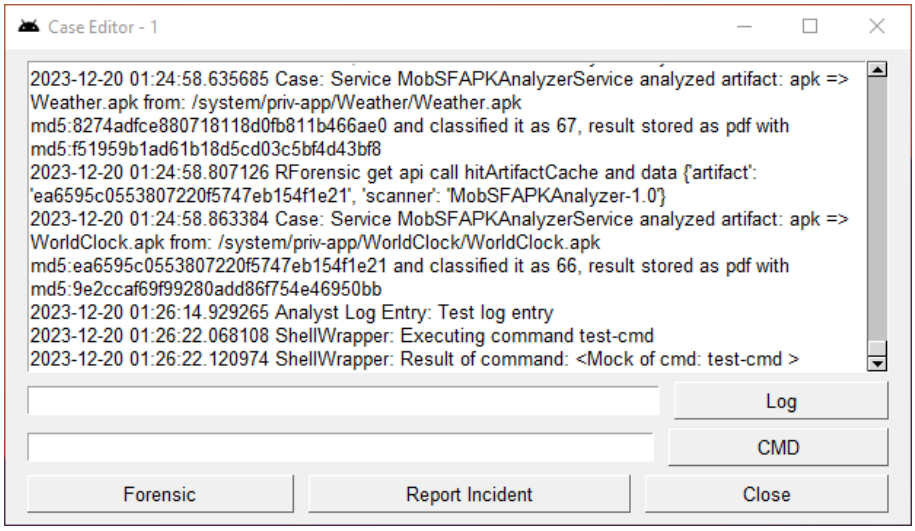

Abbildung A.8: Fall Bearbeitung GUI

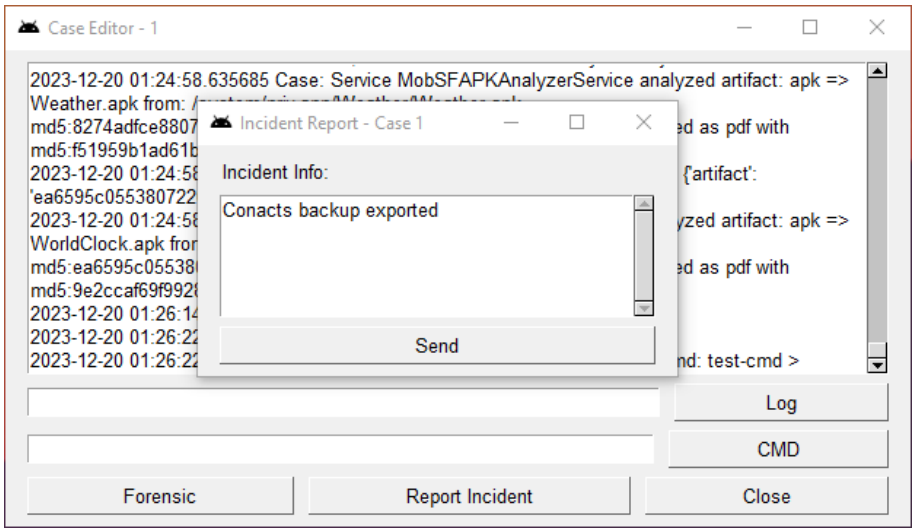

Abbildung A.9: Datenschutzvorfall Meldung GUI

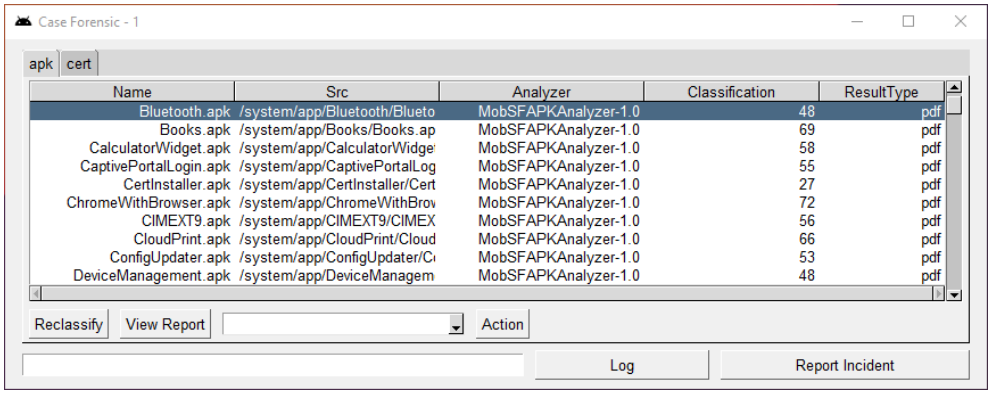

Abbildung A.10: Artefakte Übersicht GUI

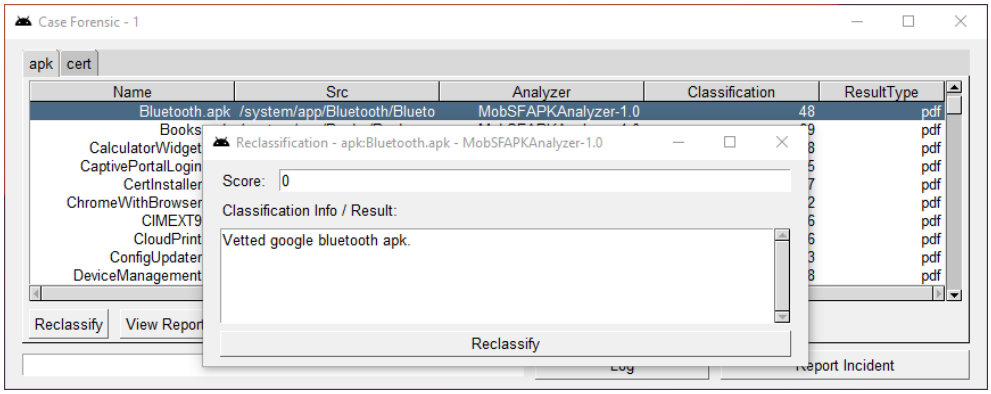

Abbildung A.11: Manuelle Artefakte Klassifizierung GUI

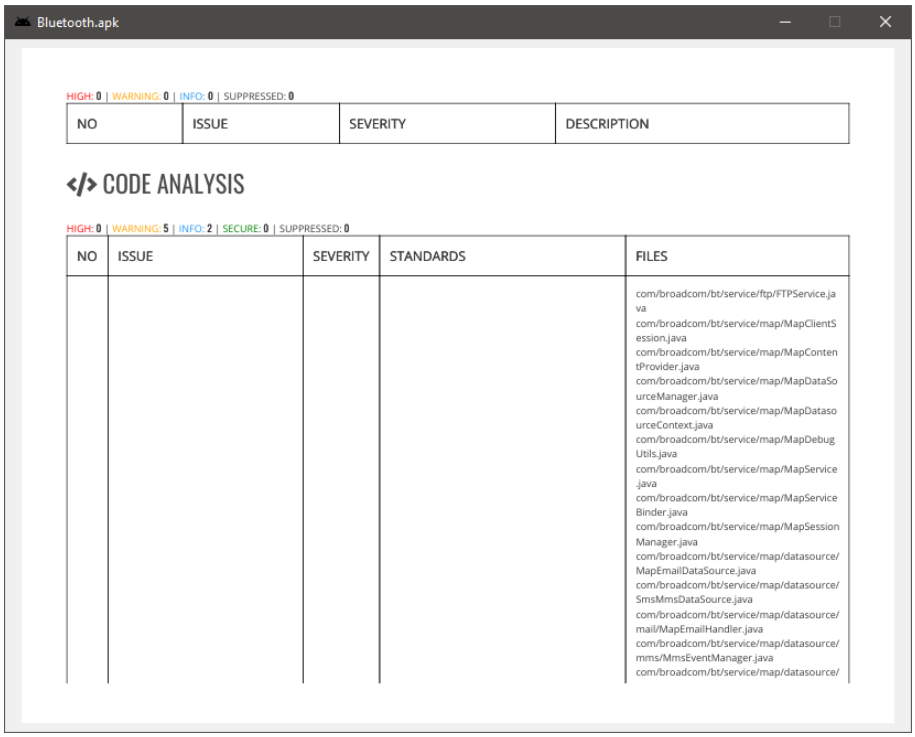

Abbildung A.12: Klassifizierungsergebnis / Report Einsicht (PDF) GUI

Abbildung A.13: Klassifizierungsergebnis / Report Einsicht (Text) GUI

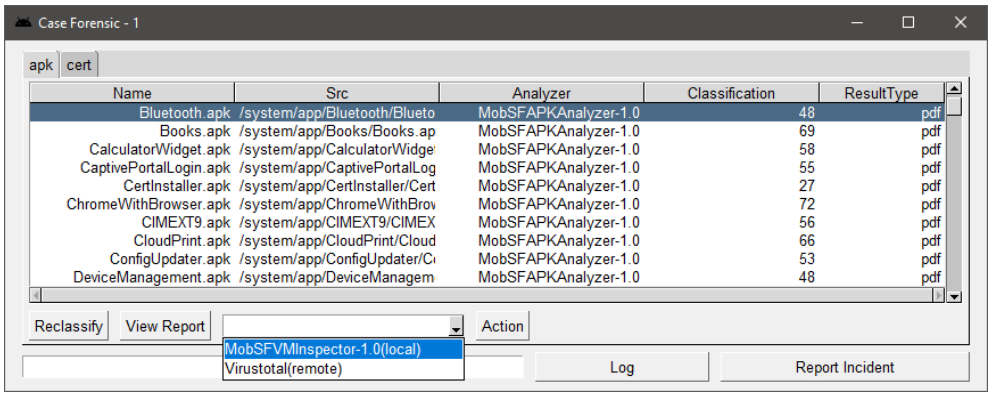

Abbildung A.14: Inspector Auswahl GUI

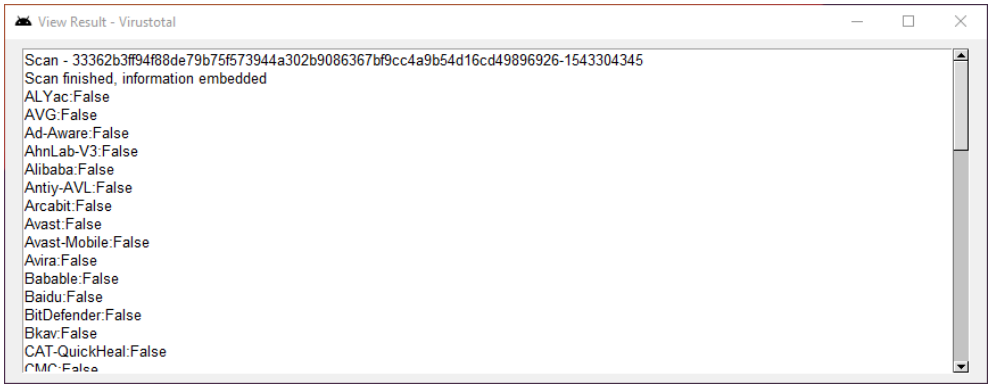

Abbildung A.15: Scanergebnis / Report Einsicht (Virustotal) GUI

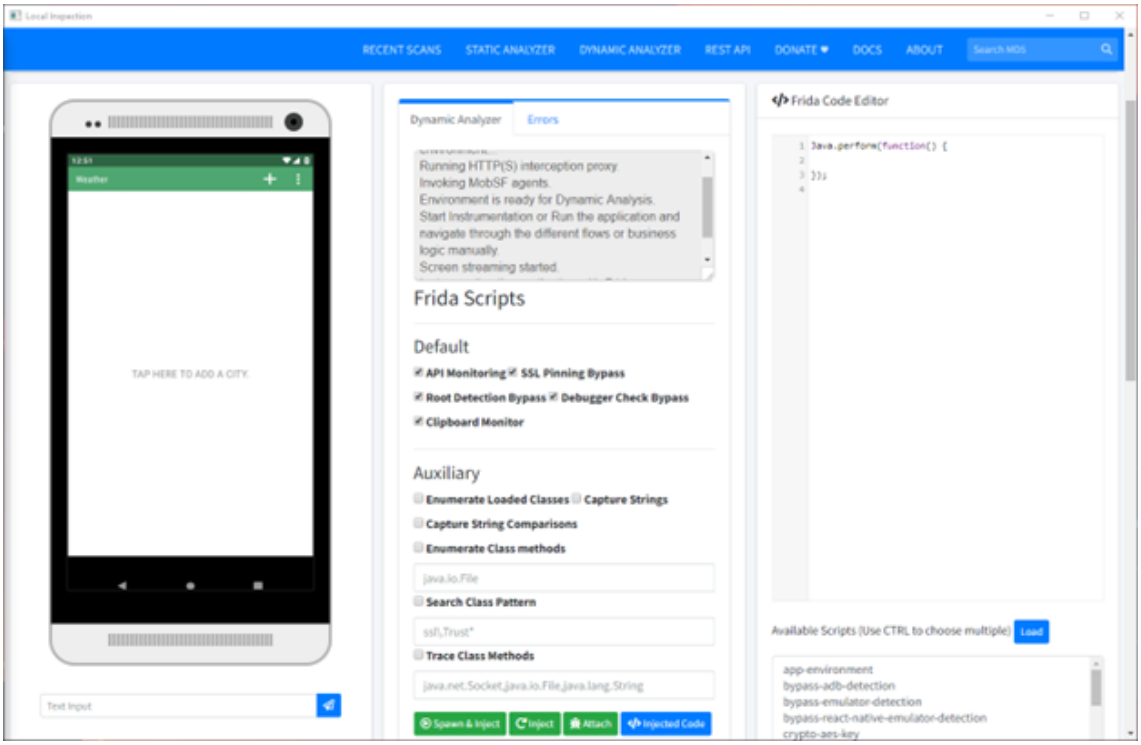

Abbildung A.16: MobSF VM Inspector GUI

Erklärung zur selbstständigen Bearbeitung

Hiermit versichere ich, dass ich die vorliegende Arbeit ohne fremde Hilfe selbständig verfasst und nur die angegebenen Hilfsmittel benutzt habe. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind unter Angabe der Quellen kenntlich gemacht.

Ort	Datum	Unterschrift im Original