FAST DETECT
SACHVERSTÄNDIGENBÜRO
FÜR IT-FORENSIK

HOCHSCHULE
MITTWEIDA
University of Applied Sciences

BACHELORARBEIT

Herr
Michael Metzger

Forensische Analyse der Chatanwendung Viber auf iOS-Geräten

Mittweida, August 2023

Fakultät **Angewandte Computer- und Biowissenschaften**

BACHELORARBEIT

Forensische Analyse der Chatanwendung Viber auf iOS-Geräten

Autor:
Michael Metzger

Studiengang:
IT-Forensik/Cybercrime

Seminargruppe:
CC19w1-B

Erstprüfer:
Prof. Ronny Bodach

Zweitprüfer:
Paul Prade, M.Sc.

Einreichung:
Mittweida, 21.08.2023

Verteidigung/Bewertung:
Mittweida, 2023

Faculty of **Applied Computer Sciences and Biosciences**

BACHELOR THESIS

Forensic analysis of the Viber chat application on iOS devices

Author:
Michael Metzger

Course of Study:
IT-Forensic/Cybercrime

Seminar Group:
CC19w1-B

First Examiner:
Prof. Ronny Bodach

Second Examiner:
Paul Prade, M.Sc.

Submission:
Mittweida, 21.08.2023

Defense/Evaluation:
Mittweida, 2023

Bibliografische Beschreibung:

Metzger, Michael:
Forensische Analyse der Chatanwendung Viber auf iOS-Geräten. – 2023. – 60 S.
Mittweida, Hochschule Mittweida – University of Applied Sciences, Fakultät Angewandte Computer- und Biowissenschaften, Bachelorarbeit, 2023.

Referat:

Ziel der Arbeit ist es, innerhalb einer forensischen Analyse die Ablagestruktur der Chatanwendung Viber unter dem Betriebssystem iOS zu analysieren. Darüber hinaus sollen Dateien analysiert werden, um relevante Informationen zu Chats zu extrahieren und auswertbar zu machen. Bei der Recherche zum aktuellen Forschungsstand wurden kaum Arbeiten gefunden, welche eine tiefgehende forensische Analyse von Viber auf iOS-Geräten zum Gegenstand haben. Für die Auswertung wurden Testdaten in Form von Einzel- und Gruppenchats auf iOS-Geräten erstellt und die Geräte anschließend Informationstechnologie (IT)-forensisch ausgelesen. Durch die Verwendung des Auslesegerätes UFED Touch2 konnte die Ablagestruktur mittels des UFED-Readers analysiert und dokumentiert werden. Die Analyse der Dateien brachte Informationen aus den zwei Hauptdatenbanken 'Settings.data' und 'Contacts.data' hervor. Hierbei wurden alle Tabellen analysiert und Informationen zu den relevanten Spalten dokumentiert. Abschließend wurden für einen Leitfaden zur Rekonstruktion von Chats SQL-Befehle erstellt, welche zum einen eine Zusammenfassung von Informationen zu Konversationen und zum anderen eine Wiederherstellung von Chatverläufen der einzelnen Konversationen möglich machen sollen.

Abstract:

The goal of the thesis is to analyze the storage structure of the chat application Viber under the operating system iOS within a forensic analysis. In addition, files are to be analyzed in order to extract relevant information about chats and make it evaluable. When analyzing the current state of research, hardly any work was found that dealt with an in-depth forensic analysis of Viber on iOS devices. For the analysis, test data was created in the form of individual and group chats on iOS devices and the devices were then read out. By using the UFED Touch2 readout device, the filing structure could be analyzed and documented through the UFED Reader. The analysis of the files brought out information from the two main databases 'Settings.data' and 'Contacts.data'. Here, all tables were analyzed and information about the relevant columns was documented. Finally, SQL commands were created for a guide to the reconstruction of chats, which on the one hand should make a reconstruction of information on conversations and on the other hand a reconstruction of chat histories of the individual conversations possible.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

API ..... Application Programming Interface

APP ..... Application

BLOB ..... Binary Large Object

DBMS ..... Database Management System

Exif ..... Exchangeable Image File Format

GIF ..... Graphics Interchange Format

Hex ..... Hexadezimal

ID ..... Identifier

IT ..... Informationstechnologie

JPEG ..... Joint Photographic Experts Group

PIN ..... Persönliche Identifikationsnummer

Plist ..... Property List

SIM ..... Subscriber Identity Module

SMS ..... Short Message Service

SQL ..... Structured Query Language

UFED ..... Universal Forensics Extraction Device

VoIP ..... Voice over Internet Protocol

Danksagung

Diese Arbeit wurde in Kooperation mit der Firma FAST-DETECT GmbH aus Unterhaching erstellt.

An dieser Stelle möchte ich mich bei allen bedanken, die mich während des Fernstudiums unterstützt haben. Zuerst möchte ich meinem Arbeitgeber FAST-DETECT dafür danken, dass mir dieses Studium parallel zur Arbeit zeitlich und auch finanziell ermöglicht wurde. Besonderen Dank möchte ich meinen Kollegen Stefan Fuchs und Finn Hohlfeld aussprechen, welche mir zu jeder Zeit mit Rat und Tat zur Seite standen und mich ermutigt haben das Fernstudium zu absolvieren.

Darüber hinaus möchte ich mich bei allen Freunden, Verwandten und Kollegen bedanken, die mich durch das Überprüfen der Bachelorarbeit auf inhaltliche, sprachliche und auch formelle Mängel unterstützt haben.

Am Ende gilt der größte Dank meiner Familie. Im Besonderen meiner Frau, die mich immer bestärkt und ermutigt hat, mir immer den Rücken freigehalten hat, um mich auf Prüfungen und die Tutorien vorbereiten zu können und nach der Geburt unserer Tochter stellenweise auf meine helfende Hand verzichten musste. Auch meiner Tochter möchte ich danken, da sie viele Wochenenden und Abende auf ihren Vater verzichten musste.

1 Einleitung

In der heutigen Zeit, dem digitalen Zeitalter, in welcher der Mensch meist einen digitalen Lebensstil anstrebt, ist der Faktor des ständigen 'im Kontakt stehen' ein großer Aspekt. Insbesondere mit dem Mobilfunknetz verbundene mobile Geräte, wie Smartphones und Tablets, tragen maßgeblich zu diesem Lebensstil bei. Dies zeigt sich an der stetig steigenden Anzahl an Smartphone-Nutzern in Deutschland. Hierbei geht aus einer von VuMA[1] veröffentlichten Statistik hervor, dass 2021 in Deutschland über 62 Millionen Menschen ein Smartphone nutzten. Laut einer weiteren Statistik nutzten 69 Prozent der deutschsprachigen Bevölkerung in Deutschland im Jahr 2022 mindestens einmal pro Woche das 'mobile Internet'. [1] [2] [3]

Ein großer Teil der Internetnutzung auf mobilen Endgeräten wird über Application (APP)s geführt. Die meist genutzte APP-Rubrik ist die der Messenger Anwendungen. Also Anwendungen, welche das sogenannte 'Instant Messaging' als Hauptfunktion haben. Bei der Betrachtung der Nutzerzahlen der meist genutzten Anwendungen, wird deutlich, welchen Anteil Messenger Anwendungen haben: WhatsApp mit ca. 2 Milliarden Nutzern, WeChat mit 1,3 Milliarden Nutzern und Facebook Messenger mit 0,9 Milliarden Nutzern. [4] [5]

1.1 Problem

Auch wenn in der oben erwähnten Statistik, mit den am weitesten verbreiteten Messenger Anwendungen, die Chatanwendung Viber mit seinen 1,3 Milliarden Nutzern (Stand September 2022) nicht aufgezählt ist, erlangt diese mit einer Anzahl von 3,34 Millionen Downloads pro Monat (Stand Januar 2022) sowie einer Nutzung durch 4 Prozent der Bevölkerung in Deutschland (Stand März 2023), immer mehr Beachtung. Kommunikationsanwendungen mit solch hohen Nutzerzahlen haben meist hohe Relevanz in der stetig steigenden Anzahl an Fällen zu Straftaten, die im Bereich Cybercrime verübt werden. [6] [7] [8] [9]

Eine tiefgehende forensische Analyse der Chatanwendung Viber unter iOS wurde bisher nicht veröffentlicht. In den meisten veröffentlichten Arbeiten, welche die forensische Analyse der Anwendung zum Gegenstand haben, wurde Viber unter dem Betriebssystem Android betrachtet. Eine forensische Analyse der Anwendung unter dem Betriebssystem iOS wurde bisher nur oberflächlich durchgeführt und veröffentlicht. Aufgrund des Marktanteils des Betriebssystems Android im Vergleich zu iOS scheint dies eine logische Konsequenz zu sein. Jedoch ist der Anteil der genutzten iOS-Geräte zu hoch, um eine Betrachtung von Viber auf diesem Betriebssystem zu ignorieren. Dies wird unter anderem durch die Betrachtung der meistverkauften Smartphone-Modelle in Deutschland im Jahr 2023 sowie die beliebtesten Smartphone-Marken in Deutschland gleichen Jahres unterstrichen. Hieraus ist ersichtlich, dass die am häufigsten verkauften Modelle von der Marke Apple vertrieben werden, welche das Betriebssystem iOS verwenden. Darüber hinaus liegt die Marke Apple bei den beliebtesten Smartphone-Marken auf dem zweiten Platz. [10] [11] [12]

[1] https://rms.de/wissen-und-planung/planungstools/vuma-touchpoints

1.2 Motivation der Arbeit

Auf dem aktuellen Stand der Technik zu bleiben, ist eine der wichtigsten Voraussetzungen für die Arbeit als IT-Forensik Analyst. Es ist notwendig, die Funktionsweise von Programmen zu verstehen und verständlich wiederzugeben. Auch die Analyse von neuen Funktionen oder Änderungen an der bisherigen Funktionsweise durch Aktualisierungen der Anwendungen ist Teil des stetigen Aufgabenbereichs eines Analysten.

Die Analyse der Chatanwendung Viber rückt aus den zuvor genannten Nutzerzahlen sowie der aus dem eigenen beruflichen Hintergrund als IT-Forensik Analyst entstandenen Erfahrung, dass Viber immer häufiger im Zusammenhang mit strafrechtlichen Verfahren steht, immer mehr in den Fokus der Analysten. Mit der Arbeit soll die Möglichkeit geschaffen werden, im beruflichen Umfeld eine automatisierte Auswertung der Chatanwendung Viber zu realisieren.

1.3 Zielstellung

Das Ziel der Arbeit soll eine tiefgehende forensische Analyse der Chatanwendung Viber auf iOS-Geräten sein. Um die Vorgehensweise und das detailliertere Ziel festlegen zu können, soll eine Analyse des aktuellen Forschungsstandes in diesem Bereich durchgeführt werden. Hierbei soll betrachtet werden, welche Erkenntnisse bereits bekannt sind und mit welcher Vorgehensweise diese erlangt wurden. Weiter sollen nicht nur Publikationen herangezogen werden, welche die forensische Analyse unter dem Betriebssystem iOS zum Gegenstand haben, sondern auch solche, welche die Anwendung Viber unter dem Betriebssystem Android analysiert haben. Die daraus gewonnenen Erkenntnisse sollen dokumentiert werden.

Während der Betrachtung des aktuellen Forschungsstandes wurde ersichtlich, dass die forensische Analyse der Chatanwendung Viber unter dem Betriebssystem iOS bisher kaum durchgeführt wurde. Daher wurde als Ziel dieser Arbeit festgelegt, dass die Ablagestruktur der Anwendung Viber im Dateisystem von iOS-Geräten analysiert und aufgezeigt werden soll. In diesem Zuge sollen Dateien in dieser Struktur analysiert und hinsichtlich ihrer Relevanz bewertet werden. Hierbei werden solche Dateien als relevant bezeichnet, welche Informationen zu Chatverläufen, Kontakten, dem lokalen Nutzer und Einstellungen der Anwendung enthalten. Relevante Dateien sollen interpretiert und analysiert werden. Festgestellte Informationen sollen dokumentiert werden. Weiter soll ein Leitfaden erstellt werden, durch welchen die Rekonstruktion von Chatverläufen möglich ist. Hierbei soll auf eine Möglichkeit zur Extraktion von relevanten Chatverläufen, Nachrichten, Dateiübertragungen und Kontaktdaten eingegangen werden.

Zusammenfassend sollen folgende Fragen durch die Arbeit beantwortet werden:

- Wie stellt sich die Ablagestruktur von Protokolldaten, Dateien und Datenbanken der Chatanwendung Viber auf iOS dar?
- In welcher Form werden Kommunikationsartefakte gespeichert?
- Wie entstehen Kommunikationsartefakte?
- Wie können Chatverläufe rekonstruiert werden?
- Welche Erkenntnisse können für strafrechtliche Verfahren gewonnen werden?
- Können vom Nutzer gelöschte Daten wiederhergestellt werden?

1.4 Aufbau der Arbeit

Zu Beginn der Arbeit wird auf den aktuellen Forschungsstand der forensischen Analyse der Chatanwendung Viber eingegangen. Hierbei wird auf die Analyse unter den Betriebssystemen iOS und Android eingegangen.

Hiernach folgt die Erörterung der technischen Grundlagen, welche in die Bereiche allgemeine Grundlagen, Betriebssysteme und Instant Messaging unterteilt sind. Im Bereich der allgemeinen Grundlagen werden digitale Spuren, Auslesemethoden von mobilen Endgeräten, Datenbanken und das SQLite-Format näher beschrieben. Im Abschnitt Betriebssysteme werden diese grundlegend erläutert und im Speziellen auf die Eigenschaften der Betriebssysteme iOS und Android eingegangen. Informationen zum Instant Messaging sind im gleichnamigen Abschnitt erläutert. Darüber hinaus sind hier Informationen zur Chatanwendung Viber und dessen Funktionen abgedruckt.

Im folgenden Kapitel wird darauf eingegangen, wie für die forensische Analyse methodisch vorgegangen wurde und welche Vorbereitungen für die Arbeit im Vorfeld nötig waren. Daraufhin wird auf die Testdatengenerierung eingegangen. Wie die Analyse der Testdaten durchgeführt worden ist, wird im letzten Abschnitt des Kapitels beschrieben.

Das Kapitel Ergebnis enthält die aus der Analyse resultierenden Erkenntnisse. Hierbei wird das Kapitel in die Ablagestruktur und Datenbankanalyse aufgeteilt. Der Abschnitt der Datenbankanalyse wird in die Ergebnisse aus den einzelnen Datenbanken unterteilt. Dabei wird zuerst auf die Informationen aus der Datenbank 'Settings.data', dann auf die aus der Datenbank 'Contacts.data' und zum Schluss auf Informationen aus weiteren festgestellten Datenbanken eingegangen.

Darauf folgend wird der Leitfaden zur Rekonstruktion von Chats erläutert. Das Kapitel ist aufgeteilt in die Rekonstruktion von allgemeinen Informationen zu Konversationen und die Rekonstruktion von Chatverläufen zu diesen Konversationen.

Abschließend wird ein Fazit zur Arbeit gegeben und betrachtet, welche Themen zukünftige forensische Analysen der Chatanwendung Viber zum Gegenstand haben könnten.

2 Verwandte Literatur

In diesem Abschnitt wird auf den aktuellen Forschungsstand hinsichtlich der forensischen Analyse der Chatanwendung Viber auf iOS-Geräten eingegangen. Darüber hinaus wird betrachtet, wie sich der aktuelle Sachstand bei der forensischen Analyse auf Android-Geräten verhält.

2.1 Forschungsstand: Viber auf iOS-Geräten

Die forensische Analyse von Viber auf iOS-Geräten ist bisher in wenigen Veröffentlichungen zum Gegenstand gemacht worden. Welche Erkenntnisse bisher veröffentlicht worden sind, wird im Folgenden erläutert.

Die erste Arbeit, welche sich mit der forensischen Analyse von Viber auf iOS-Geräten befasst ist der Artikel *'Forensics Acquisition and Analysis of Instant Messaging and VoIP Applications'* aus dem Jahr 2015. Inhalt des Artikels ist die Analyse von Anwendungen aus den Bereichen 'Instant Messaging' und 'Voice over Internet Protocol (VoIP)'. Die Autoren geben an, dass es sich bei der Datei 'Contacts.data' um die wichtigste Datei der Anwendung Viber handelt. Hierin werden Informationen zu Kontakten, ausgetauschten Dateien, Konversationen, Telefonaten und Nachrichten gespeichert. Darüber hinaus werden keine weiteren Aussagen zur Analyse von Viber auf iOS-Geräten getroffen. [13]

Die zweite Arbeit, welche Viber auf iOS-Geräten erwähnt, ist der im Jahr 2021 veröffentlichte Artikel *'What's on the Horizon? An In-Depth Forensic Analysis of Android and iOS Applications'*. Der Artikel befasst sich mit der Analyse von Anwendungen unter den Betriebssystemen iOS und Android. In diesem Dokument wird hinsichtlich der Analyse von Viber auf iOS-Geräten lediglich in einem kurzen Absatz darauf eingegangen, welche APP-Identifier (ID) der Anwendung Viber auf iOS-Geräten zugeordnet wird. Weiter wird aufgezeigt, dass Bilder, welche mittels Viber versendet oder empfangen worden sind, im Joint Photographic Experts Group (JPEG)-Format im Verzeichnis '/com.viber/AttachmentsPreview/' mit der Dateiendung '.jpg' gespeichert werden. Nachrichten, welche über Viber versendet bzw. empfangen worden sind und die Telefonnummern der Kontakte werden laut den Autoren in der Datenbank 'Contacts.data' gespeichert. Hierbei wird auf die in der Datenbank befindlichen Tabellen 'ZVIBERMESSAGE', 'VMEMBER' und 'ZPHONENUMBER' hingewiesen. Am Ende des Absatzes wird erwähnt, dass es eine Tabelle für gelöschte Nachrichten geben soll, jedoch nicht verifiziert werden konnte, ob sich gelöschte Nachrichten hieraus wiederherstellen lassen. Darüber hinaus wird auch nicht erwähnt, wie diese Tabelle benannt ist. [14]

2.2 Forschungsstand: Viber auf Android-Geräten

Der Forschungsstand in der forensischen Analyse von Viber auf Android-Geräten ist im Gegensatz zur Analyse auf iOS-Geräten weiter fortgeschritten. Aus den Publikationen geht hervor, dass die Anwendung Viber unter den beiden Betriebssystemen Android und iOS teilweise sehr unterschiedlich realisiert wurde. Dies ist zum Beispiel an der Ablagestruktur, jedoch auch an der Aufteilung der Daten in mehrere Datenbanken sowie deren unterschiedliche Strukturierung ersichtlich.

Im Folgenden werden die Erkenntnisse aus bisherigen Veröffentlichungen dargelegt, um zum einen den Unterschied näher zu beschreiben, zum anderen zu verdeutlichen, inwieweit sich die Anwendung Viber und deren Aufbau unter dem Betriebssystem Android über die Jahre hinweg verändert hat.

Eines der ersten veröffentlichten Dokumente zum Thema forensische Analyse von Viber auf Android-Geräten stammt aus dem Jahr 2013 und trägt den Titel '*Forensic Analysis of Instant Messenger Applications on Android Devices*'. In dieser Arbeit werden die Anwendungen WhatsApp und Viber IT-forensisch untersucht. Wobei für Viber lediglich die Funktionen des 'Online Voice Calls' (dt. Online Sprachanrufe) und der 'Online SMS' durchgeführt und analysiert worden sind. Im Abschnitt zur Anwendung Viber wird angegeben, dass im Pfad '/data/data/com.viber.voip/' das Verzeichnis 'databases' gefunden wurde und hierin die drei relevante Datenbanken 'viber_call_log.db', 'viber_messages' und 'viber_data' feststellbar waren. Darüber hinaus werden die in den Datenbanken enthaltenen Tabellen erwähnt. Im weiteren Verlauf der Arbeit werden feststellbare Artefakte aus den Datenbanken grob beschrieben. Jedoch wird nicht auf einzelne Spalten, deren Bezeichnung und Bedeutung eingegangen. Da lediglich zwei Funktionen der Anwendung getestet wurden, wurden auch nur Artefakte zu diesen Funktionen erwähnt. Auf Textnachrichten, Sprachnachrichten, Information zu geführten Konversationen oder der Ablagestruktur von gesendeten bzw. empfangenen Dateien wurde nicht eingegangen. [15]

Die im Jahr 2015 veröffentlichte Arbeit '*Forensics Acquisition and Analysis of Instant Messaging and VoIP Applications*' behandelt Viber unter iOS und Android. Hierbei wird darauf eingegangen, dass bei einer Extraktion der Daten über den UFED Physical Analyzer lediglich mittels Volltextsuche nach dem Suchbegriff 'Viber' Spuren feststellbar waren. Hierbei konnten nur Mediendateien zur Anwendung gefunden werden. Dies wird abschließend als initiale Identifikation der Anwendung beschrieben. Eine genauere Analyse der Anwendung wurde manuell durchgeführt. Bei der manuellen Analyse wurden die Datenbanken händisch betrachtet und hierbei konnte festgestellt werden, dass sich die Datenbanken der Chatanwendung Viber unter Android wesentlich zu den Datenbanken unter iOS unterscheiden. In diesem Zusammenhang wird auf die Ablagestruktur der Mediendateien und Datenbanken eingegangen. Hierbei werden Verzeichnisse wie 'User Photos', 'Viber Images' und 'Viber Video' erwähnt. Die Datenbanken sollen im Verzeichnis '/data/data/com.viber.voip/' auffindbar sein. Beschrieben werden die Datenbanken 'viber_message', welche Informationen zu geführten Konversationen enthält, sowie 'viber_data', welche Daten zu Telefonaten und dem Kontaktbuch beinhaltet. Es wird hier nicht auf den Aufbau der Datenbanken oder deren Tabellen eingegangen. [13]

Ein weitere im Jahr 2015 veröffentlichte Publikation trägt den Titel '*Implementation of Forensic Analysis Procedures for WhatsApp and Viber Android Applications*'. Inhalt der Arbeit ist die forensische Analyse der Chatanwendungen WhatsApp und Viber unter dem Betriebssystem Android sowie der Vergleich der Anwendungen untereinander. In der Arbeit wird auf das Datenbankschema von Viber eingegangen. Erwähnt wird das Verzeichnis '/data/data/com.viber.voip/databases/' und die darin gefundenen SQLite-Datenbanken 'viber_data' und 'viber_messages'. Die Autoren beschreiben kurz den Inhalt der Datenbanken und stellen den Aufbau der zwei Datenbanken in Bildschirmaufnahmen dar. [16]

In der aktuellsten gefundenen Arbeit aus dem Jahr 2022 mit dem Titel '*Android Device Incident Response: Viber Analysis*' wurde die Anwendung Viber am tiefgehendsten analysiert. Dies ist das einzige Dokument, welches sich allein mit der forensischen Analyse der Chatanwendung Viber befasst. Anfänglich wird auf die Ablagestruktur der Anwendung eingegangen, indem tabellarisch Verzeichnisse und deren Beschreibung aufgezeigt werden. Methodisch wurde so vorgegangen, dass Testdaten auf zwei Android-Geräten erstellt worden sind. Die Geräte wurden mehrfach zu unterschiedlichen Zeitpunkten ausgelesen. Die Zeitpunkte sowie Gründe hierfür werden in einer Tabelle dargestellt. Weiter wird in einer Tabelle detailliert aufgezeigt, welche Programme während der Analyse zu welchem Zweck eingesetzt worden sind. Die Ergebnisse, welche in der Arbeit aufgezeigt werden, werden auf die Datenbank 'viber_messages.db' zurückgeführt. Eine Erwähnung der Datenbank 'viber_data', wie in den anderen Dokumenten, findet nicht statt. Jedoch wird im Gegensatz zu den anderen Dokumenten eine detaillierte Analyse der Datenbank durchgeführt, indem aufgezeigt wird, wie viele Tabellen in der Datenbank enthalten sind, welche Tabellen als relevant erachtet werden und die relevanten Spalten zumindest einer dieser Tabelle tabellarisch mit deren Bedeutung dargestellt werden. Darüber hinaus wird ein Structured Query Language (SQL)-Befehl aufgezeigt, der die relevanten Inhalte anzeigen soll und diese teilweise auch interpretiert. Dies wird zum Beispiel durch die Differenzierung der Werte 0,1,2 in Ihrer Bedeutung 'Nachricht nicht gesendet', 'Nachricht nicht empfangen' und 'Nachricht empfangen' durchgeführt. Zu diesem SQL-Befehl wird klargestellt, dass der Befehl für die getestete Viber-Version entworfen worden ist und durch ein Update der Anwendung und etwaige Änderung am Datenbankschema durch die Entwickler, der Befehl bei neueren Versionen der Anwendung nicht mehr anwendbar sein könnte. [17]

3 Technische Grundlagen

In diesem Kapitel werden technische Grundlagen näher erläutert. Im ersten Abschnitt werden allgemeine Grundlagen erläutert. Im weiteren Verlauf wird auf das Themengebiet der Betriebssysteme eingegangen. Abschließend wird der Bereich des Instant Messagings näher beleuchtet.

3.1 Allgemeine Grundlagen

In diesem Abschnitt wird auf grundlegende Aspekte der digitalen Spuren und ihre Entstehung eingegangen. Darüber hinaus werden Auslesemethoden mobiler Endgeräte erläutert. Abschließend werden Datenbanken näher beleuchtet und auf das Format SQLite eingegangen.

3.1.1 Digitale Spuren

Digitale Spuren sind Spuren, welche in IT-Systemen gespeichert wurden oder zwischen solchen übertragen worden sind. Im Kontext der IT-Forensik und digitalen Spuren wird häufig auch vom virtuellen Tatort gesprochen. Dieser ist im Bereich der Cyberkriminalität schwer zu bestimmen, da keine physische Interaktion zwischen Täter und Opfer stattfindet. Daher wird ein sogenannter Hilfstatort bestimmt, welcher meist dort ist, wo der Schaden der Tat eintritt. Digitale Spuren entstehen hierbei beim Täter-System, beim Opfer-System und in einer potenziellen stattgefundenen Übertragung zwischen diesen Systemen. Dabei können digitale Spuren in Log-Dateien, Verbindungsdaten (beim Internetprovider), in temporären Speicherbereichen (Cache) oder aber auch in gelöschten Speicherbereichen vorliegen. Welche Art von digitalen Spuren relevant sind, muss von Fall zu Fall individuell bestimmt werden. In dieser Arbeit wird hauptsächlich auf digitale Spuren in Form von Datenbanken, Mediendateien und temporären Speicherbereichen eingegangen. [18] [19]

3.1.2 Auslesemethoden

Es gibt mehrere Möglichkeiten, Daten von mobilen Geräten mit dem Betriebssystem iOS zu extrahieren. Welche Vorgehensweise am besten geeignet ist, hängt vom vorliegendem Modell des Gerätes ab. Darüber hinaus ist auch die iOS-Version ausschlaggebend. Dies zeigt sich zum Beispiel bei den beiden Testgeräten, welche für diese Arbeit genutzt wurden.

Das iPhone 8 Plus, welches mit der iOS-Version 14.6 genutzt wurde, konnte über eine Sicherheitslücke in der Firmware des von Apple implementiertem Chips ausgelesen werden. Dieser Exploit[2] wurde von dem Hacker axiOmX unter der Bezeichnung Checkm8 - in Anlehnung an Checkmate (dt.: Schachmatt) - auf dem sozialen Netzwerk Twitter [3] veröffentlicht. Über diese Sicherheitslücke konnte das komplette Dateisystem unverschlüsselt extrahiert werden. [20]

Im Gegensatz hierzu steht das verwendete iPhone Xs mit der iOS-Version 15.4.1. Ist die Verwendung

[2] https://www.computerweekly.com/de/definition/Exploit
[3] https://twitter.com/axi0mX/status/1177542201670168576

von checkm8 noch bis zum Modell iPhone X anwendbar, ist die Sicherheitslücke im Chip des iPhone Xs nicht mehr vorhanden. Somit konnte dieses Gerät lediglich logisch ausgelesen werden.
Dies verdeutlicht die Bedeutung des vorliegenden Modells hinsichtlich der am besten anwendbaren Extraktionsmethode. Die unterschiedlichen Extraktionsverfahren sowie deren Vor- und Nachteile werden im Folgenden kurz erläutert.

Manuelle Extraktion

Diese Methode ist im forensischen Aspekt eine der unsichersten, da hierbei durch menschliche Fehler Daten gelöscht oder verändert werden können. Hierbei wird das Gerät gestartet, manuell gesichtet und Beweise durch Bildschirmaufnahmen gesichert. Bei dieser Methode können nur Daten gesichert werden, welche auf dem Bildschirm angezeigt werden. Diese sind häufig nicht vollständig, da oftmals Protokolle oder Systemdateien dem normalen Nutzer nicht zugänglich sind. Darüber hinaus gibt es keine Möglichkeit, auf gelöschte Daten zuzugreifen. Diese Methode wird häufig als Ergänzung verwendet, um Erkenntnisse aus den im Folgenden beschriebenen Methoden zu verifizieren. Auch kann sie sinnvoll sein, wenn Daten aus den anderen Methoden lediglich verschlüsselt vorliegen. [21]

Logische / Dateisystem Extraktion

Mittels dieser Methode können Daten gesichert werden, welche dem Nutzer sichtbar sind. Diese Methode ist äquivalent zu einem iTunes-Backup. Hierbei ist die Kenntnis über den Entsperrcode des Gerätes zwingend nötig. Darüber hinaus muss angegeben werden, dass das aus der Extraktion entstehende Backup verschlüsselt werden soll. Dies begründet sich auf der Tatsache, dass Entwickler von Applikationen die Sicherung ihrer Applikation in einem iTunes-Backup verhindern können, wenn das Backup nicht verschlüsselt wird. Inhalt einer solchen Extraktion sind Dateien in folgenden Formaten [22]:

- SQLite Datenbanken
- Klartext plist-Dateien
- Binär plist-Dateien
- Medien- und Text-Dateien
- Nicht standardisierte Datenformate

Logische / Vollständiges Dateisystem Extraktion

Für Modelle, die jünger als das iPhone 4s sind und älter als das iPhone Xs, kann das vollständige Dateisystem extrahiert werden. Hierbei wird das iPhone durch einen entweder temporären oder permanenten 'Jailbreak' ausgelesen. Die Extraktion kann zum Beispiel über einen bekannten Exploit (Bsp. checkm8) durchgeführt werden. Bei dem Vorgang des 'jailbreakens' eines iPhones werden die Sicherheitsmaßnahmen, welche durch Apple implementiert wurden, außer Kraft gesetzt, um Zugriff auf alle Daten auf dem Gerät zu erlangen. [23]

Physische Extraktion

Eine physische Extraktion beinhaltet alle Daten sowie die Informationen des Dateisystems auf Bit-Ebene. Diese Methode ist jedoch nur bis zum Modell iPhone 4s anwendbar, da Apple ab diesem Modell eine weitere Verschlüsselungsebene eingeführt hat. Hierbei wird die Betriebssystempartition von der Speicher-Partition getrennt, sodass der Speicher weiterhin verschlüsselt bleibt, auch wenn das Betriebssystem infiltriert wurde.

3.1.3 Datenbanken

Definition Datenbank: 'Eine Datenbank ist eine Sammlung von Daten, die untereinander in einer logischen Beziehung stehen und von einem eigenen Datenbankverwaltungssystem (Database Management System (DBMS)) verwaltet werden.'[24]
Für Datenbanken können folgende Anforderungen gestellt werden:

- Sammlung von Daten, die logisch verbunden sind
- Vermeidung von Redundanz
- Zugriffs- und Änderungsmöglichkeiten

3.1.3.1 Relationales Datenbankmodell

Datenbanken können in drei Modelle kategorisiert werden:

- Hierarchische Datenbanken
- Relationale Datenbanken
- Objektorientierte Datenbanken

Aufgrund der Relevanz für diese Arbeit, wird im Folgenden nur auf das relationale Datenbankmodell eingegangen. Relationale Datenbanken speichern Daten nach Themen (Entitäten) in Form von Tabellen ab. Hierbei können Tabellen bzw. Spalten in Beziehung zu anderen Tabellen bzw. Spalten stehen. Hierfür gibt es sogenannte Identifikationsschlüssel, Primärschlüssel und Fremdschlüssel. Hierbei stellen Spalten die Schlüssel dar, wobei auch mehrere Spalten zusammen ein Schlüssel sein können. Die Werte in der Spalte, welche als Identifikationsschlüssel einer Tabelle dient, dürfen nur einmal in der Spalte vorkommen. Auch darf sich dieser Wert, sobald einmal festgelegt, nicht mehr nachträglich ändern. Für den Primärschlüssel gelten im Grunde die gleichen Gegebenheiten, wie für den Identifikationsschlüssel. Der Unterschied besteht darin, dass der Identifikationsschlüssel auf der logischen Ebene und der Primärschlüssel auf der physikalischen Ebene arbeitet. Es kann pro Tabelle nur einen Primärschlüssel geben. Mit Hilfe eines Fremdschlüssels können Werte einer Tabelle den Werten aus einer weiteren Tabelle anhand dessen Identifikations- oder Primärschlüssel zugeordnet werden. Fremdschlüssel können beliebig oft vorkommen und beliebig viele Spalten vereinen. [25]

3.1.3.2 SQLite

Daten werden häufig in Form von Tabellen in Datenbanken gespeichert. So auch im Format SQLite in der Version 3. Bei SQLite handelt es sich um eine portable relationale Datenbank. SQLite wurde für Anwendungen konzeptioniert, welche eine schlanke Datenspeicherung nutzen wollen, da dieses Datenbankformat keinen Datenbankserver benötigt, sondern alle Daten in einer Datei speichert. Ein weiterer Vorteil dieses Formats ist die Unabhängigkeit des Betriebssystems. Das bedeutet, es kann, wie im Falle von Viber, sowohl unter dem Betriebssystem Android als auch unter iOS genutzt werden. Dies macht es für Entwickler, welche eine Anwendung auf beiden Betriebssystemen anbieten wollen, einfach, da sie nicht zwei Datenbankstrukturen entwerfen müssen. SQLite kann über eine Application Programming Interface (API)-Schnittstelle angesprochen und Daten unter Verwendung der sogenannten SQL erstellt, abgefragt oder verändert werden. [26]

3.2 Betriebssysteme

In diesem Abschnitt wird ein Überblick über die allgemeinen Funktionen eines Betriebssystems erläutert. Darüber hinaus wird näher auf die Betriebssysteme iOS und Android, welche für mobile Endgeräte entwickelt worden sind, eingegangen.

3.2.1 Betriebssysteme - Überblick

Ein Betriebssystem dient als Schnittstelle zwischen dem Anwender und der Hardware des Computers. Es ist zuständig für die Kommunikation, welche mit und zwischen der Hardware stattfindet, die Verwaltung des Speichers und die Durchführung von Anwenderprogrammen. [27]

Grundlegend hat ein Betriebssystem folgende zwei Aufgaben:

- 'Die Komplexität des darunter liegenden Rechners zu verstecken. Es soll dem Computeranwender eine leicht verständliche und gut handhabbare Schnittstelle zur eigentlichen Maschine anbieten. Der Anwender arbeitet nicht mehr mit der wirklichen Maschine, sondern mit einer virtuellen Maschine (Betriebssystem), welche wesentlich einfacher und benutzerfreundlicher ist.' [27]
- Verwaltung der einzelnen Bestandteile des Computers durch eine ordentliche Zuteilung von Prozessen, Speicherbereichen sowie Eingabe- und Ausgabe-Geräten zu den einzelnen Anwendungen. [27]

Auf Heimrechnern sind weit verbreitete Vertreter Microsoft Windows und Apple macOS [28].
Auf mobilen Endgeräten sind Google Android und Apple iOS die Betriebssysteme mit den meisten Marktanteilen. [29]

3.2.2 iOS (Apple)

iOS ist ein von Apple entwickeltes Betriebssystem für mobile Endgeräte. Es wurde im Januar 2007 von Apple Mitbegründer Steve Jobs vorgestellt. Damals wurde es von ihm mit den Worten 'iPhone runs OS Ten(X)' (Auf dem iPhone läuft OS 10(X)) betitelt [30]. Bis 2010 war das Betriebssystem für Apples Mobilgeräte als 'iPhoneOS' oder 'iPhone Software' bekannt. Erst 2010 wurde das Betriebssystem offiziell iOS getauft und fortan auch die älteren Versionen so betitelt. Die Bezeichnung iOS wird für die Betriebssystem-Versionen genutzt, welche auf den Modellen der Smartphone-Reihe iPhone und der mp3-Player-Reihe iPod Touch installiert werden. Bis zur Version 13 des Betriebssystems wurde die Bezeichnung iOS auch für die Tablet-Modelle des iPads verwendet. Ab der Version 13 bekam das Betriebssystem der iPads im Juni 2019 den eigenen Namen iPadOS. Die aktuelle Version von iOS beläuft sich auf die Nummer 16. Auf der Webseite[4] von Apple ist eine Liste mit kompatiblen Geräten zu finden. [31]

Die Abbildungen 3.1 und 3.2 zeigt die Desktopansicht der beiden Testgeräte.

[4] https://www.apple.com/de/ios/ios-16/

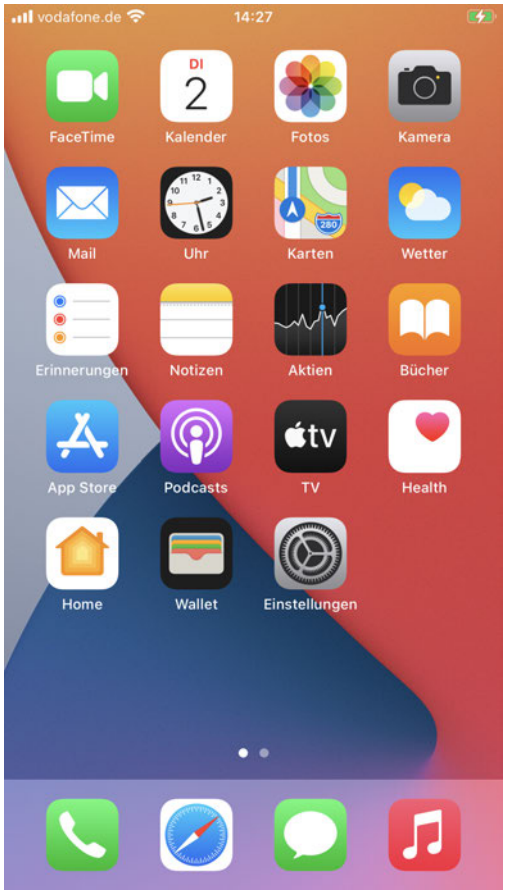

Abbildung 3.1: Desktopansicht des iPhone 8 Plus

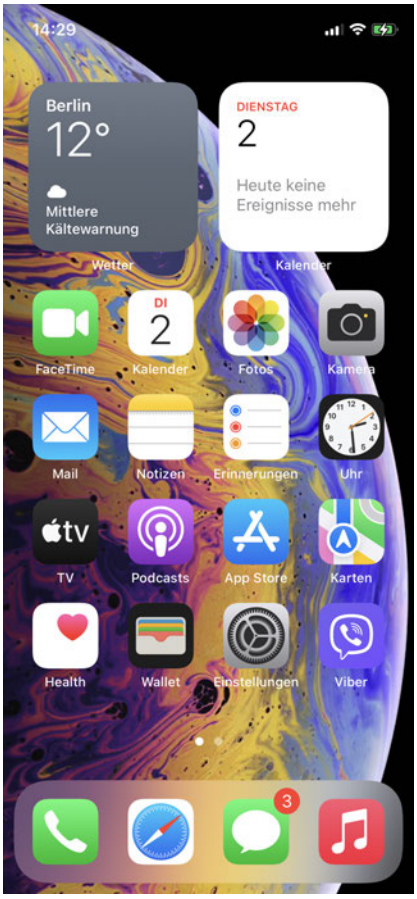

Abbildung 3.2: Desktopansicht des iPhone Xs

Die Architektur des Betriebssystems iOS ist in Abbildung 3.3 dargestellt. [21]

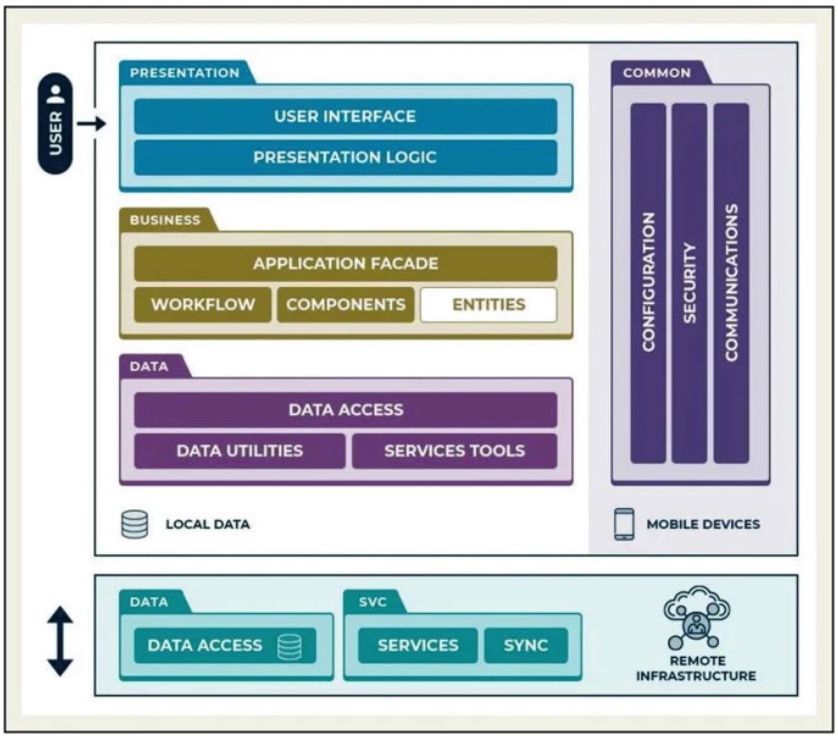

Abbildung 3.3: Architektur des Betriebssystems iOS

Das Betriebssystem iOS kann in folgende vier Schichten unterteilt werden: [21]

Cocoa Touch Schicht
Dies ist die oberste Schicht des Betriebssystems. Hierin befinden sich grundlegende Programmiergerüste für Entwickler sowie Schnittstellen für Dienste, wie die Berührungsfunktion des Berührungsbildschirms, die Multi-Berührungsfunktion sowie weitere Eingabemöglichkeiten.

Medien Schicht
Auf dieser Schicht sind Programmiergerüste für Audio-, Video- und Bildformate. Diese sollen Entwickler dabei unterstützen, grafisch eindrucksvolle Applikationen zu erstellen.

Kern Dienste Schicht
Auf dieser Schicht sind Dienste angesiedelt, welche Applikationen nutzen können. Hierunter fallen zum Beispiel der Standortdienst, Kommunikationsdienste oder die Schnittstelle zu dem von Apple bereitgestellten Cloudspeicherdienst iCloud.

Kernbetriebssystem Schicht
Auf der untersten Schicht des Betriebssystems befindet sich die Schnittstellen zur Hardware. Darüber hinaus wird hier die Speicherverwaltung und die Kommunikation sowie die Vernetzung durchgeführt.

3.2.3 Android (Google)

Android ist ein sogenanntes Open-Source-Betriebssystem (quelloffen), welches von der Firma Google entwickelt wird. Mit 33 Partnern aus der Mobilfunkbranche gründete Google die 'Open Handset Alliance', deren Bestreben war, Android als offenes Betriebssystem zu etablieren. Hierbei sollte der Fokus darauf liegen, dass Hersteller und Entwickler das Betriebssystem kostenlos nutzen können. Das Betriebssystem war anfangs für Smartphones und Tablets konzipiert, wird aber mittlerweile auch für Smart-TVs, Digitalkameras und Auto-Systeme entwickelt. [32] [33]

Die vier Schichten und deren Inhalten, in welche Android unterteilt werden kann, sind in Abbildung 3.4 dargestellt.

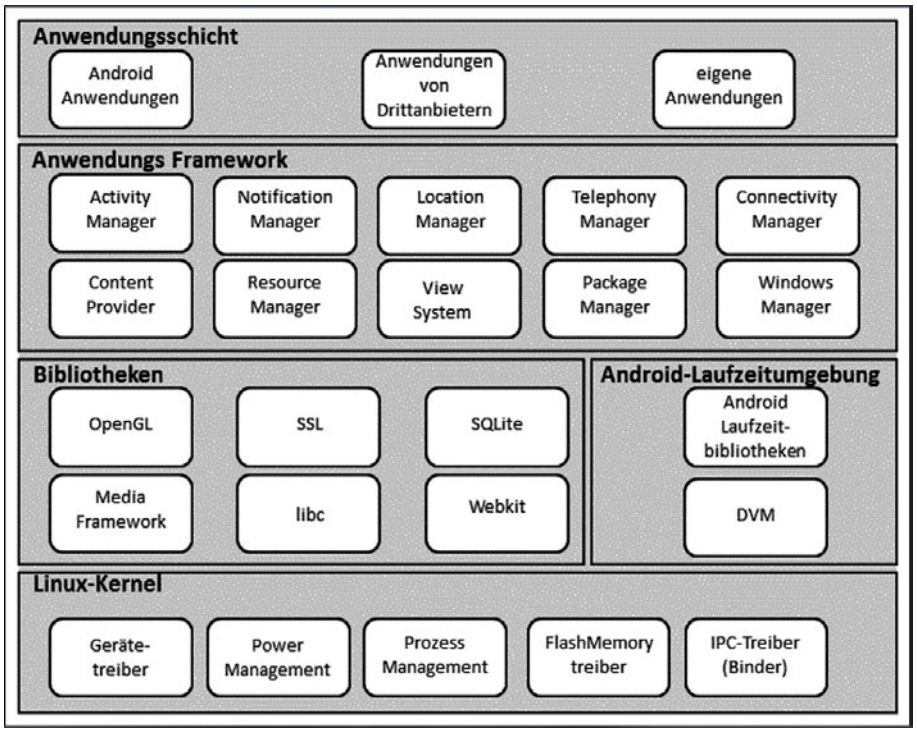

Abbildung 3.4: Schichten des Betriebssystems Android

3.3 Instant Messaging

Im Folgenden wird der Begriff Instant Messaging näher erläutert und die Funktionen der Chatanwendung Viber betrachtet.

3.3.1 Instant Messaging - Überblick

Der Begriff Instant Messaging bedeutet auf Deutsch so viel wie 'Sofortige Nachrichtenübermittlung'. Hierbei wird zumeist über Computersysteme eine synchronisierte Kommunikation von mindestens zwei Parteien geführt. Im Gegensatz hierzu steht zum Beispiel die E-Mail als asynchroner Kommunikationsansatz, bei welcher keine Möglichkeit zur direkten Antwort besteht [34].

Der von Instant Messengern heutzutage zur Verfügung gestellten Funktionsumfang umfasst folgende Funktionen:

- Textnachrichten
- Sprachnachrichten
- Mediennachrichten
- Sprach- und Videotelefonie

Voraussetzung für die Kommunikation mittels eines Instant Messengers ist die Verbindung mit dem Internet und die Nutzung eines Clients[5] des jeweiligen Anbieters (zum Beispiel Viber). Hierbei wird die Nachricht nach dem Absenden an einen Server übermittelt, welcher die Nachricht weiterleitet, sofern der Kontaktpartner mit dem Internet verbunden ist. Ein Vorteil dieser Technik ist, dass der Absender zum Zeitpunkt des Sendens der Nachricht mit dem Internet verbunden sein muss, hiernach jedoch die Verbindung kappen kann. Da die Nachricht jedoch auf dem Server zwischengespeichert wird, wird diese trotz fehlender Verbindung des Absenders an den Empfänger weitergeleitet. Die Kommunikation über solche Dienste ist zumeist verschlüsselt, sodass während der Übermittlung der Nachrichten kein Zugriff von außen stattfinden kann. Hierbei wird die Ende-zu-Ende-Verschlüsselung eingesetzt, was bedeutet, dass selbst der Anbieter des Instant Messaging Dienstes keinen Zugriff auf die Nachrichten hat.

In Abbildung 3.5 ist die Ende-zu-Ende-Verschlüsselung beispielhaft dargestellt. [35]

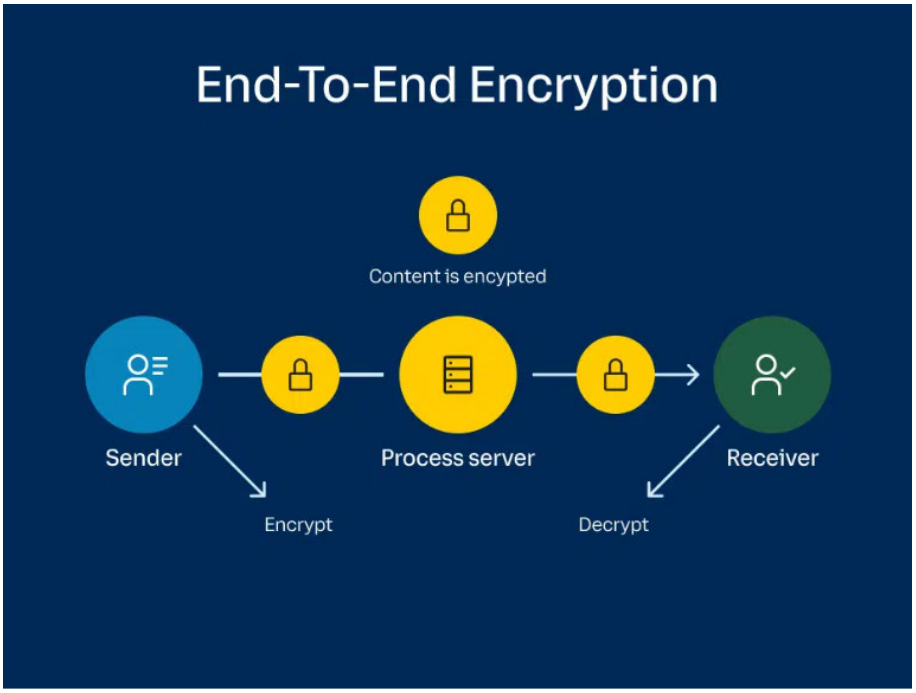

Abbildung 3.5: Darstellung einer Ende-zu-Ende-Verschlüsselung

3.3.2 Viber

Viber ist ein Instant Messaging-Dienst, welcher von der japanischen Firma Rakuten betrieben und entwickelt wird. Es ist für die mobilen Betriebssysteme Android, iOS sowie HarmonyOS verfügbar. Darüber hinaus bietet Viber die Möglichkeit, den Dienst auch auf Desktop-Computer zu nutzen. Hierfür ist ein Client für die Betriebssysteme Windows, macOS und drei Linux-Distributionen verfügbar. Im Folgenden wird auf die von Viber angebotenen Funktionen und die Sicherheit des Dienstes eingegangen. [36]

[5] https://www.it-business.de/was-ist-ein-client-a-675478/

Text- und Sprachnachrichten

Wie üblich beim Instant Messaging liegt der Fokus der Anwendung auf dem Versenden und Empfangen von Textnachrichten. Diese sind in ihrer Länge auf maximal 7.000 Zeichen begrenzt. Der Nutzer kann Textnachrichten auch formatieren (z.B. Kursiv, Fett, usw.). Nachrichten müssen jedoch nicht aus Text bestehen, sondern können auch Emoticons, Emojis [6] oder Sticker darstellen. Darüber hinaus wird dem Nutzer angezeigt, ob die Nachricht gesendet, empfangen oder gelesen wurde. Dies wird durch Haken dargestellt und ist beispielhaft in Abbildung 3.6 dargestellt. Der Anwender muss jedoch die Nachricht nicht textuell verfassen, sondern kann diese auch in Form einer Sprachnachricht verfassen. Diese kann vom Empfänger nach Erhalt abgespielt werden. [37] [38] [39]

Abbildung 3.6: Arten der Nachrichtenstatus

Medien- und Dateinachrichten

Viber bietet über die Möglichkeiten der Text- und Sprachnachrichten auch die Funktion Medien- sowie Dateinachrichten zu versenden. Hierbei können Fotos, Videos oder Audiodateien übermittelt werden. Darüber hinaus ist auch das Verschicken weiterer Dateien möglich. Viber stellt eine Liste mit verbotenen Dateiformaten zur Verfügung. [39] [37]

In der Tabelle 3.1 sind Einschränkungen für den Dateiversand beschrieben.

Tabelle 3.1: Einschränkungen für Dateiübertragungen

	Bilddateien	**Videodateien**	**Andere Dateien**
Größe	200 MB	200 MB	200 MB
Dateiformate	jpeg, png, gif	MP4, H264	Alles außer Verbotene [37]
Vorschaubild	400x400, 100 KB, jpeg	400x400, 100 KB, jpeg	Nein

Sprach- und Videoanrufe

Dem Anwender steht die Funktion der Sprachtelefonie über Internet zur Verfügung. Darüber hinaus kann der Anwender auch mit den im Gerät integrierten Kameras einen Videoanruf durchführen. Bei beiden Funktionen ist der Wechsel zum jeweils anderen auch während eines aktiven Telefonats möglich. Hierfür ist eine aktive Internetverbindung für beide Anrufarten erforderlich. [39]

[6] https://praxistipps.chip.de/emoji-und-emoticon-was-ist-der-unterschied_93465

Einzel- und Gruppenchats

Der Anwender von Viber hat die Möglichkeit in einem Einzelchat eine Kommunikation mit nur einem Kontakt zu führen. Der Gruppenchat ist für die Kommunikation mit mehreren Personen gedacht. Hierbei sind Gruppen bis zu einer Größe von 250 Teilnehmern möglich. [39] [40] Chats können ausgeblendet werden, sodass sie nicht in der Übersicht der aktiv geführten Unterhaltungen angezeigt werden. Darüber hinaus können die ausgeblendeten Chats nur mit einer zuvor vergebenen Persönliche Identifikationsnummer (PIN) geöffnet werden. [41]

Communitys und Kanäle

Eine Community ist eine Gemeinschaft, die entweder in örtlicher Nähe zueinander lebt oder eine Gemeinschaft, die ein gemeinsames Interesse, eine soziale Gruppierung oder eine Nationalität eint. In Viber ist eine Community als unlimitierte Gruppe realisiert und bietet so einen Chat für unbegrenzt viele Teilnehmer. [42] [39] [43]

Kanäle sind für eine einseitige Kommunikation gedacht. Hier können zum Beispiel Gruppierungen oder Marken Neuigkeiten veröffentlichen, welche an alle Teilnehmer verteilt werden. Auch hier ist die Teilnehmerzahl unbegrenzt. Viber bietet einen Vergleich von Gruppen, Communitys und Kanälen an, welcher Begrenzungen und Funktionen gegenüberstellend aufzeigt. [44]

Bearbeitung und Löschung von Nachrichten

Eine weitere Funktion der Chatanwendung Viber ist das Löschen von Nachrichten nach dem Versenden. Hierbei ist es dem Anwender möglich entweder die Nachricht lokal oder aber auch beim Empfänger zu löschen. Viber bietet im Gegensatz zu den meisten anderen Chatanwendungen die Möglichkeit Nachrichten, welche versendet und bereits zugestellt wurden, nachträglich zu bearbeiten. [39]

Verschwindende Nachrichten

Verschwindende Nachrichten sind sich automatisch nach einer durch den Anwender oder dessen Kontakt eingestellten Zeit löschende Nachrichten. Hierbei wird nicht eine spezielle Nachricht ausgewählt, die automatisch gelöscht wird, sondern werden mit Aktivieren dieser Funktionen alle folgenden Nachrichten mit einem Timer versehen, bis die Funktion wieder deaktiviert wird. [39]

Telefonie über das Fest- und Mobilfunknetz

Die Möglichkeit mittels der Anwendung Viber Telefonate über das Fest- und Mobilfunknetz zu führen, wird Viber Out genannt. Hierbei bietet Viber an, Rufnummern aus dem Ausland kostengünstig anrufen zu können. [45]

Notizen

Notizen können in einer Art Chat mit sich selbst gespeichert werden. Diese können digital abgehakt und auch an andere Viber-Nutzer weitergeleitet werden. Ferner kann der Anwender eine Erinnerung an eine Notiz einrichten.

GIFs und Sticker

Sticker sind größer skalierte Emoticons, welche hauptsächlich im Instant Messaging Anwendung finden. Oft werden mehrere als Paket angeboten und können starre oder auch bewegte Bilder sein. Darüber hinaus kann jedes Bild oder Foto zu einem Sticker umgewandelt und als solcher verschickt werden. Hierbei sind dem Anwender kaum Grenzen in seiner Kreativität gesetzt. Eine Anleitung zum Erstellen, Bearbeiten und Nutzen solcher Sticker bietet Viber auf ihrer Webseite an. Graphics Interchange Format (GIF) ist ein Bildformat, welches Abbildungen in einer Farbpalette von 256 Farben verlustfrei komprimiert speichern kann. Hierbei ist auch das Überlagern von mehreren Einzelbildern möglich, wodurch ein animiertes Bild entstehen kann. [46] [39] [47]

Bezahlfunktion

Viber bietet in den Ländern Deutschland und Griechenland die Möglichkeit des digitalen Bezahlens an. Hierbei können Zahlungen über Viber an einen Kontakt nahtlos getätigt werden. Eingerichtet werden können Mastercards, Visa Cards und Bankkonten.
Viber wirbt mit folgenden drei Funktionen auf ihrer Webseite [48]:

- Viber-zu-Viber Transaktionen, welche inländisch aber auch grenzübergreifend getätigt werden können
- Banküberweisungen für Routinezahlungen wie zum Beispiel Stromrechnungen
- Exklusive Rabatte und Belohnungen von lokalen Partnern

Backup

Alle Konversationen können in Viber innerhalb eines Backups gesichert werden. Dies funktioniert in Verbindung mit dem von Apple bereitgestellten Cloudspeicherdienst iCloud. Diese Sicherung ist nicht standardgemäß eingerichtet und muss durch den Anwender manuell aktiviert werden. Hierbei kann er zusätzlich folgende Einstellungen einrichten:

- Automatisches Backup
- Fotos einbeziehen
- Videos einbeziehen

Durch diese Sicherung kann ein Anwender bei einer erneuten Installation von Viber, beispielsweise auf einem neuen Gerät, alle Konversationen wiederherstellen.

Sicherheit

Viber gibt auf ihrer Webseite an, dass eine standardmäßige Ende-zu-Ende-Verschlüsselung verwendet wird. Beschrieben wird diese wie folgt: 'Von dir gesendete Nachrichten gelangen in Form eines verschlüsselten Codes, den nur das Gerät des Empfängers über einen Kodierungsschlüssel in reinen Text übersetzen kann, von deinem Gerät zum Empfänger. Kodierungsschlüssel befinden sich nur auf Benutzergeräten. So kann niemand — noch nicht einmal Viber — deine Nachrichten lesen.' Auch wenn hier lediglich auf Nachrichten eingegangen wird, ist in den Angaben zur Verschlüsselungsmethode von Viber auch die Verschlüsselung von Telefonaten beschrieben. [41] [49]

4 Methodisches Vorgehen

In diesem Kapitel wird auf das Vorgehen eingegangen, grundlegende Vorbereitungen erläutert und auf die Generierung der Testdaten eingegangen. Abschließend wird das Auslesen sowie die Analyse der Testdaten beschrieben.

4.1 Vorbereitungen

In folgenden Abschnitt wird die Auswahl sowie Einrichtung der Testgeräte und Informationen zu diesen dargestellt. Darüber hinaus wird die Organisation von zwei Subscriber Identity Module (SIM)-Karten sowie die Einrichtung der Viber-Nutzerkonten erläutert.

4.1.1 Testgeräte

Zur Generierung von Testdaten werden Testgeräte benötigt, auf welchen das Betriebssystem iOS installiert ist. Darüber hinaus müssen die Testgeräte über eine SIM-Karte eine Verbindung in das deutsche Mobilfunknetz aufnehmen können. Daher werden für die Analyse iPhones der Marke Apple genutzt.

Aufgrund des Umstandes, dass das, für das Auslesen verwendete, Gerät Universal Forensics Extraction Device (UFED) Touch2 Mobilgeräte mit der neuesten Version des Betriebssystems iOS nicht auslesen kann, muss die Version des Betriebssystems darunter liegen. Hierfür wurden zwei Geräte aus dem beruflichen Umfeld organisiert und auf Werkseinstellungen zurückgesetzt. Eine Aktualisierung der iOS-Version wurde hierbei unterdrückt, da ein Herabstufen der iOS-Version nur unter bestimmten Umständen durchführbar ist. Hierfür muss die gewollte Version von Apple für das spezielle Modell signiert sein, was jedoch meist sofort nach der Veröffentlichung einer neuen Version von Apple widerrufen wird [50].

In der Tabelle 4.1 sind Informationen zu den Testgeräten abgedruckt.

Tabelle 4.1: Geräte- und Softwareinformationen

	Testgerät 1	**Testgerät 2**
Marke	Apple	Apple
Modell	iPhone Xs (D321AP)	iPhone 8 Plus (D211AP)
iOS Version	15.4.1	14.6
Auslesemethode	Logisch (Dateisystem)	Logisch (Vollständiges Dateisystem)
Viber-Version	19.8.1	19.8.1
Viber-Installationsdatum	13.04.2023	18.04.2023

Darüber hinaus wurden weitere teils privat genutzte Geräte genutzt, um mehrere Teilnehmer an Gruppenchats, Kanälen und Communitys zu simulieren. Diese Geräte werden für diese Arbeit nicht analysiert und daher ist die Angabe von Daten obsolet.

4.1.2 SIM-Karten

Da bei der Einrichtung eines Nutzerkontos für die Chatanwendung Viber eine Mobilfunknummer verpflichtend anzugeben ist, wurden zwei SIM-Karten bei dem Supermarkt LIDL über dessen Mobilfunksparte LIDL Connect[7] organisiert. Diese wurden per POSTIDENT, einem Service der Deutschen Post, auf den Autor registriert. Sobald die Identifikation abgeschlossen war, wurden die SIM-Karten in die Testgeräte eingelegt und deren Konnektivität in das deutsche Mobilfunknetz erfolgreich getestet.

4.1.3 Viber

Auf beiden Testgeräten wurde die aktuellste Version - 19.8.1 (Stand 18.04.2023) - der Chatanwendung Viber über den von Apple bereitgestellten App Store[8] installiert. Beim ersten Aufruf der Anwendung wird automatisch der Ersteinrichtungsprozess aufgerufen und die Eingabe einer Mobilfunknummer wird gefordert. Nach Eingabe dieser Nummer wird per Short Message Service (SMS) ein Bestätigungs-Code versendet. Nach dessen Eingabe wird eine Maske zur Angabe von Name, Geburtsdatum und E-Mail-Adresse angezeigt. Die Angabe einer E-Mail-Adresse ist nicht verpflichtend. Sofern eine angegeben wird, wird eine Nachricht an diese Adresse versendet, in der die E-Mail-Adresse bestätigt werden soll.

In den Abbildungen 4.1, 4.2, 4.3 und 4.4 ist der Einrichtungsprozess sowie die erste Ansicht nach dessen Durchlauf abgedruckt.

Abbildung 4.1: Willkommen bei Viber

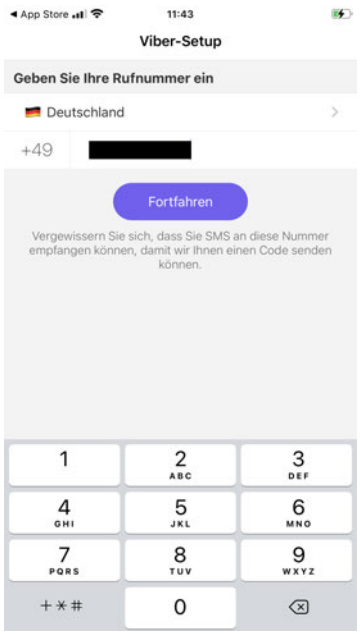

Abbildung 4.2: Einrichten der Mobilfunknummer

[7] https://www.lidl.de/c/lidl-connect/s10007717?mktc=brandpaidsearch&et_uk=3b3afdfb4ffa4d80855104d6d8595049
[8] https://apps.apple.com/de/app/viber-messenger-video-anrufe/id382617920

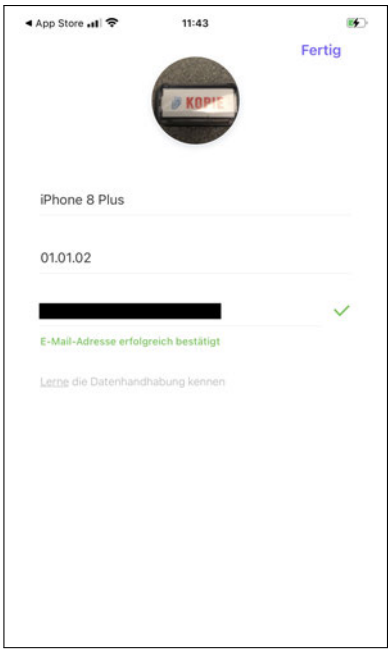

Abbildung 4.3: Angabe von persönlichen Daten

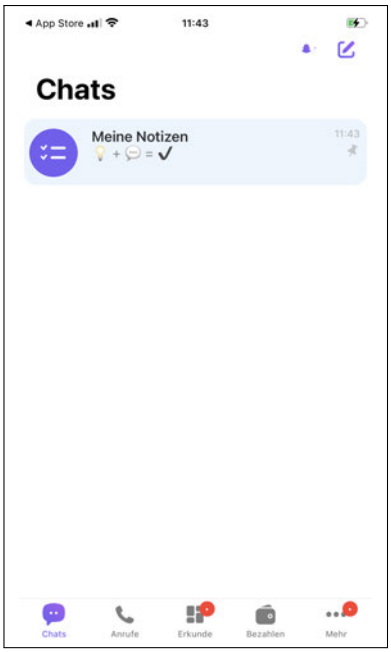

Abbildung 4.4: Ansicht nach Einrichtung von Viber

In der Tabelle 4.2 sind Informationen zu den Viber-Nutzerkonten abgedruckt.

Tabelle 4.2: Informationen zu den Viber-Nutzerkonten

	Testgerät 1	Testgerät 2	Privates Gerät 1	Privates Gerät 2
Viber-Name	iPhone Xs	iPhone 8 Plus	Michael bla	Jon Doe
Kontaktname	Testgeraet 1	Testgeraet 2	Privat 1	Privat 2

4.1.4 Auslesegerät

Der Autor ist als Analyst in dem Sachverständigenbüro für IT-Forensik FAST-DETECT angestellt und hat somit Zugang zu mehreren forensischen Auslesegeräten. Hierunter befindet sich das für Mobilgeräte konzipierte Auslesegerät UFED Touch2[9] der Firma Cellebrite.

4.2 Erster Durchlauf der Testdatengenerierung

In diesem Abschnitt wird der erste Durchlauf der Generierung der Testdaten auf beiden Testgeräten aufgezeigt und beschrieben, wie dieser durchgeführt wurde.

[9] https://cellebrite.com/de/cellebrite-ufed-de/

4.2.1 Einzelchat

Es wurden Einzelchats zwischen den Testgeräten selbst sowie zwischen den Testgeräten und dem Kontakt Privat 1 eröffnet. Darüber hinaus wurde ein Einzelchat zwischen dem Testgerät 1 und dem Kontakt Privat 2 erstellt. Dieser Test wurde mit einer PIN versehen, wodurch diese Unterhaltung lediglich nach Eingabe der PIN in der Anwendung einsehbar ist.

Inhalt der Unterhaltung zwischen Testgerät 1 und Testgerät 2 waren mehrere Textnachrichten, Sprachnachrichten, Bilder und Videos. Es wurde ein Kontakt weitergeleitet. Darüber hinaus wurden GIF-Dateien versendet. Eine Nachricht enthält einen freigegebenen Standort. Weiter wurde eine Nachricht gelöscht und eine weitere nach dem Versenden bearbeitet. Zum Testen der Funktion der selbst löschenden Nachrichten wurde für einen begrenzten Zeitraum diese Funktion aktiviert. Es wurde auf eine Nachricht reagiert sowie eine Nachricht weitergeleitet.

Inhalt der Unterhaltung zwischen Testgerät 1 und Privat 1 waren Textnachrichten, eine GIF-Datei, eine weitergeleitete Nachricht, eine Bild-Datei sowie ein Einladungs-Link in einen Kanal. Die Nachrichten sind alle vom Testgerät 1 versendet worden. Privat 1 hat keine Nachrichten versendet.

Inhalt der Unterhaltung zwischen Testgerät 2 und Privat 1 waren zwei Textnachrichten. Hierbei sollte auf dem Testgerät 2 die Textnachricht, welche durch Privat 1 versendet worden ist, nicht gelesen werden.

Inhalt der Unterhaltung zwischen Testgerät 1 und Privat 2 ist eine Textnachricht. Der Fokus hierbei liegt auf der Analyse, von per PIN-geschützten Unterhaltungen.

4.2.2 Gruppenchat

Es wurde ein Gruppenchat mit den Teilnehmern Testgerät 1, Testgerät 2, Privat 1 und Privat 2 erstellt.

Inhalt des Gruppenchats waren Text- und Sprachnachrichten. Ein Bild und ein Video wurden versendet. Weiter wurde ein Standort versendet. In dem Gruppenchat wurde durch das Testgerät 1 Nachrichten gelöscht. Hierbei wurde unterschieden zwischen ’Für mich löschen’ und ’Für alle löschen’. Vor dem Auslesen wurde eine Nachricht versendet, welche sich nach einem Tag selbst löschen wird. Somit ist sichergestellt, dass die Nachricht während dem Auslesen ungelöscht vorliegt. Dem Gruppenchat wurde zwischenzeitlich der Kontakt Privat 2 vom Testgerät 1 hinzugefügt und vom Testgerät 2 wieder entfernt.

4.2.3 Community

Es wurde eine Community mit dem Testgerät 2 erstellt. Der Name der Community lautet 'Community 1' und als Beschreibung wurde der Satz 'Community für Tests' gewählt. Teilnehmer waren Testgerät 1, Testgerät 2 und Privat 1.

Es wurden hauptsächlich Textnachrichten versendet. Darüber hinaus wurde eine Abstimmung mit den Antwortmöglichkeiten 'Ja' und 'Nein' eröffnet. Hierbei wurde von den Teilnehmern einmal Option 'Ja' und einmal Option 'Nein' gewählt. Der Teilnehmer Privat 1 wurde zum 'Super Admin' befördert und danach aus der Community ausgeschlossen.

4.2.4 Kanal

Mittels des Testgerätes 1 wurde ein Kanal mit dem Titel 'Kanal 1' und der Beschreibung 'Kanalbeschreibung' erstellt. Ein Einladungs-Link wurde an Testgerät 2 sowie Privat 1 versendet.

Inhalt der Unterhaltung waren insgesamt drei Nachrichten. Zwei Textnachrichten und eine GIF-Datei.

4.2.5 Telefonie

Es wurden mehrere Telefonate zwischen den Testgeräten geführt. Sprachanrufe und Videoanrufe wurden durchgeführt. In einem Sprachtelefonat wurde mittendrin auf Videoanruf umgeschaltet. Bei einem Anruf wurde der Kontakt Privat 1 während des Telefonats hinzugefügt. Darüber hinaus wurde bei einem Anrufversuch das Telefonat nicht angenommen.

4.3 Zweiter Durchlauf der Testdatengenerierung

Im Verlauf einer ersten Analyse der Protokolldateien der Anwendung Viber konnten Funktionen festgestellt werden, welche im ersten Durchlauf der Testdatengenerierung nicht bedacht worden sind. Darüber hinaus wurden Änderungen an den Konversationen durchgeführt, um im direkten Vergleich Unterschiede in den Datenbanken aufzeigen zu können.

4.3.1 Testgerät 1

Auf dem Testgerät 1 wurde der Kontakt Testgerät 2 blockiert. Darüber hinaus wurde die komplette Konversation, welche mit dem Kontakt Privat 1 geführt worden ist, gelöscht. Auf dem Testgerät wurde der Kanal für 30 Tage ausgeblendet.

4.3.2 Testgerät 2

Auf dem Testgerät 2 wurde der Kontakt Testgerät 1 zum Admin der Gruppe befördert. Weiter wurde die Anwendung Viber über die iOS-Funktion 'App auslagern' ausgelagert, um analysieren zu können, welche Daten nach der Auslagerung noch auffindbar sind. iOS bietet mehrere Funktionen, den Speicher auf dem Gerät zu verwalten. Die Funktion 'App auslagern' dient dazu, Speicher, welcher durch Anwendungen blockiert wird, freizugeben und verfügbar zu machen. Hierbei werden jedoch nicht alle Daten der Anwendung gelöscht, sodass bei einer Wiedereinlagerung der Anwendung, die Daten abrufbar sind. Diese Funktion muss durch den Nutzer aktiviert werden. Darüber hinaus kann der Nutzer einstellen, dass wenig genutzte Anwendungen automatisch ausgelagert werden.

4.4 Analyse der Daten

Dieser Abschnitt behandelt das Vorgehen der Analyse der Testdaten. Hierbei wird darauf eingegangen, wie die Ablagestruktur, die SQLite-Datenbanken sowie weitere relevante Dateien analysiert worden sind.

4.4.1 Ablagestruktur

Für die Analyse der Ablagestruktur von Viber auf iOS-Geräten wurden zuvor erstellte UFED-Berichte verwendet. Hierin wurde mittels Volltextsuche nach dem Suchbegriff 'Viber' gesucht. Durch die im Bericht angezeigten Suchtreffer konnten auf beiden Geräten die jeweiligen Anwendungs-Verzeichnisse identifiziert werden. Darüber hinaus wurden von Apple standardgemäß angelegte Datenbanken analysiert, um alle vorhandenen Verzeichnisse der Anwendung Viber zu identifizieren.

4.4.2 SQLite-Datenbanken

Die festgestellten Datenbanken wurden für einen ersten Überblick mit dem im UFED-Bericht integrierten Datenbank-Viewer betrachtet. Aufgrund des begrenzten Funktionsumfangs wurde die detaillierte Analyse mit dem Programm DB Browser for SQLite[10] durchgeführt. Vorteile dieses Programms sind unter anderem die einfache Filterfunktion in den Tabellen sowie die Ausführbarkeit von SQL-Befehlen. Darüber hinaus ist die Ansicht des Datenbankschemas sowie Beziehungen zwischen den Tabellen leichter erkennbar.

[10] https://sqlitebrowser.org/

4.4.3 Weitere relevante Dateien

Ein von Apple häufig verwendetes Dateiformat ist das der Property List (Plist). Diese Dateien stellen eine einfache Lösung für Anwendungen dar, Schlüssel-Wert-Paare möglichst leicht und portabel zu speichern. Im Verlauf der Analyse wurden solche Plist-Dateien aufgefunden und mit dem Texteditor Notepad++[11] analysiert. [51]

Darüber hinaus wurden sogenannten Binär-Plist-Dateien in Datenbanken festgestellt. Diese wurden mit dem Texteditor Notepad++ und dem zusätzlich installierten Plugin 'Notepad++ bplist plugin'[12] analysiert. Hierbei ist aufgefallen, dass der Inhalt der Datei von Notepad++ eine sehr lange Zeichenkette ist, welche als BASE64-Code[13] identifiziert werden konnte. Dieser Code wurde in ASCII-Code[14] decodiert. Dabei wurde festgestellt, dass der decodierte Code eine weitere Binär-Plist-Datei darstellt. Diese konnte wie zuvor mit dem Texteditor Notepad++ analysiert werden.

[11] https://notepad-plus-plus.org/
[12] https://github.com/azerg/NppBplistPlugin
[13] https://www.base64decode.org/de/
[14] https://www.ascii-code.com/de

5 Ergebnis

Gegenstand dieses Kapitels ist das Ergebnis der Analyse der Testdaten. Es wird auf die Ablagestruktur der Anwendung eingegangen. Hierbei werden Verzeichnisse aufgezeigt, welche Anwendungsdaten enthalten und Verzeichnisse welche Anhänge enthalten. Darauf folgend wird das Ergebnis der Datenbankanalyse erörtert.

5.1 Ablagestruktur

In diesem Abschnitt wird auf die Ablagestruktur der Chatanwendung Viber auf iOS-Geräten eingegangen. Hierbei sollen Stammverzeichnisse, Unterverzeichnisse und Fundstellen von relevanten Dateien aufgezeigt werden.

5.1.1 Stammverzeichnisse der Anwendung Viber

Im Laufe der Analyse konnten mehrere Stammverzeichnisse der Anwendung Viber in unterschiedlichen Pfaden festgestellt werden. In diesem Abschnitt werden Informationen hierzu erläutert.

Unter dem Betriebssystem iOS gibt es das Verzeichnis 'AppGroup' im Pfad '/root/private/var/mobile/Containers/Shared/'. Dieses Verzeichnis dient zum Teilen von Daten zwischen unterschiedlichen Anwendungen, welche zur gleichen Anwendungsgruppe gehören. Dieser Anwendungsgruppe wird eine 'APP ID' zugewiesen. Aus APP IDs kann nicht herausgelesen werden, welcher Anwendung sie zugeordnet sind. Im Verzeichnis jeder Anwendungsgruppe befindet sich die Plist-Datei '.com.apple.mobile_container_manager.metadata.plist', in welcher die Informationen zur zugehörigen Anwendung abgelegt sind. So konnte durch manuelle Analyse dieser Plist-Dateien festgestellt werden, dass der Anwendung Viber die APP ID 'E9DF8DC7-1F5A-4F06-89F1-2B9BE50B4877' zugeordnet ist. [52] [51] [53]

Zum Auffinden weiterer Stammverzeichnisse wurde die Datenbank 'applicationState.db' im Pfad '/private/var/mobile/Library/FrontBoard/' analysiert. Hierbei konnte unter Verwendung des SQL-Befehls, welcher im Quelltext 5.1 dargestellt ist, in der Spalte 'kvs.value' ein Objekt im Format einer Plist als Binary Large Object (BLOB)[15] festgestellt werden.

Quelltext 5.1: Extraktion von Stammverzeichnis-Pfaden

```
SELECT kvs.id, kvs.value
FROM kvs
    JOIN application_identifier_tab
    ON application_identifier_tab.id =
        kvs.application_identifier
WHERE application_identifier_tab.application_identifier =
    'com.viber' AND kvs.key = '1'
```

[15] https://www.sqlitetutorial.net/sqlite-data-types/

Anhand dieser Plist können folgende Pfade der Anwendung Viber zugeordnet werden:

- /private/var/containers/Bundle/Application/A4D7FCFC-F68F-43FB-9D06-9BD2EA56FF21/
- /private/var/mobile/Containers/Data/Application/40215B44-3529-435F-86BA-DA9BACB08475/

Die Bedeutung des oben genannten SQL-Befehls lautet: Es sollen die Spalten 'id' und 'value' aus der Tabelle 'kvs' ausgewählt werden. Diese sollen mit der Tabelle 'application_identifier_tab' verknüpft werden, indem die IDs der beiden Tabellen verglichen werden. Darüber hinaus soll der Wert in der Spalte 'application_identifier' in 'kvs' dem Wert 'com.viber' sowie der Wert in der Spalte 'key' in 'kvs' dem Wert '1' entsprechen.

5.1.2 Ablagestruktur der Anwendungsdaten

Bestandteil dieses Abschnittes ist die Ablagestruktur der Anwendungsdaten. Hierbei wird auf die Verzeichnisstruktur und deren Inhalten sowie Bedeutung eingegangen.

Tabelle 5.1: Ablagestruktur Anwendungsdaten

Inhalt	**Verzeichnispfad**	**Beschreibung**
Stammverzeichnis auf Testgerät 1	/mobile/Containers/Shared/AppGroup/group.viber.share.container/	Hauptverzeichnis mit relevanten Daten zur Anwendung Viber auf Testgerät 1
Stammverzeichnis auf Testgerät 2	/root/private/var/mobile/Containers/Shared/AppGroup/E9DF8DC7-1F5A-4F06-89F1-2B9BE50B4877/	Hauptverzeichnis mit relevanten Daten zur Anwendung Viber auf Testgerät 2
Contacts.data (SQLite-Datenbank)	*<Stammverzeichnis>*/com.viber/database/	Speichert alle Daten zu Konversationen, Nachrichten, Kontakten und Telefonaten
Settings.data (SQLite Datenbank)	*<Stammverzeichnis>*/com.viber/settings/	Speichert Einstellungen, welche die Anwendung Viber betreffen
Vorschaubilder für Standorte (.jpg-Dateien)	*<Stammverzeichnis>*/com.viber/CustomLocationImage/	Enthält die im Chat für versendete/empfangene Standorte angezeigten Vorschaubilder
Icons von Gruppenchats (.jpg-Dateien)	*<Stammverzeichnis>*/com.viber/Groups/*<GruppenID>*/Icons/	Enthält das Icon zur jeweiligen Gruppe
Profilbilder von Kontakten (.jpg-Dateien)	*<Stammverzeichnis>*/com.viber/ViberIcons/	Enthält die Profilbilder zu den Kontakten
Icons von Kanälen und Communities (.jpg-Dateien)	*<Stammverzeichnis>*/com.viber/Vibe/Icons/	Enthält Icons zu Kanälen und Communities

Tabelle 5.1, Fortsetzung: Ablagestruktur Anwendungsdaten

Inhalt	Verzeichnispfad	Beschreibung
Vorschaubilder der Anhänge (.jpg-Dateien)	*<Stammverzeichnis>*/com.viber/ AttachmentsPreview/	Enthält Vorschaubilder zu den versendeten und empfangenen Bild- und Video-Dateien
Große Vorschaubilder der Anhänge (.jpg-Dateien)	*<Stammverzeichnis>*/com.viber/ BigAttachmentsPreview/	Enthält Vorschaubilder zu den versendeten und empfangenen Bild- und Video-Dateien
Sticker (.png-Dateien	*<Stammverzeichnis>*/com.viber/ stickers/	Enthält in der Anwendung Viber enthaltene Sticker-Dateien

In der Tabelle 5.1 sind Informationen zum Stammverzeichnis der Anwendungsdaten, zu Unterverzeichnissen hierzu und zu darin enthaltenen relevanten Dateien abgedruckt. Bevor näher auf die abgedruckten Informationen eingegangen wird, ist zu erwähnen, dass die Tabelle nicht zu allen Verzeichnissen oder Dateien Informationen enthält. Zum einen sind einige nicht relevant, zum anderen konnte der Zweck hinter den weiteren nicht abschließend festgestellt werden.

Auf den Testgeräten liegen für die Chatanwendung Viber unterschiedliche Pfade zum Stammverzeichnis der Anwendungsdaten vor. Dies liegt zum einen an den unterschiedlichen Auslesemethoden. Das drückt sich darin aus, dass aufgrund der logischen Dateisystem-Extraktion des Testgerätes 1 der Pfadbereich '/root/private/var/' durch den UFED Touch2 nicht ausgelesen werden konnte, sondern nur die darin befindlichen Verzeichnisse. In diesem Beispiel ab dem Verzeichnis 'mobile'. Zum anderen unterscheidet sich der Verzeichnisname des Ordners, der sich im Verzeichnis 'AppGroup' befindet. In der Extraktion des Testgerätes 2 ist die APP ID 'E9DF8DC7-1F5A-4F06-89F1-2B9BE50B4877' als Verzeichnisname angegeben. Auf dem Testgerät 1 lautet der Verzeichnisname 'group.viber.share.container'. Der Unterschied hierbei entsteht sehr wahrscheinlich durch die Auslesemethode. Das Testgerät 1 wurde über ein iTunes-Backup ausgelesen. Hierbei benennt Apple das Verzeichnis nach der Anwendung, welche in der zuvor erwähnten Plist-Datei '.com.apple.mobile_container_manager.metadata.plist' angegeben ist.

In der Tabelle 5.1 wurde in der Spalte 'Verzeichnispfad' ab der dritten Zeile nicht der vollständige Pfad angegeben, da, wie zuvor erwähnt, sich die Pfade auf den Testgeräten minimal unterscheiden. Somit wurden die Pfadangaben relativ zu den Stammverzeichnissen für Anwendungsdaten angegeben und durch '*<Stammverzeichnis>*' dargestellt.

Stammverzeichnis

Im Stammverzeichnis des Testgerätes 1 für Anwendungsdaten von Viber befinden sich die beiden Verzeichnisse 'com.viber' und 'synchronizition'. Zu den Dateien im Verzeichnis 'synchronizition' konnte nicht festgestellt werden, wozu sie dienen. Augenscheinlich haben sie jedoch keine Relevanz für diese Arbeit.

Im Stammverzeichnis des Testgerätes 2 sind zusätzlich zu den beiden Verzeichnissen 'com.viber' und 'synchronizition', das Verzeichnis 'Library' und die Dateien '.com.apple.mobile_container_manager.metadata.plist', 'SecureStorage.data' und 'SecureStorage.data.backup' abgelegt. Dass diese Verzeichnisse und die Dateien lediglich auf dem Testgerät 2 vorhanden sind, ist auf die Auslesemethode zurückzuführen. Das Verzeichnis und die zwei Dateien werden von Apple standardgemäß erstellt und müssen nicht zwingend von den Anwendungen genutzt werden. [54]

Im Verzeichnis 'com.viber' befinden sich insgesamt 14 Unterverzeichnisse. Zu den 9 in der Tabelle abgedruckten Verzeichnissen konnten relevante Inhalte festgestellt werden. Die drei Verzeichnisse 'AttachmentsPreview', 'BigAttachmentsPreview' und 'stickers' liegen aufgrund der Auslesemethode lediglich auf dem Testgerät 2 vor.

Database und Settings

Die Hauptdatenbank 'Contacts.data', welche Informationen zu den Konversationen und Kontakten enthält, befindet sich im Verzeichnis 'database'. Die Einstellungen sind in der Datenbank 'Settings.data', welche sich im Verzeichnis 'settings' befindet, gespeichert. Auf den Inhalt der Datenbanken wird im späteren Verlauf der Arbeit eingegangen.

CustomLocationImage

Sofern vom lokalen Nutzer oder einem Kontakt ein Standort versendet worden ist, wird über die von Apple entwickelte Navigationsanwendung 'Apple Karten'[16] ein Vorschaubild einer Kartenansicht des Standortes erstellt und im Verzeichnis 'CustomLocationImage' im jpg-Format gespeichert.

Groups/<GruppenID>/Icons

Wenn der lokale Nutzer Teilnehmer eines Gruppenchats ist und dieser Gruppe ein Icon zugewiesen wurde, werden diese im Verzeichnis 'Groups/<GruppenID>/Icons/' im jpg-Format gespeichert. Der Pfadteil '<GruppenID>' dient hierbei als Platzhalter für die Viber-spezifische Kennung der Gruppe. Wie und wo die Kennung einer Gruppe gespeichert ist, wird im weiteren Verlauf erwähnt.

ViberIcon

Nutzer der Chatanwendung Viber können bestimmte Informationen dem eigenen Profil hinzufügen. Hierzu zählt auch die Zuweisung eines Profilbildes. Sofern sich ein Kontakt ein Profilbild zugewiesen hat, wird dieses beim lokalen Nutzer im Verzeichnis 'ViberIcons' im jpg-Format abgelegt. Auf die Zuordnung eines Profilbildes zu einem spezifischen Kontakt wird noch eingegangen.

Vibe/Icons

Im Verzeichnis 'Vibe/Icons' sind Icons gespeichert, welche Kanälen und Communitys zugeordnet worden sind. Diese Bilder sind im jpg-Format abgelegt. Hierbei muss der lokale Nutzer nicht zwingend Teilnehmer des Kanals bzw. der Community sein, da hier auch Icons von dem Nutzer von Viber vorgeschlagenen Kanäle bzw. Communitys abgelegt sind.

AttachmentsPreview

Das Verzeichnis 'AttachmentsPreview' enthält Vorschaubilder zu versendeten und empfangenen Bild- und Video-Dateien im jpg-Format. Wie die Bilder den Original-Dateien und Übertragungen zugeordnet werden, wird im weiteren Verlauf der Arbeit erläutert.

[16] https://www.apple.com/de/maps/

BigAttachmentsPreview

Vorschaubilder in einer höheren Auflösung werden im Verzeichnis 'BigAttachmentsPreview' abgelegt. Beim Analysieren der Ablagestruktur lag in diesem Verzeichnis lediglich ein Vorschaubild vor. Das Szenario, in welchem dieses Bild entstanden ist, konnte nicht reproduziert werden. Daher kann keine Aussage darüber getroffen werden, durch welche Aktion ein solches Vorschaubild entsteht.

stickers

Das Verzeichnis 'stickers' enthält Sticker, welche in die Anwendung Viber integriert sind. Hierin sind vorinstallierte und durch den Nutzer hinzugefügte bzw. heruntergeladen Sticker enthalten.

5.1.3 Ablagestruktur der Anhänge

Tabelle 5.2: Ablagestruktur Anhänge

Inhalt	Verzeichnispfad	Beschreibung
Stammverzeichnis für Anhänge auf Testgerät 1	/mobile/Containers/Data/Application/com.viber/Documents/	Hauptverzeichnisse für versendete und empfangene Anhänge auf Testgerät 1
Stammerzeichnis für Anhänge auf Testgerät 2	/root/private/var/mobile/Containers/Data/Application/40215B44-3529-435F-86BA-DA9BACB08475/Documents/	Hauptverzeichnisse für versendete und empfangene Anhänge auf Testgerät 2
Bilder und Videos (.jpg- und .mp4-Dateien)	*<Stammverzeichnis>*/Attachments/	Enthält über Viber versendete oder empfangene Bild- und Videodateien
InstantVideos (.mp4-Dateien)	*<Stammverzeichnis>*/InstantVideos/	Enthält Videos die mittels Viber aufgenommen worden sind
Sprachnachrichten (m4a-Dateien ohne Dateiendung)	*<Stammverzeichnis>*/VoiceMessages/	Enthält über Viber versendete oder empfangene Sprachnachrichten
Dateien (vd. Dateiformate)	<Stammverzeichnis>/FileMessages/	Enthält versendete und empfangene Dateien, welche als 'Datei' versendet wurden.

Auf den Testgeräten liegen für die Chatanwendung Viber unterschiedliche Pfade zum Stammverzeichnis der Anwendungsdaten vor. Dies liegt zum einen an den unterschiedlichen Auslesemethoden. Das drückt sich darin aus, dass aufgrund der logischen Dateisystem-Extraktion des Testgerätes 1 der Pfadbereich '/root/private/var/' durch den UFED Touch2 nicht ausgelesen werden konnte, sondern nur die darin befindlichen Verzeichnisse. In diesem Beispiel ab dem Verzeichnis 'mobile'. Zum anderen unterscheidet sich der Verzeichnisname des Ordners, der sich im Verzeichnis 'AppGroup' befindet. In der Extraktion des Testgerätes 2 ist die APP ID '40215B44-3529- 435F-86BA-DA9BACB08475'

als Verzeichnisname angegeben. Auf dem Testgerät 1 lautet der Verzeichnisname 'com.viber'. Der Unterschied hierbei entsteht sehr wahrscheinlich durch die Auslesemethode. Das Testgerät 1 wurde über ein iTunes-Backup ausgelesen. Hierbei benennt Apple das Verzeichnis nach der Anwendung, welche in der zuvor erwähnten Plist-Datei '.com.apple.mobile_container_manager.metadata.plist' angegeben ist.

Nach der zweiten Testdatengenerierung ist das Stammverzeichnis der Anhänge auf dem Testgerät 2 nicht mehr vorhanden. Dies begründet sich auf der Durchführung der Funktion 'App auslagern', welche innerhalb der zweiten Testdatengenerierung auf dem Testgerät 2 ausgeführt wurde. Hierbei wird zu Speicheroptimierungszwecken, das Stammverzeichnis für Anhänge gelöscht, auch wenn das Stammverzeichnis für Anwendungsdaten weiterhin vorhanden ist.

In der Tabelle 5.2 wurde in der Spalte 'Verzeichnispfad' ab der dritten Zeile nicht der vollständige Pfad angegeben, da, wie zuvor erwähnt, sich die Pfade auf den Testgeräten minimal unterscheiden. Somit wurden die Pfadangaben relativ zu den Stammverzeichnissen für Anwendungsdaten angegeben und durch '*<Stammverzeichnis>*' dargestellt. In der Tabelle 5.2 wurde in der Spalte 'Verzeichnispfad' ab der dritten Zeile aus Übersichtsgründen nicht der vollständige Pfad angegeben. Somit wurden die Pfadangaben relativ zu den Stammverzeichnissen für Anwendungsdaten angegeben und durch '*<Stammverzeichnis>*' dargestellt.

Stammverzeichnis

Im Stammverzeichnis der Anhänge auf dem Testgerät 1 befinden sich die beiden Verzeichnisse 'Documents' und 'Library'. Im zweiten Verzeichnis konnten augenscheinlich keine relevanten Informationen für diese Arbeit festgestellt werden.

Im Stammverzeichnis der Anhänge auf Testgerät 2 sind zusätzlich zu den Verzeichnissen 'Documents' und 'Library', die Verzeichnisse 'StoreKit' und 'tmp' sowie die Datei '.com.apple.mobile_container_manager.metadata.plist' abgelegt. Dass diese Verzeichnisse und die Datei lediglich auf dem Testgerät 2 vorhanden sind, ist auf die Auslesemethode zurückzuführen. Die Verzeichnisse werden von Apple standardgemäß erstellt und müssen nicht zwingend von den Anwendungen genutzt werden. [54]

Attachments

Im Verzeichnis 'Attachments' sind Bilder (jpg-Format) und Videos (mp4-Format) abgelegt, welche über Viber versendet oder empfangen worden sind. Hierbei ist zu erwähnen, dass nur heruntergeladene Dateien gespeichert werden.

InstantVideos

Im Gegensatz zum Verzeichnis 'Attachments' sind im Verzeichnis 'InstantVideos' Videos abgelegt, welche über die Chatanwendung Viber aufgenommen und versendet oder empfangen worden sind. Sie sind ebenfalls im mp4-Format gespeichert.

VoiceMessages

Sprachnachrichten, welche mittels Viber versendet oder empfangen worden sind, werden im Verzeichnis 'VoiceMessages' abgelegt. Die Dateien haben keine Dateiendung. Das Dateiformat konnte durch Öffnen der Datei in dem Hexadezimal (Hex)-Editor HxD[17] festgestellt werden.

[17] https://mh-nexus.de/en/hxd/

Hierbei war ersichtlich, dass die Datei den für m4a-Dateien üblichen Dateikopf[18] enthält. [55]

In Abbildung 5.1 ist die Hex-Ansicht des Dateikopfs einer Sprachnachricht, welche mittels Viber versendet worden ist, abgedruckt.

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Dekodierter Text
00000000	00	00	00	1C	66	74	79	70	4D	34	41	20	00	00	00	00	....ftypM4A ....
00000010	4D	34	41	20	6D	70	34	32	69	73	6F	6D	00	00	04	D9	M4A mp42isom...Ù
00000020	6D	6F	6F	76	00	00	00	6C	6D	76	68	64	00	00	00	00	moov...lmvhd....
00000030	E0	65	34	3A	E0	65	34	3A	00	00	3E	80	00	02	5C	00	àe4:àe4:..>€..\.

Abbildung 5.1: Hex-Ansicht einer Sprachnachricht

FileMessages

Im Verzeichnis 'FileMessages' werden Dateien gespeichert, die über Viber versendet oder empfangen worden sind. Hierbei ist zu erwähnen, dass auch Bilder oder Videos hierin gespeichert sein können. Dies geschieht, wenn der Nutzer ein Bild oder Video nicht über die Funktion 'Bild oder Video versenden' verschickt, sondern über die Funktion 'Datei versenden'. Hierbei ist aufgefallen, dass weiterhin Metadaten, wie z.B. Exchangeable Image File Format (Exif)-Daten, in den Bildern enthalten sind und analysiert werden können.

5.2 Datenbankenanalyse

Inhalt dieses Kapitels ist das Ergebnis aus der Analyse der festgestellten Datenbanken. Hierbei wird der Aufbau der Datenbanken, relevante Tabellen und deren Spalten sowie Datentypen näher betrachtet. Es konnten mehrere Datenbanken im Stammverzeichnis der Anwendungsdaten bzw. in Unterverzeichnissen festgestellt. Folgende Datenbanken waren feststellbar: 'cdr.sqlite', 'Contacts.data', 'events.sqlite', 'lenses.sqlite', 'search.sqlite', 'Settings.data' und 'vp.sqlite'.

Im Folgenden werden in Datenbanken enthaltene Tabellen mit einem '.' an den Namen der Datenbank angefügt. Dies sieht wie folgt aus: *<Datenbankname>.<Tabellenname>*. Als Beispiel: 'Settings.data.Data', wobei 'Settings.data' der Datenbankname und 'Data' der Tabellenname ist.

5.2.1 Settings.data

Die Datei 'Settings.data' ist eine Datenbank im SQLite-Format. Auch wenn die Datei nicht die häufig verwendeten Dateiendungen '.db' oder '.sqlite' aufweist, konnte durch eine Signaturenanalyse der Datei in Hex-Ansicht die für SQLite-Datenbanken verwendete Signatur '0x53 51 4C 69 74 65 20 66 6F 72 6D 61 74 20 33 00' festgestellt werden. [56]

Der Beginn der Datei 'Settings.data' in Hex-Ansicht ist in Abbildung 5.2 abgebildet.

[18] https://it-forensik.fiw.hs-wismar.de/index.php/Dateiheader

```
Offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  Dekodierter Text
00000000   53 51 4C 69 74 65 20 66 6F 72 6D 61 74 20 33 00  SQLite format 3.
00000010   10 00 02 02 00 40 20 20 00 00 00 11 00 00 00 15  .....@  ........
00000020   00 00 00 14 00 00 00 01 00 00 00 02 00 00 00 04  ................
00000030   00 00 00 00 00 00 00 00 00 00 00 01 00 00 00 00  ................
```

Abbildung 5.2: Hex-Ansicht der Datei Settings.data

Die Datenbank 'Settings.data' enthält eine Tabelle mit dem Titel 'Data'. Die Datenstruktur und relevante Inhalte der Tabelle werden im Folgenden näher betrachtet.

Datenstruktur der Tabelle 'Settings.data.Data'

Die Tabelle ist in die drei Spalten 'key', 'value' und 'category' gegliedert. Die Tabelle ist im Prinzip einer Schlüssel-Werte-Datenbank aufgebaut, welcher eine Kategorie 'Spalte' hinzugefügt wurde. [57] In der Tabelle 5.3 sind Informationen zu den Spalten abgedruckt.

Tabelle 5.3: Informationen zu Spalten in Tabelle in Settings.data.Data

Spalte	Typ	Bedeutung
key	TEXT	Enthält die eindeutigen Schlüssel der Einstellungen
value	BLOB	Enthält die Werte der Einstellungen
category	TEXT	Enthält die Kategorie, welcher die jeweilige Einstellung zugeordnet ist

- Die Spalte 'key' enthält die Schlüssel zu den in der Tabelle 'Settings.data.Data' gespeicherten Einstellungen. Die Bedeutungen der darin festgestellten Inhalte sind teilweise durch ihre Benennung erkennbar. Der Datentyp der Spalte ist 'TEXT'.
- Die Spalte 'value' enthält die Werte der Einstellungen. Die Inhalte der Spalte werden in BLOBs gespeichert. Da BLOBs teilweise unstrukturierte Daten enthalten können, konnte nicht zu allen Zeilen deren Bedeutung analysiert werden.
- Die Spalte 'category' enthält Informationen zu Kategorien, in denen Einstellungen zusammengefasst werden können. Bei den vorliegenden Testdaten konnten hierbei die Kategorien 'AppSettings' und 'ipcstorage' festgestellt werden.
- Die Anzahl der enthaltenen Zeilen ist variabel, was aus der unterschiedlichen Anzahl zwischen den beiden Datenbanken (erste Testdatengenerierung) auf Testgerät 1 (350 Zeilen) und Testgerät 2 (336 Zeilen) ersichtlich ist. In der zweiten Testdatengenerierung konnten auf Testgerät 1 insgesamt 353 Zeilen und auf Testgerät 2 insgesamt 338 Zeilen festgestellt werden.

In Abbildung 5.3 ist der Aufbau der Tabelle visuell dargestellt.

	key	value	category
	Filtern	Filtern	Filtern
1	_autothemeMigrated	☐	AppSettings
2	_bitmojiBottomSheetShowCount	*BLOB*	AppSettings
3	_cameraVideoFlashMode	*BLOB*	AppSettings

Abbildung 5.3: Aufbau der Tabelle Settings.data.Data

Relevante Daten in Settings.data.Data

Im Folgenden wird auf relevante Zeilen in der Tabelle eingegangen. Hierbei wurden alle Zeilen betrachtet und hinsichtlich der Feststellung von Nutzerinformationen und Anwendungseinstellungen, welche Hinweise auf eine Nutzerkenntnis von Mediendateien und deren Download liefern, als relevant oder nicht relevant bewertet.

In Tabelle 5.4 sind alle relevanten Zeilen und deren Bedeutung abgedruckt.

Tabelle 5.4: Relevante Zeilen in Settings.data.Data

key	Bedeutung von value
_autoDownloadAttachment	Einstellung, ob automatischer Download aktiv (Deaktiviert: 0, Aktiviert: 1)
_autoPlayVideo	Einstellung, ob automatisches Abspielen von Videos aktiv (Deaktiviert: 0, Aktiviert: 1)
_myCountryPhoneCode	Ländervorwahl der Telefonnummer
_myCountryCode	Ländercode des eingestellten Landes
_myPhoneNumber	Telefonnummer des lokalen Nutzers ohne Ländervorwahl
_myCanonizedPhoneNumber	Telefonnummer des lokalen Nutzers mit Ländervorwahl
_myFormattedPhoneNumber	Formatierte Telefonnummer des lokalen Nutzers mit Ländervorwahl
_myMemberID	Eindeutige Viber-Teilnehmer Kennung
_registered	Wert, ob Nutzer registriert ist (Unregistriert: 0, Registriert: 1
_registrationDate	UNIX-Zeitstempel des Zeitpunktes der Registrierung
_appVersion	Installierte Version von Viber
_currentEmail	E-Mail-Adresse des lokalen Nutzers
_isCurrentEmailVerified	Wert, ob Email-Adresse verifiziert ist (Nicht verifiziert: ’0’, Verifiziert: ’1’)
_verificationEmailSentDate	Zeitpunkt der letzten Verifizierungs-Mail
_birthdate	UNIX-Zeitstempel des vom lokalen Nutzer angegebenen Geburtstags
_myUserName	Nutzername des lokalen Nutzers
_numberOfContactsFromLastABSync	Anzahl der aus dem Adressbuch synchronisierten Kontakte

- 16 der festgestellten Zeilen wurden als relevant bewertet. Auffällig hierbei war, dass die Zeile mit dem Schlüssel '_autoDownloadAttachment' in den Testdaten der ersten Testdatengenerierung auf Testgerät 2 nicht vorhanden ist, in der zweiten Testdatengenerierung jedoch schon. Nachfolgend wird auf Informationen zu den relevanten Zeilen eingegangen.
- Der Wert aus dem Schlüssel '_autoDownloadAttachment' sagt aus, ob Dateien, welche der lokale Nutzer zugesendet bekommt, automatisch heruntergeladen und auf dem Gerät gespeichert werden sollen. Die Standardeinstellung für diesen Wert lautet '1' und aktiviert somit das automatische Herunterladen. Durch den eingetragenen Wert '0' ist ersichtlich, dass diese Option deaktiviert ist. Liegen über Viber zugesendete Dateien auf dem ausgelesenen Gerät vor und das automatische Herunterladen ist deaktiviert, kann dies ein Hinweis auf die Kenntnis des Nutzers über die Dateien sein, da der lokale Nutzer die Datei somit höchstwahrscheinlich manuell heruntergeladen hat. Es konnte keine Zeile gefunden werden, welche einen Zeitstempel enthält, zu dem diese Option aktiviert/deaktiviert wurde.
- Die Werte '0' (deaktiviert) und '1' (aktiviert) zum Schlüssel '_autoPlayVideo' geben an, ob dem lokalen Nutzer zu gesendete Videos automatisch abgespielt werden sollen. Ist diese Option aktiviert, kann dies ein Hinweis darauf sein, dass dem lokalen Nutzer der Inhalt von empfangenen Videos bekannt ist.
- Der zum Schlüssel '_myCountryCode' zugeordnete Wert in der Spalte 'value' enthält den Ländercode, in dem sich der lokale Nutzer registriert hat. In den vorliegenden Testdaten war hier 'DE' eingetragen. Somit kann dieser Wert einen Hinweis auf den Aufenthaltsort des Nutzers geben.
- Die Telefonnummer des lokalen Nutzers, mit der er sich registriert hat, liegt in mehreren Schlüssel-Werte-Paaren vor. Zum einen ist die länderspezifische Vorwahl dem Schlüssel '_myCountryPhoneCode' zugeordnet. Im Schlüssel '_myPhoneNumber' ist die Nummer ohne die länderspezifische Vorwahl gespeichert.
  In den beiden Schlüsseln '_myCanonizedPhoneNumber und '_myFormattedPhoneNumber' ist die Telefonnummer komplett gespeichert, wobei im zweiten Schlüssel die Telefonnummer formatiert vorliegt. Dies zeigt sich in der Trennung der länderspezifsichen Vorwahl, mobilfunkspezifischen Anteil und nutzerspezifischen Anteil durch ein Leerzeichen sowie ein am Anfang angefügtes '+'.
- Eine durch Viber zugeordnete eindeutige Teilnehmer-Kennung ist durch den Schlüssel '_myMemberID' heraus findbar. Diese Kennung wird intern durch Viber vergeben und genutzt, um den Teilnehmer eindeutig identifizieren zu können.
- Der Wert im Schlüssel '_registered' gibt durch die Werte '0' und '1' an, ob der Nutzer die Telefonnummer per SMS-Code registriert hat oder nicht. Im Schlüssel '_registrationDate' ist der UNIX-Zeitstempel der Registrierung abgelegt. Ein UNIX-Zeitstempel ist die Angabe der Zeit in Sekunden, welche seit dem 01.01.1970 vergangen ist. Dieser kann mittels eines Rechners in ein lesbares Datumsformat umgewandelt werden. [58]
- Die Version, in der Viber auf dem ausgelesenen Gerät installiert ist, befindet sich im Schlüssel '_appVersion'.
- Dem Schlüssel '_currentEmail' ist die vom Nutzer angegebene E-Mail-Adresse zugeordnet. Durch die Werte '0' (nicht verifiziert) und '1' (verifiziert) ist im Schlüssel '_isCurrentEmailVerified' hinterlegt, ob der Nutzer die E-Mail-Adresse verifiziert hat oder nicht. Sofern diese verifiziert worden ist, wird im Schlüssel '_verificationEmailSentDate' der Zeitpunkt der Verifikation als UNIX-Zeitstempel abgelegt

- Das vom lokalen Nutzer angegebene Geburtsdatum ist in dem Schlüssel '_birthdate' zugeordneten 'value' gespeichert. Hierdurch können, sofern richtig angegeben, Rückschlüsse auf die Identität des Nutzers gezogen werden.
- Unter welchem Nutzernamen der lokale Nutzer für andere Viber-Nutzer sichtbar ist, wird im Schlüssel-Werte-Paar '_myUserName' abgelegt. Hierdurch sind potenziell Rückschlüsse auf die Identität des lokalen Nutzers möglich.
- Der Schlüssel '_numberOfContactsFromLastABSyn' enthält die Anzahl der aus dem Adressbuch des Mobilfunkgerätes synchronisierten Kontakte. Hieraus können Rückschlüsse gezogen werden, wie viele Kontakte der lokale Nutzer potenziell außerhalb von Viber kennt.

5.2.2 Contacts.data

Die Datei 'Contacts.data' ist eine Datenbank im SQLite-Format. Auch wenn die Datei nicht die häufig verwendeten Dateiendungen '.db' oder '.sqlite' hat, konnte durch eine Signaturenanalyse der Datei in Hex-Ansicht die für SQLite-Datenbanken verwendete Signatur '0x53 51 4C 69 74 65 20 66 6F 72 6D 61 74 20 33 00' festgestellt werden. [56]

Der Beginn der Datei 'Contacts.data' in Hex-Ansicht ist in Abbildung 5.4 abgebildet.

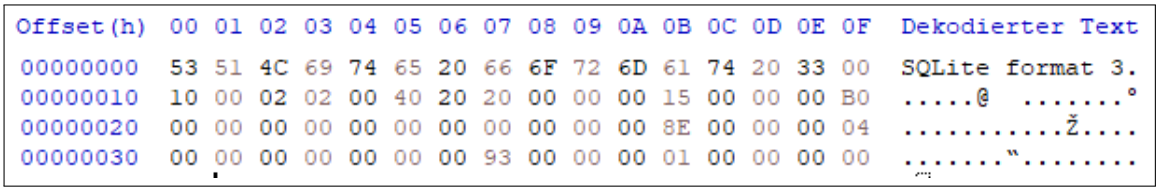

Abbildung 5.4: Hex-Ansicht der Datei Contacts.data

Die Datenbank 'Contacts.data' enthält insgesamt 43 Tabellen. Die Datenstruktur und relevante Inhalte der Tabellen werden im Folgenden näher betrachtet.

Von den insgesamt 43 festgestellten Tabellen sind 15 Tabellen relevant für diese Arbeit. Zu den weiteren 28 Tabellen wurden entweder keine relevanten Inhalte festgestellt oder der Zweck der Tabelle konnte nicht identifiziert werden.

In der Tabelle 5.5 sind Informationen zu den 15 relevanten Tabellen abgedruckt.

Tabelle 5.5: Datenstruktur der Datenbank 'Contacts.data'

Tabellenname	Spaltenanzahl	Inhalt
ZABCONTACT	18	Informationen zu den Kontakten
ZABCONTACTNUMBER	7	Weitere Informationen zu den Kontakten
ZATTACHMENT	32	Informationen zu den versendeten oder empfangenen Dateien
ZCONVERSATION	99	Informationen zu den geführten Konversationen
ZLIKE	12	Informationen zu den gelikten Nachrichten

Tabelle 5.5, Fortsetzung: Datenstruktur der Datenbank 'Contacts.data'

Tabellenname	Spaltenanzahl	Inhalt
ZMEMBER	38	Informationen zu Teilnehmern von Konversationen
ZPHONENUMBER	7	Informationen zu Telefonnummern
ZRECENT	10	Informationen zu Telefonaten
ZRECENTLINE	10	Statistische Informationen zu Telefonaten nach Kategorie
ZVIBERLOCATION	11	Informationen zu versendeten oder empfangenen Standorten
ZVIBERMESSAGE	35	Informationen zu Nachrichten
Z_1RECENTLINES	2	Information, welche Kontakte an einem Anruf teilgenommen haben
Z_10RECENTLINES	2	Information, welche Teilnehmer an einem Anruf teilgenommen haben
Z_1MEMBERS	2	Information, welcher Kontakt-Eintrag zu welchem Teilnehmer-Eintrag zugeordnet ist
Z_5BANNEDMEMBERS	2	Information, welcher Teilnehmer in welcher Konversation gesperrt ist

Die einzelnen Tabellen werden im Folgenden näher erläutert. Aufgrund der hohen Anzahl an Spalten aller Tabellen, werden in den folgenden Abschnitten nicht alle Spalten sowie deren Typ und Bedeutung abgedruckt, sondern nur jene, welche als relevant deklariert wurden.

ZABCONTACT

Nachfolgend wird auf die Datenstruktur der Tabelle 'ZABCONTACT' sowie relevante Inhalte dieser eingegangen. In der Tabelle 'ZABCONTACT' konnten insgesamt 18 Spalten festgestellt werden. In diesen Spalten sind Informationen zu den in Viber gespeicherten Kontakten abgelegt. Der Namensbestandteil 'AB' steht im Tabellenname für 'Address Book' (dt. Adressbuch).

In der Tabelle 5.6 sind die für diese Arbeit relevanten Spalten, der Datentyp sowie die Bedeutung abgedruckt.

Tabelle 5.6: Datenstruktur der Tabelle ZABCONTACT

Spalte	Typ	Bedeutung
Z_PK	INTEGER	Primärschlüssel der Tabelle 'ZABCONTACT', für die Referenzierung in weiteren Tabellen
ZISVIBERICON	INTEGER	Wert, ob Kontakt ein Profilbild hochgeladen (Nicht Vorhanden: 0, Vorhanden: 1
ZMAINNAME	VARCHAR	Nachname des Kontaktes
ZPREFIXNAME	VARCHAR	Vorname des Kontaktes

- Es wurden insgesamt 4 Spalten als relevant deklariert. Hierunter fällt, wie in allen weiteren Tabellen, der Primärschlüssel 'Z_PK' der Tabelle. Der Inhalt stellt eine fortlaufende Zahl dar, welche sich für jeden neuen Kontakt um eins erhöht. Durch diese Spalte können die Information eindeutig weiteren Informationen zu Kontakten aus anderen Tabellen oder Konversationen und Nachrichten zugeordnet werden.
- In der Spalte 'ZISVIBERICON' ist die Information gespeichert, ob dem Kontakt ein Profilbild zugeordnet ist oder nicht. Dies ist durch die Werte '0' für kein Profilbild und '1' für Profilbild abgelegt.
- Der Name, unter dem der Kontakt gespeichert ist, ist in zwei Spalten abgelegt. Zum einen ist der Nachname in der Spalte 'ZMAINNAME' gespeichert. Zum anderen ist der Vorname in der Spalte 'ZPREFIXNAME' abgelegt.

ZABCONTACTNUMBER

Nachfolgend wird auf die Datenstruktur der Tabelle 'ZABCONTACTNUMBER' sowie relevante Inhalte dieser eingegangen. Die Tabelle ist in 7 Spalten untergliedert, welche die Telefonnummern der Kontakte enthält. Der Namensbestandteil 'AB' steht im Tabellenname für 'Address Book' (dt. Adressbuch).

In der Tabelle 5.7 sind Informationen zu den relevanten Spalten abgedruckt.

Tabelle 5.7: Datenstruktur der Tabelle ZABCONTACTNUMBER

Spalte	Typ	Bedeutung
Z_PK	INTEGER	Primärschlüssel der Tabelle 'ZABCONTACTNUMBER', für die Referenzierung in weiteren Tabellen
ZCONTACT	INTEGER	Fremdschlüssel zur Tabelle 'ZABCONTACT'
ZCANONIZED-PHONENUM	VARCHAR	Telefonnummer des Kontaktes mit der Ländervorwahl
ZPHONE	VARCHAR	Telefonnummer des Kontaktes, wie eingegeben

- In der Tabelle 'ZABCONTACTNUMBER' konnten 4 relevante Spalten identifiziert werden. Der Primärschlüssel der Tabelle ist in der Spalte 'Z_PK' gespeichert.

- Die Spalte 'ZCONTACT' stellt den Fremdschlüssel zur Tabelle 'ZABCONTACT' dar. Hierbei ist der Wert des Primärschlüssels von 'ZABCONTACT' abgelegt, wodurch die Telefonnummer eines Kontaktes den weiteren Informationen dieses Kontaktes zugeordnet werden kann.
- Die Telefonnummer des Kontaktes ist in zwei Spalten enthalten. Zum einen wird in der Spalte 'ZCANONIZEDPHONENUM' die Telefonnummer mit der entsprechenden Ländervorwahl abgelegt. Zum anderen ist in der Spalte 'ZPHONE' die Telefonnummer wie vom Nutzer eingegeben gespeichert.

ZATTACHMENT

Nachfolgend wird auf die Datenstruktur der Tabelle 'ZATTACHMENT' sowie relevante Inhalte dieser eingegangen. Informationen zu versendeten oder empfangenen Anhängen, wie Bilder, Videos, Dateien usw., sind in 32 Spalten abgelegt.

In Tabelle 5.8 sind Informationen zu den relevanten Spalten abgedruckt.

Tabelle 5.8: Datenstruktur der Tabelle ZATTACHMENT

Spalte	Typ	Bedeutung
Z_PK	INTEGER	Primärschlüssel der Tabelle 'ZATTACHMENT', für die Referenzierung in weiteren Tabellen
ZFILESIZE	INTEGER	Enthält die Dateigröße des Anhangs in Byte
ZDURATION	FLOAT	Enthält die Abspieldauer der Datei, sofern Video oder Audio
ZFILENAME	VARCHAR	Enthält den Originaldateinamen, sofern als 'Datei' oder Audio versendet
ZNAME	VARCHAR	Enthält den Dateinamen auf dem Gerät
ZSTATE	VARCHAR	Enthält den Sendestatus des Anhangs (Ausstehend: pending, Abgeschlossen: complite)
ZTHUMBNAILID	VARCHAR	Enthält die Kennung des Vorschaubildes, falls vorhanden
ZTYPE	VARCHAR	Enthält den Typ des Anhangs (Bild: picture, Video: video, Audio: audio, Datei: file, Standort: customLocation, Instant Video: instantVideo)

- Von den 32 Spalten wurden 8 als relevant deklariert. Der Primärschlüssel der Tabelle ist in der Spalte 'Z_PK' gespeichert.
- Die Dateigröße, des versendeten oder empfangenen Anhangs, ist in der Spalte 'ZFILESIZE' in der Einheit Byte gespeichert.
- Sofern der Anhang eine Video- oder Audiodatei ist, ist in der Spalte 'ZDURATION' die Abspieldauer in Sekunden angegeben.
- Beim Dateinamen muss differenziert werden, ob die Datei über die Funktion 'Datei versenden' bzw. als Audio versendet worden ist oder über die anderen Funktionen. Wurde die Datei als 'Datei' oder als Audioaufnahme versendet, dann ist in der Spalte 'ZFILENAME' der Originalname vom Gerät des Senders gespeichert.
  Der Dateiname des Anhangs auf dem Gerät ist in der Spalte 'ZNAME' abgelegt.

- Aus der Spalte 'ZSTATE' ist die Information extrahierbar, in welchem Zustand die Übertragung ist. In den Testdaten konnten hier die beiden Status 'pending' für ausstehende Übertragung und 'complite' für abgeschlossene Übertragung festgestellt werden. Es kann hierbei nicht ausgeschlossen werden, dass noch weitere Status existieren, welche in dem Test nicht generiert werden konnten.
- Wurde durch die Anwendung Viber ein Vorschaubild für den Anhang erstellt, ist die Kennung zu diesem in der Spalte 'ZTHUMBNAILID' abgelegt. Über diese Kennung kann das Vorschaubild im entsprechenden Verzeichnis im Dateisystem gefunden werden.
- Die Spalte 'ZTYPE' enthält den Typ des Anhangs. In den Testdaten waren die Typen 'picture', 'video', 'audio', file', 'customLocation' und 'instantVideo' festgestellt werden. Hier kann nicht ausgeschlossen werden, dass weitere Typen existieren.

ZCONVERSATION

Nachfolgend wird auf die Datenstruktur der Tabelle 'ZCONVERSATION' sowie relevante Inhalte dieser eingegangen. In der Tabelle sind Informationen zu den geführten Konversationen enthalten.

In Tabelle 5.9 sind Informationen zu den relevanten Spalten abgedruckt.

Tabelle 5.9: Datenstruktur der Tabelle ZCONVERSATION

Spalte	Typ	Bedeutung
Z_PK	INTEGER	Primärschlüssel der Tabelle 'ZCONVERSATION', für die Referenzierung in weiteren Tabellen
ZISMUTED	INTEGER	Wert, ob Benachrichtigungen für die Konversation deaktiviert sind (Aktiviert: 0, Deaktiviert: 1)
ZISUNREAD	INTEGER	Wert, ob ungelesene Nachrichten vorhanden sind
ZUNREADCOUNT	INTEGER	Anzahl der ungelesenen Nachrichten
ZDATE	TIMESTAMP	Zeitstempel der letzten versendeten Nachricht als Apple Cocoa Core Data Zeitstempel
ZTIMEBOMB-DURATION	FLOAT	Enthält die Dauer in Sekunden, nach welcher Nachrichten automatisch gelöscht werden
ZICONID	VARCHAR	Enthält die Kennung des Icon der Konversation
ZLASTMESSAGE-TEXT	VARCHAR	Enthält den Inhalt der zuletzt versendeten Nachricht
ZLASTMESSAGE-TYPE	VARCHAR	Enthält den Typ der zuletzt versendeten Nachricht
ZNAME	VARCHAR	Enthält den Namen der Konversation
ZCOUNTRY	VARCHAR	Enthält den Ländercode, in welchem die Konversation erstellt wurde
ZTAGLINE	VARCHAR	Enthält den Beschreibungstext der Konversation

- In der Tabelle 'ZCONVERSATION' konnten 12 relevante Spalten identifiziert werden. Der Primärschlüssel der Tabelle ist in der Spalte 'Z_PK' gespeichert.
- In der Spalte 'ISMUTED' ist die Information gespeichert, ob dem lokalen Nutzer Benachrichtigungen über neue Nachrichten in der Konversation angezeigt werden. Hierbei bedeutet der eingetragene Wert '0', dass der Nutzer Benachrichtigungen angezeigt bekommt und der Wert '1', dass diese deaktiviert sind.
- Die Information, ob der lokale Nutzer ungelesene Nachrichten in der Konversation erhalten hat, ist in der Spalte 'ZISUNREAD' enthalten. Die Anzahl der ungelesenen Nachrichten ist aus der Spalte 'ZUNREADCOUNT' extrahierbar.
- Das Datum der zuletzt in der Konversation versendeten Nachricht, ist in der Spalte 'ZDATE' gespeichert. Im Gegensatz zu den bisherigen UNIX-Zeitstempeln findet hier der 'Apple Cocoa Core Data' Zeitstempel Anwendung. Dieser gibt die Sekunden an, welche seit dem 01.01.2001 vergangen sind und kann durch einen Rechner in ein lesbares Datum umgewandelt werden. [59]
- Sofern das automatische Löschen in einer Konversation aktiviert ist, ist in der Spalte 'ZTIMEBOMBDURATION' die Dauer gespeichert, nach welcher die Nachrichten automatisch gelöscht werden. Die Angabe ist in Sekunden.
- Viber bietet die Möglichkeit Gruppen, Communitys und Kanälen ein Icon zuzuweisen. Die Kennung des Bildes ist in der Spalte 'ZICONID' gespeichert.
- Die Spalte 'ZLASTMESSAGETEXT' enthält den Inhalt der zuletzt in der Konversation versendeten Nachricht. Der Typ der zuletzt versendeten Nachricht ist in der Spalte 'ZLASTMESSAGETYPE' enthalten.
- Gruppen, Communitys und Kanäle haben einen Namen, welcher in der Spalte 'ZNAME' eingetragen ist.
- Für Communitys und Kanäle ist in der Spalte 'ZCOUNTRY' der Ländercode gespeichert, in welchem diese erstellt worden sind.
- Communitys und Kanälen kann ein Beschreibungstext durch die Nutzer hinzugefügt werden. Sofern einer zugewiesen wurde, ist dieser aus der Spalte 'ZTAGLINE' extrahierbar.
- Bei der Analyse dieser Tabelle ist aufgefallen, dass die Unterhaltung, welche durch eine PIN geschützt ist, trotzdem in der Datenbank unverschlüsselt vorliegt. Somit liegt der Schluss nahe, dass die PIN lediglich zum Schutz der Konversation vor Zugriff eines Dritten, bei Bedienung des Gerätes dient.

ZLIKE

Nachfolgend wird auf die Datenstruktur der Tabelle 'ZLIKE' sowie relevante Inhalte dieser eingegangen. Die Tabelle enthält Informationen zu gelikten Nachrichten, welche in 12 Spalten aufgeteilt sind.

In Tabelle 5.10 sind Informationen zu den relevanten Spalten abgedruckt.

Tabelle 5.10: Datenstruktur der Tabelle ZLIKE

Spalte	Typ	Bedeutung
Z_PK	INTEGER	Primärschlüssel der Tabelle 'ZLIKE', für die Referenzierung in weiteren Tabellen
ZLIKEVALUE	INTEGER	Enthält die Anzahl der Likes einer Nachricht

Tabelle 5.10, Fortsetzung: Datenstruktur der Tabelle ZLIKE

Spalte	Typ	Bedeutung
ZMESSAGE-TOKEN	INTEGER	Fremdschlüssel zur Tabelle 'ZVIBERMESSAGE'
ZPREVIOUSLIKE-VALUE	INTEGER	Enthält die Anzahl der Likes, vor der letzten Änderung
ZCONVERSATION	INTEGER	Fremdschlüssel zur Tabelle 'ZCONVERSATION'
ZSENDER	INTEGER	Fremdschlüssel zur Tabelle 'ZMEMBER'
ZDATE	TIMESTAMP	Enthält den Zeitstempel der letzten Änderung als Apple Cocoa Core Data Zeitstempel

- Die Tabelle enthält 7 relevante Spalten. Der Primärschlüssel der Tabelle ist in der Spalte 'Z_PK' gespeichert.
- Die Information, wie oft eine Nachricht gelikt worden ist, ist aus der Spalte 'ZLIKEVALUE' extrahierbar. Darüber hinaus ist die Höhe der Likes, wie sie vor der letzten Änderung war, in der Spalte 'ZPREVIOUSLIKEVALUE' gespeichert.
- Der Fremdschlüssel 'ZMESSAGETOKEN' zur Tabelle 'ZVIBERMESSAGE' enthält die eindeutige Kennung der gelikten Nachricht. Über den Wert können weitere Informationen dieser Nachricht zugeordnet werden.
- In der Spalte 'ZCONVERSATION' ist die Information gespeichert, in welcher Konversation die gelikte Nachricht versendet worden ist.
- Über den Fremdschlüssel 'ZSENDER' können dem Kontakt, welcher die Nachricht gelikt hat, weitere Informationen aus der Tabelle 'ZMEMBER' zugeordnet werden.
- Zu welchem Zeitpunkt die Nachricht gelikt worden ist, ist in der Spalte 'ZDATE' in Form eines 'Apple Cocoa Core Data' Zeitstempels abgelegt.

ZMEMBER

Nachfolgend wird auf die Datenstruktur der Tabelle 'ZMEMBER' sowie relevante Inhalte dieser eingegangen. In der Tabelle sind Informationen zu Teilnehmern von auf dem Gerät gespeicherten Konversationen gespeichert. Diese sind in 38 Spalten unterteilt.

In Tabelle 5.11 sind Informationen zu den relevanten Spalten abgedruckt.

Tabelle 5.11: Datenstruktur der Tabelle ZMEMBER

Spalte	Typ	Bedeutung
Z_PK	INTEGER	Primärschlüssel der Tabelle 'ZMEMBER', für die Referenzierung in weiteren Tabellen
ZISBLOCKED	INTEGER	Wert, ob der Teilnehmer vom lokalen Nutzer blockiert ist (Nicht blockiert: 0, Blockiert: 1)
ZPUBLIC-ACCOUNTCHAT	INTEGER	Wert, ob Teilnehmer ein Öffentlicher Chat ist (Community: 6, Kanal: 7)

Tabelle 5.11, Fortsetzung: Datenstruktur der Tabelle ZMEMBER

Spalte	Typ	Bedeutung
ZDISPLAYFULL-NAME	VARCHAR	Enthält den vollständigen Namen des Teilnehmers
ZDISPLAYSHORT-NAME	VARCHAR	Enthält den Vornamen des Teilnehmers
ZICONID	VARCHAR	Enthält die Kennung des Profilbildes des Teilnehmers
ZICONSTATE	VARCHAR	Enthält die Information, ob ein Profilbild existiert (Nicht vorhanden: NULL, Vorhanden: iconExist)
ZMEMBERID	VARCHAR	Eindeutige Viber-Teilnehmer Kennung
ZNAME	VARCHAR	Enthält den Viber-Anzeigenamen des Teilnehmers

- Die Tabelle enthält 9 relevante Spalten. Der Primärschlüssel der Tabelle ist in der Spalte 'Z_PK' gespeichert.
- Ist der Teilnehmer durch den lokalen Nutzer blockiert worden, ist durch die Werte '0' für nicht blockiert und '1' für blockiert in der Spalte 'ZISBLOCKED' ersichtlich.
- In der Spalte 'ZPUBLICACCOUNTCHAT' ist die Information gespeichert, ob es sich bei dem Eintrag um einen öffentlichen Chat handelt. Aus den Testdaten waren folgende Werte extrahierbar: Community: 6 und Kanal: 7. Es kann nicht ausgeschlossen werden, dass weitere Werte für nicht getestete Funktionen existieren.
- 'ZDISPLAYFULLNAME' enthält den Vor- sowie Nachnamen des Teilnehmers. Dieser Name wird anderen Viber-Nutzer angezeigt, wenn der Teilnehmer nicht in der Kontaktliste des anderen Viber-Nutzers eingetragen ist. Hierbei ist aufgefallen, dass in der Spalte der Kontaktname eingetragen ist, sofern in der Tabelle 'ZABCONTACT' ein Eintrag zu diesem existiert. Diese Information ist in der Tabelle 'Z_1MEMBERS' enthalten.
- Der Vorname des Teilnehmers ist in der Spalte 'ZDISPLAYSHORTNAME' gespeichert. Dieser Name wird anderen Viber-Nutzer angezeigt, wenn der Teilnehmer nicht in der Kontaktliste des anderen Viber-Nutzers eingetragen ist. Auch in dieser Spalte ist der Kontakt-Vorname eingetragen, sofern in der Tabelle 'ZABCONTACT' ein Eintrag zu diesem existiert.
- Hat der Teilnehmer ein Profilbild hochgeladen, ist in der Spalte 'ZICONSTATE' diese Information durch den Wert 'iconExist' enthalten. Die Kennung des Bildes ist in der Spalte 'ZICONID' eingetragen.
- In der Spalte 'ZMEMBERID' ist die durch Viber eindeutig vergebene Teilnehmer-Kennung abgelegt.
- Die Spalte 'ZNAME' enthält den Anzeigenamen des Teilnehmers, den dieser selbst für sich bestimmt hat.

ZPHONENUMBER

Nachfolgend wird auf die Datenstruktur der Tabelle 'ZPHONENUMBER' sowie relevante Inhalte dieser eingegangen. Die Tabelle enthält Informationen zu Mobilfunknummern, welche den Teilnehmern zugeordnet sind.

In Tabelle 5.12 sind Informationen zu den relevanten Spalten abgedruckt.

Tabelle 5.12: Datenstruktur der Tabelle ZPHONENUMBER

Spalte	Typ	Bedeutung
Z_PK	INTEGER	Primärschlüssel der Tabelle 'ZPHONENUMBER', für die Referenzierung in weiteren Tabellen
ZMEMBER	INTEGER	Fremdschlüssel zur Tabelle 'ZMEMBER'
ZCANONIZED-PHONENUM	VARCHAR	Enthält die Mobilfunknummer des zugehörigen Teilnehmers ohne vorangehendem '+'
ZPHONE	VARCHAR	Enthält die Mobilfunknummer des zugehörigen Teilnehmers mit vorangehendem '+'

- Die Tabelle enthält 4 relevante Spalten. Der Primärschlüssel der Tabelle ist in der Spalte 'Z_PK' gespeichert.
- In der Spalte 'ZMEMBER' ist abgelegt, welchem Eintrag in der Tabelle 'ZMEMBER' die jeweilige Mobilfunknummer zugeordnet ist.
- Die Mobilfunknummer liegt in zwei leicht unterschiedlichen Schreibweisen vor. Zum einen ist die Nummer mit vorangehendem '+' in der Spalte 'ZPHONE' abgelegt. Zum anderen ist in der Spalte 'ZCANONIZEDPHONENUM' die Nummer ohne '+' gespeichert.

ZRECENT

Nachfolgend wird auf die Datenstruktur der Tabelle 'ZRECENT' sowie relevante Inhalte dieser eingegangen. In der Tabelle sind Informationen zu geführten Telefonaten gespeichert.

In Tabelle 5.13 sind Informationen zu den relevanten Spalten abgedruckt.

Tabelle 5.13: Datenstruktur der Tabelle ZRECENT

Spalte	Typ	Bedeutung
Z_PK	INTEGER	Primärschlüssel der Tabelle 'ZRECENT', für die Referenzierung in weiteren Tabellen
ZDURATION	INTEGER	Dauer des Telefonats in Sekunden
ZCALLLOG-MESSAGE	INTEGER	Fremdschlüssel zur Tabelle 'ZVIBERMESSAGE'
ZRECENTSLINE	INTEGER	Fremdschlüssel zur Tabelle 'ZRECENTLINE'
ZDATE	TIMESTAMP	Enthält den Zeitstempel des Telefonat-Beginns als Apple Cocoa Core Data Zeitstempel
ZCALLTYPE	VARCHAR	Enthält die Kategorie des Telefonats

- Die Tabelle enthält 6 relevante Spalten. Der Primärschlüssel der Tabelle ist in der Spalte 'Z_PK' gespeichert.
- Die Dauer des Telefonats in Sekunden ist in der Spalte 'ZDURATION' abgelegt.

- Weitere Informationen zum Telefonat sind in der Tabelle 'ZVIBERMESSAGE' abgelegt. Der Fremdschlüssel 'ZCALLLOGMESSAGE' deutet hierbei auf den Primärschlüssel der Tabelle 'ZVIBERMESSAGE'.
- Der Fremdschlüssel 'ZRECENTSLINE' verweist auf den Primärschlüssel der Tabelle 'ZRECENTSLINE'.
- In der Spalte 'ZDATE' ist der Beginn des Telefonats als 'Apple Cocoa Core Data' Zeitstempel gespeichert.
- Die Information, in welche Kategorie das Telefonat einzuordnen ist, enthält die Spalte 'ZCALLTYPE'. Aus den Testdaten sind die Kategorien 'outgoing_viber_with_video', 'incoming', 'incoming_with_video', 'missed_with_video' und 'outgoing_viber' extrahierbar. Es kann nicht ausgeschlossen werden, dass weitere Kategorien existieren.

ZRECENTSLINE

Nachfolgend wird auf die Datenstruktur der Tabelle 'ZRECENTSLINE' sowie relevante Inhalte dieser eingegangen. Die Tabelle enthält Informationen zu den Kategorien der geführten Telefonate.

In Tabelle 5.14 sind Informationen zu den relevanten Spalten abgedruckt.

Tabelle 5.14: Datenstruktur der Tabelle ZRECENTSLINE

Spalte	Typ	Bedeutung
Z_PK	INTEGER	Primärschlüssel der Tabelle 'ZRECENTSLINE', für die Referenzierung in weiteren Tabellen
ZCOUNT	INTEGER	Enthält die Anzahl der Telefonate, welche der entsprechenden Kategorie zugeordnet sind
ZDATE	TIMESTAMP	Enthält den Zeitstempel des letzten Telefonat-Beginns als Apple Cocoa Core Data Zeitstempel
ZCALLTYPE	VARCHAR	Enthält die Kategorie des Telefonats
ZPHONENUMBER	VARCHAR	Enthält die Mobilfunknummer, mit der die Telefonate geführt wurden

- Die Tabelle enthält 5 relevante Spalten. Der Primärschlüssel der Tabelle ist in der Spalte 'Z_PK' gespeichert.
- Die Spalte 'ZCOUNT' enthält die Anzahl der Telefonate, welche derselben Kategorie zugeordnet sind.
- Der Beginn des letzten Telefonats, welches der entsprechenden Kategorie zugeordnet ist, als 'Apple Cocoa Core Data' Zeitstempel ist in der Spalte 'ZDATE' gespeichert.
- Die Information, in welche Kategorie das Telefonat einzuordnen ist, enthält die Spalte 'ZCALLTYPE'. Aus den Testdaten sind die Kategorien 'outgoing_viber_with_video', 'incoming', 'incoming_with_video', 'missed_with_video' und 'outgoing_viber' extrahierbar. Es kann nicht ausgeschlossen werden, dass weitere Kategorien existieren.
- In der Spalte 'ZPHONENUMBER' ist die Mobilfunknummer des lokalen Nutzers gespeichert, mit welcher die Telefonate geführt worden sind.

ZVIBERLOCATION

Nachfolgend wird auf die Datenstruktur der Tabelle 'ZVIBERLOCATION' sowie relevante Inhalte dieser eingegangen. Die Tabelle enthält Informationen zu versendeten und empfangenen Standorten.

In Tabelle 5.15 sind Informationen zu den relevanten Spalten abgedruckt.

Tabelle 5.15: Datenstruktur der Tabelle ZVIBERLOCATION

Spalte	Typ	Bedeutung
Z_PK	INTEGER	Primärschlüssel der Tabelle 'ZVIBERLOCATION', für die Referenzierung in weiteren Tabellen
ZMESSAGE	INTEGER	Fremdschlüssel zur Tabelle 'ZVIBERMESSAGE'
ZDATE	TIMESTAMP	Enthält den Zeitstempel des Zeitpunkts der Erstellung des Standortes als Apple Cocoa Core Data Zeitstempel
ZLATITUDE	FLOAT	Enthält den Breitengrad des Standorts
ZLONGITUDE	FLOAT	Enthält den Längengrad des Standortes
ZADDRESS	VARCHAR	Enthält die interpretierte Adresse des Standortes

- Die Tabelle enthält 6 relevante Spalten. Der Primärschlüssel der Tabelle ist in der Spalte 'Z_PK' gespeichert.
- Weitere Informationen zum Standort sind in der Tabelle 'ZVIBERMESSAGE' abgelegt. Der Fremdschlüssel 'ZCALLLOGMESSAGE' deutet hierbei auf den Primärschlüssel der Tabelle 'ZVIBERMESSAGE'.
- Der Zeitpunkt der Erstellung des Standortes ist in der Spalte 'ZDATE' als 'Apple Cocoa Core Data' Zeitstempel abgelegt. Dieser unterscheidet sich minimal von dem Zeitpunkt aus der Tabelle 'ZVIBERMESSAGE', in welcher der Übertragungszeitpunkt gespeichert ist.
- Die geografischen Koordinaten in Form von Breiten- und Längengrad sind in den Spalten 'ZLATITUDE' und 'ZLONGITUDE' gespeichert.
- Da für die Erstellung des Standortes die Navigationsanwendung von Apple verwendet wird, können Informationen hieraus bezogen werden. Viber speichert die Adresse des Standortes in der Spalte 'ZADDRESS'.

ZVIBERMESSAGE

Nachfolgend wird auf die Datenstruktur der Tabelle 'ZVIBERMESSAGE' sowie relevante Inhalte dieser eingegangen. Die Tabelle enthält Informationen zu versendeten oder empfangenen Nachrichten.

In Tabelle 5.16 sind Informationen zu den relevanten Spalten abgedruckt.

Tabelle 5.16: Datenstruktur der Tabelle ZVIBERMESSAGE

Spalte	Typ	Bedeutung
Z_PK	INTEGER	Primärschlüssel der Tabelle 'ZVIBERMESSAGE', für die Referenzierung in weiteren Tabellen
ZLIKESCOUNT	INTEGER	Enthält die Anzahl, wie oft die Nachricht gelikt wurde
ZTOKEN	INTEGER	Enthält eine eindeutige Kennung der Nachricht
ZATTACHMENT	INTEGER	Fremdschlüssel zur Tabelle 'ZATTACHMENT'
ZCONVERSATION	INTEGER	Fremdschlüssel zur Tabelle 'ZCONVERSATION'
ZPHONENUM-INDEX	INTEGER	Fremdschlüssel zur Tabelle 'ZPHONENUMBER'
ZDATE	TIMESTAMP	Enthält das Datum des Übertragungszeitpunktes als Apple Cocoa Core Data Zeitstempel
ZSTATEDATE	TIMESTAMP	Enthält das Datum der letzten Status-Änderung der Nachricht als Apple Cocoa Core Data Zeitstempel
ZTIMEBOMB-DURATION	FLOAT	Enthält die Dauer in Sekunden, nach welcher Nachrichten automatisch gelöscht werden
ZCALLTYPE	VARCHAR	Enthält die Kategorie des Telefonats
ZMETADATA	VARCHAR	Enthält zusätzliche Informationen zu Übertragungen
ZSTATE	VARCHAR	Enthält den Status der Nachricht
ZSYSTEMTYPE	VARCHAR	Enthält den Typ der Nachricht, falls es eine Systemnachricht ist
ZTEXT	VARCHAR	Enthält den Inhalt der Nachricht

- In der Tabelle 'ZVIBERMESSAGE' sind 14 relevante Spalten enthalten. Der Primärschlüssel der Tabelle ist in der Spalte 'Z_PK' gespeichert.
- In der Spalte 'ZLIKESCOUNT' ist die Information gespeichert, wie oft die jeweilige Nachricht gelikt worden ist.
- Um die Nachricht eindeutig identifizieren zu können, ist in der Spalte 'ZTOKEN' eine eindeutige Kennung gespeichert. Die Kennung wird zum Beispiel in der Tabelle 'ZLIKE' genutzt.
- Ein Fremdschlüssel, welcher auf den Primärschlüssel der Tabelle 'ZATTACHMENT' verweist, ist in der Spalte 'ZATTACHMENT' abgelegt. Er dient dazu, weitere Informationen zum übertragenen Anhang der Nachricht zuzuordnen.
- In der Spalte 'ZCONVERSATION' ist der Fremdschlüssel, welcher auf den Primärschlüssel der Tabelle 'ZCONVERSATION' verweist, gespeichert. Aus dieser Spalte ist ersichtlich, in welcher Konversation die Nachricht übertragen worden ist.
- Die Information, welcher Teilnehmer die Nachricht versendet hat, wird im Fremdschlüssel 'ZPHONENUMINDEX' gespeichert. Er verweist auf den Primärschlüssel der Tabelle 'ZPHONENUMBER'.
- Der Übertragungszeitpunkt der Nachricht ist in der Spalte 'ZDATE' als Apple Cocoa Core Data-Zeitstempel abgelegt.

- Der Zeitpunkt, an dem sich der Status der Nachricht verändert hat, ist in der Spalte 'ZSTATEDATE' als Apple Cocoa Core Data-Zeitstempel abgelegt. Welchen Statuts eine Nachricht hat, ist in der Spalte 'ZSTATE' gespeichert. Hierbei konnten aus den Testdaten folgende Status identifiziert werden: 'delivered' für zugestellt, 'received' für erhalten und 'send' für gesendet.
- Wenn die Funktion der automatischen Löschung von Nachrichten aktiviert ist, ist in der Spalte 'ZTIMEBOMBDURATION' die Dauer gespeichert, nach der die Nachricht automatisch gelöscht wird.
- Wurde der entsprechende Eintrag aufgrund eines Telefonats erstellt, ist in der Spalte 'ZCALLTYPE' die Kategorie des Telefonats eingetragen.
- In der Spalte 'ZMETADATA' sind Metadaten zu Übertragungen gespeichert. In den Testdaten waren solche Metadaten zu Datei-Übertragungen, Umfragen, Reaktionen, Kanal-Nachrichten, .gif-Dateien-Übertragungen, Standorten, Einladungen, weitergeleitete Nachrichten und nachträglich veränderten Nachrichten festgestellt.
- Ist die Nachricht eine Mitteilung des Systems, ist in der Spalte 'ZSYSTEMTYPE' der Typ der Mittelung gespeichert. In den Testdaten waren die Typen 'customLocation' für Standort, 'poll' für Umfragen, 'rich' für .gif-Dateien, 'systemCallLog' für Telefonate, 'systemIconChanged' für Änderungen von Icons in den Konversationen, 'systemMemberAdded' für hinzugefügte Teilnehmer, 'systemRemovedFromGroup' für aus einer Gruppe entfernte Teilnehmer, 'systemTimebombTimerChanged' für die Veränderung der Funktion der automatisch löschenden Nachrichten, 'invite' für Einladungen zu Kanälen oder Communitys und 'pollMessageInvisible' für die Optionen der Umfrage. Der ebenfalls festgestellte Typ 'formatted' enthält Nachrichten über die Verifikation der E-Mail-Adresse oder Übertragungen von Kontakten. Darüber hinaus konnten die weiteren Typen 'systemGeneralMessageRemoved', 'systeminvalidMessage' und 'systemSelfAdded' festgestellt werden. Zu diesen konnte die Bedeutung jedoch nicht abschließend bestimmt werden.

Z_1RECENTLINES

Nachfolgend wird auf die Datenstruktur der Tabelle 'Z_1RECENTLINES' sowie relevante Inhalte dieser eingegangen. Die Tabelle wird verwendet, um Informationen aus der Tabelle 'ZABCONTACT' mit den Informationen aus der Tabelle 'ZRECENTSLINE' zu verbinden.

In Tabelle 5.17 sind Informationen zu den relevanten Spalten abgedruckt.

Tabelle 5.17: Datenstruktur der Tabelle Z_1RECENTLINES

Spalte	Typ	Bedeutung
Z_1CONTACTS	INTEGER	Fremschlüssel zur Tabelle 'ZABCONTACT'
Z_14RECENTS-LINES	INTEGER	Fremschlüssel zur Tabelle 'ZRECENTSLINE'

- Der Fremdschlüssel 'Z_1CONTACTS' verweist auf den Primärschlüssel der Tabelle 'ZABCONTACT'.
- Der Fremdschlüssel 'Z_14RECENTSLINES' verweist auf den Primärschlüssel der Tabelle 'ZRECENTSLINE'.
- Durch diese Tabelle kann herausgefunden werden, welcher Kontakt an welcher Kategorie der Telefonate mit dem lokalen Nutzer beteiligt war.

Z_10RECENTLINES

Nachfolgend wird auf die Datenstruktur der Tabelle 'Z_10RECENTLINES' sowie relevante Inhalte dieser eingegangen. Die Tabelle wird verwendet, um Informationen aus der Tabelle 'ZMEMBER' mit den Informationen aus der Tabelle 'ZRECENTSLINE' zu verbinden.

In Tabelle 5.18 sind Informationen zu den relevanten Spalten abgedruckt.

Tabelle 5.18: Datenstruktur der Tabelle Z_10RECENTLINES

Spalte	Typ	Bedeutung
Z_10MEMBERS1	INTEGER	Fremschlüssel zur Tabelle 'ZMEMBER'
Z_14RECENTS-LINES1	INTEGER	Fremschlüssel zur Tabelle 'ZRECENTSLINE'

- Der Fremdschlüssel 'Z_10MEMBERS1' verweist auf den Primärschlüssel der Tabelle 'ZMEMBER'.
- Der Fremdschlüssel 'Z_14RECENTSLINES1' verweist auf den Primärschlüssel der Tabelle 'ZRECENTSLINE'.
- Durch diese Tabelle kann herausgefunden werden, welcher Teilnehmer an welcher Kategorie der Telefonate mit dem lokalen Nutzer beteiligt war.

Z_1MEMBERS

Nachfolgend wird auf die Datenstruktur der Tabelle 'Z_1MEMBERS' sowie relevante Inhalte dieser eingegangen. Die Tabelle wird verwendet, um Informationen aus der Tabelle 'ZMEMBER' mit den Informationen aus der Tabelle 'ZABCONTACT' zu verbinden.

In Tabelle 5.19 sind Informationen zu den relevanten Spalten abgedruckt.

Tabelle 5.19: Datenstruktur der Tabelle Z_1MEMBERS

Spalte	Typ	Bedeutung
Z_1ABCONTACTS	INTEGER	Fremschlüssel zur Tabelle 'ZABCONTATC'
Z_10MEMBERS	INTEGER	Fremschlüssel zur Tabelle 'ZMEMBER'

- Der Fremdschlüssel 'Z_1ABCONTACTS' verweist auf den Primärschlüssel der Tabelle 'ZABCONTACT'.
- Der Fremdschlüssel 'Z_10MEMBERS' verweist auf den Primärschlüssel der Tabelle 'ZMEMBER'.
- Durch diese Tabelle kann herausgefunden werden, ob einem Teilnehmer ein Kontakt aus dem Adressbuch zugeordnet ist und wenn ja, welcher.

Z_5BANNEDMEMBERS

Nachfolgend wird auf die Datenstruktur der Tabelle 'Z_5BANNEDMEMBERS' sowie relevante Inhalte dieser eingegangen. Die Tabelle wird verwendet, um Informationen aus der Tabelle 'ZCONVERSATION' mit den Informationen aus der Tabelle 'ZMEMBER' zu verbinden.

In Tabelle 5.20 sind Informationen zu den relevanten Spalten abgedruckt.

Tabelle 5.20: Datenstruktur der Tabelle Z_5BANNEDMEMBERS

Spalte	Typ	Bedeutung
Z_5BANNEDIN-CONVERSATIONS	INTEGER	Fremschlüssel zur Tabelle 'ZCONVERSATION'
Z_10BANNED-MEMBERS	INTEGER	Fremschlüssel zur Tabelle 'ZMEMBER'

- Der Fremdschlüssel 'Z_5BANNEDINCONVERSATIONS' verweist auf den Primärschlüssel der Tabelle 'ZCONVERSATION'.
- Der Fremdschlüssel 'Z_10BANNEDMEMBERS' verweist auf den Primärschlüssel der Tabelle 'ZMEMBER'.
- Durch diese Tabelle kann herausgefunden werden, ob Teilnehmer in Konversationen gesperrt wurden und wenn ja, in welchen.

5.2.3 Weitere Datenbanken

Im Rahmen dieser Arbeit wurden zuvor detaillierte Informationen zu den Datenbanken 'Contacts.data' und Settings.data abgedruckt. Es konnten noch weitere Datenbanken festgestellt werden, welche nicht relevant sind oder deren Zweck nicht gänzlich festgestellt werden konnte. Im Folgenden werden die gewonnenen Erkenntnisse zu diesen Datenbanken abgedruckt.

cdr.sqlite

Die Datenbank 'cdr.sqlite' enthält augenscheinlich ein Objekt, welches Sitzungsinformationen enthält. Welche Bedeutung diese Informationen haben, konnte nicht herausgefunden werden.

events.sqlite

In der Datenbank 'events.sqlite', werden unbekannte Events mit UNIX-Zeitstempeln abgelegt. Die Ursache dieser Events war nicht einwandfrei reproduzierbar und konnte nicht analysiert werden.

lenses.sqlite

Anhand der Webseite von Viber konnte der Schluss erlangt werden, dass in der Datenbank 'lenses.sqlite' Filter gespeichert sind, mit welchen Bilder, GIFs und Videos editiert werden können. [60]

search.sqlite

Hierin werden augenscheinlich über Viber durchgeführte Suchen abgespeichert. In den Testdaten waren keine Suchen gespeichert, wodurch keine weiteren Informationen zu der Datenbank genannt werden können.

vp.sqlite

Augenscheinlich werden in der Datenbank 'vp.sqlite' Informationen zur Bezahlfunktion Viber Pay abgespeichert. In den Testdaten war diese Datenbank leer, da diese Funktion nicht getestet worden ist.

6 Rekonstruktion von Chats

In diesem Kapitel wird ein Leitfaden zur Rekonstruktion von Chatverläufen sowie weiteren wichtigen Informationen beschrieben. Hierbei sollen Informationen zu allen geführten Konversationen aus der Datenbank extrahiert werden. Weiter wird erläutert, inwiefern eine Konversation rekonstruierbar ist.

6.1 Rekonstruktion von Konversationen

Bei der Auswertung von Chatprogrammen ist eine Übersicht über Konversationen, welche der lokale Nutzer mit weiteren Nutzern der Anwendung geführt hat, hilfreich. Um für die Anwendung Viber eine solche Übersicht erstellen zu können, wird in diesem Abschnitt eine Vorgehensweise hierfür dargelegt. Dies soll als ein Leitfaden dienen und entspricht nicht der einzig möglichen Vorgehensweise. Die Informationen zu Konversationen befinden sich in zuvor beschriebenen Datenbanken. Daher wird für die Extraktion dieser Informationen mit SQL-Befehlen gearbeitet. Im Folgenden wird ein solcher Befehl aufgezeigt und dessen Funktionsweise näher beschrieben. Das Ergebnis dieses Befehls wird hiernach visuell dargestellt.

Mit dem SQL-Befehlen, welcher im Quelltext 6.1 dargestellt ist, lassen sich Informationen zu den Konversationen extrahieren. Im abgedruckten Befehl sind aus Gründen der Übersicht nicht alle relevanten Spalten enthalten, können jedoch manuell hinzugefügt werden.

Quelltext 6.1: Rekonstruktion von Konversationen

```
SELECT DISTINCT
    CASE
        WHEN ZCONVERSATION.ZNAME IS NULL
        THEN
            (SELECT DISTINCT ZMEMBER.ZDISPLAYFULLNAME
            FROM ZVIBERMESSAGE
            JOIN ZPHONENUMBER ON ZVIBERMESSAGE.ZPHONENUMINDEX =
                ZPHONENUMBER.Z_PK
            JOIN ZMEMBER ON ZPHONENUMBER.ZMEMBER = ZMEMBER.Z_PK
            WHERE ZVIBERMESSAGE.ZCONVERSATION = ZCONVERSATION.Z_PK
                AND ZPHONENUMINDEX IS NOT NULL)
        ELSE ZCONVERSATION.ZNAME
    END      AS Chatpartner_Bezeichnung,
    strftime('%d.%m.%Y␣%H:%M:%S␣',
        datetime(ZCONVERSATION.ZDATE+978307200, 'unixepoch',
        'localtime')) AS Letzte_Nachricht_Zeitpunkt,
    ZCONVERSATION.ZTIMEBOMBDURATION/3600 AS Timebomb_Dauer,
    ZCONVERSATION.ZNAME AS Name,
    ZCONVERSATION.ZTAGLINE AS Beschreibung,
    (SELECT COUNT(ZVIBERMESSAGE.Z_PK)
    FROM [ZVIBERMESSAGE]
```

```
        WHERE ZVIBERMESSAGE.ZCONVERSATION = ZCONVERSATION.Z_PK) AS
            Anzahl_Nachrichten
    FROM ZCONVERSATION
```

Im Folgenden wird die Funktionsweise des Befehls erläutert. Hierfür wird jeder Bereich des Befehls eigens beschrieben:

SELECT DISTINCT

Der erste Teil wählt die eindeutigen Zeilen aus dem Ergebnis aus.

CASE WHEN ... THEN ... ELSE ... END

Dies ist eine bedingte Anweisung. Wenn die Spalte 'ZNAME' der Tabelle 'ZCONVERSATION' den Wert NULL hat, wird eine Unterabfrage ausgeführt, um den Wert der Spalte 'ZDISPLAYFULLNAME' aus der Tabelle 'ZMEMBER' abzurufen. Hierfür müssen die Tabellen 'ZVIBERMESSAGE', 'ZPHONENUMBER' und 'ZMEMBER' über die Fremdschlüssel 'ZVIBERMESSAGE.ZPHONENUMINDEX' und 'ZPHONENUMBER.ZMEMBER' den jeweiligen Primärschlüsseln aus den anderen Tabellen zugeordnet werden. Trifft die Bedingung nicht zu, wird der Wert von 'ZCONVERSATION.ZNAME' verwendet.

Bei diesem Teil ist anzumerken, dass für Nachrichten, welche durch den lokalen Nutzer versendet worden sind, in der Spalte 'ZPHONENUMINDEX' der Wert 'NULL' eingetragen ist. Um den Chatpartner in Einzelchats zu ermitteln, wurde durch den Teil 'ZPHONENUMINDEX IS NOT NULL' der Index des Chatpartners ermittelt und hierzu der Name des Chatpartners ausgewählt.

strftime('%d.%m.%Y %H:%M:%S', datetime(ZCONVERSATION.ZDATE+978307200, 'unixepoch', 'localtime')) AS Letzte Nachricht_Zeitpunkt:

Dieser Ausdruck konvertiert das Datum und die Uhrzeit aus dem UNIX-Zeitstempel in das Format 'dd.mm.yyyy hh:mm:ss' und gibt es als Spalte 'Letzte Nachricht_Zeitpunkt' zurück. Hierbei ist zu erwähnen, dass durch die Addition von 978307200 Sekunden zum Wert aus der Spalte 'ZDATE' wird der Zeitstempel vom Apple Cocoa Core Date-Zeitstempel zum UNIX-Zeitstempel konvertiert.

ZCONVERSATION.ZTIME-BOMBDURATION/3600 AS Timebomb_Dauer

Dieser Ausdruck teilt den Wert in der Spalte 'ZTIMEBOMBDURATION' von 'ZCONVERSATION' durch 3600, um die Zeitbomben-Dauer in Stunden zu berechnen und gibt den Wert als Spalte 'Timebomb_Dauer' aus.

ZCONVERSATION.ZNAME AS Name

Hierbei soll der Wert aus der Spalte 'ZNAME' von 'ZCONVERSATION' ausgewählt werden und als Spalte 'Name' ausgegeben werden.

ZCONVERSATION.ZTAGLINE AS Beschreibung

Dieser Teil wählt den Wert der Spalte 'ZTAGLINE' von 'ZCONVERSATION' aus und gibt ihn als Spalte 'Beschreibung' zurück.

(SELECT COUNT(ZVIBERMESSAGE.Z_PK) FROM [ZVIBERMESSAGE] WHERE ZVIBERMESSAGE.ZCONVERSATION=ZCONVERSATION.Z_PK) AS Anzahl_Nachrichten

Der letzte Teil führt eine Unterabfrage aus, um die Anzahl der Nachrichten in der Tabelle 'ZVIBERMESSAGE' zu zählen, die zur entsprechenden Konversation in der Tabelle 'ZCONVERSATION' gehören. Das Ergebnis wird als Spalte 'Anzahl_Nachrichten' zurückgegeben.

Insgesamt liefert dieser SQL-Befehl eine Ergebnismenge mit den Spalten 'Chatpartner_Bezeichnung', 'Letzte Nachricht_Zeitpunkt', 'Timebomb_Dauer', 'Name', 'Beschreibung' und 'Anzahl_Nachrichten' aus der Tabelle 'ZCONVERSATION'.

In Abbildung 6.1 ist das Ergebnis abgedruckt, welches angezeigt wird, wenn der oben gezeigte Befehl auf der Datenbank 'Contacts.data' auf dem Testgerät 2 angewendet wird.

	Chatpartner_Bezeichnung	Letzte_Nachricht_Zeitpunkt	Timebomb_Dauer	Name	Beschreibung	Anzahl_Nachrichten
1	NULL	18.04.2023 13:13:53	NULL	NULL	NULL	0
2	Testgeraet 1	19.04.2023 11:04:28	NULL	NULL	NULL	41
3	Privat 1	19.04.2023 08:11:14	NULL	NULL	NULL	2
4	Privat 2	18.04.2023 15:03:52	NULL	NULL	NULL	1
5	Gruppe 1	20.04.2023 09:41:09	24.0	Gruppe 1	NULL	24
6	Community 1	19.04.2023 08:26:04	NULL	Community 1	Community für Tests	10
7	Kanal 1	19.04.2023 10:59:56	NULL	Kanal 1	Kanalbeschreibung	3

Abbildung 6.1: Ergebnis nach Ausführung des SQL-Befehls zur Rekonstruktion von Konversationen

6.2 Rekonstruktion von Chatverläufen

Das Darstellen von einzelnen Chatverläufen, der Konversationen, welche der lokale Nutzer über die Chatanwendung geführt hat, ist Gegenstand dieses Abschnittes. Hierfür wird eine Vorgehensweise erläutert, welche jedoch nicht die einzige Vorgehensweise ist, sondern als Leitfaden dienen soll. Die Informationen zu Chatverläufen befinden sich in zuvor beschriebenen Datenbanken. Daher wird für die Extraktion dieser Informationen mit SQL-Befehlen gearbeitet. Im Folgenden wird ein solcher Befehl aufgezeigt und dessen Funktionsweise näher beschrieben. Das Ergebnis dieses Befehls wird hiernach visuell dargestellt.

Mit dem SQL-Befehlen, welcher im Quelltext 6.2 dargestellt ist, lassen sich Informationen zu den Konversationen extrahieren. Im abgedruckten Befehl sind aus Gründen der Übersicht nicht alle relevanten Spalten enthalten, können jedoch manuell hinzugefügt werden.

Quelltext 6.2: Rekonstruktion von Chatverläufen

```
SELECT
    ZVIBERMESSAGE.ZCONVERSATION as Konversation,
    ZMEMBER.ZDISPLAYFULLNAME as Sender_Name,
    ZPHONENUMBER.ZPHONE as Sender_Nummer,
    strftime('%d.%m.%Y␣%H:%M:%S␣',
        datetime(ZVIBERMESSAGE.ZDATE+978307200, 'unixepoch',
        'localtime')) as Zeitpunkt,
```

```
    ZVIBERMESSAGE.ZMETADATA as Metadaten,
    ZVIBERMESSAGE.ZSTATE as Senderichtung,
    ZVIBERMESSAGE.ZTEXT as Nachricht
FROM
    ZVIBERMESSAGE
JOIN
    ZPHONENUMBER ON
    CASE
        WHEN ZVIBERMESSAGE.ZPHONENUMINDEX IS NULL
        THEN ZPHONENUMBER.Z_PK = 2
        ELSE ZVIBERMESSAGE.ZPHONENUMINDEX = ZPHONENUMBER.Z_PK
    END
JOIN
    ZMEMBER ON ZPHONENUMBER.ZMEMBER=ZMEMBER.Z_PK
WHERE
    ZVIBERMESSAGE.ZCONVERSATION = 2
ORDER BY
    ZVIBERMESSAGE.ZDATE ASC;
```

Im Folgenden wird die Funktionsweise des Befehls erläutert. Hierfür wird jeder Bereich des Befehls eigens beschrieben:

SELECT

Dieser Ausdruck wählt die angegebenen Spalten aus der Tabelle aus, um sie im Ergebnis anzuzeigen.

ZVIBERMESSAGE.ZCONVERSATION as Konversation

Der Teil des Befehls wählt die Spalte 'ZCONVERSATION' aus der Tabelle 'ZVIBERMESSAGE' aus und gibt sie als 'Konversation' im Ergebnis zurück.

ZMEMBER.ZDISPLAYFULLNAME as SenderName

Dies wählt die Spalte 'ZDISPLAYFULLNAME' aus der Tabelle 'ZMEMBER' aus und gibt sie als 'Sender_Name' im Ergebnis zurück.

ZPHONENUMBER.ZPHONE as SenderNummer

Der Teil wählt die Spalte 'ZPHONE' aus der Tabelle 'ZPHONENUMBER' aus und gibt sie als 'Sender_Nummer' im Ergebnis zurück.

strftime('%d.%m.%Y %H:%M:%S ', datetime(ZVIBERMESSAGE.ZDATE+978307200, 'unixepoch', 'localtime')) as Zeitpunkt

Dieser Ausdruck konvertiert das Datum und die Uhrzeit aus dem UNIX-Zeitstempel in das Format 'dd.mm.yyyy hh:mm:ss' und gibt es als Spalte 'Letzte Nachricht_Zeitpunkt' zurück. Hierbei ist zu erwähnen, dass durch die Addition von 978307200 Sekunden zum Wert aus der Spalte 'ZDATE' wird der Zeitstempel vom Apple Cocoa Core Date-Zeitstempel zum UNIX-Zeitstempel konvertiert.

ZVIBERMESSAGE.ZMETADATA as Metadaten

Dies wählt die Spalte 'ZMETADATA' aus der Tabelle 'ZVIBERMESSAGE' aus und gibt sie als 'Metadaten' im Ergebnis zurück.

ZVIBERMESSAGE.ZSTATE as Senderichtung

Der Teil des Befehls wählt die Spalte 'ZSTATE' aus der Tabelle 'ZVIBERMESSAGE' aus und gibt sie als 'Senderichtung' im Ergebnis zurück.

ZVIBERMESSAGE.ZTEXT as Nachricht

Dies wählt die Spalte 'ZTEXT' aus der Tabelle 'ZVIBERMESSAGE' aus und gibt sie als 'Nachricht' im Ergebnis zurück.

FROM ZVIBERMESSAGE

Hierdurch ist ersichtlich, dass die Haupttabelle, auf der die Abfrage ausgeführt wird, 'ZVIBERMESSAGE' ist.

JOIN ZPHONENUMBER ON CASE ... END

Dies führt einen JOIN auf der Tabelle 'ZPHONENUMBER' durch und verwendet eine bedingte Anweisung. Wenn 'ZVIBERMESSAGE.ZPHONENUMINDEX' den Wert NULL hat, wird die Bedingung 'ZPHONENUMBER.Z_PK = 2' verwendet, sonst wird 'ZVIBERMESSAGE.ZPHONENUMINDEX=ZPHONENUMBER.Z_PK' hergenommen.

Bei diesem Teil ist anzumerken, dass für Nachrichten, welche durch den lokalen Nutzer versendet worden sind, in der Spalte 'ZPHONENUMINDEX' der Wert 'NULL' eingetragen ist. Um die Informationen zum Sender trotzdem aus den anderen Tabellen via Fremdschlüssel zu erlangen, muss herausgefunden werden, welchen Index der lokale Nutzer in der Tabelle 'ZPHONENUMBER' hat. In den Testdaten war der Index die Nummer 2, weshalb die Bedingung 'ZPHONENUMBER.Z_PK = 2' verwendet wird.

Der Index des lokalen Nutzers lässt sich dadurch ermitteln, indem aus der Datenbank 'Settings.data' die Mobilfunknummer des lokalen Nutzers extrahiert wird und mit den Mobilfunknummern aus der Tabelle 'ZPHONENUMBER' in der Datenbank 'Contacts.data' abgeglichen wird.

JOIN ZMEMBER ON ZPHONENUMBER.ZMEMBER=ZMEMBER.Z_PK

Dies führt einen weiteren JOIN auf der Tabelle 'ZMEMBER' durch, um Informationen über den Sender aus der Tabelle abzurufen.

WHERE ZVIBERMESSAGE.ZCONVERSATION = 2

Dies legt das Filterkriterium fest, um nur Nachrichten aus der Konversation mit der ID 2 abzurufen.
Die Zahl 2 wurde hier nur als Beispiel ausgewählt. Diese Zahl kann durch einen beliebigen Index einer vorhandenen Konversation ersetzt werden.

ORDER BY ZVIBERMESSAGE.ZDATE ASC

Dies sortiert das Ergebnis nach dem Datum der Nachrichten in aufsteigender Reihenfolge.

Insgesamt liefert dieser SQL-Befehl eine Ergebnismenge mit den Spalten 'Konversation', 'Sender_Name', 'Sender_Nummer', 'Zeitpunkt', 'Metadaten', 'Senderichtung' und 'Nachricht' aus der Tabelle 'ZVIBERMESSAGE' für die angegebene Konversation zurück, wobei die Ergebnisse nach dem Datum der Nachrichten sortiert sind.

In Abbildung 6.2 ist das Ergebnis abgedruckt, welches angezeigt wird, wenn der oben gezeigte Befehl auf der Datenbank 'Contacts.data' auf dem Testgerät 2 angewendet wird.

	Konversation	SenderName	SenderNummer	Nachricht	Zeitpunkt	Metadaten	Sendestatus
1	2	Testgeraet 1	+49	Hallo	18.04.2023 13:19:01	NULL	received
2	2	Testgeraet 2	+49	Servus	18.04.2023 13:19:18	NULL	delivered
3	2	Testgeraet 1	+49		18.04.2023 13:19:41	NULL	send
4	2	Testgeraet 1	+49	Guten Morgen	19.04.2023 07:52:36	NULL	received
5	2	Testgeraet 2	+49	Servus	19.04.2023 07:52:54	NULL	delivered
6	2	Testgeraet 2	+49	(smiley)	19.04.2023 07:53:03	NULL	delivered
7	2	Testgeraet 2	+49		19.04.2023 07:55:15	{"fileInfo":...	delivered
8	2	Testgeraet 2	+49	Wer kennt es nicht	19.04.2023 07:55:24	NULL	delivered
9	2	Testgeraet 1	+49	Ich Reste mal die automatisch löschende ...	19.04.2023 07:56:28	NULL	received
10	2	Testgeraet 1	+49	Hab's wieder ausgeschaltet	19.04.2023 07:57:29	NULL	received
11	2	Testgeraet 2	+49	Oh jetzt sind alle Nachrichten weg	19.04.2023 07:59:09	NULL	delivered
12	2	Testgeraet 2	+49	NULL	19.04.2023 07:59:40	{"fileInfo":...	delivered
13	2	Testgeraet 2	+49	Hab dir mal ein Bild in Originalgrösse geschickt	19.04.2023 08:00:04	NULL	delivered
14	2	Testgeraet 2	+49	[{"iOS_bundleImageName":"share_contact_d...	19.04.2023 08:00:20	NULL	delivered

Abbildung 6.2: Ergebnis nach Ausführung des SQL-Befehls zur Rekonstruktion von Chatverläufen

7 Fazit und Ausblick

Ziel der Arbeit war es, innerhalb einer forensischen Analyse der Chatanwendung Viber auf iOS-Geräten die Ablagestruktur im Dateisystem festzustellen. Darüber hinaus sollte eine Analyse der relevanten Dateien durchgeführt werden. Die extrahierten Informationen aus der Analyse sollten in einem Leitfaden zur Rekonstruktion von Chats zur automatisierten Auswertung der Chatverläufe zum Tragen kommen. Das Ziel und das Vorgehen zum Erreichen dieses, sollten im Abgleich mit dem aktuellen Forschungsstand definiert werden.

Zu Beginn wurde verwandte Literatur zum Thema der Arbeit gesucht und analysiert. Hieraus konnte der Schluss gezogen werden, dass zwar forensische Analysen zur Chatanwendung Viber durchgeführt wurden, jedoch diese hauptsächlich unter dem Betriebssystem Android stattgefunden haben und das Betriebssystem iOS bisher nicht tiefgehend betrachtet wurde. Anhand der daraus resultierenden Freiheit in Bezug auf der Wahl des Vorgehens und dem Ziel ein automatisiertes Interpretieren der Chatanwendung Viber zu ermöglichen, wurde die Entscheidung getroffen, Testdaten auf Testgeräten zu generieren und diese mittels dazu fähigen Programmen zu analysieren.

Im Zuge der Entscheidung, welche Vorgehensweise gewählt werden soll, wurden weitere Optionen betrachtet. Zum einen bestand die Möglichkeit, mittels Reverse Engineering-Techniken und Programmen zu analysieren. Hierbei hätten tiefgehende Informationen zur Funktionsweise der Anwendung erlangt werden können. Da jedoch der Fokus dieser Arbeit nicht darauf liegt, wurde diese Vorgehensweise nicht weiter verfolgt. Zum anderen wäre es möglich gewesen, durch Hilfsprogramme, wie zum Beispiel 'Frida', SQL-Befehle, welche bei der Testdatengenerierung an die Datenbanken der Anwendung gesendet wurden, abzufangen und anhand dieser in Korrelation mit den getätigten Eingaben am Gerät Schlüsse zur Abspeicherung relevanter Daten in den Datenbanken zu ziehen. Hierbei konnte festgestellt werden, dass auch ohne diese Hilfsprogramme Informationen zu den relevanten Daten erlangt werden können. Zusätzlich wären für die beiden weiteren Möglichkeiten Root-Rechte auf den Testgeräten nötig gewesen, welche für die vorhanden Testgeräte nicht vorlagen.

Es sollten die technischen Grundlagen erläutert werden, welche zum einen das Analysieren der Anwendung möglich machen und zum anderen beim Lesen der Arbeit unterstützen sollen. Hierfür wurde unter anderem die vom Vertreiber der Chatanwendung Viber betriebene Internetseite zur Anwendung verwendet. Auch die Dokumentation für Entwickler, welche durch den Vertreiber zur Verfügung gestellt wird, wurde zurate gezogen. Hierdurch konnten viele Funktionen und Reglementierungen im Vorfeld festgestellt werden.

Bei der Durchführung der Testdatengenerierung ist aufgefallen, dass trotz vorangegangener Definition von Funktionen, welche getestet werden sollen, im ersten Durchgang nicht alle relevanten Funktionen getestet wurden. Dadurch war ein zweiter Durchlauf der Testdatengenerierung nötig. Dies sah auf den ersten Blick, wie ein Rückschlag aus, erwies sich bei der Analyse jedoch als Vorteil. Durch die Möglichkeit, unterschiedliche Stände der Datenbanken analysieren zu können, konnten weitere Erkenntnisse bei dem Vergleich der Stände gewonnen werden.

Ein Problem bei der Testdatengenerierung war jedoch die geringe Anzahl an Testgeräten. Aus eigener Erfahrung kommen in strafrechtlich relevanten Verfahren sehr häufig Gruppenkonversatio-

nen vor, welche weit mehr als nur vier Teilnehmer enthalten. Trotz der Verwendung privat genutzter Geräte zur Simulation von Gruppenchats, konnte bei der Analyse festgestellt werden, dass einige Zustände von Daten in den Datenbanken nicht erzielt oder überprüft werden konnten. Die Verwendung von mehr Testgeräten war nicht möglich, da zur Verwendung von Viber zwingend eine verfügbare Mobilfunknummer nötig ist.

Im Folgenden wird erörtert, welche Erkenntnisse in Bezug auf die, an die Arbeit im Vorfeld gestellten Fragen, erlangt werden konnten.

Wie stellt sich die Ablagestruktur von Protokolldaten, Dateien und Datenbanken der Chatanwendung Viber auf iOS dar

Die Ablagestruktur der Anwendung konnte soweit ersichtlich vollständig analysiert und dargestellt werden. Hierbei konnte festgestellt werden, dass je nach verwendeter bzw. möglicher Auslesemethode sowie vorliegendem Gerät bzw. Betriebssystem leichte Unterschiede in der Bezeichnung von Verzeichnissen vorliegen. Hauptsächlich ist zu erwähnen, dass die festgestellte APP ID der Chatanwendung Viber nicht in jeder Version identisch ist und sich in Zukunft ändern kann. Jedoch konnte erläutert werden, wie die Stammverzeichnisse der Anwendung gefunden werden können. Auch liegen je nach Auslesemethode nicht alle Verzeichnisse vor. Dies kann eine Auswertung der Anwendung einschränken.

Auch auf Dateiebene konnten sehr viele Erkenntnisse gewonnen und dargestellt werden. Hierbei ist aufgefallen, dass nicht alle relevanten Daten im Dateisystem zu finden sind, sondern auch als BLOB in Datenbanken vorliegen können. Einige Dateien liegen je nach Auslesemethode vor oder nicht und somit kann die Auswertung der Anwendung eingeschränkt sein.

In welcher Form werden Kommunikationsartefakte gespeichert

Kommunikationsartefakte werden hauptsächlich in Form von Datenbanken gespeichert. Das hierbei verwendete Datenbankformat ist das SQLite-Format. Hieraus können durch die Verwendung von SQL-Befehlen Daten ausgelesen werden. Ein weiteres verwendetes Format ist das der Plist oder bplist. In Bezug auf Dateiübertragungen konnten die Dateiformate JPEG, GIF, PNG, MP4 und M4A festgestellt werden.

Wie entstehen Kommunikationsartefakte

Kommunikationsartefakte entstehen hauptsächlich durch SQL-Befehle, welche bei der Verwendung der Anwendung an die Datenbanken gesendet werden. Dateien werden in ihren Formaten im Dateisystem von der Anwendung gespeichert.

Wie können Kommunikationsverläufe rekonstruiert werden

Es konnte ein Leitfaden erstellt werden, welcher SQL-Befehle enthält. Durch diese ist es möglich, Informationen zu Konversation und auch Chatverläufe zu diesen Konversationen zu rekonstruieren. Die Befehle sind hierbei beliebig anpassbar gestaltet, um auch weitere Daten aus den Datenbanken extrahieren zu können.

Können vom Nutzer gelöschte Daten wiederhergestellt werden

Es konnte festgestellt werden, dass in der Datenbank 'Contacts.data' die Tabelle 'ZDELETEDVIBERMESSAGE' vorhanden ist, welche auf die Speicherung von Daten, welche durch den Nutzer gelöscht worden sind, deutet. Es war nicht möglich, eine Situation zu simulieren, in welcher Daten in diese Tabelle eingetragen werden. Somit kann zu dieser Fragestellung kein Ergebnis geliefert werden. Ein Hinweis, dass Nachrichten gelöscht wurden, kann die Betrachtung der Werte des Primärschlüssel 'Z_PK' in der Tabelle 'ZVIBERMESSAGE' geben. Sollten hier Lücken in der Reihenfolge der Zahlen erkennbar sein, so wurden hier Nachrichten gelöscht.

Diese Arbeit konnte nicht alle Fragen beantworten oder alle relevanten Aspekte bearbeiten. Daher ist es notwendig, dass in Zukunft weitere Analysen der Chatanwendung Viber auf iOS-Geräten durchgeführt werden.

Bei diesen Arbeiten kann zum Beispiel die Vorgehensweise der Analyse anders gewählt und somit den Fokus auf andere Teile der Anwendung gesetzt werden. Hierdurch können weitere Informationen zu geführten Konversationen erlangt werden. Aber auch bei derselben Vorgehensweise, welche in dieser Arbeit verwendet wurde, können weitere Analysen durchgeführt werden.

Da ein wichtiger Bestandteil der forensischen Arbeit das Wiederherstellen gelöschter Artefakte ist, sollte überprüft werden, inwieweit gelöschte Daten wiederherstellbar sind und wie dies durchgeführt werden kann. Hierbei können Situationen simuliert werden, welche entsprechende gelöschte Daten in den Datenbanken hinterlegen oder vorhandene Backups könnten gelöschte Daten enthalten.

Aufgrund der begrenzten Anzahl an Testgeräten in dieser Arbeit sollten Analysen durchgeführt werden, welche Gruppenchats mit einer erhöhten Teilnehmeranzahl behandeln, um so auch realitätsnähere Unterhaltungen analysieren zu können.

Ein weiterer Aspekt, warum in der Zukunft regelmäßig die Anwendung analysiert werden sollte, ist der der Aktualisierungen. Anwendungen werden in der heutigen Zeit ständig aktualisiert. Hierbei können es einfache Problembehandlungen oder aber auch das Hinzufügen neuer Funktionen sein. Weiter ist es auch möglich, dass durch Aktualisierungen Änderungen am Datenbankschema oder der Ablagestruktur durchgeführt werden. All diese Änderungsmöglichkeiten sind ein unabdingbarer Punkt, warum weitere Analysen obligatorisch sein sollten.

Literaturverzeichnis

[1] C. Lemke und W. Brenner, „Einführung in das digitale Zeitalter“, in *Einführung in die Wirtschaftsinformatik*, Springer Berlin Heidelberg, 2014, S. 11–51. DOI: 10.1007/978-3-662-44065-0_2.

[2] F. Tenzer. „Anzahl der Smartphone-Nutzer in Deutschland in den Jahren 2009 bis 2021(in Millionen)“. (2022), Adresse: https://de.statista.com/statistik/daten/studie/198959/umfrage/anzahl-der-smartphonenutzer-in-deutschland-seit-2010/ (besucht am 06. 07. 2023).

[3] l. Lohmeier. „Anteil der wöchentlich mobilen Internetnutzer in Deutschland in den Jahren 2018 bis 2022“. (2023), Adresse: https://de.statista.com/statistik/daten/studie/762982/umfrage/anteil-der-taeglich-mobilen-internetnutzer-in-deutschland/ (besucht am 06. 07. 2023).

[4] A. Kunst. „Beliebte Arten von Smartphone-Apps in Deutschland im Jahr 2023“. (2023), Adresse: https://de.statista.com/prognosen/999785/deutschland-beliebte-arten-von-smartphone-apps (besucht am 06. 07. 2023).

[5] S. R. Department. „Ranking der größten Social Networks und Messenger nach der Anzahl der Nutzer im Januar 2023“. (2023), Adresse: https://de.statista.com/statistik/daten/studie/181086/umfrage/die-weltweit-groessten-social-networks-nach-anzahl-der-user/ (besucht am 06. 07. 2023).

[6] L. lohmeier. „Anzahl der Unique IDs bei Viber in ausgewählten Monaten von März 2015 bis September 2022“. (2023), Adresse: https://de.statista.com/statistik/daten/studie/548727/umfrage/anzahl-der-unique-nutzer-ids-bei-viber/ (besucht am 06. 07. 2023).

[7] L. Ceci. „Most popular messenger apps worldwide in January 2022, by monthly downloads(in millions)“. (2023), Adresse: https://www.statista.com/statistics/1263360/most-popular-messenger-apps-worldwide-by-monthly-downloads/ (besucht am 06. 07. 2023).

[8] A. Kunst. „Beliebteste Messenger in Deutschland im Jahr 2023“. (2022), Adresse: https://de.statista.com/prognosen/999735/deutschland-beliebteste-messenger?locale=de (besucht am 06. 07. 2023).

[9] S. R. Department. „Anzahl der Straftaten im Bereich Cybercrime in Deutschland nach Art des Delikts von 2015 bis 2021“. (2022), Adresse: https://de.statista.com/statistik/daten/studie/156866/umfrage/anzahl-der-straftaten-im-bereich-iuk-kriminalitaet-in-deutschland/?locale=de (besucht am 06. 07. 2023).

[10] F. Tenzer. „Marktanteile der mobilen Betriebssysteme am Absatz von Smartphones in Deutschland von Januar 2012 bis März 2023“. (2023), Adresse: https://de.statista.com/statistik/daten/studie/225381/umfrage/marktanteile-der-betriebssysteme-am-smartphone-absatz-in-deutschland-zeitreihe/ (besucht am 06. 07. 2023).

[11] F. Tenzer. „Marktanteile der meistverkauften Smartphone-Modelle in Deutschland im April 2023“. (2023), Adresse: https://de.statista.com/statistik/daten/studie/831655/umfrage/meistverkaufte-smartphone-modelle-in-deutschland/ (besucht am 06. 07. 2023).

[12] A. Kunst. „Beliebteste Smartphone-Marken in Deutschland im Jahr 2023“. (2023), Adresse: https://de.statista.com/prognosen/999729/deutschland-beliebteste-smartphone-marken (besucht am 06. 07. 2023).

[13] C. Sgaras, M.-T. Kechadi und N.-A. Le-Khac, „Forensics Acquisition and Analysis of Instant Messaging and VoIP Applications“, in *Computational Forensics*, Springer International Publishing, 2015, S. 188–199. DOI: https://doi.org/10.1007/978-3-319-20125-2_16.

[14] F. E. Salamh, M. M. Mirza, S. Hutchinson, Y. H. Yoon und U. Karabiyik, „What's on the Horizon? An In-Depth Forensic Analysis of Android and iOS Applications“, *IEEE Access*, Jg. 9, S. 99 421–99 454, 2021. DOI: 10.1109/ACCESS.2021.3095562.

[15] A. Mahajan, M. Dahiya und H. Sanghvi, „Forensic Analysis of Instant Messenger Applications on Android Devices“, *International Journal of Computer Applications*, Jg. 68, Nr. 8, S. 38–44, 2013.

[16] A. H. Lone, F. Ahmad, K. Ram und A. Khalique, „Implementation of Forensic Analysis Procedures for WhatsApp and Viber Android Applications“, *International Journal of Computer Applications*, Jg. 128, Nr. 12, S. 26–33, 2015.

[17] A. Vasilaras, D. Dosis, M. Kotsis und P. Rizomiliotis, „Android Device Incident Response: Viber Analysis“, in *2022 IEEE International Conference on Cyber Security and Resilience (CSR)*, IEEE, Juli 2022. DOI: 10.1109/CSR54599.2022.9850300.

[18] E. Casey, *Digital evidence and computer crime: Forensic science, computers, and the internet*. Academic press, 2011.

[19] D. Pawlaszczyk, „Digitaler Tatort, Sicherung und Verfolgung digitaler Spuren“, in *Forensik in der digitalen Welt*, Springer Berlin Heidelberg, 2017, S. 113–166. DOI: https://doi.org/10.1007/978-3-662-53801-2_5.

[20] J. Wu, G. Chen, Y. Xu, G. Li und Q. Liu, „A research of digital forensic method based on the Checkm8 heap vulnerability“, in *2021 IEEE 2nd International Conference on Information Technology, Big Data and Artificial Intelligence (ICIBA)*, IEEE, Dez. 2021. DOI: https://doi.org/10.1109/ICIBA52610.2021.9688162.

[21] M. Moreb, „Introduction to iOS Forensics“, in *Practical Forensic Analysis of Artifacts on iOS and Android Devices*, Apress, 2022, S. 37–70. DOI: https://doi.org/10.1007/978-1-4842-8026-3_2.

[22] M. Piccinelli und P. Gubian, „Exploring the iPhone Backup Made by iTunes“, *Journal of Digital Forensics, Security and Law*, 2011. DOI: https://doi.org/10.15394/jdfsl.2011.1099.

[23] F. Liu, K.-S. Liu, C. Chang und Y. Wang, „Research on the Technology of iOS Jailbreak“, in *2016 Sixth International Conference on Instrumentation & Measurement, Computer, Communication and Control (IMCCC)*, IEEE, 2016. DOI: https://doi.org/10.1109/IMCCC.2016.178.

[24] E. Schicker, *Datenbanken und SQL*. Springer Fachmedien Wiesbaden, 2017. DOI: https://doi.org/10.1007/978-3-658-16129-3.

[25] R. Steiner, *Grundkurs Relationale Datenbanken*. Springer Fachmedien Wiesbaden, 2014. DOI: 10.1007/978-3-658-04287-5.

[26] M. Owens, *The Definitive Guide to SQLite*. Apress, 2006. DOI: https://doi.org/10.1007/978-1-4302-0172-4.

[27] P. Kopacek, R. Probst und M. Zauner, „Das Betriebssystem“, S. 41–76, 1995. DOI: https://doi.org/10.1007/978-3-7091-9444-7_3.

[28] Statista Research Department. „Marktanteile der führenden Betriebssysteme weltweit von Januar 2009 bis Januar 2023“. (2023), Adresse: https://de.statista.com/statistik/daten/studie/157902/umfrage/marktanteil-der-genutzten-betriebssysteme-weltweit-seit-2009/ (besucht am 22. 04. 2023).

[29] A. Kunst. „Beliebteste Smartphone-Betriebssysteme in Deutschland im Jahr 2022“. (2023), Adresse: https://de.statista.com/prognosen/999737/deutschland-beliebteste-smartphone-betriebssysteme (besucht am 22. 04. 2023).

[30] TheTabascovideos. „FULL Steve Jobs iPhone Presentation - Steve Jobs präsentiert das neue iPhone“, Youtube. (2007), Adresse: https://www.youtube.com/watch?v=s72uTrA5EDY (besucht am 30. 04. 2023).

[31] T. Warren. „Apple reveals iPadOS for iPad with a new home screen, multitasking improvements, and more“. (2019), Adresse: https://www.theverge.com/2019/6/3/18650197/apple-ipad-os-ipados-multitasking-homescreen-features-wwdc-2019 (besucht am 30. 04. 2023).

[32] Google. „Wie aus einer Idee ein Betriebssystem für zwei Milliarden Geräte wurde“. (2023), Adresse: https://about.google/intl/ALL_de/stories/geschichte-android/ (besucht am 22. 04. 2023).

[33] D. Westhoff, „Das Betriebssystem Android“, S. 167–190, 2020. DOI: https://doi.org/10.1007/978-3-662-60855-5_7.

[34] N. S. Baron, „6. Instant messaging“, S. 135–162, Jan. 2013. DOI: https://doi.org/10.1515/9783110214468.135.

[35] ringcentral. „Introducing dynamic end-to-end encryption for RingCentral Video“. (2023), Adresse: https://www.ringcentral.com/us/en/blog/dynamic-end-to-end-encryption/ (besucht am 22. 04. 2023).

[36] Viber. „Viber Herunterladen“. (2023), Adresse: https://www.viber.com/de/download/ (besucht am 22. 04. 2023).

[37] Viber. „Viber REST API“. (2023), Adresse: https://developers.viber.com/docs/api/rest-bot-api/#message-types (besucht am 22. 04. 2023).

[38] Viber. „Chat Knowledge Base“. (2023), Adresse: https://help.viber.com/en/article/chat-knowledge-base (besucht am 22. 04. 2023).

[39] Viber. „Viber Funktionen“. (2023), Adresse: https://www.viber.com/de/features/ (besucht am 22. 04. 2023).

[40] Viber. „Viber Messenger: Video Anrufe“. (2023), Adresse: https://apps.apple.com/de/app/viber-messenger-video-anrufe/id382617920 (besucht am 22. 04. 2023).

[41] Viber. „Viber Sicherheit“. (2023), Adresse: https://www.viber.com/de/security/ (besucht am 22. 04. 2023).

[42] C. U. P.
bibinitperiod Assessment. „Community“. (2023), Adresse: https://dictionary.cambridge.org/de/worterbuch/englisch/community (besucht am 22. 04. 2023).

[43] Viber. „Communities“. (2023), Adresse: https://www.viber.com/de/communities/ (besucht am 22. 04. 2023).

[44] Viber. „Comparison: Group vs. Community vs. Channel“. (2023), Adresse: https://help.viber.com/en/article/comparison-group-vs-community-vs-channel (besucht am 22. 04. 2023).

[45] Viber. „Viber Out“. (2023), Adresse: https://www.viber.com/de/features/ (besucht am 22. 04. 2023).

[46] J. Russel. „Stickers: From Japanese craze to global mobile messaging phenomenon“. (2023), Adresse: https://thenextweb.com/news/stickers (besucht am 22. 04. 2023).

[47] Viber. „Use and Create Stickers on Viber“. (2023), Adresse: https://help.viber.com/en/article/use-and-create-stickers-on-viber (besucht am 22. 04. 2023).

[48] Viber. „Our Latest Super Feature: Viber Pay“. (2023), Adresse: https://www.viber.com/en/blog/2022-07-27/our-latest-super-feature-viber-pay/ (besucht am 22. 04. 2023).

[49] Viber. „Viber Encryption Overview“. (2023), Adresse: https://www.viber.com/app/uploads/Viber-Encryption-Overview.pdf (besucht am 22. 04. 2023).

[50] J. Mahlmann. „iOS-Update rückgängig machen - geht das?“ (2022), Adresse: https://www.heise.de/tipps-tricks/iOS-Update-rueckgaengig-machen-geht-das-7161532.html (besucht am 22. 04. 2023).

[51] Apple. „Property List“. (2023), Adresse: https://developer.apple.com/library/archive/documentation/General/Conceptual/DevPedia-CocoaCore/PropertyList.html (besucht am 14. 06. 2023).

[52] Apple. „App ID“. (2018), Adresse: https://developer.apple.com/library/archive/documentation/General/Conceptual/DevPedia-CocoaCore/AppID.html (besucht am 03. 06. 2023).

[53] Apple. „About Files and Directories“. (2023), Adresse: https://developer.apple.com/library/archive/documentation/FileManagement/Conceptual/FileSystemProgrammingGuide/Introduction/Introduction.html (besucht am 14. 06. 2023).

[54] Apple. „File System Basics“. (2023), Adresse: https://developer.apple.com/library/archive/documentation/FileManagement/Conceptual/FileSystemProgrammingGuide/FileSystemOverview/FileSystemOverview.html (besucht am 07. 08. 2023).

[55] D. Hamdi, F. Iqbal, T. Baker und B. Shah, „Multimedia File Signature Analysis for Smartphone Forensics“, in *2016 9th International Conference on Developments in eSystems Engineering (DeSE)*, IEEE, Aug. 2016. DOI: 10.1109/dese.2016.22.

[56] S. Jeon, J. Bang, K. Byun und S. Lee, „A recovery method of deleted record for SQLite database“, *Personal and Ubiquitous Computing*, Jg. 16, Nr. 6, S. 707–715, Juli 2011. DOI: https://doi.org/10.1007/s00779-011-0428-7.

[57] hfaeskornwoyke. „Key/Value-Datenbanksysteme“. (2013), Adresse: https://wikis.gm.fh-koeln.de/Datenbanken/KeyValueSysteme (besucht am 10. 06. 2023).

[58] D. Tools. „Der aktuelle Epoch Unix Timestamp“. (2023), Adresse: https://www.unixtimestamp.com/de/ (besucht am 10. 06. 2023).

[59] Misja.com. „Cocoa Core Data Timestamp Converter“. (2023), Adresse: https://www.epochconverter.com/coredata (besucht am 19. 06. 2023).

[60] Viber. „Say It All With Viber Lenses“. (2023), Adresse: https://www.viber.com/en/blog/2021-11-21/say-it-all-with-viber-lenses/ (besucht am 09. 06. 2023).

Eidesstattliche Erklärung

Hiermit versichere ich – Michael Metzger – an Eides statt, dass ich die vorliegende Arbeit selbstständig und nur unter Verwendung der angegebenen Quellen und Hilfsmittel angefertigt habe.

Sämtliche Stellen der Arbeit, die im Wortlaut oder dem Sinn nach Publikationen oder Vorträgen anderer Autoren entnommen sind, habe ich als solche kenntlich gemacht.

Diese Arbeit wurde in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegt oder anderweitig veröffentlicht.

Mittweida, 15. August 2023 | Michael Metzger

Ort, Datum | Michael Metzger