Information Security

Forensische Analyse stattgefundener Kommunikation innerhalb von Spiele-Apps

Diplomarbeit

zur Erlangung des akademischen Grades

Diplom-Ingenieur

eingereicht von

Ing. Stefan Hagl, BSc

1710619827

im Rahmen des

Studiengangs Information Security an der Fachhochschule St. Pölten

Betreuung

Betreuer/in: FH-Prof. Dipl.-Ing. Dr. Sebastian Schrittwieser, Bakk.

Mitwirkung: -

St. Pölten, 25.12.2019 _____ _____

(Unterschrift Verfasser/in) (Unterschrift Betreuer/in)

Ehrenwörtliche Erklärung

Ich versichere, dass

- ich diese Diplomarbeit selbständig verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt und mich sonst keiner unerlaubten Hilfe bedient habe.
- ich dieses Diplomarbeitsthema bisher weder im Inland noch im Ausland einem Begutachter/einer Begutachterin zur Beurteilung oder in irgendeiner Form als Prüfungsarbeit vorgelegt habe.
- diese Arbeit mit der vom Begutachter/von der Begutachterin beurteilten Arbeit übereinstimmt.

Der Studierende/Absolvent räumt der FH St. Pölten das Recht ein, die Diplomarbeit für Lehre- und Forschungstätigkeiten zu verwenden und damit zu werben (z.B. bei der Projektevernissage, in Publikationen, auf der Homepage), wobei der Absolvent als Urheber zu nennen ist. Jegliche kommerzielle Verwertung/Nutzung bedarf einer weiteren Vereinbarung zwischen dem Studierenden/Absolventen und der FH St. Pölten.

St. Pölten, 25.12.2019 *Unterschrift Autor*

Zusammenfassung

Telefongesprächsaufzeichnungen oder Anrufdatenermittlungen sind essentielle Methoden der Strafverfolgungsbehörden. Die Verbreitung von Instant Messaging Apps auf Smartphones hat Kriminellen die Möglichkeit gegeben, über andere, schwerer nachzuverfolgende Wege, zu kommunizieren. Die Forschung der letzten Jahre stürzte sich regelrecht auf die forensische Analyse von Messengern wie WhatsApp, Telegram oder dergleichen. Kriminelle Personen nutzen zunehmend alternative Kommunikationswege, welche nicht auf den ersten Blick als solche zu erkennen sind. Eine interessante und unauffällige Kommunikationsmöglichkeit bieten Spiele mit eingebauten Chatmöglichkeiten auf mobilen Geräten.

In dieser Arbeit wird untersucht, welche Daten einer stattgefundenen Kommunikation zwischen zwei SpielerInnen auf Android- bzw. iOS-Geräten direkt abgelegt und so bei einer forensischen Untersuchung aufgefunden werden können. Es gibt etliche Datenakquisitionsmethoden in der Forensik. Je nach Anwendung können diese verschiedene Informationen extrahieren, aber auch unterdrücken. Diese Arbeit stellt die erlebten Informationsunterschiede je nach Methode dar. Es werden verschiedene Spiele-Apps jeweils auf den beiden Betriebssystemplattformen Android und iOS analysiert. Nach Abschluss der Untersuchung wird dargestellt, ob bei Auffindung von Artefakten innerhalb einer Plattform, auch auf das Vorhandensein solcher, im anderen Betriebssystem geschlossen werden kann.

Die genauen Fundorte relevanter Kommunikationsartefakte werden präsentiert und mit den Funden in der jeweils anderen Betriebssystemversion verglichen. Ob überhaupt solche Artefakte innerhalb von Spiele-Apps zu finden sind, ist mit einem klaren Ja zu beantworten. In 82 Prozent der Analysen wurden entsprechende Daten aufgefunden. Auch die Untersuchung von am Gerät gespeicherten Push-Notifications ist für die Auffindung von Hinweisen für einen stattgefundenen Kontakt zwischen zwei SpielerInnen hilfreich. Je nach Betriebssystem sind verschiedene Datenextraktionsmethoden sinnvoller als andere - diese werden entsprechend herausgehoben.

Die Ergebnisse aus dieser Arbeit sollen das Bewusstsein von ForensikerInnen dafür schärfen, dass auch in Spiele-Apps Kommunikationsinhalte auffindbar sind.

Abstract

Mobile phone tapping or call data analysis are essential methods of law enforcement agencies. The increasing usage of instant messaging apps on smartphones enabled criminals to communicate through other, less traceable ways. In recent years, there was a wide range of research regarding forensic analysis of messaging apps like WhatsApp, Telegram or similar applications. Criminals are now searching for alternative ways of covert communication methods. An interesting and unobtrusive communication option could be in-game chats on mobile devices.

This thesis examines what kind of data regarding a recent communication between two players is stored on the devices and therefore can be found in a forensic investigation. There are several data acquisition methods in mobile forensics. Depending on their useage, these methods could extract but also suppress various important information. This thesis also presents experienced information differences depending on the method used for data extraction. All game apps are tested on both operating system platforms Android and iOS. Upon completion of the analysis, we will show if artifacts discovered within one platform, can be inferred with artefacts on the other operating system.

The exact locations of communication artifacts are presented and compared with the findings on the other operating system. Whether such artefacts can be found at all, can be answered with a clear yes. In 82 percent of all games analysed, corresponding data were found. The analysis of stored push notifications was also helpful in finding evidence for a contact between two players. Depending on the operating system, different data extraction methods were more useful than others.

The results of this thesis should raise the general awareness, that communication content can also be found in gaming apps.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1. Einleitung

Über die Jahre veränderten sich die Kommunikationsgewohnheiten der Menschen im Internet. Dadurch vermehrten sich auch die Möglichkeiten für Kriminelle, sich (möglichst verdeckt) über das Internet auszutauschen. Zuerst nutzte man Systeme wie den Internet Relay Chat (IRC), öffentliche Chatrooms auf Webseiten oder den AOL Service. Danach verlagerte sich die Kommunikation auf Social Media Seiten wie MySpace bzw. später Facebook. [2, pp. 3] Als zusätzliche Möglichkeit zu den Diskussionen auf Social Media Webseiten formierten sich in letzter Zeit Instant Messaging Apps, wie zum Beispiel WhatsApp, Signal und Telegram. Die eben angeführten Dienste bieten eine Ende-zu-Ende Verschlüsselung an, weshalb diese Apps auch gerne von kriminellen Organisationen zur verdeckten Kommunikation genutzt werden. [3, p. 949]

Telefongesprächsaufzeichnungen oder Anrufdatenermittlungen sind essentielle Methoden von Strafverfolgungsbehörden zum Nachweis einer Kommunikation zwischen Personen. Die Verbreitung von Instant Messaging Applikationen auf Smartphones hat jedem von uns, aber auch Kriminellen, die Möglichkeit gegeben, über andere, schwerer nachzuverfolgende Wege zu kommunizieren. [4, p. 679] Das Internet entwickelt sich ständig weiter, neue Kommunikationstechnologien entstehen und die dadurch aufgehenden sozialen Räume werden von kriminellen und extremistischen Personen mittlerweile so schnell und effektiv genützt, dass eine Reihe von Geheimdiensten diesen neuen Kommunikationswegen oberste Beobachtungspriorität einräumen. [5, p. 33]

Die Forschung der letzten Jahre stürzte sich regelrecht auf die forensische Analyse dieser Messenger Applikationen. Ein paar Beispiele: Satrya et al. [6] haben 2016 eine Untersuchung der Apps Telegram, Line und KakaoTalk auf Android Geräten durchgeführt. Ovens and Morison [7] haben im selben Jahr eine Analyse des Instant Messengers KIK auf iOS durchgeführt. Mahajan et al. [8] haben 2013 auf Android Systemen die Messenger Apps WhatsApp und Viber analysiert. Al-Saleh und Forihat [9] nahmen sich der forensischen Analyse von Skype auf Android Devices an und Husain bzw. Sridhar [10] analysierten bereits 2009 auf einem iPhone 3G AIM, Yahoo! und Google (Web Based).

Auch das Interesse der Strafverfolgungsbehörden galt bisher vorwiegend diesen Programmen. Kriminelle nutzen also zunehmend alternative Kommunikationswege, welche vielleicht nicht auf den ersten Blick als solche zu erkennen sind:

- Das FBI[1] (Federal Bureau of Investigation) veröffentlichte einen Report, welcher auf die Gefahr einer verdeckten Kommunikation innerhalb des PlayStation Networks (PSN) hinweist. [11] Im konkreten Fall kommunizierten Mitglieder einer Gang, welche unter Hausarrest gestellt wurden, miteinander über das Netzwerk.

- Das New Jersey Regional Operations Intelligence Center[2], der New Jersey State Police, wies ebenfalls in einem Bericht [12] darauf hin, dass Gangmitglieder über eine XBOX360 und einer PlayStation verdeckt miteinander kommunizieren.

- Bereits 2008 warnte die NSA[3] (National Security Agency) in einem internen Report vor der Gefährlichkeit von Online-Games. Diese hätten das Potential ein großes Kommunikationsnetzwerk zu werden [13, pp. 876-877] - Kriminelle könnten sich unter dem Deckmantel der Öffentlichkeit von Online-Games verstecken.

Eine weitere, interessante und unauffällige Kommunikationsmöglichkeit bieten Spiele mit eingebauten Chatmöglichkeiten für mobile Geräte. Auf der Suche nach Hinweisen auf eine stattgefundene Kommunikation konzentrieren sich ErmittlerInnen zuerst auf designierte Kommunikationsmittel wie SMS, WhatsApp oder sonstigen Instant Messenger Apps. Ein Chat innerhalb einer Scrabble- oder Schachapplikation könnte dadurch vielleicht unentdeckt bleiben. Um diese unauffällige Möglichkeit des Informationsaustausches aufzudecken, wurde dieses Thema zur forensischen Untersuchung von Gaming-Apps gewählt. Die Ergebnisse hieraus sollen das Bewusstsein von ForensikerInnen dafür schärfen, dass auch in Spiele-Apps Kommunikationsinhalte auffindbar sein könnten.

Einige medienwirksame Entwicklungen der letzten Jahre bekräftigen die Notwendigkeit, Spiele umfassender als bisher auf ihre Kommunikationsmöglichkeiten hin zu untersuchen:
In der deutschen Stadt Halle, im Bundesland Sachsen-Anhalt, wurde am 9. Oktober 2019 ein Anschlag durchgeführt.[14] Der mutmaßliche Täter streamte den Versuch eines Massenmordes mithilfe einer auf seinem Helm befestigten Kamera über Twitch[4] live in die gesamte Welt. Twitch ist eine Online-Plattform,

[1] https://www.fbi.gov/
[2] https://www.njsp.org/division/investigations/njroic.shtml
[3] https://www.nsa.gov/
[4] https://www.twitch.tv/

in der es SpielerInnen oder sonstigen KünstlerInnen möglich ist, Livestreams mit gleichzeitiger Chatmöglichkeit zu veranstalten. Vor dem Attentat veröffentlichte er ein "Manifest" in dem er sogenannte Achievements[5] für besondere Taten auslobt. [15]
Als Reaktion auf die Tat - und der besonders an die Spiele-Szene angelehnte Durchführung - erklärte der deutsche Innenminister Horst Seehofer [16], dass man "die Gamer-Szene stärker in den Blick nehmen" müsse. Die deutsche CDU Parteispitze präzisierte die Forderung nur einen Tag später: "Wir brauchen adäquate Möglichkeiten für Ermittlungen der Behörden im Darknet, bei der Überwachung von Messenger-Diensten, der Speicherung und Analyse relevanter Daten sowie bei Online-Durchsuchungen." [17] Der Berliner Verfassungsschutz bekräftigte die Forderungen Seehofers und wünscht sich weitere Möglichkeiten, um Kommunikationsmittel im Internet besser überwachen zu können. [18]

Dass es in der Gamer-Szene (so wie in vielen anderen Personenkreisen) auch ExtremistInnen gibt, beschreibt Julia Ebner in ihrem Buch Radikalisierungsmaschinen eingehend, als sie sich für die Recherche undercover der rechtsradikalen Troll-Armee "Reconquista Germanica" anschloss. [5, pp. 130]
In ihrer Arbeit "Cyberspace: A Venue for Terrorism" veröffentlichten David Bieda und Leila Halawi einen Bericht, in dem analysiert wurde, inwiefern sich die Kommunikationsgewohnheiten von kriminellen und extremistischen Personen im Internet verändert haben. [19, p. 38] Sie beschreiben, dass sich zum Beispiel die Propaganda- und Rekrutierungsmöglichkeiten der Terrorgruppe Al-Qaida, von der Zusendung von Videomaterial an den Nachrichtensender Al-Jazeera, über die Veröffentlichung von Inhalten in diversen Internet Foren, bis hin zu Social Media Seiten, verlagert hätten.

Veerasamy und Gobler analysierten in ihrem Paper wie das Internet und dessen Ausprägungen, terroristischen Organisationen in ihrer Kommunikation helfen und auf welche Weise diese Gruppierungen es nutzen. [20] Sie führten aus, dass diese Gruppen verschiedenste Bereiche wie Social Media, Blogs aber auch Spiele dafür missbrauchen, um unter anderem folgende Ziele zu erreichen:

- Anwerbung neuer Mitglieder
- Ausbildung
- Kommunikation
- Koordinierung von Operationen
- Propaganda

[5] https://en.wikipedia.org/wiki/Achievement_(video_gaming) (letzter Zugriff: 9.10.2019)

Auch Shah Mahmood beschrieb auf einer Konferenz [21, p. 575], dass Multiplayer Spiele von TerroristInnen zur Rekrutierung bzw. Kommunikation genutzt werden. Es wurde ebenfalls davon berichtet, dass nach den Vorfällen von 9/11 [6] die US-Regierung die Überwachung diverser Kommunikationskanäle massiv erhöht hat und TerroristInnen nun neue, nicht offensichtliche Kommunikationswege, wie zum Beispiel solcher in Online Games, nutzen.
In Österreich wurde ein 14-jähriger Jugendlicher zu zwei Jahren Haft verurteilt [22], weil ErmittlerInnen auf seiner PlayStation Bombenbaupläne gefunden hatten und ihm die Unterstützung einer terroristischen Vereinigung nachgewiesen wurde.

Welche Auswüchse die Diskussion um den Konnex zwischen Kommunikationskanälen in Spielen und der Durchführung von Terrorattacken annehmen können, zeigt ein Artikel des Forbes Magazins [23], welcher anfangs behauptet hatte, dass die Täter der Parisanschläge im Jahr 2015[7] PlayStations bzw. das PlayStationNetwork (PSN) zur Abstimmung derer Pläne verwendet hätten. Dies stellte sich nach immenser Verbreitung in verschiedensten Medien auf der ganzen Welt als Missverständnis heraus.[24]

Auch wenn sich die meisten Medienberichte betreffend der kriminellen Nutzung von Spielen um Terrorismus drehen, darf man nicht vergessen, dass innerhalb einer Spieleplattform auch andere Straftaten durchgeführt werden können, welche ebenfalls vielleicht durch eine forensische Untersuchung nachzuweisen sind. Cybermobbing oder Stalking kann nicht nur auf Facebook oder anderen Social Media Plattformen entstehen, es kann auch innerhalb von Spielen stattfinden. Taylor et al. haben aus diesem Anlass eine forensische Untersuchung [25] des Spiels Minecraft durchgeführt. In dieser Arbeit wurden entsprechend notwendige Kommunikationsartefakte ausgelesen und dargestellt.

Meist wird nach verübten Terroranschlägen der Ruf diverser Behörden oder Regierungen nach stärkerer Überwachung laut. Aufgrund der starken medialen Verbreitung von Anschlägen und dem einhergehenden, sinkenden Sicherheitsgefühl der Bevölkerung, lassen sich Forderungen nach mehr Überwachungsbefugnisse und damit verbundene Eingriffe in die Grundrechte der Menschen besser argumentieren. Aus solchen Rufen werden konkrete Pläne in Regierungen. [26]
In Österreich soll ab dem Jahr 2020 ein sogenannter "Bundestrojaner" eingesetzt werden. [27] Dieser soll den Behörden ermöglichen, Ende-zu-Ende verschlüsselte Kommunikation (wie sie zum Beispiel in WhatsApp benutzt wird) am Endgerät direkt zu überwachen. Eine fehlerfrei eingepflegte, aktuelle Verschlüsselung ist für derzeitige Computersysteme praktisch nicht überwindbar. Ein "Bundestrojaner" ist

[6] https://en.wikipedia.org/wiki/September_11_attacks (letzter Zugriff am 12.10.2019)
[7] https://en.wikipedia.org/wiki/November_2015_Paris_attacks (letzter Zugriff am 12.10.2019)

eine Software, welche auf dem zu überwachenden Endgerät läuft und über Schwachstellen im Betriebssystem Zugriff auf die Daten anderer Programme hat. Auch eine verdeckte Installation dieser Applikation kann über Schwachstellen durchgeführt werden.
Sollten solche Angriffspunkte nach deren Bekanntwerden vom Hersteller behoben werden, so würde auch der Zugriff bzw. die Installation eines "Bundestrojaners" nicht mehr funktionieren. Dieser Umstand bewegt Hersteller einer solchen Software dazu, nicht bekannte Schwachstellen in Betriebssystemen selbst aufzufinden bzw. am freien Markt einzukaufen und geheimzuhalten.

Unter SicherheitsforscherInnen besteht die vorherrschende Meinung, dass gefundene Sicherheitslücken nicht verkauft, sondern (bestenfalls nach deren Behebung) veröffentlicht werden sollten. Eine Veröffentlichung macht eine Behebung des Problems viel wahrscheinlicher - und eine geschlossene Sicherheitslücke, ist keine mehr - und niemand kann diese noch ausnutzen.
Sollten Sicherheitsbehörden also selbst Exploits entwickeln bzw. einkaufen, so wären sie selbst nicht interessiert daran, dass die ausgenutzten Schwachstellen in Betriebssystemen oder Programmen behoben werden. Die Behörde selbst würde sich damit die Möglichkeit der Überwachung nehmen. Institutionen[8] welche also eigentlich für den Schutz der BürgerInnen und kritischer Infrastruktur zuständig sind, haben also plötzlich Interesse daran, Sicherheitslücken, welche BürgerInnen, Institutionen oder Infrastruktur gefährden könnten, fortbestehen zu lassen. [26, pp. 4-5]

Ob eine solche "Online-Durchsuchung" (so wird ein Zugriff auf die Daten mittels eines "Bundestrojaners" genannt) oder der Weg dahin nun ethisch oder rechtlich in Ordnung ist, ist nicht Gegenstand dieser Arbeit. Es ist jedoch zu bedenken, dass zur Überwachung jeglicher Kommunikation einer verdächtigen Person nicht nur die Daten verschlüsselter Messenger Dienste, wie WhatsApp, Signal oder Telegram durchsucht werden müssten, sondern eben auch andere am Gerät verfügbare Daten - wie zum Beispiel die abgelegten Daten von Spielen. Eine Online-Durchsuchung müsste also, wenn diese schon angewendet wird, auf alle Applikationen am Gerät ausgeweitet werden. Nur dann könnte man auch wirklich alle Kommunikationsspuren am Gerät auffinden.

[8] Beispiele in Österreich: Bundesministerium für Inneres bzw. Bundesministerium für Landesverteidigung

1.1. Forschungsfragen

Aufgrund der Darstellung über die Nutzung "verdeckter" Kommunikationswege, welche Spiele-Apps auf mobilen Geräten kriminellen Personen bieten könnten (siehe Kapitel 1), wurden die folgenden Forschungsfragen für diese Arbeit aufgestellt:

- **Welche Kommunikationsartefakte können bei verschiedenen Spiele-Apps in Android- bzw. iOS-Systemen durch eine forensische Analyse gesichert werden?**
  Innerhalb aller zu analysierenden Gaming-Apps gibt es Chatmöglichkeiten. Es soll herausgefunden werden, welche Daten einer stattgefundenen Kommunikation zwischen zwei SpielerInnen am Gerät direkt abgelegt und so bei einer forensischen Untersuchung aufgefunden werden können. Die Fundorte sollen entsprechend dokumentiert werden.

- **Welchen Informationsunterschied gibt es je nach Extraktionsmethode der Daten?**
  Es gibt etliche Datenextraktionsmethoden in der Forensik. Je nach Anwendung könnten diese verschiedene Informationen extrahieren, aber auch unterdrücken. Diese Arbeit soll die Informationsunterschiede je nach angewandter Methode darstellen und vergleichen.

- **Kann bei Auffinden von Kommunikationsartefakten in Applikationen eines Betriebssystems auf das jeweils andere geschlossen werden?**
  In dieser Arbeit werden Spiele-Apps jeweils auf den beiden Betriebssystemplattformen Android und iOS getestet. Es soll herausgefunden werden, ob bei Auffindung von Artefakten innerhalb einer Plattform, auch auf das Vorhandensein solcher, im anderen Betriebssystem geschlossen werden kann.

1.2. Aufbau dieser Arbeit

In Kapitel 1 wurde die Motivation zur Erstellung dieser Diplomarbeit vorgestellt. Der/die LeserIn wurde in das Spannungsfeld zwischen Kriminalität, Online-Games und deren Kommunikationsmöglichkeiten eingeführt. Einige Berichte über die verdeckte Kommunikation innerhalb von Spielernetzwerken machten die Motivation von Regierungen, man müsse Spielenetzwerke besser beobachten, deutlich. Die daraus resultierenden Forschungsfragen betreffend der forensischen Analyse von Spiele-Apps auf mobilen Geräten wurden vorgestellt.

Im folgenden Kapitel 2 werden die Grundlagen für die Durchführung einer forensischen Analyse von mobilen Geräten vorgestellt. Verschiedene Extraktionsmethoden zur Akquise der notwendigen Daten

werden angeführt und miteinander verglichen. Die Datenstrukturen und aufzufindende Datentypen innerhalb von Android- bzw. iOS-Geräten werden erklärt.

Das Kapitel 3 stellt die Methodik der Durchführung der forensischen Analyse selbst vor. Damit diese Arbeit wissenschaftlich reproduzierbar ist, werden die Auswahl der Applikationen, die Vorgehensweise zur Datenextraktion sowie die Laborumgebung aufgeschlüsselt.

Die eigentlichen Ergebnisse der durchgeführten Analyse werden in Kapitel 4 vorgestellt. Die analysierten Applikationen werden in ihrer Funktion kurz vorgestellt, danach die Ergebnisse je nach Gerätetyp und Extraktionsmethode detailliert angeführt. Am Ende werden die Ergebnisse kompakt dargestellt und miteinander verglichen.

Im letzten Kapitel 5 wird ein Fazit dieser Arbeit gezogen. Welche Schlüsselergebnisse konnten aus der forensischen Analyse und dem Vergleich zwischen den Betriebssystemen gewonnen werden? Mögliche Schritte für eine weitere Forschungsarbeit in diesem Thema werden ebenfalls vorgestellt.

2. Grundlagen

Um eine forensische Analyse der gespeicherten Daten mobiler Devices durchführen zu können, müssen gewisse Grundlagen der Analysemethoden bzw. der zu analysierenden Systeme vorgestellt werden. Durch Kenntnisse der Extraktionsmethoden und den Speicherorten und -typen wichtiger Daten, kann die Analysezeit deutlich verkürzt und der Outcome verbessert werden.

2.1. Methoden der Datenextraktion

In der Literatur wird zwischen einer logischen bzw. physischen Datenextraktion unterschieden. Der Unterschied der beiden Methoden liegt einerseits in der Methodik der Durchführung und andererseits im Umfang der erwartbaren Daten bzw. gewonnenen Informationen aus eben jener. In Abbildung 2.1 findet man einen kompakten Vergleich aller Methoden.

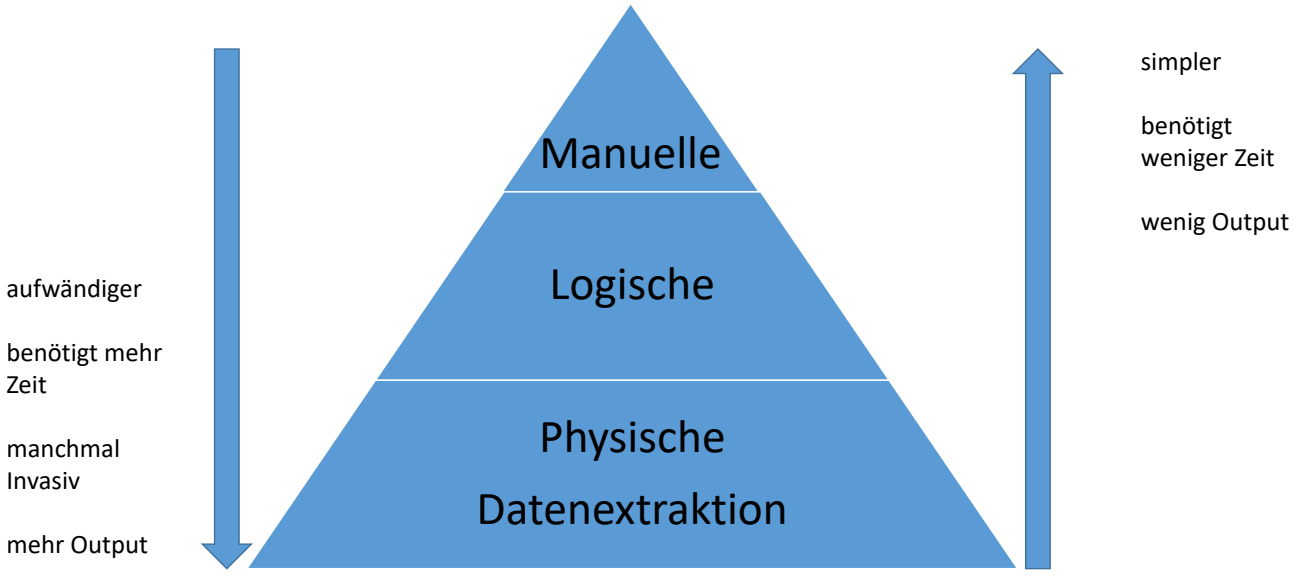

Abbildung 2.1.: Darstellung von Datenextraktionsmethoden (adaptiert von [1, p. 268])

Zusätzlich findet man auch noch Erwähnungen einer "manuellen Extraktion". [28, 29, 30] Diese Methode beinhaltet das Interagieren der ForensikerInnen mit der Benutzeroberfläche des entsperrten Geräts selbst. Das einfache Durchscrollen von Inhalten, Screenshots oder Fotos erstellen und das Verschicken von Daten direkt vom Gerät, gehören zu dieser Methode. [31, p. 606] Diese Vorgehensweise birgt eine große Fehleranfälligkeit in der Analyse. [29] Wichtige Elemente könnten aufgrund der Menge an Informationen übersehen oder bewusst versteckte Artefakte nicht in der Benutzeroberfläche aufgefunden werden. Daten können außerdem in für Menschen nicht lesbarer Form abgespeichert sein (siehe Kapitel 2.3). Aufgrund dieser Fehleranfälligkeit beschränkt sich ein Großteil der Literatur auf die logische und physische Datenakquisition - so wird die manuelle Methode auch in dieser Arbeit nicht näher beleuchtet.

2.1.1. Logische Datenextraktion

Mithilfe der logischen Datenextraktion kann man generell nicht gelöschte Daten vom Gerät extrahieren. Dafür müssen die Daten entsprechend allokiert und für das Dateisystem lesbar vorhanden sein. [32] Eine Ausnahme davon bilden Daten, welche zwar vermeintlich durch Benutzerinteraktion oder Programmroutinen gelöscht wurden, in Datenbanken aber noch vorhanden sind - also nur als "gelöscht" markiert wurden. Eine logische Datenextraktion benötigt immer eine laufende Betriebssystem- bzw. Dateisystemschicht am zu analysierenden Gerät. Diese bildet die Schnittstelle zwischen den physisch abgelegten Daten und dem auslesenden Gerät. [33, p. 4]
Da generell nur allokierte Daten ausgelesen werden können, werden die extrahierten Daten nur maximal so groß wie alle allokierten Bereiche im Gerät zusammengerechnet. Liest man also einen zu 10 Gigabyte vollen 32 Gigabyte großen Speicher aus, werden die extrahierten Daten maximal 10 Gigabyte umfassen. Die restlichen 22 GB gehen dabei verloren.

Auf iOS Geräten wurde das Erstellen von iTunes bzw. iCloud Backups als logische Datenextraktionsmethode für diese Arbeit herangezogen. Auf Android Geräten wurden drei verschiedene Methoden bzw. Stufen einer logischen Extraktion der Daten zu Vergleichszwecken durchgeführt:

1. *Dateitransfer*

   Innerhalb der Literatur ist es uneinheitlich beschrieben, ob diese Methode noch als manuell oder logisch im Sinne der Datenextraktion anzusehen ist. Jedenfalls ist hier der geringste Informationsgewinn zu erwarten. Das entsperrte Gerät wird per USB an den PC angeschlossen. Mittels MTP (Media Transfer Protocol) werden Daten zwischen dem PC und dem Android Device ausgetauscht. Es können nur wenige freigegebene Daten extrahiert werden. Das gesamte Dateisystem ist über diesen Weg nicht zugänglich, auch der Zugang zu den Applikationsdaten ist stark beschränkt. [34,

p. 49]

2. *Backup mittels ADB*[9]

   Mithilfe der Android Debug Bridge ist es ebenfalls möglich, auf verschiedensten Wegen Daten vom Gerät zu extrahieren. ADB ist ein Client-Server Programm, welches im Android SDK bzw. den "Android Platform-Tools" enthalten ist. [35, p. 155] Der Client sowie der Server laufen am PC, ein ADB-Daemon läuft am Android Device. [36, pp. 125-126] Am entsperrten Gerät muss in den Entwickleroptionen "USB-Debugging" aktiviert werden, damit dieser Daemon läuft und auf Kommandos über die USB-Verbindung wartet. Nun können über verschiedene Befehle Daten vom zu analysierenden Gerät auf den PC übertragen werden. So ist es zum Beispiel möglich mit "adb pull" einzelne Dateien zu übertragen - als Untersuchungsmethode wurde jedoch mittels "adb backup -all -apk -obb -shared" ein möglichst komplettes Devicebackup übertragen und danach analysiert. Nach Ausführen des Kommandos ist noch eine Userinteraktion am Gerät aus Sicherheitsgründen notwendig. [36, p. 136] EntwicklerInnen von Applikationen können festlegen, ob ganze Applikationen oder einzelne Dateien vom Backup ausgenommen werden sollen. [34, p. 109] [37]

3. *Backup mittels TWRP*[10]

   Als dritte Stufe, bevor das Android Gerät einer physischen Datenextraktion zugeführt wurde, wurde ein Backup über die Custom Recovery Software TWRP durchgeführt. Eine Recovery Software auf Android Geräten ist ein kleines Betriebssystem, welches eigentlich dafür verantwortlich ist, Updates nach der Auslieferung am Android Betriebssystem selbst durchführen zu können. [38, p. 7] Da die mitgelieferte Recovery Software nur offizielle Updates am System erlaubt und keinerlei Zusatzfunktionen für ForensikerInnen bietet, kann diese durch ein Third-Party Projekt wie TWRP überschrieben werden. Nach dem Flashen der neuen Software wird das Gerät in den Recovery Modus gebootet. TWRP startet sich und bietet über die Backupoption alle Partitionen am Gerät zum Download an. Bei verschlüsselten Devices wird dafür das Passwort angefordert. In dieser Stufe werden alle für das Dateisystem sichtbare Dateien am Gerät ins heruntergeladene Backup kopiert. Einzig gelöschte bzw. nicht allokierte Daten können mit dieser Methode nicht analysiert werden.

[9] Android Debug Bridge

[10] Team Win Recovery Project - https://twrp.me/

2.1.2. Physische Datenextraktion

Die physische Datenakquisition gilt im Gegensatz zur logischen Extraktion als aufwändiger, nicht immer durchführbar aber im Ergebnis als deutlich umfassender. (siehe zum Vergleich Abbildung 2.1) Physische Methoden haben die Möglichkeit, auch im Dateisystem gelöschte Daten zu extrahieren. Dateisysteme markieren Daten oft nur als gelöscht, verändern diese Datenbereiche aber nicht physisch, sodass die zuvor geschriebenen Bits in den nicht allokierten Datenbereichen noch vorhanden sind. Durch Schreiboperationen könnten nicht allokierte Bereiche jedoch jederzeit wieder durch andere Daten befüllt werden. Dies führt bei der Analyse von gelöschten Daten oft zu nicht mehr zusammenhängenden Datenfetzen.

Da mithilfe der physischen Datenextraktion alle Bereiche (auch die nicht allokierten) des Speichers ausgelesen und bitweise kopiert werden, ist der Umfang der erwartbaren Daten viel höher. Liest man also einen zu 10 Gigabyte vollen 32 Gigabyte großen Speicher aus, werden die extrahierten Daten die vollen 32 Gigabyte umfassen.

In dieser Arbeit wird das Kopieren von Datenblöcken bzw. Partitionen mittels dd [11] als einzige physische Extraktionsmethode betrachtet. Es war bei beiden Geräten (Android und iOS) aufgrund von Jailbreaks bzw. Rooten möglich, die notwendigen Speicherbereiche mit diesem Tool blockweise auszulesen. So waren keinerlei sonstige Methoden, wie zum Beispiel dem Auslöten [39] oder dem Auslesen der Speicherchips über JTAGs [12], notwendig.

2.2. Android und iOS Datenstruktur

Je nach System unterscheidet sich die Organisation der Daten am Gerät. In diesem Kapitel werden die wichtigsten Merkmale beider Systeme für die Durchführung einer forensischen Analyse von Userdaten vorgestellt.

[11] https://en.wikipedia.org/wiki/Dd_(Unix) bzw. http://man7.org/linux/man-pages/man1/dd.1.html
[12] Joint Test Action Groups

2.2.1. Android

Auf Android Geräten findet man je nach Ausprägung verschiedene Partitionskonzepte auf den Geräten. [34, p. 73][28, pp. 191] Die standardmäßige Organisation (und somit am meisten verbreitete) der am Gerät vorzufindenden Partitionen, lautet aber wie folgt:

- **Boot-Partition**

  In dieser Partition liegt alles, was der Kern eines Android System für einen erfolgreichen Start benötigt. Hier liegt der Kernel des Systems. Userdaten findet man hier keine.

- **Cache-Partition**

  In der Cache Partition werden Systemupdates oder Recovery Logs abgespeichert. Auch oft verwendete Daten anderer Paritionen kann man hier finden. Es ist also durchaus möglich, dass Daten innerhalb einer User-Applikation nicht mehr vorhanden sind, diese aber noch in der Cache Partition aufgefunden werden könnten. Bei der Durchführung dieser Arbeit, waren hier jedoch keine für die Analyse relevante Daten abgespeichert.

- **Recovery-Partition**

  Innerhalb der Recovery Partition findet man ein minimales Android Image, welches als Failsave oder für die Durchführung von systemweiten Updates verwendet wird. Wie in Kapitel 2.1.1 beschrieben ist, wird diese Partition verwendet, wenn man eine Custom Recovery Software wie TWRP auf das Gerät installiert. Diese offenbart ForensikerInnen mehrere Möglichkeiten, um auf den Datenspeicher des Geräts direkt zugreifen zu können.

- **System-Partition**

  Das generelle Android Framework, Systemkomponenten oder Programmbibliotheken findet man hier. Ohne diese Parition wäre ein Android Gerät nicht mehr lauffähig. Für die Analyse von Kommunikationsdaten ist diese Partition jedoch nicht hilfreich, da hier keine Applikationsdaten abgespeichert werden.

- **Data- / Userdata-Partition**

  Diese ist für ForensikerInnen die interessanteste Partition. Hier werden alle Applikationsdaten und somit auch Kommunikationsdaten gespeichert. In der Literatur wird diese Partition auch oft "interner Datenspeicher" genannt. Hier finden sich beispielsweise Kontakte, SMS, gewählte Rufnummern aber eben auch Kommunikationsartefakte anderer User-Applikationen wie WhatsApp oder Spielen.

Innerhalb der Userdata-Partition finden sich zwei Bereiche in denen besonders nach Kommunikationsdaten gesucht werden sollte:

- **Applikationsdaten**

  Die Daten der zu analysierenden App enthalten, aufgrund des Prinzips der Isolierung von anderen Apps, alles notwendige, um sich selbst lauffähig zu halten. [40, pp. 727-728] Es ist ihnen nicht möglich, Daten von anderen Applikationen zu verändern. Das bedeutet: Wenn innerhalb einer Applikation Kommunikation stattfindet, und diese auch offline am Gerät vorgehalten wird, so ist davon auszugehen, dass diese in den eigenen Applikationsdaten abgespeichert ist. Der Pfad im Dateisystem zu den abgelegten Appdaten lautet:

  ```
  /data/data/<Appname>/
  ```

  Innerhalb dieses Pfades müssten also in den abgelegten Dateien (siehe auch Kapitel 2.3) Kommunikationsartefakte aufzufinden sein.

- **Notifications**

  Es ist zu bedenken, dass Push Benachrichtigungen Kommunikationsartefakte beinhalten könnten. Beispiel: Wenn jemand eine Mail erhält, sendet die empfangende Mailapplikation am Smartphone (je nach Sicherheitseinstellung) eine Push Notification an eine Android API. Diese wird dann als Userbenachrichtigung am Home-Screen oder in der Benachrichtigungsleiste für die BenutzerInnen ersichtlich. Eine in Spielen beliebte Google API, welche Notifications verarbeiten kann, ist "Google Play Services". Der interne Package-Name der API lautet com.google.android.gms.[13] Der Pfad im Dateisystem zu den abgelegten Push-Notifications innerhalb der Applikationsdaten der Google Play Services lautet:

  ```
  /data/data/com.google.android.gms/files/fcm_queued_messages.ldb/
  ```

  Innerhalb dieses Pfades werden die Benachrichtigungen, welche an dieses Service geschickt wurden, aufgehalten, bis diese von den NutzerInnen des Geräts bestätigt wurden. Sobald dies durchgeführt wurde, werden die abgelegten Benachrichtigungen wieder gelöscht. Die Auffindung von Artefakten wird also wahrscheinlicher, wenn eine physische Datenextraktion durchgeführt wird und die gelöschten Benachrichtigungsdateien möglicherweise wieder rekonstruiert werden können.

[13] https://play.google.com/store/apps/details?id=com.google.android.gms

2.2.2. iOS

Die Organisation der Partitionen auf iOS-Geräten ist deutlich übersichtlicher gestaltet, als es in Android-Systemen (siehe Kapitel 2.2.1) ist. Ein iOS-Device enthält üblicherweise nur zwei Partitionen. [41, pp. 57-60][28, pp. 52-53]

- **System**

  Die Systempartition wird in der Literatur auch oft Firmware-Partition genannt. Wenn ein Gerät im Normalzustand läuft (also kein Jailbreak installiert wurde), ist diese Partition schreibgeschützt in das Dateisystem eingehängt. Die forensische Untersuchung dieser Daten in Hinblick auf Kommunikationsdaten ist also generell nicht von Interesse.

- **Userdata**

  Die User-Datenpartition enthält alle Daten, welche von BenutzerInnen des Gerätes generiert wurden. Hier werden Kontakte, Fotos oder eben Applikationsdaten von Programmen, welche von den NutzerInnen installiert wurden, gespeichert. Innerhalb dieser Umgebung findet man also auch Kommunikationsdaten, sollten diese von den Applikationen am Gerät abgelegt werden. Für eine forensische Analyse von Kommunikationsspuren sollte das Hauptaugenmerk auf diese Partition gelegt werden. Betrachtet man das Dateisystem von Root (/) aus, so ist die User-Datenpartition unter */private/var* im System eingehängt. Dies wird besser ersichtlich, wenn man sich die *fstab* eines iOS Systems ansieht. Die Systempartition welche, wie oben beschrieben ist, generell schreibgeschützt ist, steht an erster, die User-Datenpartition an zweiter Stelle:

  ```
  /dev/disk0s1s1 / hfs ro 0 1
  /dev/disk0s1s2 /private/var hfs,nosuid,nodev rw 0 2
  ```

Eine typische Ordner Aufteilung innerhalb der User-Datenpartition */private/var* sieht (hier gezeigt bei dem analysierten iOS 12.1.3 Gerät) so aus:

```
.DocumentRevisions-V100
.fseventsd
audit
backups
buddy
cache
containers
```

```
db
ea
empty
folders
hardware
installd
internal
iomfb_bics_daemon
keybags
Keychains
lib
local
lock
log
logs
Managed Preferences
mobile
MobileAsset
MobileDevice
MobileSoftwareUpdate
msgs
networkd
preferences
root
run
spool
staged_system_apps
tmp
vm
wireless
```

Aus forensischer Sicht ist das Verzeichnis *mobile* am interessantesten. Es enthält unter anderem die Applikationsdaten. Sollten Apps Kommunikationsspuren am Gerät ablegen, so sind diese bei den eigenen Applikationsdaten zu finden. Der vollständige Pfad zu diesen Dateien lautet:

```
/private/var/mobile/Containers/Data/Application/<UUID_der_App>
```

In Kapitel 2.2.1 wurde bereits auf die Wichtigkeit von Push-Notifications in Android Systemen hingewiesen. Diese könnten Kommunikationsartefakte enthalten. Seit iOS 12 werden die Benachrichtigungen auf Apples Betriebssystem unter folgendem Pfad abgelegt:

```
/private/var/mobile/Library/UserNotifications/<BundleID_der_App>
```

Innerhalb dieses Pfades werden die Benachrichtigungen nach App sortiert abgelegt. [42]

2.3. Datentypen

Innerhalb der in Kapitel 2.2 gezeigten Datenstrukturen finden sich verschiedenste Dateitypen. Die folgenden Typen beinhalten meist wichtige Informationen für die Analyse und werden daher kurz vorgestellt:

- **SQLite Datenbanken**

  SQLite Datenbanken stellen eine sehr beliebte Methode dar, um auf mobilen Geräten Informationen organisiert abzuspeichern. Aufgrund der spezifischen Anforderungen an Applikationen innerhalb von Smartphones bezüglich Arbeitsspeichernutzung bzw. Strombedarfs eignet sich diese Plattform hervorragend als Datenbanksystem für mobile Devices. [43, p. 83]

  Eine SQLite Datenbank benötigt keinen Server und keine Installation. [44, pp. 1-7] Die Daten werden innerhalb einer einzigen Datei abgelegt und können bei Bedarf auch einfach von System zu System transferiert werden. Diese Datei wird innerhalb der eigenen Applikationsdaten abgelegt. Der Ablageort einer einzigen Datei und das Nichtvorhandensein eines Servers gibt EntwicklerInnen eine simplere Möglichkeit die Datenbank gleichzeitig und konsistent mit den Applikationsdaten zu sichern, als mithilfe eines traditionellen RDBMS[14] wie beispielsweise MSSQL[15] oder Oracle[16]. [45, pp. 42-46]

[14] Relational Database Management System
[15] https://www.microsoft.com/en-us/sql-server/sql-server-2019
[16] https://www.oracle.com/database/

Daten innerhalb einer SQLite Datenbank können zusätzlich mithilfe der Open-Source Erweiterung SQLCipher[17] verschlüsselt werden. Ohne diese Verschlüsselung sind die Daten im Klartext bereits über einen einfachen HEX-Dump lesbar.

- **PLIST / JSON Dateien**

  Property Lists (kurz PLists) sind Dateien mit seriell abgespeicherten Objekten. [46, pp. 37-38] Diese Objekte werden in einer XML Struktur abgespeichert. Innerhalb dieses Aufbaus können in den einzelnen Schlüsseln Klartextdaten, sowie auch Base64 enkodierte Daten abgelegt werden. [41, p. 61] Property Lists wurden von Apple entwickelt und waren bisher vorwiegend auf den iOS Geräten eingesetzt. Ein Äquivalent in der Android Welt stellen JSON (JavaScript Object Notation) Dateien dar. Man findet bei Analysen jedoch auch PLISTs auf Android Geräten. Ein durchgeführter Vergleich beider Filetypen [47] liefert die Aussage, dass wenn man Applikationen nur für iOS Geräte entwickeln will, PLIST eine gute Wahl wäre, wenn man aber Cross-Plattform entwickelt, JSON verwenden sollte.

- **andere lesbare Textdateien**

  Property Lists bzw. JSON Dateien sind bereits lesbare Textdateien. Bei Analysen findet man außerdem häufig .xml oder .txt Dateien. Diese werden vorwiegend für die Speicherung von Einstellungen verwendet, welche sich nicht allzu oft ändern. Logdateien (praktischerweise oft im Dateisystem mit .log Endung) bilden hier die Ausnahme. Diese werden häufig beschrieben - meist seriell fortgesetzt. In Logdateien können sich (je nach Design der EntwicklerInnen) oft nützliche Informationen verstecken.

- **Binärdateien**

  Einen großen Teil der aufzufindenden Dateien bilden Binärfiles. Diese Files sind für AnalystInnen nicht auf Anhieb lesbar. Auch ist für ForensikerInnen nur anhand des Dateinamens erkennbar, ob der Inhalt für eine tiefer gehende Analyse sinnvoll wäre. Wenn sich EntwicklerInnen einer Applikation, dazu entscheiden, keine der bereits in der Programmiersprache implementierten Funktionen für die Bearbeitung von PLIST- oder JSON-Dateitypen zu nutzen, bzw. eigene Datenstrukturen innerhalb von Dateien am Gerät ablegen zu wollen, so werden Binärdateien genutzt. Eine Möglichkeit binäre Strukturen zu analysieren, liegt im Reverse-Engineering der Applikation. Hiermit kann herausgefunden werden, mit welchen Datenstrukturen die Dateien befüllt werden. Nach Auffinden solcher Informationen ist man selbst in der Lage ein Programm zu entwickeln, welches aus den Dateien die gleichen Datenstrukturen wieder auslesen und für Menschen ver-

[17] https://www.zetetic.net/sqlcipher/

ständlich darstellen kann.

2.4. Netzwerk- bzw. Serverartefakte

Wenn in Spielen über mobile Devices hinweg kommuniziert wird, müssen Daten über ein Netzwerk (2) und meist auch über Server (3) zum anderen Gerät geschickt werden. (siehe Abbildung 2.2) Aufgrund dieses Aufbaus könnten je nach Implementierung auch Kommunikationsartefakte in diesen beiden Bereichen bei einer forensischen Untersuchung aufgefunden werden. Ziel dieser Arbeit ist es jedoch, direkt am Gerät (1) abgelegte Daten aufzufinden und zu dokumentieren. Eine in anderen Arbeiten durchzuführende, weitere Analyse des Netzwerkverkehrs oder der beteiligten Server wäre wünschenswert und als Ergänzung äußerst aufschlussreich.

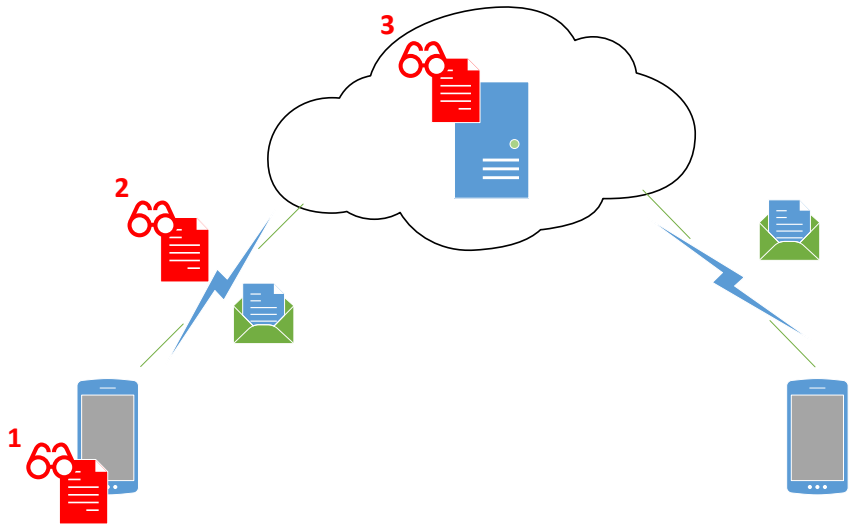

Abbildung 2.2.: Darstellung möglicher Analysepunkte

3. Methodik

In diesem Kapitel wird die Auswahl der zu analysierenden Applikationen erklärt. Warum wurden gerade diese Apps gewählt? Außerdem wird die Vorgehensweise der Generierung künstlicher Kommunikationsartefakte auf beiden Geräten vorgestellt und mit welcher Methodik jedes System analysiert wurde. Um alle zuvor vorgestellten Analysemethoden durchführen zu können, war eine kontrollierte Laborumgebung notwendig. Das Android Gerät wurde dafür gerootet und eine Custom Recovery Software aufgespielt. Am iOS Device wurde ein Jailbreak durchgeführt. Diese Voraussetzungen werden sich natürlich nicht zu 100 Prozent mit den Möglichkeiten auf zur Analyse sichergestellten Smartphones decken.

3.1. Auswahl der Apps

Die Motivation dieser Diplomarbeit ist es, verstecke Kommunikationsmöglichkeiten in Spiele Apps per forensischer Analyse sichtbar zu machen. Die Auswahl der Apps erfolgte nach einer manuellen Suche im Google Play Store[18] mit den Schlüsselwörtern "game" und "chat" - diese triviale Suche könnten auch BenutzerInnen durchführen, die das Ziel verfolgen, schnelle und einfache Chatmöglichkeiten in Spielen nutzen zu wollen.

Da diese Diplomarbeit außerdem das Ziel verfolgt, die selben Chatmöglichkeiten gleichzeitig auf den Plattformen Android sowie iOS zu analysieren, wurde nach Auffinden eines Treffers, dieser auch im Apple App Store [19] gesucht. War ein Spiel vom selben Hersteller in beiden Stores verfügbar, so wurde diese Applikation ausgewählt.

Da die jeweiligen Downloadzahlen nur im Google Play Store öffentlich ersichtlich sind, wurden die finalen Apps aus der Vorauswahl nach dieser Zahl ausgewählt.

Die nachfolgend analysierten Applikationen "Playstation Messages" (siehe Kapitel 4.5) sowie "XBOX" (siehe Kapitel 4.10) sind zwar keine Spiele-Apps per se, wurden aber aufgrund deren Verbreitung und plattformübergreifenden Chatmöglichkeiten innerhalb derer Spielnetzwerke zusätzlich ausgewählt. In Kapitel 1 waren einige Berichte über den Gebrauch von XBOX bzw. PlayStation zur verdeckten Kom-

[18] https://play.google.com/store

[19] https://www.apple.com/ios/app-store/

munikation aufgelistet - diese waren zusätzlich ausschlaggebend die Applikationen forensisch zu untersuchen.

3.2. Vorgehensweise

Nach dem Vorbereiten der Testgeräte (siehe Kapitel 3.3) wurden die zuvor ausgewählten Applikationen installiert. Um eine eindeutige Suche nach Artefakten durchführen zu können, wurden vor dem Start der Apps verschiedene Maßnahmen festgelegt:

- **Username / Nickname**

  Damit man innerhalb der Spiele erkannt wird, bekommt jede/r SpielerIn einen Username oder "Nickname" zugewiesen. Oft kann man diesen anfangs selbst festlegen, oder nach Start des ersten Spiels ändern. Für das Android Benutzerkonto sowie dem iOS Konto wurden jeweils eindeutige Namen zugewiesen, welche in jedem Spiel verwendet wurden: *anddipl2019* (Username innerhalb einer Android App) und *iosdipl2019* (Username innerhalb einer iOS App)

- **E-Mail**

  Für jedes der beiden Konten wurde eine eigene E-Mailadresse erstellt. Diese war für viele Registrierungen notwendig und je Festlegung eines Usernamen einzigartig erforderlich.

- **Testnachrichten**

  Um die Suche nach Kommunikationsartefakten zu erleichtern, wurde pro Spiel und Konto eine genaue Testnachricht mit folgendem Schema festgelegt: "Test[OS-Typ][appname]2019"
  Eine durch die Untersuchung aufgefundene Nachricht (auch ohne Zuordnung mit dem Usernamen) am Androidgerät, lautend auf "Testanddiplchess2019", beschreibt also eine vom Android Konto selbst gesendete Nachricht im Schachspiel. Ein gefundener Testauszug, lautend auf "Testiosdiplpool2019", stellt eine vom iOS Konto im Pool-Billard Spiel gesendete Nachricht dar. Sollten innerhalb von Spielen Gruppenchats zur Verfügung stehen, so wurde in diesen Gruppen die Nachricht "Test[OS-Typ][appname]group2019" verschickt, um auch solche Artefakte eindeutig zu einem Chatroom zuordnen zu können.

Die jeweiligen Apps wurden auf beiden Systemen in etwa gleichzeitig ausgeführt und die Registrierung der User wie oben beschrieben durchgeführt sowie dokumentiert. Je nach Spiel waren verschiedene Anforderungen zu erfüllen um die Chatmöglichkeit freischalten zu können. Dauerte die Freischaltung eines Chatrooms länger als eine Stunde oder war diese nur mit Geld zu erreichen, wurde das Spiel aus der Auswahl ausgeschlossen. Dies war leider bei den beliebten Applikationen "Angry Birds Evolution"[20] [21] und "Clash of Clans"[22] [23] der Fall.

Im Spiel wurden sich gegenseitig die zuvor festgelegten Nachrichten geschickt und der Zeitpunkt sowie ordnungsgemäße Empfang dokumentiert. Nachdem innerhalb aller zu analysierenden Applikationen die Testnachrichten verschickt bzw. empfangen wurden, war der nächste Schritt die Datenakquisition:

3.2.1. Android

Um einen stufenweisen Informationsgewinn je nach Datenextraktionsmethode nachzuweisen, wurde wie folgt vorgegangen:

1. **manuelle Extraktion - Dateitransfer**
   Das Gerät wurde mittels USB an den PC angeschlossen. Am Android Gerät wurde der Dateitransfer mittels MTP aktiviert. Am PC wurden alle - über die USB-Verbindung sichtbaren - Daten zur späteren Analyse abgespeichert.

2. **logische Extraktion - ADB Backup**
   Am PC wurden das Android SDK bzw. die Plattform Tools installiert. Mithilfe von ADB wurde wie in Kapitel 2.1.1 beschrieben eine vollständige Gerätesicherung vom Android Device auf dem Rechner transferiert. Mit dem Android Backup Extractor im "Android Backup Toolkit" in der Version 20180521 [24] wurde diese Sicherung aus dem verschlüsselten Archiv für die spätere Analyse entpackt.

3. **logische Extraktion - TWRP Backup**
   Wie in Kapitel 2.1.1 beschrieben, wurde das Smartphone in den Recovery Mode gebootet, in welchem TWRP als Custom Recovery Software bei der Vorbereitung des Geräts installiert war. TWRP bietet die Möglichkeit, direkt die vorhandenen Partitionen zu sichern. Zur Analyse wurde

[20] https://play.google.com/store/apps/details?id=com.rovio.tnt
[21] https://apps.apple.com/us/app/angry-birds-evolution/id1104911270
[22] https://play.google.com/store/apps/details?id=com.supercell.clashofclans
[23] https://apps.apple.com/us/app/clash-of-clans/id529479190
[24] https://sourceforge.net/p/android-backup-toolkit/

die /data Partition auf den PC extrahiert. Der Informationsgewinn gegenüber der ADB Methode besteht darin, dass die gesamte Partition (jedoch nur allokierte Bereiche) gesichert wird, und keine Android-Betriebssystem Schicht mehr für die Extraktion genutzt wird.

4. **physische Extraktion - DD**
   Mit einer Root Shell wurde mit dem Kommando "dd" die Partition /data blockweise auf den PC extrahiert. Der Informationsgewinn gegenüber der TWRP Methode besteht in der zusätzlichen, vollständigen Extraktion von nicht-allokierten Datenblöcken, welche die Möglichkeit zur Rekonstruktion gelöschter Daten bietet.

3.2.2. iOS

Beim iOS-Gerät wurden zwei verschiedene Arten einer Datenextraktion durchgeführt:

1. **logische Extraktion - Backup**
   Die logische Extraktion wurde mittels unverschlüsseltem und verschlüsseltem iTunes-Backup durchgeführt. Da sich das verschlüsselte Backup nur in einigen definierten Bereichen [48] zum unverschlüsselten unterscheiden sollte, wurde diese Behauptung auch betreffend der Kommunikationsdaten innerhalb der Apps überprüft. EntwicklerInnen wird jedoch die Möglichkeit gegeben, sensitive Dateien in den Applikationsdaten zu verschlüsseln und den Schlüssel dafür in der eigenen Keychain zu sichern. Beim unverschlüsselten Backup wird diese Keychain dann im Backup nicht berücksichtigt. [49, pp. 20-24] Es stellt sich bei diesem Vorgehen jedoch die Frage, ob diese Dateien überhaupt gesichert und nicht per Design ausgeschlossen werden sollen.

   An dieser Stelle kann bereits eines vorweg genommen werden: Die Behauptung hält auch in dieser Arbeit - es waren betreffend der Kommunikationsartefakte keine Unterschiede zwischen dem unverschlüsselten und dem verschlüsselten Backup erkennbar.

2. **physische Extraktion - Jailbreak und DD**
   Um für die forensische Analyse trotzdem Daten zu erhalten, welche nicht im Backup enthalten sind, weil diese von EntwicklerInnen beispielsweise explizit ausgenommen wurden [50], muss das gesamte Dateisystem extrahiert werden. Um dies zu ermöglichen, muss das Gerät mit einem Jailbreak versehen werden, um eigene Programme ausführen zu können. Nach einem erfolgreichen Ausbruch können Programme wie dd am Gerät ausgeführt werden - der gesamte Speicher des iOS Geräts kann nun auf den Computer bitweise übertragen werden. Dies ermöglicht die Analyse des gesamten Dateisystems bzw. Rekonstruktion von möglicherweise gelöschten Daten.

3.3. Laborumgebung

Die beiden Geräte wurden vor Beginn vollständig zurückgesetzt und gelöscht, um zufällige, alte Datenfunde bestmöglich zu vermeiden.
Der Bootloader des Android Geräts (Google Nexus 6) wurde in Hinblick auf eine spätere physische Datenakquisition geöffnet, das Gerät selbst mit einer Custom Recovery Software (TWRP) ausgestattet und mit der neuesten, vom Device noch unterstützten, Stock-ROM[25] Android Version bespielt. Das Apple iPhone 6 wurde auf die iOS Version 12.1.3 aktualisiert.

Folgende mobile Geräte wurden zur Analyse verwendet:

- **Google Nexus 6**
  Android 7.1.1
  Unlocked Bootloader / Root mit Nexus Root Toolkit v2.1.9 [26]
  TWRP 3.2.3 Recovery System [27]

- **Apple iPhone 6**
  iOS 12.1.3
  Jailbreak mit Unc0ver 3.3.8 [28]

Für die Analyse von Dateien, Backups bzw. Dumps wurden verschiedene Programme verwendet:

- **Android Backup Toolkit 20180521**[29]
  Mithilfe des Android Backup Toolkits lassen sich ADB-Backups bearbeiten. Backup Dateien können entschlüsselt werden und in einer weiterverwendbaren .tar Datei abgelegt werden.

- **Android Plattform Tools R26**[30]
  Die Android Plattform Tools beinhalten unter anderem ADB zur Erstellung von Backups.

- **Autopsy 4.12.0 / The Sleuth Kit** [31]
  Das grafische Interface von Sleuth Kit genannt Autopsy ermöglicht das Durchsuchen von physischen Dumps oder großen Ansammlungen von Dateien. Es können gelöschte Dateien wiederhergestellt, oder eine Volltextsuche inklusive einer Indexierung über alle Daten durchgeführt werden.

[25] Vom Hersteller selbst ausgelieferter und somit unterstützter Softwarestand.
[26] https://www.wugfresh.com/nrt/
[27] https://twrp.me/
[28] https://github.com/pwn20wndstuff/Undecimus
[29] https://sourceforge.net/p/android-backup-toolkit/
[30] https://developer.android.com/studio/releases/platform-tools
[31] https://www.sleuthkit.org/

- **DB-Browser for SQLite 3.11.2**[32]

  Mit diesem Tool können SQLite Datenbanken grafisch dargestellt und durchsucht werden. Es können direkt SQL Abfragen auf die Datenbankdatei durchgeführt werden.

- **iBackupBot 5.6.1**[33]

  Backups welche mit iTunes gemacht wurden, wurden damit durchsucht. Einzelne Domains oder Dateien können zur Analyse extrahiert werden.

- **iBackupViewer 4.12.0**[34]

  Ein Alternativprogramm zu iBackupBot.

- **PList Explorer 1.0**[35]

  Mit diesem Programm können Property Lists durchsucht und komfortabel dargestellt werden.

[32] https://sqlitebrowser.org/

[33] https://www.icopybot.com/itunes-backup-manager.htm

[34] http://www.imactools.com/

[35] www.ithmbconverter.com/plist/plistexplorer.zip

4. Auswertung

In diesem Kapitel werden Inhalt und Sinn der zur Analyse ausgewählten Applikationen kurz beschrieben, welche Verbreitung diese weltweit genießen und wie viel Aufwand es ist, sich zu registrieren und schlussendlich mit anderen SpielerInnen kommunizieren zu können. Danach werden die Ergebnisse der forensischen Analyse jeder Applikation detailliert vorgestellt. Am Ende des Kapitels werden Gemeinsamkeiten zwischen den Applikationen bzw. der jeweiligen Betriebssysteme dargestellt.

Wenn von "Unterschieden zwischen auf verschiedenem Wege gewonnenen Informationen oder Daten" geschrieben wird, so ist dies immer in Hinblick auf die Frage, ob und was zwei Personen miteinander kommuniziert haben, zu sehen. Andere, für diese Frage irrelevante Datenunterschiede, werden in diesem Vergleich nicht hervorgehoben.

4.1. 8 Ball Pool

8 Ball Pool ist ein klassisches Single- und Multiplayer Billard Spiel. Die Android Version wurde über 500 Millionen mal installiert, hat 15,9 Millionen Bewertungen und ist als Editors' Choice im Google Play Store gekennzeichnet. [36] Die Version für iOS hat bereits 1.3 Millionen Bewertungen und ist ebenfalls als Editors' Choice im App Store gekennzeichnet. [37]

In 8 Ball Pool registriert man sich zuerst mit einer Mailadresse, einem Nicknamen und legt ein Passwort fest. Danach können SpielerInnen über den Nicknamen gesucht und herausgefordert werden. Im Spiel selbst kann man nun mit dem Gegenüber chatten. Der Aufwand für die Herstellung der Chatbereitschaft lag bei dieser Applikation zwischen 5 und 10 Minuten.

[36] https://play.google.com/store/apps/details?id=com.miniclip.eightballpool&hl=en
[37] https://apps.apple.com/us/app/8-ball-pool/id543186831

4.1.1. Android

Die Android Version des Spiels "8 Ball Pool" brachte einen signifikanten Informationsunterschied in den Daten je nach gewählter Extraktionsmethode mit sich. Nur nach der Datenakquise mittels TWRP und DD waren Hinweise auf eine Kontaktaufnahme zu finden, nicht aber ein Chatverlauf.

```
[Pfad nach Extraktion mittels TWRP bzw. physischer Extraktion]
/data/data/com.miniclip.eightballpool/files/Contents/Documents/
friendsList2891014915.plist
```

In dieser Datei des Typs PLIST finden sich alle eingegangenen Freundschaften des Android Kontos wieder. Der Dateiname enthält die eigene UserID 2891014915. In dieser Liste findet sich auch das iOS Konto mit der UserID 2891013603 und Informationen wie viele Spiele gemeinsam gespielt wurden, wann das letzte Spiel durchgeführt und wann dieses Konto zuletzt online gesehen wurde. Die Frage nach dem Wann wird jeweils in der Form eines Unixtimestamps beantwortet.

```
[Pfad nach Extraktion mittels TWRP bzw. physischer Extraktion]
/data/data/com.miniclip.eightballpool\files\Contents\Documents\
__user_defaults__/addAsFriend.plist
```

Der Inhalt dieser Datei zeigt alle eingegangenen Freundschaften. In diesem Fall war der Inhalt "2891013603addAsFriendTimeKey2891014915" also [*UserIDiOSUser*]addAsFriendTimeKey[*eigeneUserID*].

4.1.2. iOS

In der iOS Version des Spiels "8 Ball Pool" war kein Informationsunterschied zwischen der logischen Extraktion mittels iTunes-Backup oder dem physischen Dump der Daten, betreffend der Frage nach einer stattgefundenen Kontaktaufnahme zwischen zwei SpielerInnen, erkennbar. Die analysierten Daten geben einen Hinweis auf eine Kontaktaufnahme, nicht aber einen Chatverlauf wieder.

```
[Pfad nach logischer Extraktion]
/com.miniclip.8ballpoolmult/Documents/friendsList2891013603.plist
bzw.
[Pfad nach physischer Extraktion]
/private/var/mobile/Containers/Data/Application/
7C511F21-7F2A-4212-A611-76F1341B0BB7/friendsList2891013603.plist
```

In dieser Datei des Typs PLIST finden sich alle FreundInnen des iOS Kontos wieder. Der Dateiname enthält die eigene UserID 2891013603. In dieser Liste findet sich auch das Android Konto mit der UserID 2891014915 und die Informationen wie viele Spiele gemeinsam gespielt wurden, wann das letzte Spiel durchgeführt und wann dieses Konto zuletzt online gesehen wurde. Die Frage nach dem Wann wird jeweils in der Form eines Unixtimestamps beantwortet.

```
/com.miniclip.8ballpoolmult/Documents/__user_defaults__/
addAsFriend.plist
bzw.
/private/var/mobile/Containers/Data/Application/
7C511F21-7F2A-4212-A611-76F1341B0BB7/Documents/
__user_defaults__/addAsFriend.plist
```

Der Inhalt dieser Datei ergibt alle eingegangenen Freundschaften. In diesem Fall war der Inhalt "2891014915addAsFriendTimeKey2891013603" also [*UserIDAndroidUser*]addAsFriendTimeKey[*eigeneUserID*].

4.1.3. Unterschiede bzw. Gemeinsamkeiten

In beiden Versionen wurden die selben Informationen an gleicher Stelle gefunden. Nur in der iOS Version waren diese Daten auch nach der logischen Extraktion bereits vorhanden. In keiner Version konnten Chatverläufe gefunden, bzw. wiederhergestellt werden. Eine Kontaktaufnahme zwischen den SpielerInnen wurde jedoch nachgewiesen.

4.2. ChessTime

Chess Time ist ein klassisches Single- und Multiplayer Schach Spiel. Die Android Version wurde über 1 Million mal installiert und hat rund 41.000 Bewertungen im Google Play Store. [38] Die iOS Version des Spiels hat nur rund 700 Bewertungen im App Store. [39]

In Chess Time registriert man sich ebenfalls anfangs mit einer Mailadresse, wählt einen Nicknamen und legt ein Passwort fest. Danach können andere SpielerInnen über den gewählten Namen gesucht und herausgefordert werden. Im Spiel selbst kann man nun mit dem Gegenüber chatten. Der Aufwand für die Herstellung der Chatbereitschaft lag bei dieser Applikation ebenfalls zwischen 5 und 10 Minuten.

[38] https://play.google.com/store/apps/details?id=com.haptic.chesstime&hl=en
[39] https://apps.apple.com/us/app/chess-time-multiplayer-chess/id455602152

4.2.1. Android

Die Android Version des Spiels "ChessTime" ermöglichte ab der logischen Extraktionsmethode ADB-Backup eine Rekonstruktion des Chatverlaufs. Nach einer Extraktion mittels TWRP wurden weitere Merkmale einer Kontaktaufnahme (wie offene Spiele bzw. Spielabbrüche) sichtbar.

```
[Pfad nach Extraktion mittels ADB]
/apps/com.haptic.chesstime/f/36873381/_gchat_36873381.json
bzw.
[Pfad nach Extraktion mittels TWRP bzw. physischer Extraktion]
/data/data/com.haptic.chesstime/files/36873381/_gchat_36873381.json
```

In dieser Datei ist ein kompletter Chatverlauf abgebildet. Auszug aus der Datei:

```
"C1":"{\"cid\":\"104881545\",\"txt\":\"Testanddiplchess2019\",
\"dte\":\"1552303079000\",\"fid\":\"2703631\"}",
```

- fid = friendID - UserID des Senders
- txt = Der gesendete Text
- dte = unix Timestamp (mit 13 Stellen) der gesendeten bzw. empfangenen Nachricht

Ordnet man diese Datei nach dem Schlüssel *dte*, so erhält man einen chronologischen Chatverlauf.

Die weiteren Artefakte wurden erst ab der Extraktionsmethode mittels TWRP auffindbar:

```
[Pfad nach Extraktion mittels TWRP bzw. physischer Extraktion]
/data/data/com.haptic.chesstime/files/offlinestorage.json
```

In dieser Datei sind Metadaten wie Name, ID, Ranghöhe und Countrycode der anderen GegnerInnen aller offenen Spiele abgespeichert.

```
[Pfad nach Extraktion mittels TWRP bzw. physischer Extraktion]
/data/data/com.google.android.gms/files/fcm_queued_messages.ldb/
000029.ldb
```

In dieser Android Push-Notification Cache Datei sieht man, dass ein aktives Spiel mit einem anderen User gecancellt wurde. Auszug aus der Datei:

```
msg, !'  against iosdipl2019 has been cancelled
```

4.2.2. iOS

In der iOS Version des Spiels "ChessTime" war kein Informationsunterschied zwischen der logischen Extraktion oder dem physischen Dump der Daten, betreffend der Frage nach der Kontaktaufnahme zwischen zwei SpielerInnen, erkennbar. Die Gesamtheit der extrahierten Daten geben einen Hinweis auf eine Kontaktaufnahme, nicht aber einen Chatverlauf wieder.

```
[Pfad nach logischer Extraktion]
/com.hapticapps.chess/Documents/pageCache_Friendlist_2703630.plist
bzw.
[Pfad nach physischer Extraktion]
/private/var/mobile/Containers/Data/Application/
9D0FC3AF-3445-4591-B738-FFBCFC30F665/Documents/
pageCache_Friendlist_2703630.plist
```

Aus dieser Datei wurde folgender Inhalt entnommen:

```
"countryCode":"AT","id":2703631,"name":"anddipl2019"
```

Hier lässt sich das Android Konto mit dem Nicknamen und der zugehörigen UserID 2703631, sowie dem Staat in welchem es registriert wurde, entnehmen. Dem Dateinamen zu folge ist dieses Konto in der eigenen Freundesliste. Außerdem gibt dieser Name auch die eigene UserID 2703630 preis. Aufgrund der vergebenen Nummern, bei fast gleichzeitiger Registrierung, kann darauf geschlossen werden, dass diese in einfacher, aufsteigender Reihenfolge vergeben werden.

4.2.3. Unterschiede bzw. Gemeinsamkeiten

Je nach Plattform ist die App ChessTime mehr oder weniger forensisch interessant. Nur in der Android Version konnte ein kompletter Chatverlauf rekonstruiert werden. Eine Freundesliste konnte jedoch nur in den Daten der iOS-Version gefunden werden.

4.3. Dice with Buddies

Dice with Buddies ist ein Multiplayer Würfel Spiel. Die Android Version wurde über 5 Millionen mal installiert und hat circa 162.000 Bewertungen Google Play Store. [40] Die Version für iOS hat rund 19.400 Bewertungen im App Store. [41]

In Dice with Buddies muss man sich anfangs durch ein Tutorial spielen. Nach dem Tutorial bekommt man einen Nicknamen zugewiesen, welchen man anschließend ändern kann. Es ist nicht notwendig sich mit einer Mailadresse zu registrieren. Andere SpielerInnen können jetzt über deren Nicknamen gesucht werden. Nun kann man auch außerhalb eines Spiels mit dem Gegenüber chatten. Der Aufwand für die Herstellung der Chatbereitschaft lag bei dieser Applikation zwischen 10 und 15 Minuten.

4.3.1. Android

Die Android Version des Spiels "Dice with Buddies" ergab, ab der Extraktionsmethode mittels ADB, Hinweise auf am Gerät gespeicherte Chatverläufe bzw. Freundeslisten. Die extrahierten Daten beinhalteten jedoch trotzdem keine konkreten Kommunikationsdaten. Es wurden Dateien mit hinweisgebenden Namen gefunden, welche jedoch nicht ausgelesen werden konnten. Es handelt sich um Binärdateien mit folgenden Namen:

```
[Pfad nach logischer Extraktion mittels ADB]
/apps/com.withbuddies.dice.free/f/

[Pfad nach Extraktion mittels TWRP bzw. physischer Extraktion]
/data/data/com.withbuddies.dice.free/files/

[Dateinamen]
BuddiesList-v2.dat
BuddyList-v1.dat
BuddiesMessages.dat
```

[40] https://play.google.com/store/apps/details?id=com.withbuddies.dice.free&hl=en
[41] https://apps.apple.com/us/app/dice-with-buddies-social-game/id432750508

Aufgrund der Benennung dieser Dateien und der jeweils aufgefundenen Dateigröße wird vermutet, dass sich der Chatverlauf (BuddiesMessages.dat) sowie die Freundesliste (BuddiesList-v2.dat bzw. BuddyList-v1.dat) am Gerät befinden. Es müsste jedoch die Datenstruktur innerhalb der Binärdateien durch Reverse Engineering herausgefunden werden um diese Vermutung zu bestätigen.

```
[Pfad nach Extraktion mittels TWRP bzw. physischer Extraktion]
/data/data/com.google.android.gms/files/fcm_queued_messages.ldb/
000018.ldb

[Inhalt]
Iosdipl2019 [...] eine Nachricht gesendet [...]
```

In dieser Push-Notification Cache Datei sieht man, dass einem das iOS Konto eine Nachricht geschickt hat, nicht aber den Inhalt.

4.3.2. iOS

In der iOS Version des Spiels "Dice with Buddies" war ein Unterschied zwischen der logischen Extraktion oder dem physischen Dump der Daten ersichtlich. Im iTunes-Backup waren keinerlei hinweisgebenden Spuren enthalten. Die physisch extrahierten Daten waren umfangreicher, geben jedoch trotzdem keinen konkreten Hinweis auf eine Kontaktaufnahme oder einen Chatverlauf. Es wurden lediglich Dateien mit hinweisgebenden Namen gefunden, welche jedoch nicht ausgelesen werden konnten. Es handelt sich um Binärdateien mit folgenden Namen:

```
[Pfad nach physischer Extraktion]
/private/var/mobile/Containers/Data/Application/
38319BF4-8ABF-43AE-9E26-04E0F863B90A/Documents/

[Dateinamen]
BuddiesList-v2.dat
BuddyList-v1.dat
BuddiesMessages.dat
```

Aufgrund der Benennung dieser Dateien und der jeweils aufgefundenen Dateigröße wird vermutet, dass sich der Chatverlauf (BuddiesMessages.dat) sowie die Freundesliste (BuddiesList-v2.dat bzw. BuddyList-v1.dat) am Gerät befinden. Es müsste jedoch die Datenstruktur innerhalb der Binärdateien durch Reverse Engineering herausgefunden werden um diese Vermutung zu bestätigen.

4.3.3. Unterschiede bzw. Gemeinsamkeiten

Die Strukturen und Namen der abgelegten Daten der Android Version gleichen der iOS Version. Da jedoch nur interessant benannte Binärdateien gefunden wurden, müssten diese per Reverse Engineering noch lesbar gemacht werden. Es ist aufgrund der Ähnlichkeiten naheliegend, dass wenn Binärdateien einer Version dieses Spiels entsprechend analysiert wurden, diese auch am anderen Betriebssystem in der gleichen oder zumindest sehr ähnlichen Weise ausgelesen werden können.

Die erhaltenen Informationen auf eine stattgefundene Kommunikation durch die forensische Analyse waren bei beiden Version jedoch minimal bis nicht vorhanden. Nur eine auf Android gecachte Push-Benachrichtigung ergab einen Hinweis auf Kontaktaufnahme.

4.4. Plato

Plato ist an sich kein eigenes Spiel, sondern eine Chat Plattform mit vielen integrierten Mini-Spielen. Man kann sich über die App vernetzen und kurzweilige Spiele gegeneinander spielen. Die Android Version wurde über 5 Millionen mal installiert und hat circa 48.500 Bewertungen Google Play Store. [42] Die Version für iOS hat rund 39.400 Bewertungen im App Store. [43]

Nach der Installation von Plato bekommt man eine ID zugewiesen, welche man jedoch selbst ändern kann. Nach einer Suche von IDs anderer SpielerInnen kann sofort gechattet bzw. gespielt werden. In dieser App ist es auch möglich Gruppen zu erstellen und mit mehreren Personen gleichzeitig zu chatten. Der Aufwand für die Herstellung der Chatbereitschaft lag bei dieser Applikation bei maximal 5 Minuten.

4.4.1. Android

Die Android Version von "Plato" lieferte erst bei einer Extraktion mittels TWRP bzw. DD Hinweise auf eine stattgefundene Kommunikation. Es wurden unter anderem in einer SQLite Datenbank der komplette Chatverlauf und auch andere Kommunikationsmetadaten aufgefunden.

```
[Pfad nach Extraktion mittels TWRP bzw. physischer Extraktion]
/data/data/com.plato.android/shared_prefs/androidMe.xml

[Inhalt]
```

[42] https://play.google.com/store/apps/details?id=com.plato.android&hl=en
[43] https://apps.apple.com/us/app/plato-find-fun/id1054747306

```
<map>
<string name="mePublicUUIDKey">kut4wtyzolex-1ltu89sp4pz0t</string>
<string name="meUUIDKey">29bbanqkoikvq-gvt4hcst3nwd</string>
<int name="meDeviceIDKey" value="16" />
<string name="mePlatoIdKey">anddipl2019</string>
<long name="meAuthTokenKey" value="-897142273392624035" />
</map>
```

- meUUIDKey: Dies ist die im Plato System einzigartige ID für den Android User.
- mePlatoIdKey: Der vergebene Nickname des Android Users.

```
[Pfad nach Extraktion mittels TWRP bzw. physischer Extraktion]
/data/data/com.plato.android/databases/d
```

Die Datei mit dem einfachen Namen "d" ist eine SQLite Datenbank. In dieser finden sich einige Tabellen mit Inhalt, welche auf eine Kommunikation mit dem Android Konto hinweisen bzw. diese sogar komplett darstellt. Die gewählten Spaltennamen innerhalb der Tabellen sind nicht aussagekräftig. Nur anhand des Inhalts konnte hier eine Interpretation der Daten erfolgen.

Der gesamte Chatverlauf findet sich in der Tabelle "t0":

d	f	j	k
Testiosdiplplato2019	117qvcuhe7znd-2i8ymqjz8q1nd	1552311696011	
Testanddiplplato2019		1552311709515	
Testiosdiplplatogroup2019	117qvcuhe7znd-2i8ymqjz8q1nd	1552311964467	1unea62sh9plm-12lqo2zy01oi4
Testanddiplplatogroup2019	29bbanqkoikvq-gvt4hcst3nwd	1552311982751	1unea62sh9plm-12lqo2zy01oi4

Tabelle 4.1.: Plato Android Version: SQLite Tabelle mit Chatverlauf

- d: Die gesendeten oder empfangenen Nachrichten.
- f: UserID des sendenden Kontos. Wenn das Feld leer ist, dann interpretiert die App das offensichtlich als eigene Nachricht. Die eigene ID steht nur in diesem Feld wenn es eine eigene Nachricht an eine Gruppe ist. (siehe 4. Zeile)
- j: Unixtimestamp (13 stellig). Zeitpunkt der Nachricht.
- k: Systemweite Gruppen-ID einer Chatgruppe.

Eine UserID zu Nickname Zuordnung findet man in "t4":

a	c	ut
117qvcuhe7znd-2i8ymqjz8q1nd	iosdipl2019	1552311680552

Tabelle 4.2.: Plato Android Version: SQLite Tabelle mit Zuordnung UserID zu Nickname

- a: Systemweite UserID des Kontos (vgl. Tabelle 4.1)
- c: vergebener Nickname des Kontos
- ut: Unixtimestamp (13 stellig). Zeitpunkt der Erstellung des Kontos.

```
[Pfad nach Extraktion mittels physischer Extraktion]
/data/data/com.plato.android/databases/d-journal
```

Die gelöschte Datei "d-journal" ist eine Journal Datei der SQLite Datenbank. Zum Zeitpunkt der Extraktion befanden sich hier die selben Einträge wie in der Datenbank selbst.

4.4.2. iOS

In der iOS Version der Chat Plattform "Plato" war kein Informationsunterschied zwischen der logischen Extraktion oder dem physischen Dump der Daten ersichtlich. Bereits im iTunes-Backup wurde in einer SQLite Datenbank der komplette Chatverlauf und auch andere Kommunikationsmetadaten aufgefunden.

```
[Pfad nach logischer Extraktion]
/com.platoapp.Plato/Documents/plato.sqlite
bzw.
[Pfad nach physischer Extraktion]
/private/var/mobile/Containers/Data/Application/
62593F3D-DDB7-4648-B7E3-5597CDB06E9F/Documents/plato.sqlite
```

In dieser SQLite Datenbank finden sich einige Tabellen mit Inhalt, welche auf eine Kommunikation mit dem Android Konto hinweisen oder diese sogar komplett darstellt.

Datenauszug Tabelle u_43FA58CA_72B7_4239_A4D9_B0B82C22AED9_conversations:

name	addressee_id	time_stamp
anddipl2019	u-29bbanqkoikvq-gvt4hcst3nwd	1552311921
Diplplatogroup2019	p-1unea62sh9plm-12lqo2zy01oi4	1552311983

Tabelle 4.3.: Plato iOS Version: SQLite Tabelle mit Zuordnung UserID zu Nickname/Gruppe

- name: Name des Kontos bzw. der Chatgruppe
- addressee_id: interne UUID des Kontos bzw. Gruppe
- time_stamp: Unixzeitstempel des ersten Kontakts

Datenauszug Tabelle u_43FA58CA_72B7_4239_A4D9_B0B82C22AED9_messages:

data	sender	time	room_id
Testiosdiplplato2019	u-117qvcuhe7znd-2i8ymqjz8q1nd	1552311695848	
Testanddiplplato2019	u-29bbanqkoikvq-gvt4hcst3nwd	1552311709952	
Testiosdiplplato2019	u-117qvcuhe7znd-2i8ymqjz8q1nd	1552311806401	
Testiosdiplplatogroup2019	u-117qvcuhe7znd-2i8ymqjz8q1nd	1552311964276	p-1unea62sh9plm-12lqo2zy01oi4
Testanddiplplatogroup2019	u-kut4wtyzolex-1ltu89sp4pz0t	1552311983006	p-1unea62sh9plm-12lqo2zy01oi4

Tabelle 4.4.: Plato iOS Version: SQLite Tabelle mit Chatverlauf

- data: Die verschickten bzw. empfangenen Nachrichten.
- sender: Absender ID der Nachricht (addressee_id in Tabelle u_43FA58CA_72B7_4239_A4D9_B0B82C22AED9_conversations).
  Auffällig war der Unterschied, dass in der Android Version ein leeres Feld gleichzeitig eine selbst verschickte Nachricht darstellte, in der hier gezeigten iOS Version ist die eigene ID stets eingetragen.
- time: Unixzeitstempel des Vorgangs
- room_id: ID der Chatgruppe (addressee_id in Tabelle u_43FA58CA_72B7_4239_A4D9_B0B82C22AED9_conversations)

4.4.3. Unterschiede bzw. Gemeinsamkeiten

In beiden Versionen von "Plato" konnte der gesamte Chatverlauf rekonstruiert werden. Bei der Android Version dieser App war eine Extraktion ab dem Level TWRP notwendig, die iOS Version sicherte die notwendigen Daten bereits im normalen Backup ab. Die entsprechenden Datenbanken sind in der internen Benennung grob unterschiedlich, vom Aufbau jedoch sehr ähnlich. Die Benennung der Dateien bzw. der internen Datenbankstrukturen in der Android Version weisen möglicherweise auf eine schlichte Obfuskation hin, welche in der iOS Version nicht vorhanden ist.

4.5. Playstation Messages

Playstation Messages ist eigentlich eine reine Chat Plattform. Trotzdem wurde die App innerhalb dieser Arbeit durch ihre vorwiegende Nutzung innerhalb von Spielenetzwerken und der in Kapitel 1 entsprechend vorgestellten Berichte über deren kriminellen Nutzung, zur Analyse ausgewählt. Die Android Version wurde über 10 Millionen mal installiert und hat circa 96.400 Bewertungen Google Play Store. [44] Die Version für iOS hat rund 3.600 Bewertungen im App Store. [45]

Um bei Playstation Messages teilnehmen zu können muss ein Playstation Konto [46] eröffnet werden. Zur Registrierung wird eine Mailadresse und ein Passwort benötigt. Nach Bestätigung der E-Mailadresse kann man sich einen Nicknamen frei definieren. Über eine Suche kann man andere SpielerInnen finden und direkt anschreiben. Der Aufwand für die Herstellung der Chatbereitschaft lag bei dieser Applikation bei 10 Minuten.

4.5.1. Android

Die Android Version der "Playstation Messages" App brachte ab der Extraktion mittels TWRP Hinweise auf am Gerät gespeicherte Chatverläufe bzw. Freundeslisten. Die extrahierten Daten ergaben jedoch keine konkreten Kommunikationsdaten. Es wurden Dateien mit hinweisgebenden Namen gefunden, welche jedoch nicht ausgelesen werden konnten.

```
[Pfad nach Extraktion mittels TWRP bzw. physischer Extraktion]
/data/data/com.playstation.mobilemessenger/databases/messages
```

[44] https://play.google.com/store/apps/details?id=com.playstation.mobilemessenger&hl=en
[45] https://apps.apple.com/us/app/playstation-messages/id1053285387
[46] https://id.sonyentertainmentnetwork.com/create_account/

Es dürfte sich bei dieser Datei um eine SQLite Datenbank handeln, welche jedoch verschlüsselt ist. In der vollständigen Extraktion mittels DD konnte noch eine gelöschte Datei sichergestellt werden welche Hinweise auf die Verschlüsselung enthält:

```
[Pfad nach physischer Extraktion mittels DD]
/data/data/com.playstation.mobilemessenger/databases/messages-journal

[Inhalt]
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
<string name="1"></string>
<string name="KEY">pTm677s3osFQluZ5lN6MDw==</string>
</map>
```

Auch die vorhandene Datei "/app/com.playstation.mobilemessenger-1/lib/armlibsqlcipher.so" weist auf den Einsatz einer Verschlüsselung mit SQLCipher hin. Verschiedenste Versuche die Datenbank mit dem aufgefundenen Hinweis zu entschlüsseln schlugen jedoch fehl. Ein Reverse-Engineering der Applikation selbst, könnte weitere Hinweise auf die konkrete Verwendung dieser Verschlüsselung liefern.

4.5.2. iOS

In der iOS Version der "Playstation Messages" App war ein Informationsunterschied zwischen der logischen Extraktion und dem physischen Dump der Daten ersichtlich. Nach der physischen Datenextraktion konnte zwar kein Chatverlauf rekonstruiert werden, ein Hinweis auf eine Kontaktaufnahme konnte den Daten jedoch entnommen werden.

```
[Pfad nach physischer Extraktion]
/private/var/mobile/Containers/Data/Application/
7867A4AA-8802-4B23-A018-6EE98186C3AE/Library/Caches/
com.playstation.eu.mobilemessages/Cache.db

[Inhalt Tabelle "cfurl_cache_receiver_data"]
[{"onlineId":"anddipl2019","accountId":"378735757002354429",
[...] "languagesUsed":["de"], [...] ,"friendRelation":"no", [...]
"blocking":false,"following":false}],[...]
```

In dieser SQLite Datenbank findet sich die Tabelle "cfurl_cache_receiver_data". In dieser Tabelle können die letzten Abfragen anderer Profile entnommen werden. Konkret sieht man die AccountID bzw. den Accountnamen des Android Kontos und mit diesem eine Freundschaft eingegangen wurde oder geblockt ist.

4.5.3. Unterschiede bzw. Gemeinsamkeiten

Die App "Playstation Messages" unterscheidet sich in der Android Version grob von der iOS Version. Es wurden unterschiedlichste Hinweise in beiden Versionen gefunden. Nur in der Android Version wird vermutet, dass der Chatverlauf verschlüsselt abgespeichert wird - die iOS Version gibt dagegen zumindest konkrete Hinweise auf eine Bekanntschaft der beiden SpielerInnen.

4.6. QuizUp

QuizUp ist ein Multiplayer Wissensspiel. Die KontrahentInnen müssen Fragen jeweils richtig beantworten und sammeln so Punkte. Die Android Version wurde über 10 Millionen mal installiert und hat circa 662.000 Bewertungen Google Play Store. [47] Die Version für iOS hat rund 7.900 Bewertungen im App Store. [48]

In QuizUp registriert man sich mit einer Mailadresse und legt ein Passwort sowie einen Nicknamen fest. Danach können andere SpielerInnen über deren Nicknamen gesucht werden. Anderen Konten muss für die Freischaltung der Chatmöglichkeit gefolgt werden. Der Aufwand für die Herstellung der Chatbereitschaft lag bei dieser Applikation bei 5 Minuten.

4.6.1. Android

Die Android Version des Spiels "QuizUp" ermöglichte ab der Extraktionsmethode mittels TWRP die Rekonstruktion von am Gerät gespeicherten Chatverläufen. Es konnten zumindest alle eingehenden Nachrichten rekonstruiert werden.

```
[Pfad nach Extraktion mittels TWRP bzw. physischer Extraktion]
/data/data/com.quizup.core/cache/store/
cache-conversations/cache-conversations_921687115.0
```

[47] https://play.google.com/store/apps/details?id=com.quizup.core&hl=en
[48] https://apps.apple.com/us/app/quizup/id718421443

```
und

/data/data/com.quizup.core/cache/store/
cache-chat-history/cache-chat-history_2117137294.0
```

In diesen beiden Dateien wurden jeweils die selben eingehenden Nachrichten abgespeichert. Zusätzliche Attribute verraten die genaue Uhrzeit, ob die Nachricht gelesen wurde und die genauen PlayerIDs beider Konten:

```
[Auszug der Daten]
"created":"2019-03-11T12:59:08.741","from_player":
"1638384774936530366" [...] "read":false,
"text":"Testiosdiplquiz2019","to_player":"1638381904111600061"
```

Für diese beiden Dateien wurden zusätzlich gleichnamige aber gelöschte .tmp Dateien wiederhergestellt. Diese Dateien hatten genau den gleichen Inhalt:

```
[Pfad nach physischer Extraktion mittels DD]
/data/data/com.quizup.core/cache/store/
cache-conversations/cache-conversations_921687115.0.tmp

und

/data/data/com.quizup.core/cache/store/
cache-chat-history/cache-chat-history_2117137294.0.tmp
```

Auch diese weiteren Dateien zeigten eingehenden Nachrichten:

```
[Pfad nach Extraktion mittels TWRP bzw. physischer Extraktion]
/media/0/Android/data/com.quizup.core/cache/responses/
09f7e939e44fc57cf469fe001d196e7c.1

/media/0/Android/data/com.quizup.core/cache/responses/
836474d46b7b9c9be85efc24d8287b51.1

/media/0/Android/data/com.quizup.core/cache/responses/
92263a7022ce8ea52d77c71f1431fbb0.1/0
```

4.6.2. iOS

In der iOS Version des Spiels "QuizUp" war kein Informationsunterschied zwischen der logischen Extraktion oder dem physischen Dump der Daten betreffend der Frage nach der Kontaktaufnahme zwischen zwei SpielerInnen zu finden. Die extrahierten Daten geben einen Hinweis auf eine Bekanntschaft der beiden SpielerInnen, nicht aber einen Chatverlauf wieder.

```
[Pfad nach logischer Extraktion]
/com.plainvanillacorp.quizup/Library/Preferences/
com.plainvanillacorp.quizup.plist

[Pfad nach physischer Extraktion]
/private/var/mobile/Containers/Data/Application/
78433A8B-9A10-450D-8E17-63E9CF2147D1/Library/Preferences/
com.plainvanillacorp.quizup.plist
```

In dieser PLIST Datei finden sich innerhalb eines Binärblocks die Usernamen der beiden Konten inklusive eines Datums wieder. Um diese Zeit folgten die beiden Profile einander.

4.6.3. Unterschiede bzw. Gemeinsamkeiten

Die beiden Versionen des Spiels "QuizUp" könnten in deren Datenhaltung unterschiedlicher nicht sein. In der Android Version konnten nach einer physischen Extraktion alle eingehenden Nachrichten wiederhergestellt werden - diese wurden sogar mehrmals in vielen verschiedenen Dateien abgelegt. In der iOS Version konnte nur eine Bekanntschaft zwischen den SpielerInnen nachgewiesen werden konnte - diese dafür bereits ab der logischen Extraktion mittels Backup.

4.7. Scrabble

Scrabble ist ein Multiplayer Spiel in dem man aus vorhandenen Buchstaben Wörter legen muss. Die Android Version wurde über 10 Millionen mal installiert und hat circa 324.700 Bewertungen Google Play Store. [49] Die Version für iOS hat rund 135.100 Bewertungen im App Store. [50]

[49] https://play.google.com/store/apps/details?id=com.ea.game.scrabblemattel_bv&hl=en
[50] https://apps.apple.com/us/app/scrabble/id501724085

Die untersuchte Scrabble-Version ist eine Applikation der Firma Electronic Arts. Um in diesem Spiel mit anderen Personen spielen bzw. chatten zu können, muss es mit einem Facebook- oder Electronic-Arts-Konto verbunden werden. Bei der Anlage eines Electronic-Arts-Kontos [51] wird ein Verifizierungscode an die angegebene E-Mailadresse verschickt um diese zu bestätigen.
In der mobilen App selbst kann man keine anderen SpielerInnen suchen. Dies muss über den Electronic Arts Desktop Client "Origin" [52] passieren. In diesem Client kann man andere SpielerInnen suchen und Freundschaftsanfragen verschicken, annehmen oder ablehnen. Erst danach tauchen diese Freundschaften auch im mobilen Client auf. Nun kann man ein Spiel mit dem Gegenüber starten. Im Spiel selbst ist ein Chat jetzt möglich. Der Aufwand für die Herstellung der Chatbereitschaft lag bei dieser Applikation bei circa 20 Minuten.

4.7.1. Android

In der Android Version des Spiels "Scrabble" war kein Informationsunterschied zwischen einer logischen Extraktion oder dem physischen Dump der Daten (betreffend der Frage nach der Kontaktaufnahme zwischen zwei SpielerInnen) zu finden. Die extrahierten Daten geben bereits ab der Extraktion mittels einfachem Dateitransfer einen Hinweis auf eine Kontaktaufnahme, nicht aber einen Chatverlauf wieder.

```
[Pfad nach manueller Extraktion mittels einfachem Dateitransfer]
/Android/data/com.ea.game.scrabblemattel_bv/files/UserCache/
users.json

[Pfad nach logischer Extraktion mittels ADB]
/apps/com.ea.game.scrabblemattel_bv/files/UserCache/
users.json

[Pfad nach Extraktion mittels TWRP bzw. physischer Extraktion]
/data/data/com.ea.game.scrabblemattel_bv/files/UserCache/
users.json
```

In der Datei "users.json" findet man alle dem Spiel bekannten SpielerInnern. Es finden sich die Daten des eigenen Kontos, sowie alle jemals kontaktierten User. Zusätzliche Attribute umfassen die eindeutigen User-IDs sowie ein Datum wann sich die SpielerInnen registriert haben.

[51] https://www.ea.com/register
[52] https://www.origin.com

```
[Pfad nach manueller Extraktion mittels einfachem Dateitransfer]
/Android/data/com.ea.game.scrabblemattel_bv/files/MatchesCache/
SessionOrigin.json

[Pfad nach logischer Extraktion mittels ADB]
/apps/com.ea.game.scrabblemattel_bv/files/MatchesCache/
SessionOrigin.json

[Pfad nach Extraktion mittels TWRP bzw. physischer Extraktion]
/data/data/com.ea.game.scrabblemattel_bv/files/MatchesCache/
SessionOrigin.json
```

Dieser "SessionOrigin.json" Datei kann man Metadaten der zuletzt gespielten Matches entnehmen. Der oder die GegnerIn wird durch die "MayhemUserID", welche zuvor im File users.json ersichtlich war, identifiziert. Auch erkennt man über den Key "lastUpdated" einen UnixTimestamp, welcher den letzten Kontaktzeitpunkt zwischen den KontrahentInnen identifiziert. Ob in diesem Spiel auch anders als durch reines Buchstabenlegen kommuniziert wurde, ist nicht ersichtlich.

4.7.2. iOS

In der iOS Version des Spiels "Scrabble" war kein Informationsunterschied zwischen der logischen Extraktion oder dem physischen Dump gegeben. Die extrahierten Daten geben einen konkreten Hinweis auf eine Kontaktaufnahme, nicht aber einen Chatverlauf wieder.

```
[Pfad nach logischer Extraktion]
/ea.scrabblefreemattel.bv/Documents/UserCache/users.json

[Pfad nach physischer Extraktion]
/private/var/mobile/Containers/Data/Application/
6C60FCA5-B8F6-4FD8-A961-94DA45276D34/Documents/UserCache/
users.json
```

In users.json findet man alle dem Spiel bekannten SpielerInnen. Es finden sich die Daten des eigenen sowie alle jemals kontaktierten User. Zusätzliche Attibute umfassen die eindeutigen User-IDs sowie einem Datum, wann sich das Konto registriert hat.

```
[Pfad nach logischer Extraktion]
/ea.scrabblefreemattel.bv/Documents/MatchesCache/
SessionOrigin.json

[Pfad nach physischer Extraktion]
/private/var/mobile/Containers/Data/Application/
6C60FCA5-B8F6-4FD8-A961-94DA45276D34/Documents/MatchesCache/
SessionOrigin.json
```

Der Datei "SessionOrigin.json" kann man Metadaten der zuletzt gespielten Matches entnehmen. Der oder die GegnerIn wird durch die "MayhemUserID", welche zuvor im File users.json ersichtlich war, identifiziert. Auch erkennt man über den Key "lastUpdated" einen UnixTimestamp, welcher den letzten Kontaktzeitpunkt zwischen den KontrahentInnen identifiziert. Ob in diesem Spiel auch anders als durch Buchstabenlegen kommuniziert wurde, ist nicht ersichtlich.

4.7.3. Unterschiede bzw. Gemeinsamkeiten

Die beiden Betriebssystemversionen des Spiels "Scrabble" könnten gleicher nicht sein. Die abgelegten Daten sowie deren Speicherorte sind nahezu ident. Alle Hinweise auf Kontaktaufnahme konnten bereits nach einer logischen Extraktion der Daten aufgefunden werden. Es konnte jedoch kein konkreter Chatverlauf rekonstruiert werden.

4.8. Words with Friends

Words with Friends ist ein Multiplayer Spiel in dem man wie bei Scrabble aus vorhandenen Buchstaben Wörter legen muss. Die Android Version wurde über 50 Millionen mal installiert und hat circa 1,5 Millionen Bewertungen Google Play Store. [53] Die Version für iOS hat rund 27.300 Bewertungen im App Store. [54]

Die Herstellung der Chatbereitschaft ist bei der App "Words with Friends" deutlich einfacher und schneller möglich als bei der vergleichbaren App "Scrabble". Zuerst muss man sich innerhalb der App mit einer Mailadresse registrieren und einen Nickname auswählen. Über eine Suche können andere SpielerInnen herausgefordert werden. Nachdem jede/r SpielerIn einen Zug gespielt hat, ist ein Chat innerhalb des

[53] https://play.google.com/store/apps/details?id=com.zynga.words&hl=en
[54] https://apps.apple.com/us/app/words-with-friends-classic/id321916506

Spiels möglich. Der Aufwand für die Herstellung der Chatbereitschaft lag bei dieser Applikation bei 10 Minuten.

4.8.1. Android

Die Android Version des Spiels "Words with Friends" ermöglichte ab der Extraktionsmethode ADB-Backup eine Rekonstruktion des Chatverlaufs. Nach einer Extraktion mittels TWRP wurden weitere Merkmale einer Kontaktaufnahme (wie offene Spiele oder Einladungen zu Spielen) sichtbar.

```
[Pfad nach logischer Extraktion mittels ADB]
/apps/com.zynga.wwf2.free/db/wf_database.sqlite

[Pfad nach Extraktion mittels TWRP bzw. physischer Extraktion]
/data/data/com.zynga.wwf2.free/databases/wf_database.sqlite
```

In dieser SQLite Datenbank findet sich in der Tabelle "users" eine Auflistung aller diesem Konto bekannten SpielerInnen.

Auszug aus der Tabelle "users":

name	email_address	created_at	zynga_account_id
Anddipl2019	anddipl2019@gmail.com	2019-03-07T16:06:53Z	74736282070
Iosdipl2019			74736263004

Tabelle 4.5.: Words with Friends Android Version: SQLite Tabelle mit Username - ID Zuordnung

- name: gewählter Username der SpielerInnen
- email_address: Emailadresse des lokalen Kontos
- created_at: Unixtimestamp - Datum der Registrierung
- zynga_account_id: eindeutige ID des jeweiligen Kontos

Den kompletten Chatverlauf findet man in der selben Datenbank in der Tabelle "messages":

created_at	zynga_account_id	text
1552303954223	74736282070	Testanddiplwords2019
1552303965412	74736263004	Testiosdiplwords2019

Tabelle 4.6.: Words with Friends Android Version: SQLite Tabelle mit Chatnachrichten

- created_at: Unixtimestamp - Datum der Nachricht
- zynga_account_id: eindeutige ID der jeweiligen Person
- text: Chatnachricht in Klartext

Zwischengespeicherte Push-Notifications des Android Systems zeigen ebenfalls die Kontaktaufnahme zwischen zwei SpielerInnen an:

```
[Pfad nach Extraktion mittels TWRP bzw. physischer Extraktion]
/data/data/com.google.android.gms/files/fcm_queued_messages.ldb/
000005.ldb und 000010.ldb

[lesbarer Auszug]
alert,Iosdipl2019 hat dich zum Spielen eingeladen!
```

4.8.2. iOS

In der iOS Version des Spiels "Words with Friends" war ein Informationsunterschied zwischen der logischen Extraktion und dem physischen Dump der Daten ersichtlich. Erst die Daten der physischen Akquisition machten eine Rekonstruktion des kompletten Chatverlaufs möglich.

```
[Pfad nach logischer Extraktion]
/com.newtoyinc.NewWordsWithFriendsFree/Documents/
RCTAsyncLocalStorage_V1

[Pfad nach physischer Extraktion]
/private/var/mobile/Containers/Data/Application/
57F4EE79-3EC5-49D0-8D59-B28D7A794EB5/Documents/
RCTAsyncLocalStorage_V1
```

In dieser Datei findet sich der Username des Android Kontos und ein Unixtimestamp mit dem Namen "lastOnlineTime". Wirklich interessant wird es aber in Dateien, welche nur in den Daten der physischen Extraktion auffindbar sind, da diese den Chatverlauf wiedergeben:

```
[Pfad nach physischer Extraktion]
/private/var/mobile/Containers/Data/Application/
57F4EE79-3EC5-49D0-8D59-B28D7A794EB5/Documents/
.WF/WFConversationStorageModel
```

Bei der Datei "WFConversationStorageModel" handelt es sich um eine SQLite Datenbank. In der Tabelle "ZWFCONVERSATIONMESSAGEMODEL" wird der komplette Chatverlauf abgespeichert:

ZZID	ZCREATEDTIMESTAMP	ZTEXT
74736282070	1552303954.22379	Testanddiplwords2019
74736263004	1552303965.41297	Testiosdiplwords2019

Tabelle 4.7.: Words with Friends iOS Version: SQLite Tabelle mit Chatverlauf

- ZZID: eindeutige ID des sendenden Kontos
- ZCREATEDTIMESTAMP: Unixtimestamp der Nachricht
- ZTEXT: Nachricht im Klartext

Auszug aus der Tabelle "ZWFCONVERSATIONMODEL":

ZCREATEDTIMESTAMP	ZCONVERSATIONID
1551974892.35826	74736263004:74736282070

Tabelle 4.8.: Words with Friends iOS Version: SQLite Tabelle mit Chatmetadaten

- ZCREATEDTIMESTAMP: Unixtimestamp bei Erstellung des Chatrooms
- ZCONVERSATIONID: ID des Chatrooms gebildet aus den beiden ZZIDs der User (siehe Tabelle 4.7)

```
[Pfad nach physischer Extraktion]
/private/var/mobile/Containers/Data/Application/
57F4EE79-3EC5-49D0-8D59-B28D7A794EB5/Documents/.WF/WFStorageModel
```

"WFStorageModel" ist eine weitere SQLite Datenbank. In dieser findet sich in einer Tabelle die Zuordnung zwischen Username und der oben genannten ZZID welche die eindeutige ID des Kontos in "Words with Friends" darstellt.

Auszug aus der Tabelle "ZWFBASEUSER":

ZZID	ZUSERNAME
74736263004	Iosdipl2019
74736282070	Anddipl2019

Tabelle 4.9.: Words with Friends iOS Version: SQLite Tabelle mit ID - Usernamen Zuordnung

- ZZID: eindeutige ID des Kontos
- ZUSERNAME: gewählter Username des Kontos

4.8.3. Unterschiede bzw. Gemeinsamkeiten

Das Spiel "Words with Friends" glich sich in den abgelegten Daten nicht zu 100 Prozent aber es wurde ein sehr ähnliches Modell festgestellt. In beiden Versionen konnte der komplette Chatverlauf in einer SQLite Datenbank aufgefunden werden. Erst eine physische Extraktion bei iOS brachte gute Ergebnisse, wohingegen bei Android bereits ein logisches Backup mittels ADB ausreichte um die stattgefundene Kommunikation darstellen zu können.

4.9. World of Tanks Blitz MMO

World of Tanks Blitz MMO ist ein Multiplayer Panzerschlachten-Simulationsspiel. Die Android Version wurde über 50 Millionen mal installiert und hat 2,5 Millionen Bewertungen im Google Play Store. [55] Die Version für iOS hat rund 40.700 Bewertungen und ist als Editors' Choice im App Store gekennzeichnet.[56]

In World Of Tanks Blitz MMO müssen SpielerInnen für eine erfolgreiche Registrierung ein Google Konto bekannt geben. Dieses Konto wird mit dem Spiel verknüpft. Nun muss man ein circa 15-minütiges Tutorial durchspielen. Erst jetzt kann man sich einen Nicknamen definieren. Die Android Version lädt während des Spiels in Pausen immer wieder viele Pakete aus dem Internet nach - circa 2 Gigabyte an Daten wurden beim einfachen Test geladen. Die iOS Version des Spiels ist bereits beim Download 2,4 Gigabyte groß und lädt nach erfolgter Installation keine Pakete mehr nach. Nach Ende des Tutorials (und dem Nachladen der Pakete) können andere SpielerInnen in den eigenen "Zug" eingeladen werden. Ein Chat ist nun möglich. Der Aufwand für die Herstellung der Chatbereitschaft lag bei dieser Applikation bei ungefähr 30 Minuten.

4.9.1. Android

Die Android Version des Spiels "World of Tanks Blitz MMO" liefert ab der Datenextraktion mittels TWRP Hinweise auf einen Kontakt zwischen zwei SpielerInnen.

```
[Pfad nach Extraktion mittels TWRP bzw. physischer Extraktion]
/data/data/net.wargaming.wot.blitz/files/.Fabric/
com.crashlytics.sdk.android.crashlytics-core/log-files/
crashlytics-userlog-5C8690F101F3-0001-182D-676F32A45292
```

In dieser Log-Datei findet sich der Username des iOS Kontos inklusive einer Uhrzeit wieder. Leider ist kein Datum vorhanden, das Änderungsdatum der Datei kann hier aber aushelfen. Es dürfte sich hier um Metadaten von eröffneten Spielen handeln.

4.9.2. iOS

Das Spiel "World of Tanks Blitz MMO" in der iOS Version ist trotz der gewaltigen Datenmenge äußerst sparsam mit gecachten Kommunikationsdaten. Weder in den Daten der logischen noch in denen der

[55] https://play.google.com/store/apps/details?id=net.wargaming.wot.blitz&hl=en
[56] https://apps.apple.com/us/app/world-of-tanks-blitz-mmo/id859204347

physischen Extraktion wurden Hinweise auf einen Informationsaustausch zwischen zwei SpielerInnen gefunden.

4.9.3. Unterschiede bzw. Gemeinsamkeiten

Die extrahierten Daten von "World of Tanks Blitz MMO" lassen keine Rückschlüsse auf eine Kommunikation zwischen zwei Konten zu. Lediglich in der Android Version konnte in einer Log-Datei ein Hinweis auf ein gemeinsames Spiel gefunden werden.

4.10. XBOX

Die App XBOX ist eine Chat Plattform innerhalb des Microsoft XBOX Live[57] Netzwerks. Trotzdem wurde diese Applikation, wie auch die App "PlayStation Messages", innerhalb dieser Arbeit durch ihre vorwiegende Nutzung innerhalb von Spielenetzwerken und der in Kapitel 1 entsprechend vorgestellten Berichte über deren kriminellen Nutzung, zur Analyse ausgewählt. Die Android Version wurde über 10 Millionen mal installiert und hat rund 679.600 Bewertungen Google Play Store. [58] Die Version für iOS hat rund 446.300 Bewertungen im App Store. [59]

Es muss ein XBOX bzw. Microsoft Konto [60] erstellt werden. Für eine erfolgreiche Registrierung muss eine E-Mailadresse bekanntgegeben werden, an welche ein Verifizierungscode geschickt wird. Nach Eingabe dieses Codes wird zusätzlich ein Captcha abgefragt. Jetzt kann ein sogenannter "Gamertag" gewählt werden.

Andere SpielerInnen können nun über deren Gamertag gesucht und als "FreundIn" hinzugefügt werden. Ein Chat ist jedoch nicht sofort möglich, zuvor fordert Microsoft einen Reputation Check. Man muss eine Telefonnummer angeben, auf welche eine SMS mit einem Verifizierungscode geschickt wird. Nach Hinzufügen der Telefonnummer kann eine sogenannte "XBOX Party" eröffnet werden. Hier ist ein Voice Chat mit mehreren TeilnehmerInnen möglich. In diesem Gruppenchat können aber auch klassische Chatnachrichten verschickt werden.

Außerhalb dieser "XBOX Party" kann mit anderen SpielerInnen direkt gechattet werden. Aus unbekannten Gründen war es in diesem Setup am Android Gerät nicht möglich, aktiv zu chatten - Nachrichten

[57] https://www.xbox.com/en-US/live
[58] https://play.google.com/store/apps/details?id=com.microsoft.xboxone.smartglass&hl=en
[59] https://apps.apple.com/us/app/xbox/id736179781
[60] https://signup.live.com

konnten jedoch empfangen werden. Ein weiterer Weg zu kommunizieren ist öffentlich über den "Feed" anderer SpielerInnen. Dieser Feed ist wie eine öffentliche Profilseite auf derer man Texte erstellen bzw. kommentieren kann.

4.10.1. Android

Die Android Version von "XBOX" zeigte ab der Extraktionsmethode mittels ADB einen gespeicherten Chatverlauf.

```
[Pfad nach logischer Extraktion mittels ADB]
/apps/com.microsoft.xboxone.smartglass/sp/xlesetting.xml

[Pfad nach Extraktion mittels TWRP bzw. physischer Extraktion]
/data/data/com.microsoft.xboxone.smartglass/shared_prefs/xlesetting.xml
```

Im Key "MESSAGESDB_SKYPE_CONVERSATION_MESSAGE_CACHE" dieser XML Datei finden sich die Chatnachrichten. Es handelt sich nicht um die Gespräche bzw. Chats innerhalb einer "XBOX Party", sondern nur um die direkten Chats zwischen zwei SpielerInnen.

Auszug der Daten:

```
8:xbox:2535414777516234 [...]
originalarrivaltime [...] Mar 12, 2019 11:32:33 AM [...]
content&amp;quot;:&amp;quot;Testiosdipl209
```

Anfangs erkennt man die ID des iOS Kontos, danach wird als "originalarrivaltime" der Timestamp und unter "content" die Nachricht selbst im Klartext abgespeichert.

4.10.2. iOS

Die iOS Version von "XBOX" speichert bereits im normalen Backup den gesamten privaten Chatverlauf in einer SQLite Datenbank ab. Es handelt sich wie in der Android Version nicht um die Gespräche bzw. Chats innerhalb einer "XBOX Party", sondern nur um die direkten Chats zwischen zwei SpielerInnen.

```
[Pfad nach logischer Extraktion]
/com.microsoft.smartglass/Library/Application Support/
SmartGlass-2535414777516234.sqlite
```

```
[Pfad nach physischer Extraktion]
/private/var/mobile/Containers/Data/Application/
55A8BDCD-28E1-4CD0-8241-704598F852BB/Library/Application Support/
SmartGlass-2535414777516234.sqlite
```

Auszug aus der Tabelle "ZSGUSER":

ZGAMERTAG	ZRAWXUID
anddipl2019	2535443634407665

Tabelle 4.10.: XBOX iOS Version: SQLite Tabelle mit Zuordnung UID zu Gamertag

- ZGAMERTAG: gewählter Username bzw. Gamertag des Android Kontos
- ZRAWXUID: eindeutige ID des Android Kontos

In dieser Tabelle sieht man alle anderen SpielerInnen (aufgelistet mit deren Gamertag sowie der eindeutigen ID) mit denen interagiert wurde.

Die Tabelle "ZSGMESSAGE" hingegen, speichert den Chatverlauf selbst ab:

ZTEXT1	ZTHREADID	ZTIMEINTERVAL
Testiosdipl209	8:xbox:2535443634407665	1552386798.0

Tabelle 4.11.: XBOX iOS Version: SQLite Tabelle mit Chatverlauf

- ZTEXT1: private Chatnachricht
- ZTHREADID: ID gebildet aus 8:xbox:ZRAWXUID (siehe Tabelle 4.10) des Empfängers
- ZTIMEINTERVAL: Unixtimestamp der Nachricht

Anmerkung: Wie in Kapitel 4.10.1 erwähnt, war es nicht möglich vom Android Gerät eine Nachricht zu verschicken. Deshalb taucht in dieser Tabelle auch nur die eigene Nachricht auf.

4.10.3. Unterschiede bzw. Gemeinsamkeiten

Die App "XBOX" bietet viele verschiedene Kommunikationsmöglichkeiten. Es wird jedoch nur der private Chat zwischen zwei SpielerInnen am Gerät selbst abgespeichert. Dieser ist in beiden Versionen der App bereits ab einer logischen Extraktion vorhanden. Auffällig ist die gegensätzliche Speicherung der

Daten. In der iOS Version wird dafür eine SQLite Datenbank benutzt, in der Android Version lediglich ein XML File.

4.11. Yahtzee

Yahtzee ist, wie die ebenfalls von der Firma Scopely hergestellte und in dieser Arbeit analysierte App "Dice with Buddies", ein Multiplayer Würfel Spiel. Die Android Version wurde über 5 Millionen mal installiert und hat rund 111.900 Bewertungen Google Play Store. [61] Die Version für iOS hat rund 123.900 Bewertungen im App Store. [62]

Das Spiel "Yahtzee" wird genau wie "Dice with Buddies" eingerichtet. Auch in "Yahtzee" muss man sich anfangs durch ein Tutorial spielen. Nach dem Tutorial bekommt man einen Nicknamen zugewiesen, welchen man jedoch ändern kann. Es ist nicht notwendig sich mit einer Mailadresse zu registrieren. Nun können anderere SpielerInnen über deren Nicknamen gesucht werden. Jetzt kann man auch außerhalb des Spiels mit dem Gegenüber chatten. Der Aufwand für die Herstellung der Chatbereitschaft lag bei dieser Applikation zwischen 10 und 15 Minuten.

Es ergab sich jedoch aufgrund der gleichzeitig installierten Applikation "Dice with Buddies" der Vorteil, dass der Nickname aus dem anderen Spiel übernommen wurde. Ein Chat zwischen den beiden Applikationen war jedoch nicht möglich.

4.11.1. Android

Die Android Version des Spiels "Yahtzee" brachte ab der Extraktionsmethode mittels ADB Hinweise auf am Gerät gespeicherte Chatverläufe bzw. Freundeslisten. Die extrahierten Daten ergaben jedoch trotzdem keine konkreten Kommunikationsdaten. Es wurden zwar, wie in der Applikation "Dice with Buddies", Dateien mit hinweisgebenden Namen gefunden, welche jedoch nicht ausgelesen werden konnten. Es handelt sich um Binärdateien mit folgenden Namen:

```
[Pfad nach logischer Extraktion mittels ADB]
/apps/com.scopely.yux/f

[Pfad nach Extraktion mittels TWRP bzw. physischer Extraktion]
```

[61] https://play.google.com/store/apps/details?id=com.scopely.yux&hl=en
[62] https://apps.apple.com/us/app/yahtzee-with-buddies-dice/id1206967173

```
/data/data/com.scopely.yux/files/

[Dateinamen]
BuddiesList-v2.dat
BuddyList-v1.dat
BuddiesMessages.dat
```

Aufgrund dieser Namen und der aufgefundenen Dateigrößen wird vermutet, dass sich der Chatverlauf (BuddiesMessages.dat) sowie die Freundesliste (BuddiesList-v2.dat bzw. BuddyList-v1.dat) am Gerät befinden. Es müsste jedoch die Datenstruktur innerhalb der Binärdateien durch Reverse Engineering herausgefunden werden, um diese Vermutung bestätigen zu können.

Mithilfe der physischen Extraktion konnte noch eine weitere Datei rekonstruiert werden:

```
[Pfad nach Extraktion mittels TWRP bzw. physischer Extraktion]
/data/data/com.google.android.gms/files/fcm_queued_messages.ldb/
000026.ldb

[Auszug]
AndroidText":"Iosdipl2019 hat dir eine Nachricht gesendet."
```

In dieser Push-Notification Cache Datei sieht man, dass einem das iOS Konto eine Nachricht geschickt hat, nicht aber den Inhalt.

4.11.2. iOS

In der iOS Version des Spiels "Yahtzee" war ein Unterschied zwischen der logischen Extraktion und dem physischen Dump der Daten ersichtlich. Im iTunes-Backup waren keinerlei hinweisgebenden Spuren enthalten. Die physisch extrahierten Daten waren umfassender, geben jedoch trotzdem keinen Hinweis auf eine konkrete Kontaktaufnahme oder einen Chatverlauf. Es wurden zwar Dateien mit hinweisgebenden Namen gefunden, welche jedoch nicht ausgelesen werden konnten. Es handelt sich um Binärdateien mit folgenden Namen:

```
[Pfad nach physischer Extraktion]
/private/var/mobile/Containers/Data/Application/
6D6D32C7-C215-4DEE-AA4B-A47357304502/Documents/
```

```
[Dateinamen]
BuddiesList-v2.dat
BuddyList-v1.dat
BuddiesMessages.dat
```

Aufgrund dieser Namen und der aufgefundenen Dateigrößen wird vermutet, dass sich der Chatverlauf (BuddiesMessages.dat) sowie die Freundesliste (BuddiesList-v2.dat bzw. BuddyList-v1.dat) am Gerät befindet. Es müsste jedoch die Datenstruktur innerhalb der Binärdateien durch Reverse Engineering herausgefunden werden, um diese Vermutung bestätigen zu können.

4.11.3. Unterschiede bzw. Gemeinsamkeiten

Vergleicht man die Ergebnisse des Spiels "Yahtzee" mit deren der App "Dice with Buddies" (siehe 4.3), so gleichen sich diese zu 100 Prozent. Diese beiden Apps wurden von den selben EntwicklerInnen erschaffen. Die Spiele selbst ähneln sich auch in der Spielmechanik sehr. Es wird vermutet, dass hier lediglich das Design und ein paar Kleinigkeiten verändert wurden und daraus ein neues Spiel entstanden ist.

Die erhaltenen Informationen auf eine stattgefundene Kommunikation durch die forensische Analyse waren bei beiden Version jedoch minimal bis nicht vorhanden. Nur eine auf Android gecachte Push-Benachrichtigung ergab einen Hinweis auf Kontaktaufnahme.

4.12. Zusammenfassung der Ergebnisse

Um Schlüsse aus den erhaltenen Ergebnisse ziehen zu können, wurden diese in den folgenden Tabellen kompakt dargestellt. Die Tabelle 4.12 zeigt alle analysierten Applikationen.
Die aufgefundenen Artefakte wurden in die Kategorien "Hinweis auf Kontakt" (Beispiel: Der/die KommunikationspartnerIn wird in einer Freundesliste gefunden oder das entsprechende Profil wurde aktiv gesucht.) und "Chatverlauf" (hier wurde ein Chatverlauf rekonstruiert) aufgeteilt. Beim Auffinden von den jeweiligen Artefakten wurde die Betriebssystemversion in dieser Tabelle eingetragen. Die Darstellung ermöglicht außerdem eine schnelle Gegenüberstellung der Betriebssysteme.

App	Hinweis auf Kontakt	Chatverlauf
8 Ball Pool	Android / iOS	- / -
ChessTime	Android / iOS	Android / -
Dice with Buddies	- / -	- / -
Plato	Android / iOS	Android / iOS
Playstation Messages	- / iOS	- / -
QuizUp	Android / iOS	Android / -
Scrabble	Android / iOS	- / -
Words with Friends	Android / iOS	Android / iOS
World of Tanks Blitz MMO	Android / -	- / -
XBOX	Android / iOS	Android / iOS
Yahtzee	- / -	- / -

Tabelle 4.12.: Auflistung der Findings nach Kategorie

In 9 von 11 analysierten Spielen (rund 82 Prozent) wurden Artefakte aufgefunden, welche zumindest einen Kontakt zwischen SpielerInnen nahelegt. Datenfunde bestätigen zum Beispiel gemeinsame Spiele oder das Hinzufügen von Personen in Freundeslisten.

In 7 von den 9 Fällen (rund 78 Prozent) wurde ein Hinweis auf einen Kontakt in beiden Betriebssystemversionen der Applikation entdeckt. Jeweils einmal konnte man von der Android Version nicht auf die iOS Version schließen bzw. vice versa.

In 5 von 11 Spielen (45 Prozent) konnte der gesamte Chatverlauf rekonstruiert werden. In 2 von den 5 Fällen (40 Prozent) konnte nur am Android Gerät die stattgefundene Kommunikation entdeckt werden.

Um weitere Aussagen bezüglich der Frage nach dem Informationsgewinn, der in Kapitel 2.1 angeführten Extraktionsmethoden, treffen zu können, wurde die Tabelle 4.13 erstellt. Sobald in der jeweiligen Kategorie "Hinweis auf Kontakt" bzw. "Chatverlauf" innerhalb einer Extraktionsmethode Artefakte auffindbar waren, wurden diese in der Tabelle eingetragen. Die niederwertigste Methode wurde in der Tabelle belassen. Beispiel: Wenn auf einem Android Gerät im ADB-Backup bereits entsprechende Artefakte gefunden wurden, so waren diese naturgemäß auch im TWRP-Backup bzw. DD-Image vorhanden. Diese höherwertigen Methoden wurden aus Gründen der Lesbarkeit jedoch aus der Tabelle entfernt.

App	Hinweis auf Kontakt	Chatverlauf
8 Ball Pool - Android Version	TWRP	-
8 Ball Pool - iOS Version	Backup	-
ChessTime - Android Version	ADB	ADB
ChessTime - iOS Version	Backup	-
Dice with Buddies - Android Version	-	-
Dice with Buddies - iOS Version	-	-
Plato - Android Version	TWRP	TWRP
Plato - iOS Version	Backup	Backup
Playstation Messages - Android Version	-	-
Playstation Messages - iOS Version	Physische Extraktion	-
QuizUp - Android Version	TWRP	TWRP
QuizUp - iOS Version	Backup	-
Scrabble - Android Version	FileTransfer	-
Scrabble - iOS Version	Backup	-
Words with Friends - Android Version	ADB	ADB
Words with Friends - iOS Version	Backup	physische Extraktion
World of Tanks Blitz MMO - Android Version	TWRP	-
World of Tanks Blitz MMO - iOS Version	-	-
XBOX - Android Version	ADB	ADB
XBOX - iOS Version	Backup	Backup
Yahtzee - Android Version	-	-
Yahtzee - iOS Version	-	-

Tabelle 4.13.: Auflistung der Findings ab Level der Extraktionsmethode

Auf Android Geräten war in 50 Prozent der Auffindungen (4 von 8 Fällen) zumindest eine Extraktion des gesamten Dateisystems mittels TWRP notwendig. Anderenfalls wäre kein Hinweis auf eine Kommunikation aufgefunden worden. In 37,5 Prozent der Auffindungen (3 von 8 Fällen) war bereits ein ADB Backup ausreichend. Lediglich in einem einzigen Fall (12,5 Prozent) war bereits ein Dateitransfer für den Hinweis eines Kontakts ausreichend.

In keinem einzigen Fall war es für die Wiederherstellung eines kompletten Chatverlaufes entscheidend, dass eine höherwertige Extraktionsmethode im Gegensatz zu bereits Hinweis gebenden, niederwertigeren Methoden, gewählt wurde. Anders ausgedrückt: Wenn auf Android Geräten in einem ADB Backup bereits Hinweise auf einen Kontakt gefunden wurden, so war der Einsatz von höherwertigeren Methoden einer Extraktion (wie TWRP oder DD) zur Auffindung eines Chatverlaufes unnötig.

Auf iOS Geräten war dies in einem Analysefall anders. Hier beinhaltete die physische Extraktion im Gegensatz zum logischen Backup einen Chatverlauf. Auch war die physische Extraktion zur Auffindung von Kontakthinweisen in einem Fall notwendig - das Backup beinhaltete hier keine Spuren.

Generell lassen sich aus diesen Ergebnissen also folgende Aussagen ableiten:

- Es gibt eine sehr hohe Wahrscheinlichkeit (82 Prozent), dass innerhalb der am Gerät abgelegten Daten, zumindest Hinweise auf einen Kontakt gefunden werden können.
- Wenn Hinweise auf eine stattgefundene Kommunikation innerhalb einer Betriebssystemversion einer Applikation gefunden wurden, so waren diese auch in rund 78 Prozent der Fälle in der anderen Version auffindbar.
- Die Chance einer Rekonstruktion eines kompletten Chatverlaufes kann in dieser Untersuchung mit 45 Prozent beziffert werden.
- Auf iOS Geräten war zur Auffindung von Kommunikationsartefakten in fast allen Fällen (7 von 8 Analysen) ein einfaches (auch unverschlüsseltes) Backup ausreichend.
- Auf Android Geräten war im Vergleich ein (mittels ADB durchgeführtes) Gerätebackup, nur in 3 von 8 Fällen ausreichend. Die Hälfte aller Auffindungen waren erst in der Sicherung des gesamten Dateisystems mittels TWRP sichtbar.

- Die physische Extraktion von *nicht allokierten* Daten brachte in dieser konkreten Analyse keinen Informationsgewinn. Wären die Spiele jedoch vor der Analyse vom Gerät gelöscht worden, so wäre es nur mit dieser Methode gelungen, Daten wiederherzustellen.

5. Fazit

Wie in Kapitel 4 gezeigt wurde, waren in fast allen analysierten Applikationen (genauer: 9 von 11) Kommunikationsartefakte auffindbar. In den restlichen zwei Apps wird aufgrund von abgelegten Binärdateien, deren Name auf Kommunikationsdaten hinweisen (zum Beispiel "BuddiesMessages.dat"), ebenfalls vermutet, dass nach einem erfolgreichen Reverse Engineering, diese Daten ausgelesen werden könnten.

Zur Auffindung von Artefakten war eine simple Volltextsuche über die extrahierten Daten meist ausreichend. Die genaue Rekonstruktion eines Chatverlaufes musste mithilfe von SQLite Abfragen bzw. manueller Analyse und Verknüpfung der aufgefundenen Daten durchgeführt werden.

Diese Arbeit hat gezeigt, dass für die Speicherung von Kommunikationsdaten innerhalb von Spiele-Apps entweder SQLite Datenbanken (geräteübergreifend) oder Property Lists (iOS) bzw. JSON Dateien (Android) verwendet werden. Zusätzlich wurde deutlich, dass alle bis auf eine Applikation (siehe Kapitel 4.5 - Playstation Messages) Nachrichten nicht verschlüsselt abspeichern.

Für die Frage ob ein Kontakt zwischen zwei SpielerInnen stattgefunden hat, war, wie in Kapitel 2.2 vermutet, die Analyse bzw. Rekonstruktion der am Gerät abgespeicherten Push-Notifications hilfreich. Während der genauen Untersuchung des Android Gerätes (siehe Detailergebnisse in den Kapiteln 4.2.1, 4.3.1, 4.8.1 und 4.11.1) wurden entsprechende Push-Notifications aufgefunden. Am iOS Gerät konnten leider keine brauchbaren Benachrichtigungen rekonstruiert werden. Dies könnte jedoch daran gelegen haben, dass erst einige Wochen, nach dem Setzen der künstlichen Kommunikationsartefakte, ein Jailbreak (als Vorarbeit für eine physische Extraktion) am Gerät durchgeführt wurde. Die dafür notwendigen Schreiboperationen könnten bereits gelöschte Push-Notifications wieder überschrieben haben.

Diese Arbeit zeigte, dass auf Android Geräten in 50 Prozent der Fälle mindestens das gesamte Dateisystem mittels TWRP gesichert und analysiert werden musste, bevor Kommunikationsartefakte aufgefunden wurden. Die Analyse von Android Gerätesicherungen, welche in dieser Arbeit mit ADB angefertigt wurden, war also nur in der Hälfte aller analysierten Fälle sinnvoll.

Deutlich "offener" geben sich anscheinend die EntwicklerInnen von iOS Applikationen. Hier war in 7 von 8 Fällen bereits die Analyse von einfachen Geräte-Backups für den Nachweis von Kommunikationsartefakten ausreichend. Die Tatsache, ob ein iTunes-Backup verschlüsselt wird oder nicht, ist für diese Art der Daten unerheblich.

Eine weitere Erkenntnis aus der durchgeführten Analyse ist, dass wenn Hinweise auf eine stattgefundene Kommunikation in einer Betriebssystemversion einer App gefunden wurden, auch die forensische Analyse der Version am anderen Betriebssystem zu rund 78 Prozent erfolgreich war. Es gibt also eine hohe Wahrscheinlichkeit, dass eine Applikation auf beiden Betriebssystemversionen die gleichen oder sehr ähnliche Daten ablegt.

Diese Arbeit warf einige Themen auf, welche in anderen, zukünftigen Forschungsarbeiten behandelt werden sollten:

- Wie in Kapitel 2.4 dargestellt, müssten Kommunikationsartefakte nicht nur am Gerät selbst, sondern auch auf Servern bzw. dem Netzwerk sichtbar sein. Eine eingehende Analyse und Gegenüberstellung mit den Ergebnissen dieser Arbeit wäre daher wünschenswert.
- Wie in Kapitel 4 beschrieben, wurden während der Analyse Binärdateien aufgefunden, welche aufgrund ihrer Benennung höchstwahrscheinlich Kommunikationsdaten enthalten. In diesen Fällen wäre zusätzlich ein Reverse-Engineering der Applikationen durchzuführen, um die Vermutung zu bestätigen. Außerdem wäre die Erstellung eines entsprechenden Leitfadens, für ein Reverse Engineering von in Binärdateien abgelegten Kommunikationsdaten, sinnvoll.
- Einige Spiele haben bei der erstmaligen Registrierung angeboten, sich mit dem Google oder Facebook Konto zu verknüpfen. Im Fall einer Verknüpfung mit Facebook könnten die FreundInnen ins Spiel importiert werden. Es wäre zu untersuchen, ob und wie sich diese Verknüpfung auf die Speicherung von Kommunikationsdaten auswirkt.
- Das hilfreiche Open-Source Analyse Tool "iPhone-Backup-Analyzer-2 - kurz: ipba2" [63] ist leider für Analysen von Backups ab iOS Version 10 (aufgrund von massiven Änderungen im Aufbau des Backups - Stichwort manifest.db statt manifest.mbdb) nicht mehr geeignet. Dieses bis dahin sehr brauchbare Werkzeug könnte in zukünftigen Forschungsarbeiten aktualisiert werden.

[63] https://github.com/PicciMario/iPhone-Backup-Analyzer-2

Aufgrund der mittlerweile gut erforschten forensischen Analyse von designierten Kommunikationsapps und Anwendung dieser durch die Strafverfolgungsbehörden, ist es möglich, dass kriminelle Personen auf der Suche nach verdeckter Kommunikation, auf andere Mittel umsteigen. Dies könnten, wie gezeigt, durchaus Spiele-Apps auf mobilen Geräten sein.

ForensikerInnen müssen sich auf der Suche nach Kommunikationsdaten in Zukunft nicht nur Messenger Apps wie WhatsApp, Skype, Telegram oder iMessage genauer ansehen, sondern auch alle anderen Applikationen in Betracht ziehen, in denen eine Kommunikation möglich ist. Solange diese Apps die Daten unverschlüsselt bzw. in Klartext abspeichern, ist dies mithilfe einer Volltextsuche und manueller Verknüpfung von aufgefundenen Informationen noch eine vergleichbar einfache Aufgabe. Würden zukünftige Applikationen stärker auf die Verwendung von Binärdaten oder verschlüsselten SQLite Datenbanken für die Speicherung von Chatprotokollen setzen, wird die Aufgabe der AnalystInnen deutlich zeitintensiver werden.

A. Glossar

ADB Die Android Debug Bridge[64] ist ein Client-Server Programm welches im Android SDK bzw. den Android Plattform Tools[65] enthalten ist. Die ADB stellt eine Schnittstelle zwischen PC und Android Smartphone dar. Es kann damit zum Beispiel auf gewisse Daten des Geräts zugegriffen, verschiedene Befehle ausgeführt bzw. ein Gerätebackup erstellt werden. 10, 21–23, 28, 30, 44, 47, 50, 52, 56, 57, 59

JTAG Joint Test Action Group[66] - Dieser Begriff wird häufig als Synonym für eine physische Möglichkeit des direkten Zugriffs auf integrierte Schaltungen von Geräten (eigentlich als Debug Maßnahme integriert) verwendet. Der Zugriff auf die JTAG Pins ermöglicht möglicherweise ein Auslesen von Speicherchips. 11

MTP Media Transfer Protocol - Ein Datenaustauschprotokoll welches zwischen PC und Android Devices verwendet wird. Die Verbindung zwischen Smartphone und PC über USB Kabel beim Transfer von Dateien wird mithilfe von MTP gesteuert. 9, 21

[64] https://developer.android.com/studio/command-line/adb
[65] https://developer.android.com/studio/releases/platform-tools
[66] https://en.wikipedia.org/wiki/JTAG

PLIST Property List - Ein von Apple entwickelter, und vor allem auf iOS Geräten weit verbreiteter Dateityp. In einem solchen Typ lassen sich seriell abgespeicherte Objekte gut verwalten. 17, 26, 27, 40

TWRP Team Win Recovery Project - eine Custom Recovery Software für Android Devices. Eine Recovery Software auf Android Geräten ist ein minimales Betriebssystem, welches auf einer eigenen Partition abgespeichert ist. Das ab Produktion ausgelieferte Recovery System ist als Failsave bzw. für die Durchführung von systemweiten Updates zuständig. Eine modifizierte Version ermöglicht beispielsweise AnalystInnen direkten Zugriffe auf den Gerätespeicher. 10, 12, 21–23, 26, 28, 32, 36, 38, 44, 48, 56, 57, 59

Literaturverzeichnis

[1] K. A. Alghafli, A. Jones, and T. A. Martin, "Forensics data acquisition methods for mobile phones," in *2012 International Conference for Internet Technology and Secured Transactions*. IEEE, 2012, pp. 265–269.

[2] J. R. Nurse and M. Bada, "The group element of cybercrime: Types, dynamics, and criminal operations," *arXiv preprint arXiv:1901.01914*, 2019.

[3] R. Umar, I. Riadi, and G. M. Zamroni, "Mobile forensic tools evaluation for digital crime investigation," *International Journal on Advanced Science, Engineering and Information Technology*, vol. 8, no. 3, pp. 949–955, 2018.

[4] F.-C. Tsai, E.-C. Chang, and D.-Y. Kao, "Whatsapp network forensics: Discovering the communication payloads behind cybercriminals," in *2018 20th International Conference on Advanced Communication Technology (ICACT)*. IEEE, 2018, pp. 679–684.

[5] J. Ebner, *Radikalisierungsmaschinen - Wie Extremisten die neuen Technologien nutzen und uns manipulieren*. Suhrkamp Verlag, Sep. 2019.

[6] G. B. Satrya, P. T. Daely, and S. Y. Shin, "Android forensics analysis: Private chat on social messenger," in *2016 Eighth International Conference on Ubiquitous and Future Networks (ICUFN)*. IEEE, 2016, pp. 430–435.

[7] K. M. Ovens and G. Morison, "Forensic analysis of kik messenger on ios devices," *Digital Investigation*, vol. 17, pp. 40–52, 2016.

[8] A. Mahajan, M. Dahiya, and H. Sanghvi, "Forensic analysis of instant messenger applications on android devices," *International Journal of Computer Applications*, vol. 68, no. 8, pp. 38–44, 2013.

[9] M. I. Al-Saleh and Y. A. Forihat, "Skype forensics in android devices," *International Journal of Computer Applications*, vol. 78, no. 7, 2013.

[10] M. I. Husain and R. Sridhar, "iforensics: forensic analysis of instant messaging on smart phones," in *International Conference on Digital Forensics and Cyber Crime*. Springer, 2009, pp. 9–18.

[11] FBI, "Bronx bloods members communicating through playstation network (psn)," FEDERAL BUREAU OF INVESTIGATION - New York Division, techreport, May 2011.

[12] NJRIOC, "Ms-13 using gaming consoles to conduct business," NJ REG'L OPERATIONS INTELLIGENCE CTR, techreport, Sep. 2010.

[13] M. S. Ruskin, "Playing in the dark: How online games provide shelter for criminal organizations in the surveillance age," *Ariz. J. Int'l & Comp. L.*, vol. 31, pp. 875–908, 2014.

[14] T. Haselton and M. Graham. (2019, Oct.) About 2,200 people watched the german synagogue shooting on amazon's twitch. [Online]. Available: https://www.cnbc.com/2019/10/09/the-german-synagogue-shooting-was-streamed-on-twitch.html (Accessed 2019-10-09).

[15] Biermann. Er plante seine taten wie computerspiele. [Online]. Available: https://www.zeit.de/gesellschaft/zeitgeschehen/2019-10/attentaeter-halle-internet-radikalisierung-memes/komplettansicht (Accessed 2019-10-10).

[16] (2019, Oct.) Seehofer will rechtsextremisten auf gaming-plattformen bekämpfen. Süddeutscher Verlag. [Online]. Available: https://www.sueddeutsche.de/politik/seehofer-gaming-halle-twitch-1.4639479 (Accessed 2019-10-13).

[17] S. Krempl. (2019, Oct.) Nach halle-attentat: Cdu fordert umfangreiches Überwachungspaket fürs internet. Heise Medien GmbH. [Online]. Available: https://www.heise.de/newsticker/meldung/Nach-Halle-Attentat-CDU-fordert-umfangreiches-Ueberwachungspaket-fuers-Internet-4555401.html (Accessed 2019-10-14).

[18] D. Koller. (2019, Nov.) Deutscher verfassungsschutz will gamer und internet-chaträume stärker beobachten. STANDARD Verlagsgesellschaft m.b.H. [Online]. Available: https://www.derstandard.at/story/2000111473848/deutscher-verfassungsschutz-will-gamer-und-internet-chatraeume-staerker-beobachten (Accessed 2019-11-25).

[19] D. Bieda and L. Halawi, "Cyberspace: A venue for terrorism," *Issues in Information Systems*, vol. 16, no. 3, pp. 33–42, 2015.

[20] N. Veerasamy and M. Grobler, "Terrorist use of the internet: Exploitation and support through ict infrastructure," in *Proceedings of the International Conference on Information Warfare and Security, Washington, DC*, 2011, pp. 260–267.

[21] S. Mahmood, "Online social networks: The overt and covert communication channels for terrorists and beyond," in *2012 IEEE Conference on Technologies for Homeland Security (HST)*. IEEE, 2012, pp. 574–579.

[22] Nasralla. (2015, May) Teenager in austrian 'playstation' terrorism case gets two years. Reuters. [Online]. Available: https://www.reuters.com/article/us-mideast-crisis-austria/teenager-in-austrian-playstation-terrorism-case-gets-two-years-idUSKBN0OB0LK20150526 (Accessed 2019-10-16).

[23] P. Tassi. (2015, Nov.) How isis terrorists may have used playstation 4 to discuss and plan attacks. Forbes. [Online]. Available: https://www.forbes.com/sites/insertcoin/2015/11/14/why-the-paris-isis-terrorists-used-ps4-to-plan-attacks/#66a2248c7055 (Accessed 2019-10-01).

[24] J. Temperton. (2015, Nov.) There is no link between the ps4 and the paris attacks. Wired. [Online]. Available: https://www.wired.co.uk/article/ps4-connected-paris-attacks-isis (Accessed 2019-09-16).

[25] D. P. J. Taylor, H. Mwiki, A. Dehghantanha, A. Akibini, K. K. R. Choo, M. Hammoudeh, and R. Parizi, "Forensic investigation of cross platform massively multiplayer online games: Minecraft as a case study," *Science & Justice*, vol. 59, no. 3, pp. 337–348, 2019.

[26] H. Gnauer, "Weshalb der sogenannte bundestrojaner eine schlechte idee ist," *Medienimpulse*, vol. 55, no. 3, 2017.

[27] M. Al-Youssef. (2019, Aug.) Bundestrojaner: Ministerium will Überwachungssoftware teils selbst entwickeln. STANDARD Verlagsgesellschaft mbH. [Online]. Available: https://www.derstandard.at/story/2000106929089/bundestrojaner-innenministerium-will-ueberwachungssoftware-selbst-entwickeln (Accessed 2019-10-31).

[28] R. Tamma, O. Skulkin, and S. Bommisetty, *Practical Mobile Forensics*. Packt Publishing, 2018.

[29] K. Barmpatsalou, D. Damopoulos, G. Kambourakis, and V. Katos, "A critical review of 7 years of mobile device forensics," *Digital Investigation*, vol. 10, no. 4, pp. 323–349, 2013.

[30] G. Grispos, T. Storer, and W. B. Glisson, "A comparison of forensic evidence recovery techniques for a windows mobile smart phone," *Digital Investigation*, vol. 8, no. 1, pp. 23–36, 2011.

[31] N. R. Roy, A. K. Khanna, and L. Aneja, "Android phone forensic: Tools and techniques," in *2016 International Conference on Computing, Communication and Automation (ICCCA)*. IEEE, 2016, pp. 605–610.

[32] A. Hoog, *Android forensics: investigation, analysis and mobile security for Google Android*. Elsevier, 2011.

[33] D. Abalenkovs, P. Bondarenko, V. K. Pathapati, A. Nordbø, D. Piatkivskyi, J. E. Rekdal, and P. B. Ruthven, "Mobile forensics: Comparison of extraction and analyzing methods of ios and android," *Gjovik University College, Gjovik, Norway*, 2012.

[34] Tamma, *Learning Android Forensics : Analyze Android Devices with the Latest Forensic Tools and Techniques*, 2nd ed. Packt Publishing Ltd, 2018.

[35] F. Freiling, M. Spreitzenbarth, and S. Schmitt, "Forensic analysis of smartphones: The android data extractor lite (adel)," in *ADFSL Conference on Digital Forensics, Security and Law, 2011*, 2011, pp. 151–160.

[36] R. Regupathy, *Unboxing Android USB: A hands on approach with real world examples*. Apress, 2014.

[37] Back up user data with auto backup. Google Inc. [Online]. Available: https://developer.android.com/guide/topics/data/autobackup (Accessed 2019-11-27).

[38] S.-T. Sun, A. Cuadros, and K. Beznosov, "Android rooting: Methods, detection, and evasion," in *Proceedings of the 5th Annual ACM CCS Workshop on Security and Privacy in Smartphones and Mobile Devices*. ACM, 2015, pp. 3–14.

[39] S. Willassen, "Forensic analysis of mobile phone internal memory," in *IFIP International Conference on Digital Forensics*. Springer, 2005, pp. 191–204.

[40] I. Mohamed and D. Patel, "Android vs ios security: A comparative study," in *2015 12th International Conference on Information Technology-New Generations*. IEEE, 2015, pp. 725–730.

[41] M. Epifani and P. Stirparo, *Learning iOS forensics*. Packt Publishing Ltd, 2016.

[42] (2019, Aug.) ios 12 - delivered notifications and a new way to parse them. D20 Forensics Blog. [Online]. Available: https://blog.d204n6.com/2019/08/ios-12-delivered-notifications-and-new.html (Accessed 2019-12-03).

[43] C. Bi, "Research and application of sqlite embedded database technology," *wseas transactions on computers*, vol. 8, no. 1, pp. 539–543, 2009.

[44] J. Kreibich, *Using SQLite*, M. Loukides, Ed. O'Reilly Media, Inc., 2010.

[45] A. Grant and M. Owens, *The Definitive Guide to SQLite, Second Edition*, J. Gennick, Ed. Apress, 2010.

[46] M. Piccinelli and P. Gubian, "Exploring the iphone backup made by itunes," *Journal of Digital Forensics, Security and Law*, vol. 6, no. 3, p. 4, 2011.

[47] (2011, May) Json versus plist, the ultimate showdown. Cocoanetics. [Online]. Available: https://www.cocoanetics.com/2011/03/json-versus-plist-the-ultimate-showdown/ (Accessed 2019-11-19).

[48] (2019, Oct.) About encrypted backups on your iphone, ipad, or ipod touch. Apple Inc. [Online]. Available: https://support.apple.com/en-us/HT205220 (Accessed 2019-10-09).

[49] Apple, "ios security guide - ios 12.1," Apple, techreport, Nov. 2018.

[50] (2016, May) Technical q&a - how do i prevent files from being backed up to icloud and itunes? [Online]. Available: https://developer.apple.com/library/archive/qa/qa1719/_index.html (Accessed 2019-10-09).