HOCHSCHULE
MITTWEIDA
UNIVERSITY OF
APPLIED SCIENCES

BACHELORARBEIT

Frau
Hannah Wiedeking

Reverse Engineering von Messengerdaten auf Mobiltelefonen am Beispiel der Datenbanken des Dienstes TamTam

2019

Fakultät **Angewandte Computer- und Biowissenschaften**

BACHELORARBEIT

Reverse Engineering von Messengerdaten auf Mobiltelefonen am Beispiel der Datenbanken des Dienstes TamTam

Autorin:
Hannah Wiedeking

Studiengang:
Allgemeine und Digitale Forensik

Seminargruppe:
F016w5-B

Erstprüfer:
Prof. Dr.-Ing. Toralf Kirsten

Zweitprüfer:
Philip Schütz, B.Sc.

Mittweida, November 2019

Faculty of **Angewandte Computer- und Biowissenschaften**

BACHELOR THESIS

Reverse Engineering of messenger data on mobile phones using the example of databases generated by the service TamTam

Author:
Hannah Wiedeking

Study Programme:
Allgemeine und Digitale Forensik

Seminar Group:
F016w5-B

First Referee:
Prof. Dr.-Ing. Toralf Kirsten

Second Referee:
Philip Schütz, B.Sc.

Mittweida, November 2019

Bibliografische Angaben

Wiedeking, Hannah: Reverse Engineering von Messengerdaten auf Mobiltelefonen am Beispiel der Datenbanken des Dienstes TamTam, 121 Seiten, 48 Abbildungen, Hochschule Mittweida, University of Applied Sciences, Fakultät Angewandte Computer- und Biowissenschaften

Bachelorarbeit, 2019

Referat

Messengerdaten auf Mobiltelefonen sind häufig für Ermittlungs- und Strafverfahren relevant. Eine händische Untersuchung dieser ist für einen digitalen Forensiker jedoch sehr arbeits- und zeitintensiv. Aus diesem Grund wird in dieser Bachelorarbeit ein möglichst effektives Verfahren zur Auswertung und Aufbereitung von Messengerdaten auf Mobiltelefonen vorgestellt, welches digitalen Forensikern als Leitfaden dienen soll. Das vorgestellte Verfahren kann für jeden Messenger verwendet werden und basiert auf der Methode des Reverse Engineerings.
Um dessen Anwendung zu demonstrieren und seine Funktionalität unter Beweis zu stellen, werden die Daten des Messengers TamTam, insbesondere die Datenbanken, exemplarisch ausgewertet und aufbereitet.

Abstract

Messenger data on mobile phones often contains relevant information in regards to police investigations and criminal proceedings. However, it is very labor-intensive and time-consuming to examine messenger data manually. That is the reason why this thesis presents a procedure for the evaluation and processing of messenger data, which minimizes the labor and time needed. The procedure presented can be used for any messenger and is intended to serve as a guideline for digital forensic experts. It is based on a method called Reverse Engineering.
Afterwards the execution of the presented procedure is demonstrated using the databases generated by the service TamTam as an example. This will also prove the functionality of this procedure.

I. Inhaltsverzeichnis

II. Abbildungsverzeichnis

III. Tabellenverzeichnis

1 Einleitung

Messengerdienste spielen für viele Menschen eine große Rolle bei der alltäglichen Kommunikation. Noch nie war es so einfach, sich mit Menschen auf der ganzen Welt auszutauschen und Nachrichten, Bilder, Sprachnachrichten oder Videos über das Internet zu teilen.
Bei einer Umfrage von *Bitkom* gaben 89% der befragten Deutschen im Jahr 2018 an, Messengerdienste zu nutzen (siehe Abbildung 1.1). Wird die Altersgruppe von 14-29 Jahren betrachtet, so liegt die Zahl der Messengernutzer sogar bei 98%. [1]
Auch im Jahr 2019 hat die Popularität von Messengerdiensten nicht abgenommen. Eine Umfrage zur Häufigkeit der Nutzung von Messengern von *statista.com* ergab, dass lediglich 13% der teilnehmenden Deutschen keine Messenger nutzen. 56% der Umfrageteilnehmer hingegen verwenden Messenger mehrmals täglich (siehe Abbildung 1.2). [2]
Die große Beliebtheit von Messengern hat zur Folge, dass von Nutzern auch Daten verschickt werden, die für Ermittlungs- und Strafverfahren von Relevanz sind. Diese Daten können von Tätern, Opfern oder Zeugen stammen und in Verbindung mit verschiedenen Verbrechen stehen. Es ist beispielsweise möglich, dass ein verschicktes Bild in Zusammenhang mit Kinderpornografie steht oder ausgetauschte Nachrichten einen Hinweis auf einen geplanten Terroranschlag enthalten.

Infolge ihrer Relevanz für Ermittlungs- und Strafverfahren zählt es häufig zu den Aufgaben eines digitalen Forensikers, gespeicherte Messengerdaten zu analysieren und die Ergebnisse der Auswertung übersichtlich darzustellen. Teilweise kann diese Aufgabe von forensischen Tools übernommen werden. Allerdings unterstützen diese Tools nicht alle Messenger in allen Versionen. Zudem birgt die Verwendung von automatischen Auswertetools verschiedene Nachteile, auf die im Abschnitt „Herausforderungen bei der Auswertung und Aufbereitung von Messengerdaten“ des zweiten Kapitels näher eingegangen wird. Aus diesem Grund ist es erforderlich, dass ein digitaler Forensiker in der Lage ist, Messengerdaten händisch auszuwerten. Da es sich hierbei um einen Prozess mit hohem Arbeits- und Zeitaufwand handelt, wird in dieser Arbeit ein Verfahren beschrieben, welches einem digitalen Forensiker ein effizientes Vorgehen bei der Untersuchung von Messengerdaten ermöglicht. Es soll als Leitfaden für digitale Forensiker dienen und ist auf alle Messenger anwendbar.
Bevor auf dieses Verfahren eingegangen wird, werden zunächst die nötigen Grundkenntnisse für das Verständnis dieser Arbeit vermittelt. Diese sind insbesondere für Leser wichtig, die mit Messengern, ihren Funktionen und ihrer Auswertung nicht im Detail vertraut sind. Im Anschluss an die Erläuterung des Verfahrens wird seine Anwendung in Kapitel 4 am Beispiel des russischen Messengers „TamTam“ demonstriert. Dies soll dem Leser die einzelnen Verfahrensschritte in ihrer Umsetzung näher bringen und zeigen, dass das Verfahren in der Praxis einsetzbar ist.

In Kapitel 5 wird das vorgestellte Verfahren und die durchgeführte Auswertung von TamTam bewertet. Zudem wird die präsentierte Vorgehensweise mit ähnlichen Herangehensweisen zur Analyse von Messengerdaten aus der Literatur verglichen. Daraufhin wird ein Ausblick auf weitere Möglichkeiten bei der Auswertung von Messengern gegeben, bevor ein abschließendes Fazit gezogen wird.

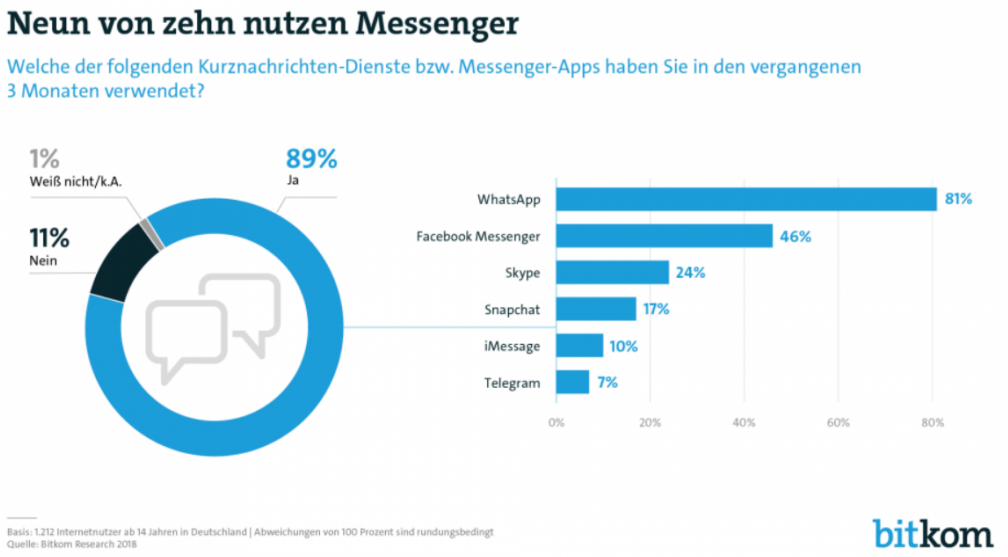


Abbildung 1.1: Statistik zur Nutzung von Messengerdiensten in Deutschland [1]

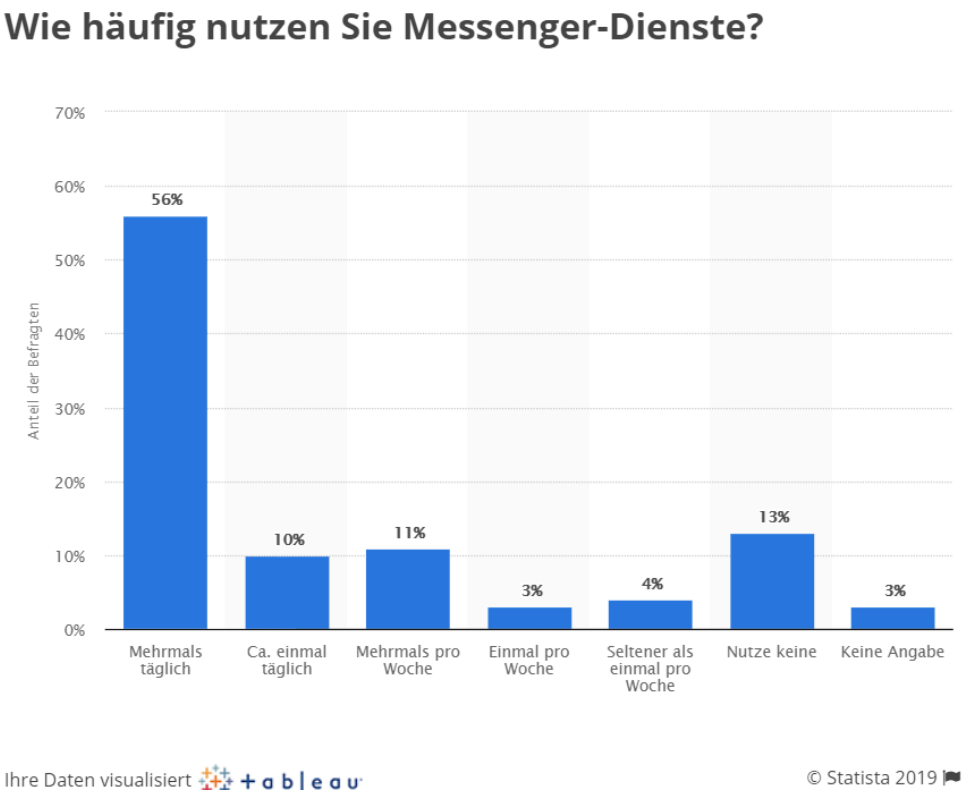


Abbildung 1.2: Statistik zur Häufigkeit der Nutzung von Messengerdiensten in Deutschland [2]

2 Grundlagen

Bevor auf ein Verfahren zur Datenauswertung von Messengerdiensten auf Mobiltelefonen eingegangen werden kann, müssen zunächst die dafür erforderlichen Grundlagen erläutert werden. Zum einen wird dabei auf Mobiltelefonmessenger und die Auswertung und Aufbereitung von Messengerdaten eingegangen. Zum anderen wird das Reverse Engineering vorgestellt. Hierbei handelt es sich um eine Analysemethode, die die Basis für das Auswerten von Messengerdaten in der digitalen Forensik darstellt und im weiteren Verlauf der Arbeit Verwendung findet.

2.1 Mobiltelefonmessenger und ihre Funktionen

Viele Messenger, wie beispielsweise WhatsApp, Telegram, KakaoTalk, LINE, Threema, Wire oder TamTam, ermöglichen es, Chats und Gruppenchats zu verwenden, um darin Text- und Sprachnachrichten oder Medieninhalte, wie Bilder und Videos, mit anderen Nutzern des entsprechenden Messengers auszutauschen. Häufig haben sie zusätzlich die Funktion der Telefonie, beziehungsweise der Videotelefonie.
Obwohl alle diese Messenger oberflächlich betrachtet über die gleichen Grundfunktionen verfügen, weichen die Messenger bei näherer Begutachtung dennoch stark voneinander ab.
Der offensichtlichste Unterschied zwischen den einzelnen Messengern ist die individuelle Benutzeroberfläche. Zudem bieten viele der Messenger zusätzliche Funktionen, die bei anderen Messengern nur gelegentlich oder gar nicht implementiert wurden. Bei Telegram, Threema oder KakaoTalk können beispielsweise Abstimmungen durchgeführt werden, bei WhatsApp oder TamTam ist dies nicht möglich [5] [6] [7]. KakaoTalk verfügt über einen in den Messenger integrierten Terminkalender, die anderen hier angeführten Messenger nicht [8].
KakaoTalk, LINE und Telegram bieten zudem die Möglichkeit, mit anderen Nutzern über private Chats zu kommunizieren [9]. Bei KakaoTalk und Telegram werden diese privaten Chats als geheime Chats bezeichnet, bei LINE als versteckte Chats. Ihr Ziel ist es, die Unterhaltungen zwischen den Nutzern privater und sicherer zu machen. Bei KakaoTalk und Telegram ist eine wichtige Eigenschaft dieser privaten Chats, dass diese im Gegensatz zu den normalen Chats Ende-zu-Ende verschlüsselt sind [10] [11]. Bei Telegram und LINE bieten die privaten Chats zudem zusätzliche Funktionen, wie zum Beispiel einen Selbstzerstörungstimer für Nachrichten [12] [13]. Bei LINE können die Nutzer mit Hilfe der “Letter Sealing“-Funktion eine Ende-zu-Ende-Verschlüsselung für alle Nachrichten aktivieren [14]. Da diese demnach nicht auf versteckte Chats begrenzt ist, handelt es sich, im Gegensatz zur Verschlüsselung bei den privaten Chats von KakaoTalk und Telegram, um kein spezielles Feature dieser Chats.

Bei Threema, WhatsApp oder Wire können keine privaten Chats angelegt werden. Trotzdem achten auch diese Messenger auf die Privatsphäre ihrer Nutzer, da alle Nachrichten Ende-zu-Ende verschlüsselt sind [15] [16] [17]. Somit ist ein solches Feature innerhalb eines privaten Chats bei diesen Messengern nicht mehr notwendig.
Eine zusätzliche Funktion von Telegram und TamTam sind die Kanäle, die von Nutzern angelegt oder abonniert werden können. Kanäle bei Telegram funktionieren dabei äquivalent zu Kanälen in TamTam, welche in Abschnitt 4.1.2 genauer erläutert werden.
Neben der Benutzeroberfläche und den zusätzlichen Funktionen ist ein weiterer und wesentlicher Unterschied die individuelle Implementierung gleicher Funktionen bei unterschiedlichen Messengern. Diese stellt eine große Herausforderung bei der Auswertung und Aufbereitung von Messengerdiensten auf Mobiltelefonen dar und beeinflusst, auf welche Art und Weise Daten gespeichert oder nicht gespeichert werden.
Welche weiteren Herausforderungen bei einer Analyse und Aufbereitung von Messengerdaten bestehen und welche Messenger zum aktuellen Zeitpunkt aufbereitet werden können, wird im folgenden Kapitel erläutert.

2.2 Auswertung und Aufbereitung von Messengerdaten

Um ein besseres Verständnis bei der Untersuchung von Messengerdaten zu gewährleisten, wird zunächst definiert, wie die Begriffe Auswertung und Aufbereitung in dieser Arbeit zu verstehen sind.

Im Rahmen einer Auswertung beziehungsweise Analyse von Messengerdaten werden die im Mobiltelefon gespeicherten Daten untersucht und interpretiert. Für diese Interpretation werden Regeln erstellt, mit deren Hilfe Daten Informationen zugeordnet werden können. Beispielsweise kann anhand der Auswertung festgestellt werden, dass eine Funktion A des Messengers bei seiner Verwendung Element B an Position C im Speicher erzeugt. Die Speicherposition ist in diesem Kontext kein absolutes Offset in einer physikalischen Datensicherung eines Mobiltelefons. Vielmehr bezeichnet sie den Speicherpfad beziehungsweise die Position eines Elements in einer Speicherstruktur, wie zum Beispiel einer Datenbank.
Mit Hilfe des Elements B kann bei der Untersuchung fremder Messengerdaten ermittelt werden, dass Funktion A verwendet wurde.
Für die Auswertung kann ein Programm geschrieben werden, welches die festgestellten Interpretationsregeln auf Messengerdaten anwendet.
Um eine korrekte Interpretation von Messengerdaten zu gewährleisten, dürfen sich die Speicherposition und die Struktur des zu analysierenden Speicherelements allerdings nicht verändern. Dies gilt für eine händische, aber auch für eine programmgestützte Auswertung. Werden die Interpretationsregeln ohne eine Anpassung auf veränderte Messengerdaten angewandt, hat dies unvollständige oder fehlinterpretierte Informationen zur Konsequenz. Der Grund dafür ist, dass sich die erstellten Regeln auf Stellen im

Speicher beziehen, an denen eine Information mit verändertem Speicherort nicht mehr gefunden werden kann. Wird dennoch versucht an dieser Stelle im Speicher eine Interpretationsregel anzuwenden, werden entweder keine interpretierbaren Daten gefunden oder es findet eine Auswertung von falschen Informationen statt.
Sind Daten zwar an der selben Stelle im Speicher abgelegt, jedoch in einem anderen Datenformat oder unter einem anderen Namen abgespeichert, so können diese ebenfalls nicht korrekt interpretiert werden. Ein verändertes Datenformat sorgt besonders bei programmbasierten Auswertungen dafür, dass die Daten gar nicht oder nicht korrekt verarbeitet werden können. Für einen digitalen Forensiker können sie unter Umständen noch lesbar sein. Ändert sich beispielsweise der Datentyp eines Datenbankfeldes von Integer auf String, so kann die Information von einem digitalen Forensiker immer noch gelesen und interpretiert werden. Ein Programm hingegen kann diese Information nicht mehr korrekt verarbeiten und stürzt im schlimmsten Fall sogar ab.
Wird der Name eines Speicherelements verändert, kommt dies in seinen Auswirkungen einer Veränderung des Speicherorts gleich. Die zu interpretierenden Daten können nicht mehr gefunden werden und Programme sowie digitale Forensiker suchen an der falschen Stelle im Speicher.

Die Aufbereitung von Daten wird durchgeführt, nachdem eine Auswertung dieser stattgefunden hat. Sie hat zum Ziel, die bei einer Analyse gewonnenen Informationen übersichtlich darzustellen und auch für einen Laien zugänglich zu machen. Das bedeutet, dass beispielsweise eine PDF-Datei, ein HTML-Dokument oder eine Exceltabelle erstellt wird, die den Kommunikationsverlauf des Nutzers sowie weitere Informationen illustriert. In den meisten Fällen geschieht dies mit Hilfe eines Programms, um auch größere Datenmengen in kurzer Zeit verarbeiten zu können.

Auswertung und Aufbereitung hängen eng miteinander zusammen. Bei der Aufbereitung werden die interpretierten Ergebnisse der Auswertung weiterverarbeitet. Liefert die Auswertung fehlerhafte oder unvollständige Informationen, hat ein Programm zur Aufbereitung lediglich diese Informationen zur Verfügung. Das Dokument, welches nach der Aufbereitung in solch einem Fall erzeugt wird, enthält keine, keine verwertbaren oder sogar fehlinterpretierte Informationen.

2.2.1 Herausforderungen bei der Auswertung und Aufbereitung von Messengerdaten

Bei der Auswertung und infolgedessen bei der Aufbereitung von Messengerdatenbanken wird ein digitaler Forensiker mit unterschiedlichen Herausforderungen konfrontiert. Die unterschiedliche Implementierung ähnlicher Funktionen sowie das Angebot zusätzlicher Funktionen mancher Messenger erschweren die Arbeit eines digitalen Forensikers. Dies kommt besonders zum Tragen, wenn er einen bisher unbekannten Messenger analysieren und aufbereiten möchte. Die Implementierung beeinflusst, welche

Daten von den Messengern auf Mobiltelefonen gespeichert und in welcher Form sie abgelegt werden. Dies führt dazu, dass die Erkenntnisse, die bei der Untersuchung eines Messengers gewonnen werden, nicht ohne Weiteres auf einen anderen Messenger übertragbar sind. Die Daten jedes Messengers müssen somit individuell ausgewertet und aufbereitet werden.
Dieser Umstand zeigt sich auch an den Datenbanken der verschiedenen Messenger, welche in dieser Arbeit besonders im Vordergrund stehen. Es ist nicht ausreichend, die Datenbanken eines Messengers zu kennen, um die Datenbanken eines anderen Messengers zu interpretieren.
Einer der möglichen Unterschiede bei Datenbanken von unterschiedlichen Messengern besteht in der Anzahl der Datenbankdateien, die auf einem Mobiltelefon abgelegt werden. Dies ist beispielsweise bei einem Vergleich der Datenbanken von Telegram (v3.4.2, Android) mit lediglich einer Datenbank, LINE (v5.10.0, Android) mit 18 verschiedenen Datenbanken und KakaoTalk (v5.3.3, Android) mit drei unterschiedlichen Datenbanken zu beobachten [9].
Ein weiterer Unterschied von Messengerdatenbanken ist die abweichende Struktur von Datenbanken, die zwar ähnliche Attribute und damit einen ähnlichen Inhalt aufweisen, bei ihren Attributnamen und deren Datentypen jedoch stark von einander abweichen. Als Beispiel können hier die Messenger TamTam (v2.6.2, iOS) und WhatsApp (v2.19.31, iOS) und die Art und Weise, wie sie den Inhalt verschickter oder empfangener Nachrichten in ihre Datenbanken ablegen, angeführt werden. WhatsApp legt diese Daten auf iOS-Geräten in ein dafür definiertes Feld *ZTEXT* vom Typ VARCHAR in Tabelle *ZWAMESSAGE* der Datenbank *ChatStorage.sqlite* ab. TamTam hingegen speichert den Nachrichteninhalt bei iOS-Geräten in der Datenbank *app.db* in Tabelle *database2* innerhalb eines Feldes *data* vom Datentyp BLOB[1].
Ein Überblick über die Datenbank *ChatStorage.sqlite* von WhatsApp (v.2.12.13) auf iOS-Geräten wird von S. Tahiri in [19, S. 117 - 123] gegeben. Dort wird sich zwar nicht auf die zuvor erwähnte WhatsApp Version bezogen, allerdings ist auch dort ersichtlich, dass das Attribut *ZTEXT* den Nachrichteninhalt als Text speichert und nicht wie TamTam als BLOB.
Enthält eine Datenbank in einer ihrer Spalten BLOBs, so handelt es sich um eine zusätzliche Herausforderung bei der Analyse von Messengerdatenbanken und die BLOBs müssen erst aufwändig analysiert werden, um ihren Inhalt interpretieren zu können. Dies wird am Beispiel der Untersuchung von TamTam in Kapitel 4 deutlich.
Zusätzlich zur abweichenden Anzahl und Struktur der Datenbanken, speichern die verschiedenen Messenger Informationen über ihre Nutzung auch unterschiedlich ausführlich ab. Das bedeutet, dass vergleichbare Messenger mit ähnlichen Funktionen nicht zwangsläufig äquivalente Daten abspeichern. Der Grund dafür besteht darin, dass manche Messenger einen großen Wert darauf legen, besonders viele Daten auf Servern und nicht lokal zu speichern, wohingegen andere Messenger, zu denen unter anderem der Messenger TamTam gehört, möglichst viele Daten auch offline anbieten möchten. Die

[1] Ein BLOB (Binary Large Object) verfügt über keine einheitlich definierte Struktur und besteht aus Binärdaten [18].

Konsequenz daraus ist, dass einige Messenger viele Zusatzinformationen speichern, wohingegen andere Messenger darauf verzichten die Daten von grundlegenden Funktionen zu hinterlegen. Als Beispiel für einen Messenger, der grundlegende Daten nicht auf dem Mobiltelefon des Nutzers ablegt, kann der „Facebook Pages“-Seitenmanager (v.77.0.0.17.68) angeführt werden. Dieser ermöglicht es, Facebook-Seiten zu verwalten und Nachrichten zu lesen und zu schreiben [20]. Bei der Analyse einer Testdatenbank (*threads_db2_<Facebook UserKey>*) konnte festgestellt werden, dass die Tabelle *threads* zwar die Anzahl der Nachrichten speichert, die zwischen zwei Nutzern ausgetauscht wurden, der Inhalt dieser Nachrichten ist jedoch nicht in der genannten Datenbank dokumentiert.
Weitere Herausforderungen bei der Analyse und Aufbereitung von Messengerdiensten sind die Abweichungen der Daten, welche durch das verwendete Betriebssystem und die unterschiedlichen Versionen eines Messengers hervorgerufen werden. Beide Schwierigkeiten betreffen, im Gegensatz zu den zuvor genannten Herausforderungen, nicht die Untersuchung verschiedener Messenger, sondern die Auswertung eines einzelnen Messengers. Obwohl es sich um den gleichen Messenger handelt, führen unterschiedliche Betriebssysteme, wie Android oder iOS, oft zu schwerwiegenden Abweichungen bei der Hinterlegung der Daten. Dies betrifft unter anderem die Datenbanken, die durch einen Messenger auf dem Mobiltelefon gespeichert werden und hat zur Folge, dass die Analyse für jedes Betriebssystem individuell notwendig ist.
Dies wird unter anderem von C. Anglano im Rahmen seiner Untersuchung von WhatsApp auf Android-Geräten dargelegt. Er stellt in seinem Fazit fest, dass die Ergebnisse seiner Publikation lediglich für Android gelten, da bei anderen Betriebssystemen, wie iOS, die durch WhatsApp gespeicherten Informationen sowie ihr Format variieren. [21]
Auch Sudozai et al. beschreiben, dass die Daten des von ihnen untersuchten Messengers IMO bei Android- und iOS-Geräten voneinander abweichen, weshalb die Daten getrennt von einander betrachtet werden [22].
Im Fall des Messengers TamTam trifft dieser Umstand ebenfalls zu. Die Datenbanken der verschiedenen Betriebssysteme (Android und iOS) weichen sowohl in der Anzahl, als auch in ihrer Struktur voneinander ab (siehe Kapitel 4.3).
Die Version des Messengers kann die Elemente, welche von einem Messenger abgelegt werden, ebenfalls beeinflussen. Eine Veränderung der Spuren ist nicht bei jedem Versionswechsel eines Messengers gegeben, kann aber unvorhergesehen auftreten. Bei der Untersuchung der Artefakte der App Telegram auf Android-Geräten durch C. Anglano, M. Canonico und M. Guazzone konnten beispielsweise keine Unterschiede für die Telegramversionen v.3.15 bis v.3.18 festgestellt werden und die Analyseergebnisse konnten auf alle genannten Versionen übertragen werden [23]. Im Gegensatz dazu zeigt die forensische Software „UFED Physical Analyzer“ (Fa. Cellebrite) [24] [2], dass es Unterschiede zwischen den Telegramversionen v.3.15 bis v.3.18 und den Versionen v3.1.1 und v3.9.1 gibt.

[2] Bei dem UFED Physical Analyzer handelt es sich um ein Tool zur automatischen Auswertung und Aufbereitung von Mobiltelefonsicherungen. Durch ihn können die Daten verschiedener Apps, darunter auch eine Vielzahl an Messengerdiensten, interpretiert und aufbereitet werden.

Der UFED Physical Analyzer kann verschiedene Versionen von Telegram analysieren und automatisch aufbereiten, darunter auch die von Anglano et al. betrachteten Versionen v.3.15 bis v.3.18. Eine Aufbereitung der Telegramversionen v3.1.1 oder v3.9.1 ist durch die Software jedoch nicht möglich. Dies spricht für die Tatsache, dass die Software zwar grundsätzlich mit den Artefakten und Strukturen von Telegram umgehen kann, nicht jedoch mit diesen Versionen, da hier die vom Messenger abgelegten Artefakte von denen aus den analysierbaren Versionen abweichen.
Dieses Beispiel zeigt, dass nicht alle Versionen eines Messengers vom UFED Physical Analyzer unterstützt werden. Welche Messenger in welcher konkreten Version vom UFED Physical Analyzer automatisch ausgewertet werden können, kann einer speziellen Liste der Firma Cellebrite entnommen werden.
Grundsätzlich kann davon ausgegangen werden, dass mit zunehmender Größe eines Messengerupdates, beispielsweise von v.3.X auf v.4.X, die Wahrscheinlichkeit steigt, dass sich die abgelegten Artefakte ändern.
Besonders, wenn bei einem Messenger neue Funktionen hinzugefügt werden, müssen sich die Artefakte des Messengers zwangsläufig ändern. Diese Änderungen können allerdings mehr oder weniger gravierend sein. Im besten Fall kann an die vorherigen Untersuchungsergebnisse für einen Messenger angeknüpft werden und die neue Funktion, die mit einer neuen Version in den Messenger Einzug erhält, kann gezielt analysiert und nachträglich in Auswertetools hinzugefügt werden. Dies ist zutreffend, wenn die Entwickler beispielsweise lediglich ein neues Attribut zu einer Tabelle einer Datenbank hinzugefügt haben. Beeinflusst die neue Funktion allerdings bestehende Funktionen, ist diese Vorgehensweise häufig nicht anwendbar. Bei einer Veränderung der Speicherposition oder der Struktur der zu analysierenden Daten muss eine erneute Untersuchung und Anpassung der Auswertetools stattfinden. Wie genau die Speicherposition und die Struktur die Interpretation der Daten beeinflussen, wurde zuvor im Rahmen der Erläuterung des Begriffs Auswertung (siehe Kapitel 2.2) dargelegt, weshalb an dieser Stelle nicht mehr darauf eingegangen wird.

2.2.2 Stand der Technik bei der Interpretation von Messengerdaten

Je nach Datenmenge und Komplexität der Artefakte, die ein Messenger anlegt, kann eine händische Untersuchung und Aufbereitung sehr viel Zeit beanspruchen und muss möglicherweise auf die prominentesten Funktionen des Messengers beschränkt werden.
Für viele Messengerdienste können derzeit bereits Tools wie der UFED Physical Analyzer (Fa. Cellebrite) [24] oder Magnet AXIOM (Fa. Magnet Forensics) [25] [3] zur automatischen Analyse und Aufbereitung verwendet werden. Beide Tools decken die beliebtesten Messenger wie WhatsApp, Telegram, LINE oder KakaoTalk ab, unterstützen jedoch auch viele weitere Messenger.

[3] Bei Magent Axiom handelt es um eine forensische Software, die Datensicherungen von Mobiltelefonen ähnlich wie der UFED Physical Analyzer auswertet und aufbereitet.

Sowohl für den UFED Physical Analyzer, als auch für Magnet Axiom stehen Listen zur Verfügung, aus denen hervorgeht, welche Messenger konkret ausgewertet werden können und in welchen Fällen es zu den bereits erwähnten Versionslücken kommt (siehe Abschnitt 2.2.1). Kann ein Messenger von einem Tool wie dem UFED Physical Analyzer oder Magnet Axiom aufbereitet werden, minimiert sich der Arbeitsaufwand für digitale Forensiker. Zum einen muss sich nicht jeder Sachbearbeiter in die Artefakte der verschiedenen Messenger einarbeiten, um Verfahrensfragen beantworten zu können. So ist es möglich, Kommunikationen, ohne Kenntnisse über die Struktur der dahinterstehenden Datenbank, einzusehen. Zum anderen wird auch die Aufgabe der Aufbereitung der Daten des entsprechenden Messengers von den Tools übernommen. Ein digitaler Forensiker muss sich nicht mehr damit auseinandersetzen, die bei einer Analyse gewonnenen Informationen für Ermittlungs- und Strafverfahren verwertbar zu machen und sie in eine für Fachfremde lesbare Form zu übertragen.
Da sowohl die Analyse, als auch die Aufbereitung der Daten von den Tools übernommen werden, entfällt für den digitalen Forensiker eine langwierige Vorarbeit und er kann sich schneller detaillierteren Fragestellungen des betreffenden Falls widmen.
Obwohl die Nutzung von Auswertungs- und Aufbereitungstools die Arbeit eines digitalen Forensikers erleichtert, gibt es einen massiven Nachteil bei ihrer Verwendung: Der digitale Forensiker muss sich auf die Daten und Informationen, die das entsprechende Tool anzeigt, verlassen. Er selbst verfügt unter Umständen nicht mehr über die Kenntnisse, die für eine Auswertung erforderlich sind und kann die Interpretation des Tools nicht ohne einen großen Arbeitsaufwand überprüfen.
Bei Ermittlungs- und Strafverfahren ist es von größter Wichtigkeit, Fehlinterpretationen zu vermeiden und Daten korrekt und nachvollziehbar auszuwerten. Aus diesem Grund müssen Ergebnisse einer Auswertung immer hinterfragt werden.
Im Falle von renommierten proprietären Tools wie dem UFED Physical Analyzer oder Magnet Axiom kann zwar davon ausgegangen werden, dass von ihnen ausgewertete Messenger zu großen Teilen korrekt interpretiert werden, allerdings kann es auch bei diesen Tools zu Fehlern kommen. Im Rahmen eigener Untersuchungen eines Samsung Galaxy S6s wurde zum Beispiel beobachtet, dass bei dem UFED Physical Analyzer (v.7.18.106) Broadcastnachrichten des Messengers WhatsApp (v.2.16.133) fehlerhaft ausgewertet wurden. Broadcasts sind Nachrichten, die WhatsApp an mehrere Kontakte gleichzeitig sendet. Bei Magnet Axiom wurde eine entsprechende Broadcastnachricht korrekt interpretiert. Der UFED Physical Analyzer hingegen ordnet die Broadcastnachricht lediglich einem der Adressaten als Einzelnachricht zu. Wären die Ergebnisse vom UFED Physical Analyzer nicht hinterfragt und durch ein weiteres Tool (hier Magnet Axiom) und anhand der WhatsApp-Datenbank überprüft worden, so wäre der Fehler unentdeckt geblieben.
Unbekanntere Auswertungstools müssen ebenfalls kritisch betrachtet werden. Solange nicht nachgewiesen wurde, dass diese Tools aus vertrauenswürdiger Quelle stammen und Daten zuverlässig aufbereiten, sollten diese für Ermittlungs- und Strafverfahren nicht verwendet werden beziehungsweise nur als Unterstützung für die eigenen Untersuchungen dienen. Als zuverlässig können Tools bezeichnet werden, wenn die genaue

Arbeitsweise einzusehen ist und klar dokumentiert wurde, wann welche Daten aufbereitet werden.
Ein weiteres Problem bei der Verwendung von Auswertungs- und Aufbereitungstools ist die Abhängigkeit von ihrem Angebot. Besonders bei unbekannteren Messengern besteht die Gefahr, dass diese durch Tools nicht ausgewertet werden können. Ebenfalls gibt es Situationen, in denen ein Tool zwar prinzipiell die Auswertung eines bestimmten Messengers ermöglicht, wichtige Funktionen jedoch nicht unterstützt werden. Somit liefert das Tool eine unvollständige Auswertung. Zudem kann es vorkommen, dass bestimmte Informationen, die für eine Ermittlung relevant sind, in der Analyse des Tools nicht enthalten sind. Dies kann daran liegen, dass es sich nicht um die Auswertung von Grundfunktionen handelt, sondern um spezifische Informationen, für die das Tool nicht detailliert genug arbeitet.
Kann ein Messenger nicht in dem benötigten Rahmen ausgewertet werden, so ist ein digitaler Forensiker, der die Messengerdaten nicht händisch analysieren kann, dazu gezwungen, auf ein Update des Tools zu warten, bis dies der Fall ist. Da allerdings unbekannt ist, ob und wann ein Messenger in Zukunft von einem Tool unterstützt wird, ist diese Strategie für Strafverfahren ungeeignet. Besonders bei zeitkritischen Ermittelungen, bei denen es beispielsweise um einen geplanten Terroranschlag geht, ist ein Abwarten nicht möglich.
Die hier aufgeführten Nachteile bei der Verwendung von Tools zur automatischen Auswertung und Aufbereitung von Messengern zeigen, dass es wichtig ist, als digitaler Forensiker die Fähigkeit zu besitzen Messengerdaten zu interpretieren, um auf eine händische Analyse und Auswertung zurückgreifen zu können. Um diesen Prozess zu optimieren, wird ein effizientes Vorgehen benötigt, an dem sich ein digitaler Forensiker in solch einem Fall orientieren kann. Ein solches Verfahren soll in dieser Arbeit vorgestellt und anhand der Daten des Messengers TamTam überprüft werden.

2.2.3 Informationsquellen für eine Auswertung und Aufbereitung von Messengerdaten

Als Informationsquelle für die Auswertung und Aufbereitung von Messengerdaten sind besonders Datenbanken gut geeignet. Sie speichern häufig zahlreiche Informationen über den Accountinhaber, seine Kommunikation und die Gesprächspartner. Diese gesammelten Daten können, sofern ihr Inhalt analysiert wird, wertvolle Informationen für Ermittlungs- und Strafverfahren liefern.
In Fällen, in denen die gewünschten Informationen in den Datenbanken des Messengers fehlen, besteht die Möglichkeit, weitere Elemente, die der Messenger im Speicher eines Mobiltelefons erzeugt, zu untersuchen. Zu den Elementen, die neben den Datenbanken des Messengers in jedem Fall ausgewertet werden sollten, zählen Bilder, Videos und andere Dateien, die über den Messenger verschickt wurden.
Weitere Elemente, die im Speicher abgelegt werden, sind für jeden Messenger individuell und können aus diesem Grund nicht aufgezählt werden. Telegram (v.3.15 bis

v.3.18, Android) legt beispielsweise auf Android-Geräten eine XML-Datei mit Informationen über den Nutzeraccount an [23].
Eine zusätzliche Informationsquelle ist zudem der Programmcode des Messengers. Dieser kann bei der Auswertung hilfreich sein, da dort nachvollziehbar ist, wie die Nutzerdaten abgelegt werden. Verwendet eine Datenbank beispielsweise BLOBs, so kann der Quelltext Aufschluss darüber geben, welche Informationen an welcher Stelle des BLOBs abgelegt wurden. Zudem bietet der Quelltext die Möglichkeit, Erkenntnisse, die bei einer manuellen Analyse gewonnen wurden, zu verifizieren.
Problematisch sind Messenger, deren Quelltext nicht öffentlich zugänglich ist. Bei diesen muss der Quelltext erst mittels Reverse Engineering aus der Installationsdatei des Messengers (APK) rekonstruiert werden. Um dies durchzuführen, sind tiefgreifende Kenntnisse und ein hoher Arbeitsaufwand erforderlich, somit ist diese Methodik nicht für jeden zugänglich.

2.3 Reverse Engineering

Allgemein versteht man unter dem Reverse Engineering einen Prozess, dessen Ziel es ist, ohne Vorkenntnisse Wissen über ein existierendes System und seine Strukturen zu erhalten. Im Gegensatz zu dem Forward Engineering steht also nicht die Entwicklung eines Systems zur Lösung eines Problems im Vordergrund, sondern der Rückweg von dem Endsystem zu seiner Entstehung. [3]
Den Unterschied zwischen Forward und Reverse Engineering verdeutlicht Abbildung 2.1.

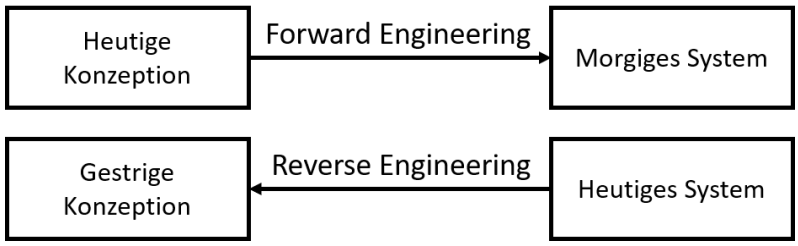


Abbildung 2.1: Der Unterschied zwischen Forward und Reverse Engineering basierend auf [3], Grafik orientiert sich an [4]

Reverse Engineering stammt ursprünglich aus dem Bereich der Hardwareanalyse. Dort war und ist die Intention des Reverse Engineerings, anhand des fertigen Produkts Erkenntnisse über seine Konzeption zu gewinnen. Somit können eigene Produkte optimiert und Konkurrenzprodukte untersucht und nachgebildet werden. [3]
Mit der Zeit hat sich die Verwendung von Reverse Engineering ausgeweitet und bezieht sich nicht mehr nur auf die Hardware- sondern auch auf die Softwareanalyse.
Mittels Software Reverse Engineering ist es beispielsweise möglich, Signaturen in Malware festzustellen. Diese können im Anschluss von Antivirenprogrammen verwendet

werden, um Viren beziehungsweise Malware zu identifizieren und unschädlich zu machen. Zudem wird Software Reverse Engineering genutzt, um Fehler, die während des Forward Engineerings entstehen, entdecken und beheben zu können, bevor sie Schaden am System anrichten. Des Weiteren eignet sich Software Reverse Engineering für das Untersuchen und Verstehen von fremden Quelltexten. So kann von den Produkten und Systemen anderer gelernt werden. [26, S. ix - x]
Auch in der digitalen Forensik wird Reverse Engineering sowohl im Hard- als auch im Softwarebereich eingesetzt. Reverse Engineering von Software wird zur Analyse der Daten von IT-Asservaten verwendet. Hierbei ist es häufig das Ziel, herauszufinden unter welchen Umständen welche Daten im Speicher eines IT-Asservats generiert werden. So ist es möglich, Informationen über den Geräteinhaber und sein Nutzungsverhalten abzuleiten. In dieser Arbeit bildet das Reverse Engineering von Software die Grundlage des Verfahrens, welches in Kapitel 3 vorgestellt wird.
Reverse Engineering von Hardware findet in der digitalen Forensik dann statt, wenn ein Gerät sichergestellt wurde und herausgefunden werden muss, welche Funktion dieses innehat.
Ein weiterer Zweck für die Verwendung von sowohl Hard- als auch Software Reverse Engineering ist das Aufdecken von Sicherheitslücken. In der digitalen Forensik werden diese Sicherheitslücken unter anderem genutzt, um Datensicherungen von IT-Asservaten anzufertigen.
Werden die vorgestellten Einsatzmöglichkeiten des Reverse Engineerings betrachtet, wird deutlich, dass diese Methodik einen großen Nutzen für die Forschung sowie die digitale Forensik hat. Allerdings ist es ebenfalls möglich, das Reverse Engineering für illegale Zwecke zu verwenden. Im besonderen Fokus steht hierbei die Spionage. Reverse Engineering eignet sich gut dazu, Konzeptionen anderer zu stehlen und zu verkaufen beziehungsweise für eigene Zwecke zu nutzen. Auch können die zuvor angesprochenen Sicherheitslücken benutzt werden, um unberechtigten Zugriff auf ein fremdes System zu erlangen und Schaden anzurichten.
Elliot Chikofsky vergleicht die Methodik des Reverse Engineerings mit der Verwendung eines Stethoskops. Dieses kann für medizinische Zwecke verwendet werden. Es ist jedoch ebenfalls möglich, dass es von einem Einbrecher genutzt wird, um einen Safe zu öffnen. Für Chikofsky ist das Stethoskop und somit auch das Reverse Engineering, aus diesem Grund weder von Natur aus gut, noch schlecht. Es kommt darauf an, wie es verwendet wird. [26, S. vii - ix]

3 Methoden

3.1 Leitfaden zur Untersuchung von Messengerdiensten

Das Verfahren, das in dieser Arbeit präsentiert und angewandt wird, lässt sich in drei Phasen einteilen: die Vorbereitungsphase, die Auswertungs- beziehungsweise Analysephase und die Aufbereitungsphase. Die Begriffe Auswertung und Aufbereitung wurden bereits im Vorfeld in Kapitel 2.2 definiert. Im Folgenden wird nun auf konkrete Schritte eingegangen, die im Rahmen einer Auswertung und Aufbereitung durchzuführen sind, um eine möglichst effiziente Umsetzung zu gewährleisten. Das Verfahren kann für die Auswertung und Aufbereitung verschiedener Elemente verwendet werden. Eine Übersicht über die einzelnen Verfahrensschritte ist in Abbildung 3.1 dargestellt.
Die Durchführung des Verfahrens wird exemplarisch an den Datenbanken des Messengers TamTam demonstriert.
Für das Verfahren ist eine individuelle Durchführung für jedes zu untersuchende Betriebssystem erforderlich. Aus diesem Grund ist darauf zu achten, dass nur Daten von Mobiltelefonen mit dem gleichen Betriebssystem analysiert werden.
Da auch die Version eines Messengers eine Rolle bei der Interpretation der Messengerdaten spielen kann, sollte neben dem gleichen Betriebssystem auch die gleiche Messengerversion zur Untersuchung verwendet werden.

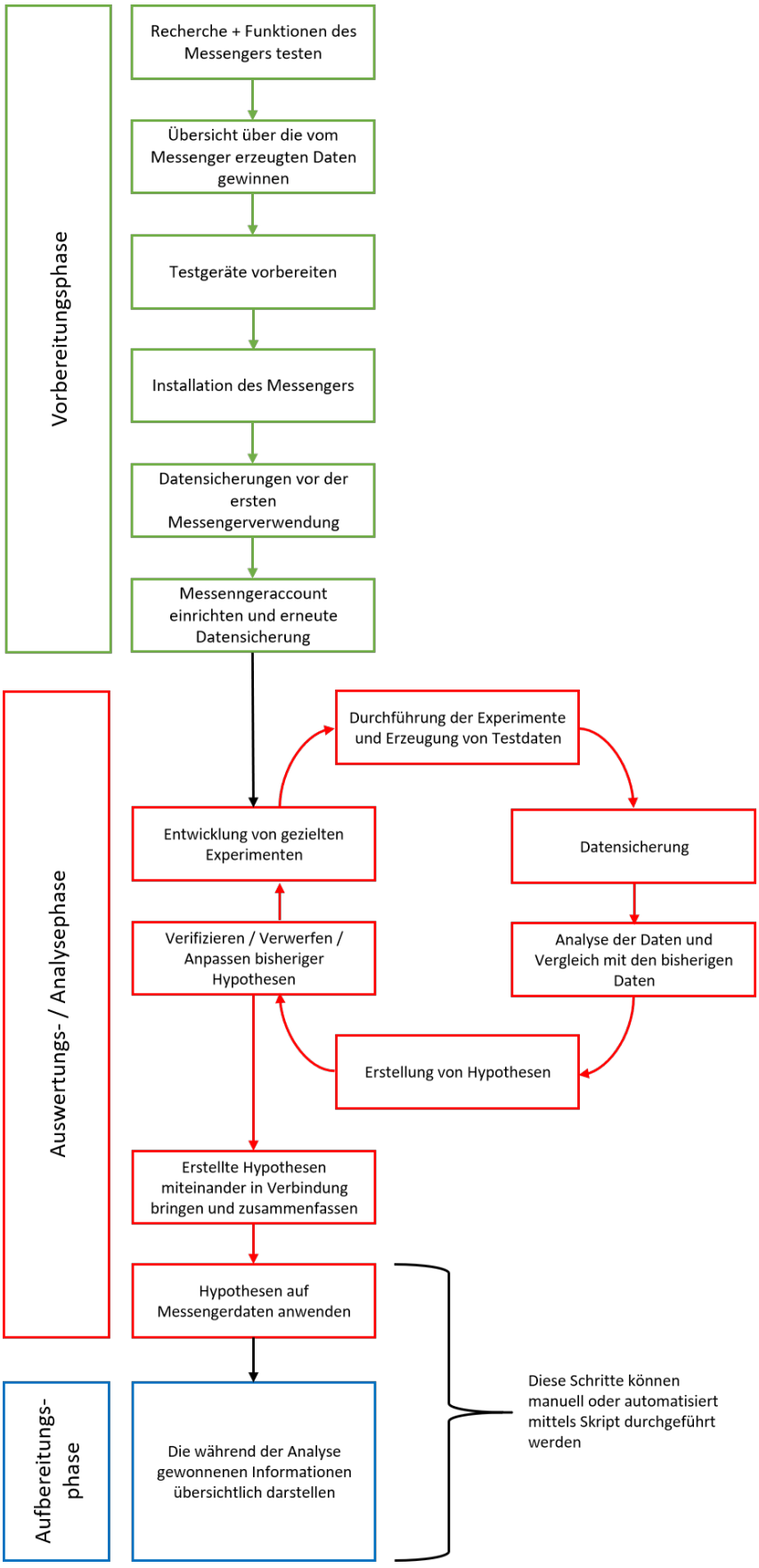


Abbildung 3.1: Verfahrensschritte bei der Untersuchungen von Messengerdiensten

3.1.1 Die Vorbereitungsphase

Im ersten Schritt der Vorbereitungsphase macht sich ein digitaler Forensiker mit dem zu analysierenden Messenger und seinen Funktionen vertraut. Hierfür kann er den Messengerdienst auf einem Testgerät installieren und die Funktionen selbst testen. Auch können die Funktionen und Hintergründe recherchiert werden. Durch diesen Schritt wird eine wichtige Grundlage für die Analysephase geschaffen, da detaillierte Kenntnisse über den Messengerdienst notwendig sind, um diesen im späteren Verlauf analysieren zu können.
Der nächste Schritt dient dazu, sich einen ersten Eindruck über die Elemente zu verschaffen, die der Messenger auf einem Mobiltelefon ablegt. So kann bereits im Vorfeld eine erste Annahme getroffen werden, welche Elemente bei der späteren Analyse von Relevanz sind. Zu diesem Zweck ist es möglich, eine Datensicherung des zuvor verwendeten Testgeräts zu untersuchen. Da die Datenbanken bei einer Auswertung von besonderem Interesse sind, ist es zudem ratsam, ein ER-Diagramm der vorhandenen Datenbanken zu erstellen. Das Diagramm hilft dabei, die Datenbanken zu visualisieren und einen Überblick über die Tabellen und ihre Attribute zu erhalten, ohne sich bereits mit den konkreten Daten auseinandersetzen zu müssen.
Nach Abschluss der Voruntersuchungen wird eine Ausgangssituation für die Analysephase geschaffen. Hierzu zählt das Vorbereiten von Testgeräten, deren Daten in der Auswertungsphase untersucht werden sollen. Eine wesentliche Komponente des Untersuchungsablaufs besteht im Anfertigen von Datensicherungen, daher ist bei der Wahl der Testgeräte darauf zu achten, dass diese gesichert werden können. Zunächst sollten die ausgewählten Mobiltelefone in den Werkszustand zurückgesetzt werden. So wird verhindert, dass sich Daten auf dem Gerät befinden, die die Untersuchungen verfälschen. Bei dem anschließenden Einrichten der Geräte ist darauf zu achten, dass die Funktion „Automatische Updates“ im Appstore des jeweiligen Mobiltelefons deaktiviert wird. So kann manuell gesteuert werden, wann ein Update für eine App installiert wird. Da ein Versionsupdate möglicherweise einen Einfluss auf die Daten hat, die auf dem Mobiltelefon abgelegt werden (siehe Abschnitt 2.2.1), ist es wichtig, alle Untersuchungen mit derselben Messengerversion durchzuführen. Im Anschluss kann der zu untersuchende Messenger installiert werden. Dabei ist darauf zu achten, dass nicht immer alle Versionen eines Messengers in dem jeweiligen offiziellen AppStore zur Verfügung stehen. Soll eine ältere Version untersucht werden, so muss die APK des Messengers über Seiten wie *https://en.uptodown.com* heruntergeladen oder von Testgeräten, auf denen diese Version bereits installiert ist, bezogen werden.
Bereits vor der ersten Nutzung des Messengers sollte eine Datensicherung des Testgeräts erstellt werden. So ist es möglich, zwischen den Standarddaten des Messengers und Daten, die durch eine Nutzeraktivität generiert werden, zu unterscheiden. Zudem ist es sinnvoll, eine weitere Datensicherung durchzuführen, nachdem der Messenger das erste Mal geöffnet wurde, aber noch kein Nutzeraccount eingerichtet ist, da manche Standarddaten erst dann erzeugt werden.
Nach dem Durchführen dieser Datensicherungen wird der Messenger mit einem Nut-

zeraccount verknüpft. Im Anschluss wird erneut eine Datensicherung durchgeführt, um festzustellen, welche Daten der Messenger bei dem Anlegen eines Nutzeraccounts speichert. Dieser Schritt ist der Abschluss der Vorbereitungsphase. Nun folgt die Auswertungs- und Analysephase.

3.1.2 Die Auswertungs- und Analysephase

Basis für die Auswertungs- und Analysephase ist das Reverse Engineering. Es werden gezielt Experimente durchgeführt, die auf den Testgeräten Daten erzeugen. Der digitale Forensiker versucht im Anschluss, eine Verbindung zwischen einer Messengerfunktion und bestimmten Daten herzustellen.
Für eine Untersuchung eines Messengers werden mindestens zwei Testgeräte benötigt. Andernfalls ist eine Auswertung einer Kommunikation nicht möglich. Da die meisten Messenger die Erstellung von Gruppen zur Kommunikation ermöglichen, wird die Verwendung von mindestens drei Testgeräten empfohlen.
Der Ablauf des Reverse Engineerings soll nun in seinen einzelnen Schritten erläutert werden. Zunächst wird festgelegt, welche Funktion des Messengers das Ziel der Untersuchungen ist. Im Anschluss wird ein Experiment entwickelt, bei dem die verschiedenen Einstellungen und Ausgangssituationen dieser Funktion getestet werden. Die Durchführung dieses Experiments erzeugt Daten, die nach einer Datensicherung untersucht werden können. Es ist wichtig, die Ausgangssituation und die Maßnahmen, die durchgeführt werden, zu dokumentieren. Nur so ist im weiteren Verlauf nachvollziehbar, wie bestimmte Daten erzeugt wurden. Wurden bei einer Untersuchung Erkenntnisse gewonnen, so können Hypothesen erstellt werden, die die Untersuchungsergebnisse widerspiegeln.
Handelt es sich bei dem Untersuchungsgegenstand beispielsweise um die Gesprächsrichtung von Textnachrichten, so müssen von einem Testgerät verschiedene Nachrichten gesendet und empfangen werden. Wird dieses Testgerät gesichert, so können die Daten ausgewertet und Unterschiede zwischen den gesendeten und empfangenen Nachrichten dokumentiert werden. Bei dem Messenger WhatsApp ist auf Android-Geräten in der Datenbank *msgstore.db* in der Tabelle *messages* in der Spalte *key_from_me* eine Abweichung bei gesendeten und empfangenen Nachrichten erkennbar. Eine Hypothese, die anhand dieser Unterschiede entwickelt werden kann, lautet: Bei gesendeten Textnachrichten erhält das Attribut *key_from_me* den Wert *1*, bei empfangenen Textnachrichten erhält das Attribut den Wert *0*. Zu dieser Erkenntnis kommt auch C. Anglano bei seiner Auswertung von WhatsApp auf Android-Geräten [21].
Bei komplexen Funktionen kann es gegebenenfalls notwendig sein, dass Experimente in mehreren Schritten durchgeführt werden müssen. In diesem Fall muss nach jedem Schritt eine Datensicherung angefertigt werden, um den Zustand vor einem bestimmten Schritt mit dem Zustand nach einem bestimmten Schritt zu vergleichen. Als Beispiel kann hier ein Experiment für das Verlassen von Konversationen durch den Accountinhaber angeführt werden. Zunächst wird eine Konversation erstellt, bei der der Nutzer

Mitglied ist. Anschließend wird eine Datensicherung durchgeführt, bevor der Nutzer aus der Konversation austritt. Daraufhin findet eine erneute Sicherung des Geräts statt und die Daten vor und nach dem Austritt können gegenübergestellt werden. Somit ist ein digitaler Forensiker in der Lage Hypothesen zu erstellen, durch die Konversationen mit unterschiedlichem Mitgliedsstatus voneinander abgegrenzt werden können.
Nach dem Erstellen von Hypothesen werden diese im nächsten Schritt überprüft. Hierzu können die Daten und Erkenntnisse verwendet werden, die durch das neue Experiment gewonnen wurden. Führt das Anwenden einer Hypothese zu einem Fehler, so muss diese entweder verworfen oder angepasst werden.
Wann eine Auswertung beendet wird, kann selbst entschieden werden. Es ist jedoch ratsam, Experimente zu wiederholen, um eine größere Datenmenge unter den gleichen Bedingungen zu erzeugen. Dies erleichtert die Suche nach Zusammenhängen. Auch bestärkt es die Aussagekraft der Hypothesen, da kein Einzelfall verallgemeinert wird. Zudem werden Abweichungen, die aus einem anderen Grund als dem Untersuchungsgegenstand auftreten, besser erkannt und falsche Hypothesen können vermieden werden. Eine vollständige Analyse eines Messengers ist aufgrund der Komplexität vieler Dienste extrem aufwändig, weshalb es teilweise sinnvoll sein kann, sich auf Basisfunktionen oder relevante Funktionen zu beschränken. Wird eine Untersuchung begrenzt, so kann diese zu jedem Zeitpunkt mit der Analyse weiterer Funktionen fortgesetzt werden, da es sich bei dem Auswertungsverlauf um einen Kreislauf handelt. Es muss jedoch darauf geachtet werden, dass alle Daten, die voneinander abhängig sind, bei der Auswertung gemeinsam betrachtet und nicht zu schnell verallgemeinert werden. Andernfalls wird riskiert, dass aufgrund von Informationsmangel falsche Schlüsse gezogen werden. Sind beispielsweise lediglich die eingespeicherten Kontakte für eine Ermittlung relevant, so kann gefahrlos darauf verzichtet werden, die Konversationen und ihren Inhalt auszuwerten. Die spätere Auswertung des Attributs *msg_delivered_status* (Zustellungs- und Lesestatus einer Konversation) von TamTam wird zeigen, dass seine Werte für Kanäle und Gruppen / Privatchats unterschiedlich zu interpretieren sind. Es besteht also eine Abhängigkeit des Konversationstyps und des Zustellungs- und Lesestatus. Wird das Attribut lediglich für Gruppen ausgewertet und die Ergebnisse im Anschluss auf Kanäle übertragen, so entstehen Fehlinterpretationen. Diese Abhängigkeiten zu erkennen ist Aufgabe des digitalen Forensikers, welcher an der Auswertung der Messengerdaten arbeitet.
Um eine Untersuchung abzuschließen, werden die gesammelten Hypothesen miteinander in Verbindung gebracht. Mehrere Hypothesen, die zu einer Funktion gehören, können so zusammengefasst werden. Die finalen Hypothesen werden in dieser Arbeit als Interpretationsregeln bezeichnet.
Nach dem erfolgreichen Erstellen der Interpretationsregeln, können diese im letzten Schritt des Verfahrens auf Messengerdaten, die von sichergestellten Mobiltelefonen stammen, angewandt werden. Die Ergebnisse der Analyse sind im Anschluss für Ermittlungen verwendbar.
Es ist nützlich, die Auswertung von Messengerdaten mit einem Programm zu automatisieren, dennoch ist auch eine manuelle Auswertung möglich. Durch die Verwendung

eines Programms wird der Zeitaufwand, der für die Analyse großer Datenmengen notwendig ist, reduziert. Zudem kann das Programm an andere digitale Forensiker weitergegeben werden, sodass auch sie Daten des Messengers auswerten können.
Wird ein Programm zur Auswertung von Messengerdaten entwickelt, so ist der erste Schritt, eine Schnittstelle zu den Messengerdaten herzustellen. Dies ist essentiell, da das Auswerteprogramm auf die Messengerdaten zugreifen und diese verarbeiten muss. Ein weiterer wesentlicher Bestandteil des Programms ist die Umsetzung der Interpretationsregeln. Diese müssen in das Programm implementiert und auf die Messengerdaten angewandt werden. Nach der Auswertung der Messengerdaten anhand der Interpretationsregeln müssen die Ergebnisse gespeichert und übersichtlich dargestellt werden. Dies geschieht in der Aufbereitungsphase.

3.1.3 Die Aufbereitungsphase

Nachdem die Messengerdaten im letzten Schritt der Analysephase verarbeitet und ausgewertet wurden, dient die Aufbereitungsphase dazu, die gewonnen Erkenntnisse übersichtlich darzustellen. Da es sich in den meisten Fällen um eine große Datenmenge handelt, ist es empfehlenswert, die Aufbereitung, genau wie die Auswertung, mit Hilfe eines Programms zu realisieren. Es ist sogar ratsam beide Aufgaben in einem Programm zu vereinen, um die Ergebnisse aus der Auswertung ohne Zwischenschritt in der Aufbereitung weiterverarbeiten zu können.
Die Verarbeitung sieht vor, die Ergebnisse der Auswertung auch für Fachfremde zugänglich zu machen. So können die Messengerdaten auch für Ermittlungs- und Strafverfahren verwendet werden. Besonders ansprechend ist es, wenn die Darstellung der Daten der Messengeransicht auf einem Mobiltelefon ähnelt. Dies ist jedoch nicht zwingend notwendig. Auch eine rudimentäre Aufbereitung in Tabellen- oder Listenform ist zweckdienlich, solange darauf geachtet wird, dass die Ausgabe übersichtlich gestaltet wird. Um dies zu gewährleisten, ist es sinnvoll, die Daten in verschiedene Kategorien einzuteilen und mehrere Tabellen oder Listen zu erstellen.
Das Ausgabeformat der Informationen kann frei gewählt werden, sinnvoll ist es jedoch, ein Format zu wählen, dessen Dateien auf jedem Computer mit Hilfe eines Standardprogramms geöffnet und ausgedruckt werden können. Dateiformate, die für eine forensische Aufbereitung in Frage kommen, sind beispielsweise *.html* oder *.pdf*.

4 Forensische Auswertung von TamTam

In diesem Kapitel wird das zuvor beschriebene Verfahren exemplarisch am Messenger TamTam vorgeführt. Seine Nutzer können die Inhalte, die sie mittels TamTam gesendet und empfangen haben, auch ohne Internetverbindung aufrufen. Das bedeutet, dass die hierfür erforderlichen Daten auf dem Mobiltelefon des Nutzers gespeichert sein müssen und von einem digitalen Forensiker ausgewertet werden können. Aus diesem Grund eignet sich TamTam besonders gut, um das in Kapitel 3 vorgestellte Verfahren zu demonstrieren.
Ein weiterer Grund für die Auswahl des Messengers TamTam besteht darin, dass dieser nach aktuellem Kenntnisstand bisher nicht im Rahmen von Veröffentlichungen analysiert wurde.
Zudem wurde die automatische Auswertung und Aufbereitung des Messengers durch Tools wie den UFED Physical Analyzer oder Magnet Axiom zu Beginn der Untersuchungen nicht unterstützt. Somit bestand Bedarf an einer Untersuchung, um die Daten von TamTam in Ermittlungs- und Strafverfahren verwenden zu können.
Laut Releasenote ist Magnet Axiom seit der Softwareversion 3.2.1.14548 in der Lage, TamTam in der Version v2.5.0 auf Android-Geräten auszuwerten und aufzubereiten. Seit der Magnet Axiom Version 3.3.1 wird zudem auch TamTam (v.2.6.1) auf iOS-Geräten unterstützt. Da in dieser Arbeit TamTam v2.6.0 auf Android-Geräten betrachtet wird, wurde überprüft, ob Axiom auch Daten dieser Messengerversion interpretiert. Dies ist der Fall. Allerdings ist die Auswertung durch Magnet Axiom sowohl für Android- als auch für iOS-Geräte unvollständig, so wird beispielsweise die Analyse von ausgetauschten Medien für Android-Geräte nicht unterstützt.[4]
Weiterhin ist es aufgrund der vorgestellten Herausforderungen wünschenswert, Messenger unabhängig von Tools auswerten zu können und ein Verständnis über die abgelegten Spuren im Speicher zu besitzen.
Infolgedessen ist die Unterstützung von TamTam durch Magnet Axiom kein Grund, die Untersuchungen für diese Arbeit einzustellen. Vielmehr ist dies ein Vorteil für die Untersuchungen, da die ausgearbeiteten Ergebnisse überprüft und verifiziert werden können.
Zum besseren Verständnis bei der Auswertung der Messengerdaten wird TamTam zunächst vorgestellt und auf Hintergründe und Funktionen dieses Messengers eingegangen.

[4] Auf die Qualität der Interpretation von TamTam-Daten auf Android-Geräten durch Magnet Axiom wird im späteren Verlauf dieser Arbeit in Kapitel 5.1 noch einmal genauer eingegangen.

4.1 Vorstellung TamTam

TamTam ist ein russischer Messengerdienst, der sowohl auf Mobiltelefonen (Android, iOS) als App, als auch auf Computern als Web- beziehungsweise Desktopversion (Windows, MacOS) verwendet werden kann [27]. TamTam wurde von der Mail.ru Group entwickelt und im Mai 2017 veröffentlicht [28] [29]. Der Messenger ist kostenlos und werbefrei. [30] [31]
Weitere Messengerdienste der Mail.ru Group sind Mail.ru Agent und ICQ. [32]
Dem Jahresrückblick 2018 des Blogs auf der offiziellen TamTam-Webseite [33] zufolge hatte TamTam im Jahr 2018 bis zu 50.000 neue Registrierungen pro Tag und am Ende des Jahres mehr als 6 Millionen Nutzer aus verschiedenen Ländern. Zu diesen Ländern zählen unter anderem Russland, Kasachstan, Weißrussland oder Kirgistan, aber auch im Iran, in Indien und den Vereinigten Arabischen Emiraten wurde TamTam verwendet. Eigenen Aussagen zufolge lag TamTam im Februar 2019 in den Google PlayStore Charts der Vereinigten Arabischen Emirate auf Platz zwei der kostenlosen Apps in der Kategorie Kommunikation und überholte damit WhatsApp [33]. Laut Mail.ru Group ist die Menge der registrierten TamTam-Nutzer im Jahr 2019 weiter angestiegen und liegt nun bei über 7 Millionen Nutzern [32]. Im Google PlayStore wurde die App bis Juni 2019 bisher mehr als 5 Millionen Mal heruntergeladen [30].
TamTam beruht auf der Architektur des Messengers „OK Messages“ [28] [34] [35]. Dieser soll laut Yuri Buyanov, dem Senior iOS Entwickler bei TamTam [36], ein separater Messenger für das soziale Netzwerk OK.ru (auch Odnoklassniki) sein. Aufgrund des Erfolgs von „OK Messages“ wurde auf seiner Basis der Messenger TamTam entwickelt. Dieser kann jedoch im Gegensatz zu seinem Prototypen unabhängig von OK.ru und als eigenständiger Messenger verwendet werden [35]. Ein Zusammenhang zwischen TamTam und „OK Messages“ ist auch an den Messengerdaten auf Android-Geräten ersichtlich, denn der Ordner, in dem unter anderem die TamTam-Datenbanken gespeichert sind, lautet „*ru.ok.messages*“.

4.1.1 Der Entwickler von TamTam: Mail.ru Group

Bei der Mail.ru Group handelt es sich um ein Unternehmen, welches 2005 gegründet wurde und zu den größten Internetkonzern Russlands gehört. [37] [38] [39]
Die Tochterunternehmen und Produkte der Mail.ru Group sind vielseitig und umfassen neben den Messengerdiensten TamTam, ICQ und Mail.ru Agent unter anderem den E-Mail Service Pochta Mail.ru, den Cloud Service Cloud Mail.ru, die Suchmaschine Poisk Mail.ru, den Online-Marktplatz Youla und den Kartendienst MAPS.ME. [40]
Besonders hervorzuheben sind auch die sozialen Netzwerke, die zur Mail.ru Group gehören: VK.com, OK.ru (auch Odnoklassniki) und Moy Mir gehören zu den beliebtesten sozialen Netzwerken Russlands [41]. Laut *statista.com* liegen VK.ru und OK.ru in Russland auf dem zweiten beziehungsweise vierten Platz der meistbenutzten Social Media Platformen im zweiten und dritten Quartal 2018 [42].

Auch bei *alexa.com* liegen VK.com und OK.ru auf Platz drei und zehn der im letzten Monat am häufigsten besuchten Webseiten Russlands. Die *alexa.com* Statistik für Deutschland zeigt, dass die sozialen Netzwerke der Mail.ru Group auch in Deutschland eine Rolle spielen. Hier liegen Vk.com auf Platz 11 und Ok.ru auf Platz 26 der im letzten Monat am häufigsten besuchten Webseiten. [43] [44]

4.1.2 Funktionen von TamTam

TamTam bietet die Möglichkeit, im Rahmen von Privatchats, Chats/Gruppenchats und Kanälen mit anderen Nutzern per Text- und Sprachnachricht zu kommunizieren, Bilder, Videos, Gifs, Sticker und Standorte zu versenden oder zu empfangen und Dateien bis zu 2 GB auszutauschen. Zusätzlich ist es auch möglich, mit anderen Nutzern Telefonate beziehungsweise Videotelefonate zu führen. [45] [30] [31]
Um einen TamTam-Account auf einem Mobiltelefon einzurichten, muss dieser entweder mit der Telefonnummer oder einem Google-Account und der dazugehörigen Gmail-Adresse verknüpft werden. Im Anschluss kann der Messenger ohne eine erneute Anmeldung verwendet werden. Mit einem Internetbrowser ist es zusätzlich möglich, TamTam mit einem Account des sozialen Netzwerks OK.ru (Odnoklassniki) zu verknüpfen und TamTam für eine Kommunikation innerhalb OK.ru zu verwenden. Da es sich hierbei um einen Sonderfall handelt, der bei einer Verwendung der TamTam Messenger App aus dem Google PlayStore und dem Apple AppStore nicht auftritt, wird diese Art der Kommunikation in dieser Arbeit nicht betrachtet. [46]
Laut eigener Aussage auf der offiziellen TamTam-Webseite werden Nachrichten, die mittels TamTam verschickt werden, mit einem eigenen Datenübertragungsprotokoll übertragen und mit Hilfe des TLS-Protokolls verschlüsselt. Gespeichert werden die Daten in einem verteilten Servernetzwerk. [45]
Besteht keine Internetverbindung und somit auch keine Verbindung zu den TamTam-Servern, kann der Nutzer seine gesendeten und empfangenen Inhalte weiterhin im sogenannten Offline-Modus einsehen. Die dafür erforderlichen Informationen und Daten müssen demzufolge auf dem Gerät des Nutzers abgelegt sein. Im Offline-Modus verschickte Nachrichten werden ebenfalls gespeichert und versandt, sobald wieder eine Internetverbindung zur Verfügung steht. [45] [30] [31]
Eine weitere Funktion von TamTam besteht darin, dass sich Nutzer, die sich in der Nähe voneinander befinden, über Bluetooth verbinden und einen ersten Kontakt herstellen können.
Ob es sich bei einer Konversation um einen Privatchat, Chat/Gruppenchat oder Kanal handelt, hängt davon ab, welche dieser Rubriken bei dem Erstellen der Konversation ausgewählt wurde. Die Eigenschaften der verschiedenen Konversationsarten sollen im Folgenden aufgeführt und von einander abgegrenzt werden.

Chats und Gruppenchats

Findet eine Konversation nur zwischen dem Nutzer und einer anderen Person statt, wird diese Unterhaltung als Chat bezeichnet. Eine Konversation zwischen dem Nutzer und mehr als einer Person wird Gruppenchat genannt. Durch das Hinzufügen oder Entfernen von Nutzern innerhalb eines Chats oder Gruppenchats können diese ineinander umgewandelt werden. Sie besitzen aus diesem Grund die gleichen Eigenschaften, weshalb in dieser Arbeit fortan der Begriff „Chat“ auch Gruppenchats mit einschließt.
Der Beschreibung des Messengers im Apple AppStore zufolge ist es möglich, Gruppenchats mit bis zu 20.000 Mitgliedern und 50 Administratoren zu betreiben [31]. Diese Angabe wurde im Rahmen dieser Arbeit nicht überprüft.
Chats können als private Chats in Verbindung mit einem Einladungslink, als öffentliche Chats in Verbindung mit einem öffentlichen Link oder als Chats ohne Link angelegt werden.
Für alle drei Chatarten besteht die Möglichkeit, von anderen Nutzern, die bereits Mitglied in einem Chat sind, zu diesem Chat hinzugefügt zu werden. Bei dem Hinzufügen eines neuen Teilnehmers durch ein anderes Chatmitglied kann ausgewählt werden, ob dem neuen Teilnehmer der gesamte Chatverlauf angezeigt werden soll oder ob er lediglich Nachrichten, die nach seinem Eintritt verschickt werden, einsehen darf.
Für Chats ohne Link ist das Eintreten durch Hinzufügen der einzige Weg, um Zugriff auf den Chat zu erhalten. Bei privaten und öffentlichen Chats besteht durch die chatspezifischen Links noch eine weitere Möglichkeit einem Chat beizutreten. Diese Links sind wesentliche Bestandteile privater und öffentlicher Chats und ihre jeweiligen Eigenschaften begründen, warum die entsprechenden Chats als privat beziehungsweise öffentlich bezeichnet werden.
Einem privaten Chat kann ein Nutzer nur beitreten, wenn er von Nutzern des Chats hinzugefügt wird oder wenn er im Besitz des speziellen Einladungslinks dieses privaten Chats ist. Der Einladungslink für private Chats lautet *https://tt.me/join/XXXX*. *„XXXX“* ist eine 43-stellige Kombination aus Buchstaben, Zahlen und Sonderzeichen und kann wie folgt aussehen: *https://tt.me/join/Td2RLDSWwz5AMffQMjZiC_SIhPwUdQultaL3lf0G-rk*. Hierbei handelt es sich um den Einladungslink eines, zu Testzwecken erzeugten, privaten Chats mit dem Namen „Chatexperiment2“.
Für öffentliche Chats wird ein Link nach dem Schema *tt.me/chat* angelegt. *„chat“* kann vom Nutzer frei gewählt werden, muss allerdings über eine Mindestlänge von vier Zeichen verfügen und darf maximal 64 Zeichen lang sein. Zu den zulässigen Zeichen gehören lateinische Buchstaben, Ziffern, Unter- und Bindestriche. Zudem muss der gewählte Link eines öffentlichen Chats eindeutig sein, das heißt, er darf nicht bereits von einem anderen öffentlichen Chat genutzt werden. Für einen, zu Testzwecken erstellten, öffentlichen Chat mit Namen „Chatexperiment3“ lautet der eindeutige Link beispielsweise: *tt.me/Chatexperiment3_Link*.
Ein öffentlicher Chat kann über die „Chatsuche“ Funktion gefunden werden, das heißt es können auch Nutzer beitreten und auf die Inhalte zugreifen, die nicht von einem der Chatmitglieder eingeladen wurden. Die Kenntnis von *„chat“* ist für eine Suche ausrei-

chend. Dies ist ein wesentlicher Unterschied zu privaten Chats oder Chats ohne Link, da bei diesen Chatarten kontrolliert werden kann, wer auf die Chats und ihre Inhalte zugreifen darf. [45]
Innerhalb eines Chats werden den Chatmitgliedern verschiedene Rollen zugeteilt. Diese sind: Chateigentümer, Chat Super Administrator, Chat Administrator und Chatteilnehmer. In Abhängigkeit der Rolle kann ein Nutzer beispielsweise die Chatinformationen wie Profilbild, Chatname und Beschreibung verändern, Teilnehmer hinzufügen und entfernen oder Nutzer zu (Super-)Administratoren ernennen. Der Chatteilnehmer hat die wenigsten Rechte, der Chateigentümer die meisten. Bei dem Chateigentümer handelt es sich um den Nutzer, der den betreffenden Chat erstellt hat oder dem diese Rolle von dem bisherigen Chateigentümer übertragen wurde. Die zweitmeisten Rechte besitzen Chat Super Administratoren, darauf folgen die Chat Administratoren. Im Anhang befindet sich eine detaillierte Auflistung der Rollen und Rechte, die einem Nutzer in einem Chat zur Verfügung stehen (siehe Kapitel A.1).

Privatchat

Um mit einem seiner Kontakte zu kommunizieren, muss nicht zwingend ein Chat erstellt werden. Alternativ kann zur Kommunikation ein Privatchat verwendet werden, an dem der Accountinhaber und eine weitere Person teilnehmen können. Ein Privatchat wird von TamTam automatisch angelegt, sobald das erste Mal eine direkte Nachricht an einen Kontakt verfasst wird. Der angezeigte Name des Privatchats wird von TamTam generiert und besteht aus dem Namen des Kommunikationspartners. Für einen Privatchat gibt es keine Rollenverteilung und beide Teilnehmer besitzen die gleichen Rechte. Ebenso gibt es keinen Link, der geteilt werden kann, um einem Privatchat beizutreten. Teilnehmer eines solchen Privatchats können sich Nachrichten und andere Inhalte senden, sowie aus dem Chat heraus Anrufe beziehungsweise Videoanrufe tätigen. Zudem können sie den Verlauf oder den ganzen Chat löschen, den Kontakt des Kommunikationspartners löschen oder sperren, sowie einen Standort anfordern. [45]

Kanäle

Neben Chats und Privatchats bietet TamTam seinen Nutzern auch die Möglichkeit, Kanäle zu erstellen oder sie zu abonnieren. Bei Kanälen liegt der Fokus weniger auf der Kommunikation zwischen den Nutzern, sondern mehr auf der Verbreitung von Informationen und Nachrichten. Aus diesem Grund sind in einem Kanal nur ausgewählte Nutzer berechtigt, Inhalte zu veröffentlichen, die meisten Nutzer können sich die Inhalte lediglich ansehen. [45]
Insgesamt sollen, laut dem Jahresrückblick 2018 auf der offiziellen TamTam-Webseite, bis Ende 2018 498.551 Kanäle in TamTam erstellt worden sein [33].
Der Beschreibung des Messengers im Apple AppStore zufolge kann eine unbegrenzte Anzahl an Nutzern an Kanälen teilnehmen [31]. Diese Angabe wurde im Rahmen dieser

Arbeit nicht überprüft.
Kanäle können, wie Chats, an einen Link gebunden sein und privat beziehungsweise öffentlich sein oder ohne Link angelegt werden.
Einem Kanal ohne Link kann, äquivalent zu Chats ohne Link, nur beigetreten werden, wenn man von anderen Nutzern direkt hinzugefügt wird.
Auf einen öffentlichen Kanal kann jeder Nutzer zugreifen und dessen Inhalte ansehen. Um die Inhalte eines privaten Kanals einzusehen, muss ein Nutzer diesen abonniert haben. Ein Abonnement bei einem öffentlichen oder privaten Kanal bewirkt, dass ein Nutzer diesen in seiner Kommunikationsübersicht angezeigt bekommt und über neue Inhalte, die in dem abonnierten Kanal veröffentlicht wurden, informiert wird. [45]
Um einen privaten Kanal zu abonnieren, muss ein Nutzer über einen Einladungslink zu diesem Kanal verfügen. Dieser heißt *https://tt.me/join/XXXX*. Bei *„XXXX“* handelt es sich um einen Platzhalter für eine 43-stellige Folge aus Buchstaben, Zahlen und Sonderzeichen, die von TamTam generiert wird. Der Einladungslink für einen privaten Testkanal namens „Kanalexperiment2“ lautet beispielsweise:
https://tt.me/join/1ToSgQ9ladDi9ore dYBwdRV8euoJv1P1CH8P14YDbso.
Der Link für einen öffentlichen Kanal heißt *tt.me/Kanalname*. *„Kanalname“* kann frei gewählt werden und ermöglicht es den TamTam-Nutzern, den entsprechenden Kanal zu suchen und zu finden. Die Bedingungen für *„Kanalname“* entsprechen dabei den Bedingungen für *„chat“* bei öffentlichen Chats, das heißt, es dürfen 4-64 Zeichen aus lateinischen Buchstaben, Ziffern und Unter- oder Bindestrichen verwendet werden. Zudem wird der Link, bevor er mit einem öffentlichen Kanal verknüpft werden kann, zunächst durch TamTam geprüft. Er muss eindeutig sein und darf nicht bereits von anderen Kanälen oder Chats verwendet werden. Ein öffentlicher Link eines Kanals „Kanalexperiment3“, der zu Testzwecken erstellt wurde, lautet beispielsweise: *tt.me/Kanalexperiment3_Link*.
Die Rollen für Kanalmitglieder ähneln den Rollen, welche Chat- und Gruppenmitgliedern zugeordnet werden. Ein Unterschied besteht jedoch darin, dass alle Nutzer eines Kanals zusätzlich in aktive und passive Nutzer unterteilt werden können. Zu den aktiven Nutzern gehören Kanalmitglieder in der Rolle als Kanaleigentümer, Kanal Super Administrator und Kanal Administrator. Sie sind in einem Kanal aktiv berechtigt, Inhalte zu veröffentlichen und den Kanal zu verändern. Zu den passiven Kanalmitgliedern gehören die Abonnenten eines Kanals. Diese dürfen die Inhalte eines Kanals nur passiv einsehen und keine Veränderungen am Kanal durchführen. Nutzer, die auf einen öffentlichen Kanal zugreifen, ohne ihn abonniert zu haben, besitzen die gleichen Rechte wie Abonnenten eines Kanals, ihnen wird jedoch keine eigene Rolle zugewiesen. Im Rahmen dieser Arbeit wird diese Art von Kanalnutzern als Kanalbesucher bezeichnet.
Die meisten Rechte besitzt der Kanaleigentümer. Er ist das Äquivalent zu einem Chateigentümer und kann alle für einen Kanal verfügbaren Aktionen durchführen. Ein passives Kanalmitglied hat die wenigsten Rechte und darf einen Kanal ausschließlich abonnieren, abbestellen und die Inhalte des Kanals einsehen. Eine detaillierte Übersicht über die Rollen und Rechte von TamTam-Nutzern in Kanälen befindet sich im Anhang (siehe Kapitel A.2).

Eine besondere Art von Kanälen sind verifizierte Kanäle. Diese gehören bekannten Marken oder Personen des öffentlichen Lebens und verfügen über einen blauen Haken neben ihrem Kanalnamen. Mit Hilfe des blauen Hakens können andere TamTam-Nutzer entscheiden, ob es sich bei dem Kanaleigentümer wirklich um die prominente Person beziehungsweise Marke handelt, die sie behauptet zu sein, oder ob es sich lediglich um ein Person handelt, die sich als solche ausgibt. [45]
Um verifiziert zu werden, muss ein Kanal von TamTam im Rahmen eines Verifikatiosprozesses ("Channel Verification Procedure") überprüft werden. Die Schritte für diesen Prozess sind festgelegt und können auf der Webseite von TamTam eingesehen werden. Unter anderem muss der Nutzer seine Identität nachweisen. Zudem muss er auf mindestens zwei weiteren Plattformen mit dem selben Usernamen verifiziert sein. Hierfür kommen die meisten sozialen Netzwerke wie Twitter, Instagram oder YouTube in Frage, da eine Nutzerverifizierung für soziale Netzwerke mittlerweile zum Standard gehört [47] [48] [49]. [45]

4.1.3 Vergleich von TamTam und Telegram

TamTam ähnelt in seinem Aussehen und seinen Funktionen sehr stark dem Messenger Telegram. Dieser wurde ebenfalls in Russland entwickelt und kann seit 2013 für iOS- und Android-Geräte heruntergeladen werden. Mittlerweile hat Telegram seinen Sitz in Dubai. [50]
Telegram bietet seinen Nutzern, genau wie TamTam, die Möglichkeit, mit Kontakten in Privatchats und Gruppenchats zu kommunizieren sowie Kanäle zu erstellen und diese zu abonnieren.
Des Weiteren werden die Gruppen und Kanäle bei beiden Messengern über Links organisiert. Mit ihrer Hilfe kann ein Nutzer in Gruppen und Kanäle eintreten oder nach diesen suchen, sofern sie öffentlich sind. Diese Links weisen sehr starke Gemeinsamkeiten auf und weichen nur geringfügig voneinander ab: Bei Telegram lautet der Link für einen öffentlichen Kanal beispielsweise *t.me/Kanalname*, bei TamTam lautet er *tt.me/Kanalname*. *„Kanalname"* kann bei beiden Messengern frei gewählt werden. Für einen privaten Kanal lautet ein Link bei Telegram *https://t.me/joinchat/XXXX* und bei TamTam *https://tt.me/join/XXXX*, wobei es sich bei *„XXXX"* um eine von den Messengern generierte Folge von Zahlen, Buchstaben und unter Umständen Unter- beziehungsweise Bindestrichen handelt. Diese Folge ist bei Telegram und TamTam unterschiedlich lang. Testkanäle bei Telegram verfügten über eine 22-stellige Kombination, wohingegen Testkanäle bei TamTam eine 43-stellige Folge von Zeichen aufwiesen. Bei öffentlichen Gruppen lautet ein Link bei Telegram t.me/Link, bei TamTam heißt er *tt.me/chat*, „Link" und „chat" können dabei frei gewählt werden. Für private Gruppen bei Telegram wird ein Einladungslink mit *https://t.me/joinchat/XXXX* generiert. Der Einladungslink für private TamTam (Gruppen-)Chats lautet *https:/tt.me/join/XXXX*. *„XXXX"* ist, wie bei den privaten Kanälen, eine Kombination aus Buchstaben, Zahlen und unter Umständen Unter- beziehungsweise Bindestrichen. Die Kombination war bei zu Test-

zwecken erstellten Kanälen bei Telegram 22-stellig, bei TamTam 43-stellig.
Telegram entstand drei Jahre vor TamTam und wurde laut dem am 22.03.2018 veröffentlichten Blogeintrag von Gründer Pavel Durov im Februar und März von 200 Millionen Nutzern aktiv verwendet [51]. Aufgrund der starken Ähnlichkeiten zwischen den Messengern kann davon ausgegangen werden, dass sich die Mail.ru Group bei der Entwicklung von TamTam an Telegram orientiert hat. Besonders die Tatsache, dass TamTam ebenfalls die Verwendung von Kanälen unterstützt, ist ein starkes Argument für diese These, da es sich bei den Kanälen von Telegram um eine Funktion handelt, für die der Messenger bekannt ist. Laut M. Zelensky [28] nutzt TamTam seine Ähnlichkeit zu Telegram mit Slogans wie „eine komplette Kopie von Telegram“ [28] oder „ein weiterer Messenger mit Kanälen“ [28] sogar für Werbezwecke. Die seit April 2018 drohende Sperrung von Telegram in Russland nutzte TamTam zudem, um sich als Alternative für Telegram zu präsentieren. Dafür wurde Werbung in Zeitungen, aber auch gezielt in Telegramkanälen geschaltet. Um den Nutzern einen weiteren Anreiz für einen Wechsel zu TamTam zu bieten, wurden in diesem Zeitraum zudem mehrere bekannte Telegramkanäle von TamTam kopiert. [28]
Für einen Nutzer sind lediglich kleinere Unterschiede zwischen den Messengern ersichtlich. Beispielsweise erlaubt Telegram die Verwendung von geheimen Chats, welche in Kapitel 2.1 genauer beschrieben wurden, diese können bei TamTam nicht genutzt werden. TamTam verfügt stattdessen über eine Funktion, mit deren Hilfe sich TamTam-Nutzer, die nah beieinander sind, finden und anschreiben können. Zudem können sie sich so gegenseitig zu ihren Kontakten hinzufügen. Für diese Funktion wird Bluetooth verwendet. Eine ähnliche Funktion gibt es bei Telegram aktuell nicht. Weitere, kleinere Unterschiede zwischen den Messengern sind beispielsweise die maximale Größe für Dateien, die Nutzer austauschen oder die maximale Anzahl an Benutzern, die an Gruppen teilnehmen können. Bei Telegram kann eine Datei bis zu 1,5 GB groß sein und bis zu 200.000 Nutzer können Mitglied in einer Gruppe sein [50], bei TamTam ist eine Dateigröße von bis zu 2 GB und eine Anzahl von bis zu 20.000 Gruppenmitgliedern möglich [31].

4.2 Auswertung und Aufbereitung am Beispiel des Messengers TamTam

Im Folgenden wird das in Kapitel 3 vorgestellte Verfahren am Messenger TamTam demonstriert. Der Fokus der Untersuchung liegt auf den Datenbanken des Messengers, da sich diese, wie im Vorfeld beschrieben, sehr gut als Informationsquelle eignen.
Bei TamTam handelt es sich um einen Messenger mit vielen Funktionen und Einstellungsmöglichkeiten, welcher auf Android- und iOS-Geräten verwendet werden kann.
Aufgrund der Komplexität von TamTam können im Rahmen dieser Arbeit nicht alle Funktionen des Messengers auf beiden Betriebssystemen untersucht werden. Um einen Eindruck von dem in Kapitel 3 vorgestellten Verfahren zu erhalten, wird die Untersuchung

von TamTam auf Android-Geräte beschränkt.
Es wird Version 2.6.0 des Messengers untersucht, welche im Google PlayStore zuletzt am 16.05.2019 aktualisiert wurde. Hierbei handelt sich um die aktuellste Version, die zu Beginn der Untersuchungen für diese Arbeit zur Verfügung stand. Es findet eine detaillierte, jedoch keine vollständige Auswertung der Messengerdaten statt. Eine ausführlichere Betrachtung wäre dem Rahmen dieser Arbeit nicht angemessen. Fehlende Funktionen können mit Hilfe des vorgestellten Verfahrens jederzeit untersucht und hinzugefügt werden.

4.2.1 Vorbereitungsphase

Die einzelnen Schritte der in Kapitel 3 vorgestellten Vorbereitungsphase wurden befolgt. Da diese sehr genau beschrieben wurden, wird im weiteren Verlauf lediglich auf die Ergebnisse der Voruntersuchungen sowie Besonderheiten bei der Einrichtung der Testgeräte eingegangen.

Ergebnisse der Voruntersuchung

Die Untersuchung der Testdaten aus den Voruntersuchungen zeigt, dass auf Android-Geräten für den Messenger TamTam vier Datenbanken angelegt werden. Diese lauten:

- *androidx.workdb*
- *cache.db*
- *google_app_measurement_local.db*
- *mytracker_6203910701518717586 6.db*

Die genannten Datenbanken werden im Ordner *databases* im Pfad *userdata (ExtX)/Root/data/ru.ok.messages/* abgespeichert.
Eine Übersicht der Datenbanken, die bei Benutzung von TamTam auf einem Testgerät mit Android generiert wurden, ist der Abbildung 4.1 zu entnehmen. Die Übersicht wurde nach einer physikalischen Sicherung des Testgeräts mit Hilfe des UFED Physical Analyzers erstellt.

Name	Reihenanzahl	↓ Größe (Bytes)	Pfad
cache.db	564	1175552	userdata (ExtX)/Root/data/ru.ok.messages/databases/cache.db
androidx.work.workdb	20	77824	userdata (ExtX)/Root/data/ru.ok.messages/databases/androidx.work.workdb
mytracker_6203910701518717586 6.db	3	28672	userdata (ExtX)/Root/data/ru.ok.messages/databases/mytracker_6203910701518717586 6.db
google_app_measurement_local.db	1	16384	userdata (ExtX)/Root/data/ru.ok.messages/databases/google_app_measurement_local.db

Abbildung 4.1: Übersicht über die von TamTam generierten Datenbanken auf Android-Geräten

Die Übersicht lässt vermuten, dass es sich bei der Datenbank *cache.db* um die Hauptdatenbank von TamTam auf Android-Geräten handelt. Sie ist die größte Datenbankdatei mit der höchsten Reihenanzahl. Das erstellte ER-Diagramm für diese Datenbank (siehe Abbildung 4.2) zeigt zudem, dass die *cache.db* Tabellen und Attribute enthält, die mit hoher Wahrscheinlichkeit Informationen über den Nutzer und seine Kommunikation speichern. Eine oberflächliche Betrachtung der Daten bestätigt diese Annahme.

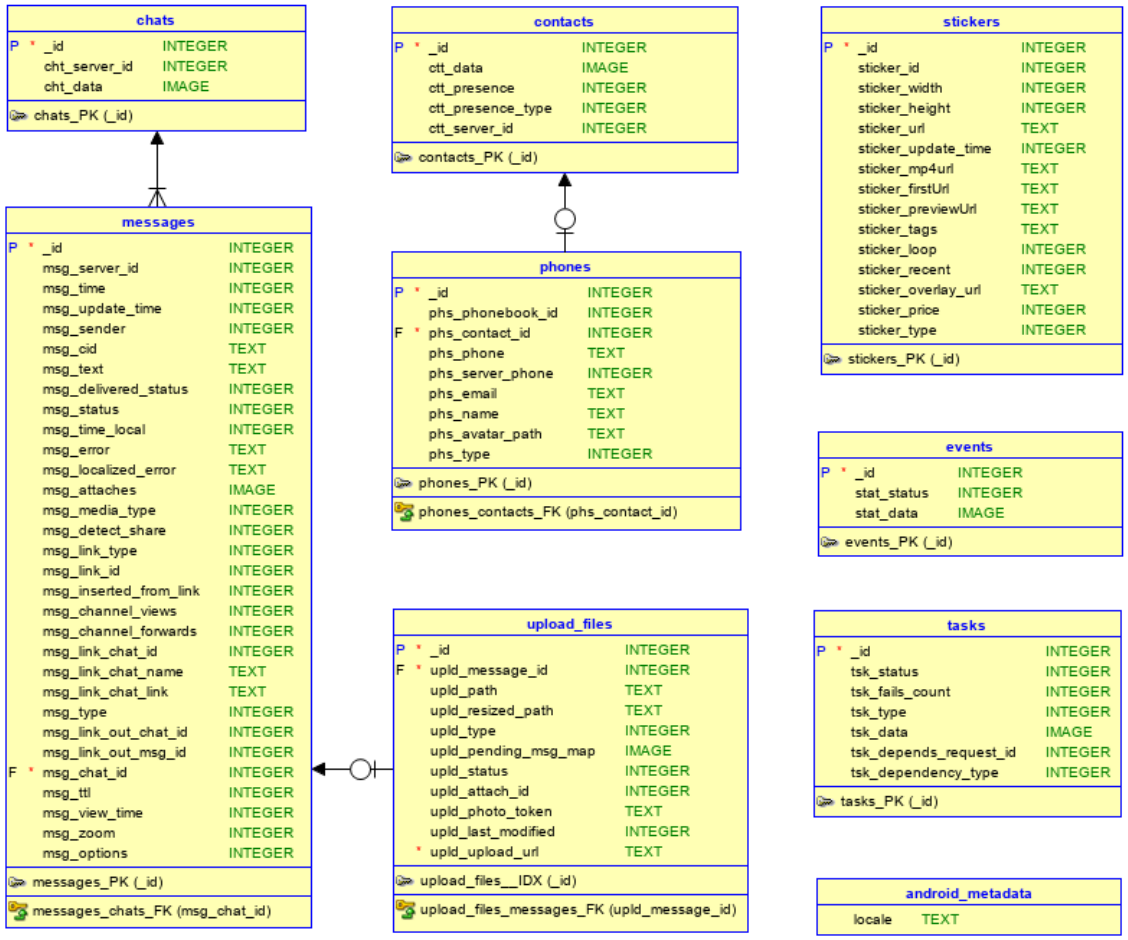


Abbildung 4.2: ER-Diagramm der Datenbank *cache.db* (erstellt mit Oracle SQL Developer)

Besonderheiten bei der Vorbereitung der Testgeräte für die Analyse

Damit die verwendeten Testgeräte leichter voneinander unterschieden werden können, wurden ihnen Aliasse zugeordnet. Ihr Alias wird im weiteren Verlauf auch als Name für den Inhaber des TamTam-Accounts auf dem entsprechenden Mobiltelefon verwendet. Eine Übersicht der Testgeräte sowie des jeweiligen Betriebssystems und des zugeordneten Alias lässt sich Tabelle 4.1 entnehmen. Details zu den TamTam-Versionen, die auf den einzelnen Geräten installiert sind, sind in Tabelle 4.2 festgehalten.

Hersteller	Modell	Betriebssystem	Alias
Apple	iPhone 5S (A1457)	iOS 12.3.1, 16GB	Birthe Wie
Apple	iPhone 6 (A1586)	iOS 12.3.1, 64GB	Dana Wie
Samsung	Galaxy Note 4 Edge (SM-N915FY)	Android 6.0.1	Finja Wie
Samsung	Galaxy S5 (SM-G900F)	Android 6.0.1	Hannah Wie
Samsung	Galaxy J3 2016 (SM-J320F/DS)	Android 5.1.1	Jule Wie
Samsung	Galaxy S9+ (SM-G965F)	Android 9	Lea Wie

Tabelle 4.1: Übersicht der verwendeten Mobiltelefone

Alias	Version	Quelle	Aktualisiert	Heruntergeladen
Birthe Wie	2.6.2	Apple AppStore	02.05.2019	19.06.2019
Dana Wie	2.6.2	Apple AppStore	02.05.2019	19.06.2019
Finja Wie	2.6.0	Google PlayStore	05/2019	19.06.2019
Hannah Wie	2.6.0	Google PlayStore	05/2019	19.06.2019
Jule Wie	2.7.0	Google PlayStore	09/2019	25.09.2019
Lea Wie	2.6.0	Google PlayStore	05/2019	09.07.2019

Tabelle 4.2: Übersicht der TamTam-Versionen

Eine Besonderheit bei der Einrichtung der Android-Geräte besteht darin, dass das Mobiltelefon „Hannah Wie“ gerootet wurde. Verwendet wurde dazu die *„CF-Auto-Root-klte-kltexx-smg900f.zip“* (*https://autoroot.chainfire.eu/*, verfügbar am 17.06.2019) zusammen mit Odin3 (v3.13.1). Durch das Rooten können Dateien gezielt aus dem Speicher extrahiert werden. Somit muss keine vollständige Datensicherung durchgeführt werden, um die Veränderungen an den Messengerdatenbanken zu untersuchen.

4.2.2 Auswertungs- und Analysephase

Auch die Auswertung der Messengerdaten erfolgte wie bei dem zuvor beschriebenen Verfahren. Zunächst wird die Herangehensweise bei der Erstellung von Experimenten zur Auswertung von TamTam dargelegt, bevor die durchgeführten Experimente sowie ihre Ergebnisse erläutert werden.
Aufgrund der Vielzahl der Experimente wird dabei darauf verzichtet, diese Schritt für Schritt zu beschreiben. Es wird lediglich auf ihr Konzept eingegangen. Als Ergebnisse werden in diesem Kontext die Hypothesen bezeichnet, von denen nach Abschluss der Untersuchungen festgestellt wurde, dass sie korrekt sind. Diese Hypothesen können als Interpretationsregeln auf neue Messengerdaten angewandt werden, um sie zu analysieren.
Um dennoch aufzuzeigen, wie die Durchführung eines Experiments und die anschließende Auswertung im Detail ablaufen, wird dies am Beispiel der Zeitstempel der *cht_data*-BLOBs der Tabelle *chats* in der Datenbank *cache.db* von Android-Geräten demonstriert. Da BLOBs ein wesentlicher Bestandteil der TamTam-Datenbanken auf Android-Geräten

sind, ist dieses Experiment für eine Demonstration besonders gut geeignet. Es vermittelt dem Leser einen Eindruck davon, welche Unterexperimente bei einer Auswertung eines BLOBs erforderlich sein können und zeigt, dass eine Auswertung dieser häufig eine Herausforderung darstellt.

Herangehensweise bei der Erstellung der Experimente

Es wurden im Rahmen der Auswertung der TamTam-Messengerdaten zwei Herangehensweisen verwendet, um Experimente zu erstellen.
Die erste Herangehensweise basiert darauf, einer Funktion Daten und Eigenschaften aus der Datenbank zuzuordnen. Bei der zweiten Herangehensweise wird den Daten und Eigenschaften aus der Datenbank eine Funktion zugewiesen.
Sollen einer Funktion Daten einer Datenbank zugeordnet werden, so wird zunächst eine Messengerfunktion oder Einstellung ausgewählt, die untersucht werden soll. Als nächstes wird festgestellt, welche Unterfunktionen und Einstellungsmöglichkeiten bei einer Funktion zur Verfügung stehen. Im Anschluss werden für diese Funktion geeignete Experimente entwickelt und durchgeführt. Es ist darauf zu achten, dass alle bekannten Möglichkeiten beziehungsweise Einstellungen im Rahmen eines Experiments abgebildet werden. So kann vermieden werden, dass Informationsmangel zu falschen Hypothesen führt.
Soll im Rahmen einer Untersuchung beispielsweise festgestellt werden, welche Kontaktdaten ein Messenger speichert, so müssen verschiedene Kontakte miteinander bekannt gemacht werden. Diese Nutzer sollten sowohl über gleiche, als auch über abweichende Eigenschaften verfügen. Im Anschluss wird eine Datensicherung durchgeführt. So kann durch einen Vergleich der Einfluss von Kontaktname, Profilbild, Kontaktbeschreibung oder Telefonnummer auf die erzeugten Daten untersucht werden.
Bei der zweiten Herangehensweise, bei der den Daten und Eigenschaften einer Datenbank Funktionen zugeordnet werden, wird die Struktur der Datenbank betrachtet, um Experimente zu entwickeln. Anhand von Tabellen- und Attributnamen sowie ihrem Datentypen können Vermutungen darüber angestellt werden, welche Informationen diese speichern. Gibt es beispielsweise, wie bei TamTam auf Android-Geräten, eine Tabelle mit dem Namen *chats*, so lässt sich vermuten, dass diese Informationen über die Konversationen eines Nutzers speichert.
Da diese Tabelle lediglich die Attribute *_id*, *cht_server_id* und *cht_data* enthält, ist es wahrscheinlich, dass es sich bei den ersten zwei Attributen um eindeutige Identifikatoren einer Konversation handelt und in *cht_data* die Hauptinformationen über eine Konversation gespeichert sein müssen.
Die Tatsache, dass es sich bei dem Attribut *cht_data* um einen BLOB handelt, bestärkt diese Vermutung. BLOBs enthalten häufig nicht nur eine, sondern eine Vielzahl an Informationen.
Besteht anhand des Namens eine Vermutung darüber, was ein bestimmtes Attribut in einer ausgewählten Tabelle speichert, so ist es möglich, gezielte Experimente zu erstel-

len, die auf dieses Attribut abzielen. Der Attributname *msg_sender* der Tabelle *messages* legt zum Beispiel nahe, dass dort gespeichert ist, welcher Nutzer eine Nachricht verschickt hat. Wird nun ein Experiment durchgeführt, bei dem mehrere Nutzer Textnachrichten verschicken und enthält das Attribut *msg_sender* je nach Nachrichtensender einen spezifischen Eintrag, so kann die Bedeutung des Attributs verifiziert werden.
Ein Vorteil dieser Herangehensweise ist es, dass Funktionen oder Einstellungsmöglichkeiten des Messengers entdeckt werden können, die durch die Verwendung des Messengers nicht gefunden wurden. So ist es möglich, Experimente zu Funktionen zu ergänzen und Wissen über Daten zu erhalten, deren Bedeutung durch die erste Herangehensweise unbekannt geblieben wäre.
Es ist ratsam, bei der Auswertung beide Herangehensweisen zu kombinieren, um möglichst effektive Experimente durchzuführen und keine Funktionen oder Einstellungsmöglichkeiten zu übersehen.
Werden Daten einer Messengerfunktion in einem BLOB abgespeichert, so ist ein wesentlich höherer Untersuchungsaufwand als bei Attributen des Datentyps Integer oder String notwendig. Während Daten dieser Datentypen direkt zugänglich sind und lediglich eine Information enthalten, sammeln BLOBs häufig mehrere Informationen und können als Attributansammlung betrachtet werden. Diesen Attributen fehlt jedoch eine individuelle Bezeichnung. Sie besitzen lediglich einen gemeinsamen Attributnamen, weshalb dieser keine oder nur sehr begrenzte Informationen über den Inhalt des BLOBs preisgeben kann.
Die Herausforderung bei BLOBs besteht deshalb darin, dass dem digitalen Forensiker unbekannt ist, welche Informationen in einem BLOB enthalten sind, wo diese innerhalb des BLOBs abgespeichert werden und wie ihre Werte zu interpretieren sind.
Demzufolge müssen zunächst die Informationen eines BLOBs identifiziert werden, bevor eine Interpretation stattfinden kann.
Hilfreich bei der Auswertung von BLOBs ist es, sich zunächst zu überlegen, welche Daten ein BLOB enthalten könnte. Da der Attributname keine oder sehr begrenzte Informationen über den Inhalt des BLOBs liefert, muss auf das Wissen über die Funktionen und Einstellungsmöglichkeiten bei dem zu untersuchenden Messenger zurückgegriffen werden.
Anhand der Struktur der Tabelle *contacts* kann festgestellt werden, dass die Hauptinformationen über Kontakte auf Android-Geräten mit Hilfe des BLOBs *ctt_data* gespeichert werden. Wird auf das Wissen über TamTam zurückgegriffen, so ist es wahrscheinlich, dass in dem BLOB unter anderem der Name des Kontakts, seine Beschreibung sowie ein Hinweis auf sein Profilbild gespeichert werden. Nach diesen Informationen kann in dem BLOB gezielt gesucht werden.

Experimente zur Auswertung des Messengers TamTam

Die durchgeführten Experimente werden in diesem Unterkapitel konzeptionell beschrieben. Die erstellten Experimente dienen dazu, Informationen aus drei Kategorien zu gewinnen: Nutzerinformationen (einschließlich Informationen über den Accountinhaber), Informationen über die Konversationen, die geführt wurden und Informationen über die ausgetauschten Nachrichten.

Nutzerinformationen

Die Experimente für die Nutzerinformationen beliefen sich darauf, verschiedene TamTam-Nutzer mit dem Nutzer „Hannah Wie“ bekannt zu machen und in die Kontakte dieses Nutzers hinzuzufügen. Dabei wurde darauf geachtet, dass die Nutzer über verschiedene Eigenschaften verfügen, um diese im Rahmen der Analyse untersuchen und voneinander abgrenzen zu können. Die untersuchten Nutzer zeigt Tabelle B.1 im Anhang.
Nach einer Bekanntmachung mit „Hannah Wie“ wurde das Mobiltelefon, auf dem sich dieser Nutzeraccount befindet, gesichert und die Datenbanken ausgewertet. Eine Sicherung und Auswertung wurde wiederholt durchgeführt, nachdem einige Profile aktualisiert und verändert wurden. So konnten mögliche Zeitstempel für eine Profilaktualisierung sowie die Auswirkungen der neuen Informationen auf die Datenbank ermittelt werden. Des Weiteren wurde mit den verschiedenen Kontakten mehrfach auf TamTam zugegriffen und im Anschluss eine Datensicherung angefertigt. Ziel dieses Schritts war es, herauszufinden, wo in der Datenbank die Eigenschaft „zuletzt online“ abgespeichert wird.
Zudem wurde zusätzlich das Mobiltelefon von „Finja Wie“ gesichert und auch dort wurden die Kontakte überprüft. So war es möglich, die Erkenntnisse, die bei der Untersuchung der Kontakte von „Hannah Wie“ gewonnen wurden, zu verifizieren.

Informationen über die Konversationen

Um Konversationen zu untersuchen, wurde eine Vielzahl an Privatchats, Chats und Kanälen angelegt. Diese unterschieden sich unter anderem in der Konversationsart (öffentlich/privat), dem Namen, der Beschreibung, dem Ersteller der Konversation sowie der Rolle der einzelnen Teilnehmer.
Alle Eigenschaften der Konversationen wurden tabellarisch dokumentiert (siehe Kapitel B.1.2 im Anhang).
Bei dem Erstellen der Konversationen wurde darauf geachtet, dass der Nutzer, dessen Mobiltelefon gesichert wurde (hier: „Hannah Wie“), in jeder Rolle bei jeder Konversationsart (öffentlich/privat) jedes Konversationstyps (Privatchat, Chat, Kanal) mindestens einmal auftrat. Zudem wurden auch den anderen Nutzern variierende Rollen zugewiesen. So konnte die Auswirkung der Rolle des Eigentümers und der Einfluss der Rolle

anderer Teilnehmer auf die Konversationsdaten untersucht werden. Zusätzlich wurden in einigen der Konversationen Testnachrichten verschickt. So konnte geprüft werden, welchen Einfluss Nutzeraktionen auf eine Konversation haben.
Damit die Auswirkungen der Rechte, die ein Nutzer besitzen kann, untersucht werden konnten, wurde der Chat „Chatexperiment18“ erstellt. Den Nutzern dieser Konversation wurden verschiedene Rechte zugewiesen und entzogen. Als nächstes fand eine Datensicherung statt. Im Anschluss wurden die gewonnenen Kenntnisse auf die Nutzerrechte in Kanälen angewandt und überprüft, ob die Erkenntnisse dort ebenfalls zutreffen.
Um die Erstellung der Konversationen übersichtlich zu gestalten, wurden den Konversationen Namen und Beschreibungen nach einem festgelegten Schema zugewiesen. Name und Beschreibung sind durchnummeriert und enthalten eine Information darüber, um welchen Konversationstypen (Chat, Kanal) es sich bei der Unterhaltung handelt. So lautet der Name für den ersten Chat beziehungsweise Kanal *Chatexperiment1* / *Kanalexperiment1*. Die Beschreibung für den Kanal mit dem Namen *Kanalexperiment1* heißt *Kanalexperiment1_Beschreibung*.
Aufgrund der gewählten Namen konnten die Konversationen in der Datenbank gut identifiziert werden. Zudem war es mit ihrer Hilfe möglich, die Eigenschaften aus der angelegten Tabelle mit den Daten in der Datenbank in Verbindung zu bringen und gezielt zu analysieren.

Informationen über die ausgetauschten Nachrichten sowie Anrufe

Nachdem für die vorherigen Experimente verschiedene Konversationen erstellt wurden, konnten diese nun verwendet werden, um ausgetauschte Nachrichten zu untersuchen. Zu diesem Zweck wurden in den erstellten Privatchats, Chats und Kanälen zunächst Textnachrichten verschickt. Für den Inhalt der Nachrichten wurde ein Schema gewählt, welches es ermöglicht, die Nachrichten bei der anschließenden Auswertung leichter zu identifizieren und zu vergleichen. Das Schema lautet für Android-Geräte wie folgt:

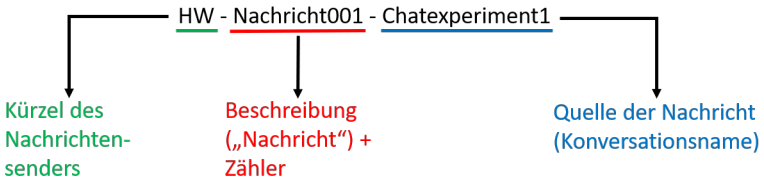


Abbildung 4.3: Schema der verschickten Textnachrichten

Ein weiterer Vorteil dieses Schemas besteht darin, dass eine Dokumentation der durchgeführten Aktionen stattfindet. Wird die Konversation auf dem Mobiltelefon betrachtet, so kann auch im Nachhinein eingesehen werden, welche Nachrichten von welchem Nutzer verschickt wurden. Dies ist insbesondere bei Nachrichten in Kanälen ohne eine Dokumentation häufig nicht möglich.

Mit Hilfe dieser Experimente ließen sich die Grundeigenschaften von Textnachrichten auswerten: der Nachrichtensender, der Zeitpunkt des Verschickens, der Nachrichteninhalt, die Konversation, die einer Nachricht zugeordnet werden kann sowie der Typ der Konversation (Privatchat, Chat, Kanal).
Im Anschluss an die Untersuchung einfacher Textnachrichten wurden Sonderfunktionen für diese untersucht. Im Rahmen weiterer Experimente wurden folgende Aspekte untersucht:

- die Bearbeitung von Nachrichten
- die Verwendung der Antwort-Funktion bei Nachrichten
- der Lese-Status von Nachrichten (gelesen beziehungsweise ungelesen)
- das Verschicken von Offline-Nachrichten (Nachrichten, die ohne Internetverbindung verschickt wurden)

Nach Abschluss der Untersuchungen für klassische Textnachrichten wurden verschickte Mediennachrichten untersucht. Diesbezüglich wurde ein Chat Namens „Medientest“ erstellt. In diesem wurden zwischen zwei Teilnehmern („Hannah Wie“ und „Finja Wie“) Mediennachrichten ausgetauscht. Es wurde darauf geachtet, dass alle bekannten Möglichkeiten zum Austausch eines Medientyps in ein Experiment umgesetzt wurden. Zudem wurden einige der Austauschmöglichkeiten mehrfach wiederholt, um leichter Gemeinsamkeiten und Unterschiede zu identifizieren. Auf den Wechsel eines Medientyps wurde zunächst mit einer klassischen Nachricht hingewiesen. Dies dient zur Dokumentation und erleichtert die Orientierung in der Datenbank, wenn die Mediennachrichten zusammen mit den klassischen Textnachrichten in einer Tabelle gespeichert werden. Dies ist bei TamTam auf Android-Geräten der Fall.
Eine Übersicht der durchgeführten Experimente ist den Tabellen in Kapitel B.1.3 des Anhangs zu entnehmen. Tabelle B.9 zeigt eine Übersicht der untersuchten Mediennachrichten. Jede aufgeführte Mediennachricht wurde mindestens einmal verschickt. Gesichert wurden die Mobiltelefone von „Hannah Wie“. „Ausgehend“ steht aus diesem Grund für den Nachrichtensender „Hannah Wie“, „eingehende“ Nachrichten können dem Nutzer „Finja Wie“ zugeordnet werden.
Weitere Nachrichten, die untersucht wurden, waren Systemnachrichten. TamTam dokumentiert einige Nutzeraktionen mit Systemnachrichten, die in einer Konversation grau hinterlegt sind und mittig in einer Konversation stehen (siehe Abbildung B.1 im Anhang). Zunächst wurde festgestellt, bei welchen Nutzeraktionen TamTam diese Systemnachrichten generiert. Im Anschluss wurden sowohl für Chats als auch für Privatchats und Kanäle Systemnachrichten generiert und in der Datenbank ausgewertet. Zu den untersuchten Systemnachrichten gehören zum Beispiel „Nutzer A hat einen Chat erstellt“, „Kanal erstellt“, „Nutzer A hat Nutzer B dem Chat hinzugefügt“ und „Nutzer A hat das Foto des Chats geändert“. Welche weiteren Systemnachrichten ermittelt werden konnten, kann den Ergebnissen entnommen werden.
Eine weitere Möglichkeit, wie Nutzer miteinander kommunizieren können, sind Anrufe. Diese können bei TamTam lediglich zwischen zwei Personen stattfinden und werden

den Privatchats der Nutzer zugeordnet. Anrufe können eingehend und ausgehend sein, sowie entgangen, durchgestellt oder abgewiesen. Zudem sind sowohl Anrufe, als auch Videoanrufe möglich. Jede Anrufmöglichkeit wurde im Rahmen der Experimente für Anrufe mindestens einmal getestet und ausgewertet. Eine Übersicht der durchgeführten Experimente zeigt Tabelle B.10 im Anhang. Als „ausgehend“ werden Anrufe bezeichnet, die der Nutzer „Hannah Wie“ getätigt hat. „Eingehende“ Anrufe sind „Finja Wie“ zuzuordnen.
Zu Dokumentationszwecken wurden Details zu einem Anruf mit Hilfe einer Textnachricht festgehalten. Ein Vorteil dieser Dokumentation besteht darin, dass die Anrufe leichter analysiert und verglichen werden können. Zudem erleichtert dies die Orientierung in der Datenbank.

Exemplarische Durchführung und Auswertung eines Experiments

Um einen Eindruck davon zu erhalten, wie die Auswertung eines BLOBs ablaufen kann, wird diese im Folgenden am Beispiel des *cht_data*-BLOBs der Tabelle *chats* in der Datenbank *cache.db* auf Android-Geräten demonstriert. Insbesondere wird dabei auf die enthaltenen Zeitstempel eingegangen.
Um den Aufbau der *cht_data*-BLOBs zu ermitteln, wurden im ersten Schritt verschiedene BLOBs dieses Attributs miteinander verglichen. Diese BLOBs wurden im Rahmen der Experimente für Konversationen von TamTam generiert (siehe Kapitel B.1.2 im Anhang). Aufgrund dieser Vergleiche konnten verschiedene Strukturen innerhalb der BLOBs identifiziert werden, die ausgewählte Informationen speichern. Zu diesen Informationen gehören unter anderem der Name sowie die Beschreibung einer Konversation. Eine weitere Struktur, die identifiziert werden konnte, besteht aus sechs Bytes und endet mit dem Byte *0x2D*. Für das fünfte Byte der untersuchten Struktur konnten die Werte *0xBA* und *0xBB* beobachtet werden. Es wurde festgestellt, dass die vollständige Struktur, in der Little Endian Notation betrachtet, eine aufsteigende Zahlenfolge bildet. Angesichts dieser Eigenschaften wurde die Hypothese aufgestellt, dass es sich bei dieser Struktur um einen Zeitstempel handelt, welcher die vergangenen Zeiteinheiten seit einem festgelegten Zeitpunkt abspeichert. Besonders die Tatsache, dass keine Veränderungen bei dem letzten Byte der untersuchten Struktur beobachtet wurden, spricht für diese Hypothese. Zeitstempel ähneln sich häufig, wenn sie in einer nicht allzu großen Zeitspanne erzeugt wurden. Wird beispielsweise der Unix-Zeitstempel betrachtet, so beginnen im Zeitraum vom 11.06.2019 - 21:24:48 Uhr bis zum 23.12.2019 - 00:45:03 Uhr alle Unix-Zeitstempel in Little Endian Schreibweise mit *0x5D*.
Zunächst wurde analysiert, in welchen Zeiteinheiten der Zeitstempel die vergangene Zeit abspeichert und zu welchem Zeitpunkt die Zählung beginnt. Die anfängliche Überprüfung, ob es sich um ein bekanntes Zeitstempelformat, wie den Unix-Zeitstempel, handelt, schlug fehlt. Wurden Bytefolgen dieser Struktur als Unix-Zeitstempel interpretiert, so handelte es sich um unrealistische Zeitangaben, die sich in der Zukunft befanden (zum Beispiel 21.05.2448 - 08:19:04 Uhr).

Aus diesem Grund wurde im Anschluss versucht, die Parameter des Zeitstempelformats zu berechnen. Zu diesem Zweck wurde aus jedem *cht_data*-BLOB ein vermeintlicher Zeitstempel ausgewählt, welcher sich jeweils an einer vergleichbaren Stelle innerhalb der *cht_data*-BLOBs befindet. Da die *cht_data*-BLOBs keine absoluten Offsets verwenden, kann die genaue Position des verwendeten Zeitstempels nicht angegeben werden. Vor dem ausgewählten Zeitstempel steht der Hexadezimalwert *0x30*. Das erste Byte nach dem Zeitstempel lautet in allen beobachteten Fällen *0x3A*. Nach dem Wert *0x3A* folgt der Name der Konversation sowie seine Länge.
Die Auswahl des Zeitstempels begründet sich damit, dass sich dieser bei einer wiederholten Datensicherung der Konversationen nicht änderte. Zudem fand die erste Datensicherung einer Konversation kurz nach ihrer Erstellung statt. Das bedeutet, dass der Zeitstempel dauerhaft eine Zeit speichert, die dem Erstellungszeitpunkt entspricht beziehungsweise zeitnah an diesem liegen muss.
Um im Anschluss das Zeitstempelformat zu berechnen, wurde eine Liste der ausgewählten Zeitstempel erstellt. Jedem dieser Zeitstempel wurde die Erstellungszeit seiner jeweiligen Konversation zugeordnet. Diese wurde im Rahmen der Experimente für Konversationen in den Tabellen B.3 und B.6 im Anhang dokumentiert. Besonders zu beachten ist, dass der ausgewählte Zeitstempel bei öffentlichen Chats nicht in dem dazugehörigen BLOB enthalten ist. Diese Chats wurden bei der Untersuchung der Zeitstempel nicht betrachtet.
Die Versuche, anhand der Zuordnungen von Realzeiten und Zeitstempeln die verwendete Zeiteinheit sowie die Ausgangszeit zu ermitteln, schlug fehl, da die betrachteten Zeiten über keinen linearen Zusammenhang verfügen. Diesen Umstand verdeutlicht Tabelle 4.3.

Realzeit B - Realzeit A (in Sekunden)	**Zeitstempel B - Zeitstempel A (Differenz in Dezimal)**	**Differenz Zeitstempel / Differenz Realzeit**
600	2443777	4072,96
900	12052365	13391,52
1200	13243120	11035,93
1680	6499540	3868,77
1920	15985441	8325,75
2220	17424503	7848,88
2640	18755142	7104,22
66240	524877555	7923,88
83220	2816068847	33838,85

Tabelle 4.3: *cht_data*-BLOBs der Tabelle *chats* in der Datenbank *cache.db*

Die erste Spalte dieser Tabelle zeigt die Differenz zweier Erstellungsdaten in Sekunden. Die zweite Spalte enthält die Differenz der dazugehörigen Zeitstempel aus dem BLOB *cht_data*. Um diese Differenz zu berechnen, wurden die Zeitstempel zunächst als Little Endian Werte interpretiert und in Dezimalzahlen umgewandelt. Die letzte Spalte der

Tabelle zeigt den Quotienten aus der Differenz der Zeitstempel und der Differenz der Erstellungsdaten. Würde eine Linearität vorliegen, so müsste die Division einen Quotienten ergeben, der für alle Berechnungen gleich ist. Dies ist bei dieser Untersuchung nicht der Fall. Die Werte schwanken stark, weshalb auch etwaige Ungenauigkeiten bei der Dokumentation des Erstellungszeitpunkts einer Konversation nicht für die fehlende Linearität verantwortlich sein können. Die fehlende Linearität ließ drei Hypothesen zu:

1. Die Ausgangshypothese, dass es sich bei der untersuchten Struktur um einen Zeitstempel handelt, trifft nicht zu
2. Die Zuordnung der Zeitstempel zur entsprechenden Realzeit ist falsch
3. Der Zeitstempel codiert die zugeordnete Realzeit anders als zunächst vermutet

Aufgrund der zuvor angeführten Begründungen, weshalb hinter der Struktur ein Zeitstempel vermutet wird sowie die Zuordnung der Zeitstempel zur entsprechenden Realzeit, wurde zunächst die dritte Hypothese betrachtet.
Hierzu fand eine detaillierte Untersuchung der Hexadezimalfolgen der Zeitstempel statt. Im Rahmen dieser Untersuchungen war auffällig, dass die Zeitstempel einen eingeschränkten Wertebereich verwenden. Alle Bytewerte eines Zeitstempels, mit Ausnahme des letzten Bytes, enthalten in Hexadezimal einen Wert größer oder gleich *0x80*. Bei dem letzten Byte eines Zeitstempels wurde bisher in allen Fällen der Wert *0x2D* beobachtet, welcher kleiner als *0x80* ist. Werden die einzelnen Bytes eines Zeitstempels in Binärdarstellung betrachtet, zeigt sich zudem, dass das erste Bit eines Bytes, welches nicht das Ende eines Zeitstempels darstellt, immer den Wert *1* besitzt. Das letzte Byte eines Zeitstempels verfügt als Most Significant Bit über den Wert *0*.
Die festgestellten Auffälligkeiten bestärkten die Richtigkeit der dritten Hypothese. Infolgedessen wurden die Hexadezimalwerte weiter untersucht. Schlussendlich konnte ein Verfahren ermittelt werden, welches die Hexadezimalzahlen der Zeitstempel codiert.
Findet eine Decodierung eines vermeintlichen Zeitstempels für die Erstellungszeit statt, so erhält man einen Wert, welcher als Unix-Zeitstempel interpretiert, genau diese speichert. Dieser Umstand bestätigt folgende Hypothesen:

1. Bei der untersuchten Struktur handelt es sich um Zeitstempel
2. Die Zeitstempel sind mit einem speziellen Verfahren codiert und entsprechen nach ihrer Decodierung einem Unix-Zeitstempel
3. Der Zeitstempel im BLOB *cht_data* an der Position zwischen dem Wert *0x30* dem Wert *0x3A* speichert die Erstellungszeit einer Konversation
4. Die Erstellungszeit wird nicht in *cht_data*-BLOBs von öffentlichen Chats gespeichert

Zusätzliche Auswertungen zeigten, dass dieses Verfahren nicht nur für Zeitstempel, sondern darüber hinaus auch für andere Werte im BLOB *cht_data* genutzt wird. Hierzu zählt unter anderem die Länge des Namens und der Beschreibung einer Konversation. Zudem wird das Verfahren auch in weiteren BLOBs der Datenbank *cache.db* angewandt. Wie genau das Verfahren funktioniert und welche Vorteile es bietet, wird im folgenden Abschnitt erläutert.

Ergebnisse der Auswertung des Messengers TamTam auf Android-Geräten

Die Datenbank *cache.db* ist die Hauptinformationsquelle für die Analyse von TamTam auf Android-Geräten. Eine Übersicht über ihre Tabellen und Attribute wurde bereits in Abbildung 4.2 gegeben. Die *cache.db* enthält an mehren Stellen in der Datenbank BLOBs, welche ausführlich analysiert werden mussten. Bevor auf die Ergebnisse der zuvor beschriebenen Experimente eingegangen werden kann, wird das zuvor erwähnte Verfahren erläutert, welches in BLOBs von TamTam verwendet wird, um Werte zu codieren. Dieses Verfahren konnte aufgrund der Experimente für Zeitstempel der *cht_data*-BLOBs der Tabelle *chats* in der *cache.db* identifiziert werden.

Das Tex-Verfahren

Das Verfahren, welches in BLOBs der Datenbank *cache.db* Zeitstempel und weitere Werte codiert, wird im weiteren Verlauf dieser Arbeit als Tex-Verfahren bezeichnet. Dieser Name ist selbstgewählt und setzt sich aus den Worten „TamTam“ und „Hex(adezimal)“ zusammen.
Bei dem Tex-Verfahren wird das Ende einer Bytefolge und somit eines Werts durch ein Byte im BLOB markiert, welches in seiner Binärdarstellung eine *0* als Most Significant Bit besitzt. Alle Bytes, die vor diesem Endmarker liegen und zu dem Wert gehören, müssen über eine *1* als Most Significant Bit verfügen. Somit ist der größtmögliche Wert eines Bytes mit einer *0* als Most Significant Bit der Wert *0x7F* in Hexadezimal beziehungsweise *127* in Dezimal. Soll ein größerer Wert codiert werden, so muss dieser aus mindestens zwei Byte bestehen, von denen nur das letzte Byte eine *0* als Most Significant Bit besitzen darf. Diese Codierung bewirkt, dass das Most Significant Bit nicht mehr frei zur Zahlendarstellung verwendet werden kann und lediglich als Marker dient. Somit ist es für eine Decodierung ausreichend, lediglich die letzten sieben Bit eines Bytes zu betrachten. Es ist jedoch nicht möglich, einen Zahlenwert ohne weiteres aus diesen Bits abzulesen. Dies begründet sich durch die Verwendung von Überläufen, um Zahlenwerte zu codieren, die größer als *127* sind. Hierfür zählt das nachstehende Byte, wie häufig das vorherige Byte die *127* überschritten hat. Eine Veranschaulichung der Systematik bietet Abbildung 4.4. Der Vorteil des Tex-Verfahrens besteht darin, dass die Länge eines Werts nicht im Voraus festgelegt werden muss. So belegt ein Wert nicht mehr Speicherplatz als wirklich benötigt.

Um Bytefolgen, bei denen dieses Verfahren verwendet wurde, in klassische Dezimalwerte umzurechnen, kann die folgende mathematische Formel verwendet werden:

$$\sum_{i=1}^{n-1} (d_i \mod 128) \cdot 128^{i-1}$$

Mit $b_1 b_2 ... b_n$ als Bytedarstellung der zu decodierenden Hexadezimalzahl und $d_1 d_2 ... d_n$ als Folge der in Dezimalwerte umgewandelten Bytes. Es gilt demnach: $d_i := (b_i)_{dez}$

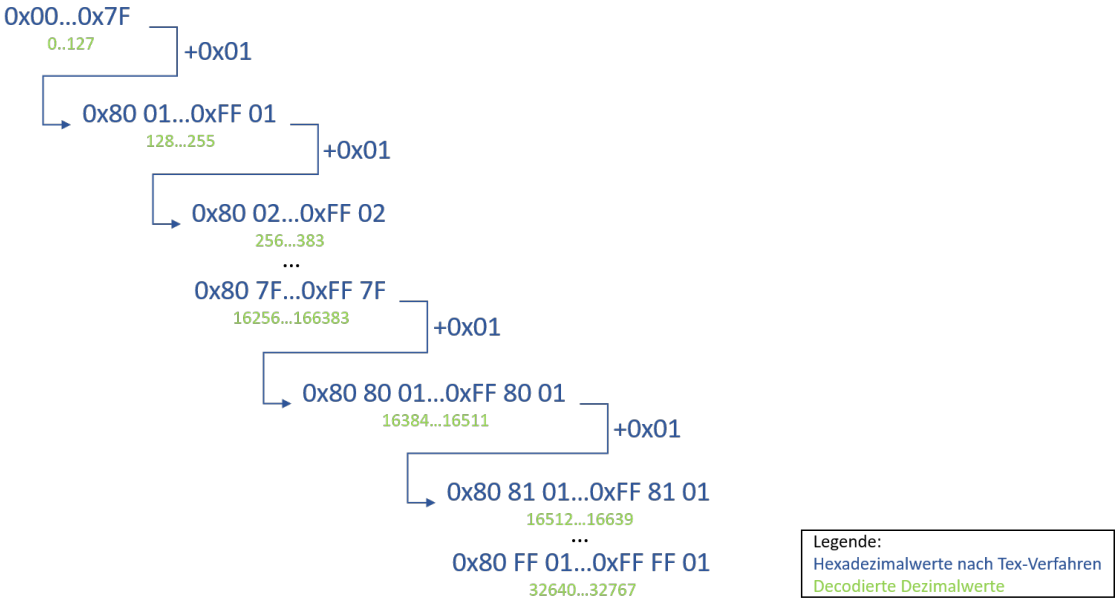


Abbildung 4.4: Veranschaulichung des Tex-Verfahrens

Die URLs in den BLOBs der Datenbank *cache.db*

In den BLOBs der Datenbank *cache.db* treten häufig URLs auf. Diese stehen im Zusammenhang mit Profilbildern von TamTam-Nutzern und Konversationen sowie Mediendateien (Sprachnachrichten, Bilder, Videos und Sticker). Im Vorfeld der Präsentation der Untersuchungsergebnisse wird dargelegt, wie durch diese URLs ein Zugriff auf die genannten Dateien möglich ist.
Zunächst muss die URL aus einem BLOB extrahiert und als ASCII-Text interpretiert werden. Gehört die URL zu einer Sprachnachricht, so kann diese in einen Webbrowser eingegeben und direkt angehört oder heruntergeladen werden. Das Vorschaubild eines gesendeten oder empfangenen Videos sowie ausgetauschte Sticker sind auf diese Weise ebenfalls einsehbar.
Auf die Profilbilder von TamTam-Nutzern und Konversationen sowie ausgetauschte Bilder ist ein Zugriff über die entsprechende URL nicht ohne Weiteres möglich. Hierfür muss die URL angepasst werden. Dies wurde mit Hilfe der iOS-Datenbank *manifest.sqlite* festgestellt. Die in dieser Datenbank enthalten URLs können direkt ein-

gesehen werden und besitzen im Gegensatz zu den URLs der *cache.db* als Endung „*&fn=sqr_192*“ oder „*&fn=w_1440*“. Wird diese Endung auch an die URLs der *cache.db* angefügt, so erlauben auch diese einen Zugriff auf die Bilddateien.
Die jeweilige Endung „*&fn=sqr_192*“ oder „*&fn=w_1440*“ beeinflusst, in welchem Format ein Bild angezeigt wird. „*&fn=sqr_192*“ zeigt ein quadratisches Bild mit 192x192 Pixeln, „*&fn=w_1440*“ ein rechteckiges Bild mit einer Höhe von 747 Pixeln und einer variablen Breite.
Da „*&fn=w_1440*“ zu einem Bild in einem größeren Format führt, ist diese Einstellung empfehlenswert. Zudem zeigt ein quadratisches Bild bei ursprünglich rechteckigen Bildern nicht den vollständigen Bildinhalt.
Es besteht die Möglichkeit, dass in einem BLOB zwei verschiedene URLs zu einer Bilddatei gespeichert sind. Dies ist beispielsweise bei Profilbildern von Konversationen gegeben. Die erste URL speichert ein quadratisches Profilbild, die zweite URL ein rechteckiges. Mit „*&fn=w_1440*“ wird in beiden Fällen der vollständige Bildinhalt angezeigt. Da bei „*&fn=w_1440*“ die Breite des Bildes variabel ist, wird wie bei „*&fn=sqr_192*“ bei URL1 ein quadratisches Bild angezeigt. Das Bild mit der Endung „*&fn=w_1440*“ ist jedoch wesentlich größer.
Um die Anpassung der URLs zu verdeutlichen, werden die verschiedenen URLs für das Profilbild der Konversation „Chat-Titel“ (Datenbank *cache.db*, Tabelle *chats*, BLOB *cht_data*) sowie das dazugehörige Profilbild exemplarisch im Anhang aufgeführt (siehe Kapitel B.2.1).

Nutzerinformationen

Aufgrund der durchgeführten Experimente konnte ermittelt werden, dass für die Nutzerinformationen besonders die Tabellen *contacts* und *phones* aus der Datenbank *cache.db* von Interesse sind.
Die Tabelle *contacts* speichert Informationen über den Accountinhaber und seine Kontakte. In der Tabelle *phones* werden Informationen zu Kontakten, die TamTam mit einer Telefonnummer verknüpft haben, festgehalten. Besitzt ein Kontakt keine Telefonnummer, werden in dieser Tabelle keine Informationen über ihn gespeichert.
In der Tabelle *contacts* wird jedem Kontakt eine ID (*_id*) zugeordnet. Bei dieser ID handelt es sich um den Primärschlüssel der Tabelle. Mit seiner Hilfe kann eine Verknüpfung zur Tabelle *phones* hergestellt werden, da diese die ID als Fremdschlüssel in der Spalte *phs_contact_id* enthält. Der Nutzer, der die ID mit dem Wert *1* besitzt, ist der Accountinhaber.
Tabelle 4.4 zeigt eine Übersicht über die Attribute der Tabellen *contacts* und *phones*, deren Bedeutung bekannt ist.

Tabelle	Attribut	Bedeutung
contacts	_id	lokale NutzerID, die nur innerhalb der untersuchten *cache.de* eindeutig ist
contacts	ctt_data	BLOB mit Nutzerinformationen
contacts	ctt_presence	Zeit, zu der der Nutzer das letzte Mal online war (als Unix-Zeitstempel)
contacts	ctt_server_id	globale NutzerID, ein Nutzer kann mit dieser ID sowohl in Datenbanken von Android-Geräten, als auch in Datenbanken von iOS-Geräten identifiziert werden
phones	phs_phone	Telefonnummer des Nutzers mit Ländervorwahl wie +49
phones	phs_phone	Telefonnummer des Nutzers ohne + als Vorzeichen
phones	phs_name	Kontaktname
phones	phs_contact_id	KontaktID, die mit der lokalen NutzerID (_id) aus der Tabelle *contacts* übereinstimmt

Tabelle 4.4: Bedeutung der Attribute der Tabellen *contacts* und *phones* der *cache.db*

Im weiteren Verlauf dieser Arbeit wird die globale NutzerID mit NutzerID bezeichnet. Bezieht sich eine Information auf die lokale NutzerID wird dies explizit erwähnt.
Die größte Menge an Nutzerinformationen speichert der BLOB *ctt_data* aus der Tabelle *contacts*.
Er enthält eine Vielzahl an Markern, die als Indikator für eine bestimmte Information stehen. Nicht jeder Marker taucht zwangsläufig in jedem BLOB auf. Der BLOB enthält nur Marker, zu denen eine Information bekannt ist. Hat ein Nutzer seinen Account beispielsweise nicht mit einer Telefonnummer verknüpft, so fehlt der Indikator für die Telefonnummer.
Einem Marker können weitere Marker untergeordnet sein. Diese werden als Submarker bezeichnet. Enthält auch dieser Marker eigene Marker, so heißen diese Subsubmarker. Die Werte aller Marker einer Hierarchiestufe sind aufsteigend und im Tex-Verfahren codiert.
Viele der Werte, die den Markern zugeordnet sind, sind ebenfalls nach diesem Verfahren codiert. Wird das Tex-Verfahren nicht verwendet, so folgt nach dem Marker häufig eine Angabe über die Länge seines Werts, damit auch hier das Ende bestimmt werden kann. Da Marker sowie Beginn und Ende ihres zugeordneten Werts leicht erkannt werden können, ist es möglich, alle vorhandenen Marker und ihre Werte zu ermitteln, auch wenn unbekannt ist, welche Informationen die Werte speichern.
Tabelle 4.5 zeigt eine Übersicht der bekannten Marker und ihrer Bedeutung. Für eine Auswertung sind die identifizierten Marker zusammen mit ihren untergeordneten Markern und ihrer Bedeutung im Anhang detailliert aufgeführt (siehe Kapitel B.2.2).

Marker	Bedeutung
0x08	NutzerID
0x40	Profilaktualisierung
0x48	Telefonnummer
0x50	Kontaktstatus (wurde ein Nutzer blockiert?)
0x72	Nutzername
0x8201	Beschreibung des Nutzers
0x8A01	selbstgewählter Link des Nutzers
0xA201 und 0xAA01	URLs zum Profilbild des Nutzers

Tabelle 4.5: Übersicht der Marker des BLOBs *ctt_data* der Tabelle *contacts* der Datenbank *cache.db*

Eine exemplarische Analyse eines *ctt_data* BLOBs anhand der aufgeführten Marker zeigt Abbildung 4.5.

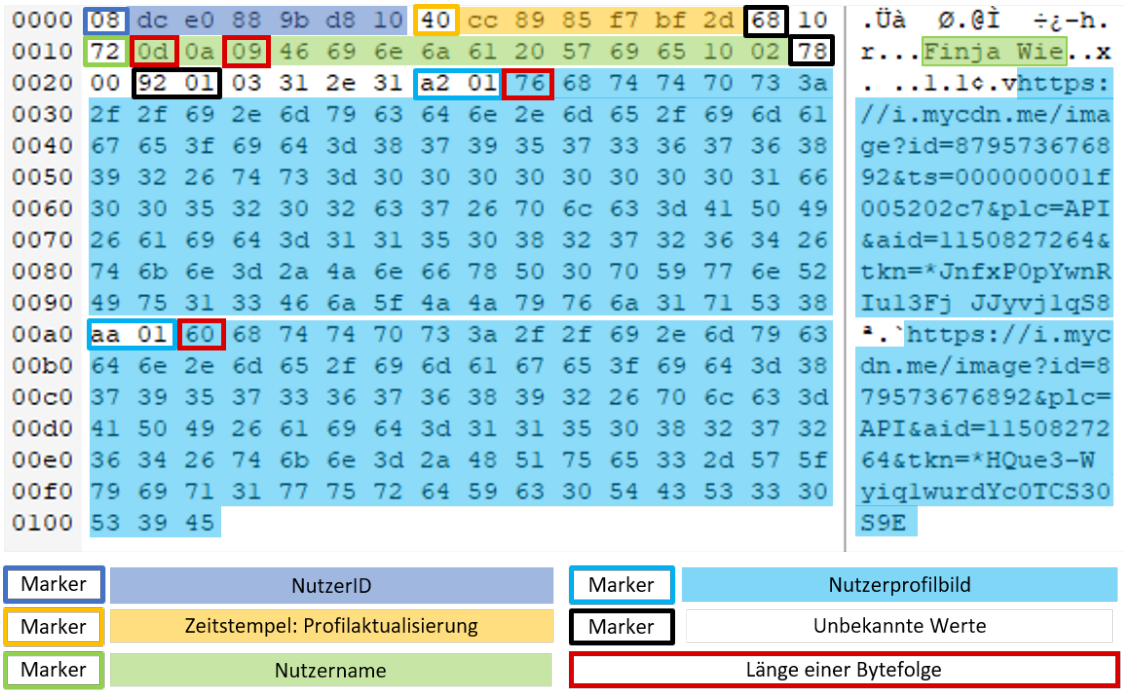


Abbildung 4.5: Exemplarische Auswertung eines *ctt_data* BLOBs aus der Tabelle *contacts*

Nach Abschluss der Auswertung kann dem BLOB entnommen werden, dass der Nutzer der ID *0xdce0889bd810* (ID codiert im Tex-Verfahren, als Dezimalzahl lautet die ID *573434900572*) „Finja Wie“ heißt. Der Zeitstempel für die letzte Profilaktualisierung, codiert im Tex-Verfahren, lautet *0xcc8985f7bf2d*. In einen Dezimalwert umgerechnet und als Unix-Zeitstempel interpretiert, steht dieser Wert für den 17.07.2019 09:41:44 Uhr. Weiterhin besitzt der Nutzer „Finja Wie“ ein selbstgewähltes Profilbild, zu dem zwei URLs bekannt sind.

Wird die zweite URL des BLOBs angepasst (*https://i.mycdn.me/image?id=879573676892&plc=API&aid=1150827264&tkn=*HQue3-W_yiq1wurdYc0TCS30S9E&fn=w_1440*), lässt sich das Profilbild von „Finja Wie“ ermitteln (siehe Abbildung 4.6)

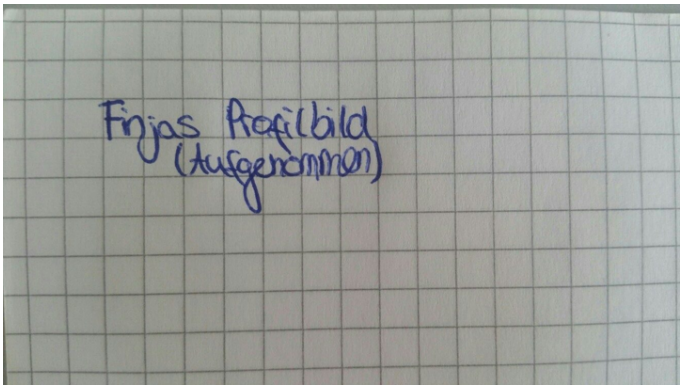


Abbildung 4.6: Profilbild des Nutzers „Finja Wie“

Informationen über die Konversationen

Anhand der durchgeführten Experimente ließ sich feststellen, dass TamTam die Informationen über Konversationen in der Tabelle *chats* der *cache.db* abspeichert. Diese Tabelle besitzt drei Attribute: *_id*, *cht_server_id* und *cht_data*. Auf die Bedeutung der Attribute wird in Tabelle 4.6 eingegangen.

Tabelle	Attribut	Bedeutung
chats	_id	lokale ID der Konversation
chats	cht_server_id	globale ID der Konversation
chats	cht_data	BLOB mit Informationen über die Konversation

Tabelle 4.6: Bedeutung der Attribute der Tabelle *chats* der *cache.db*

Die Struktur des BLOBs *cht_data* stimmt mit der Struktur des BLOBs *ctt_data* überein. Das heißt, es werden ebenfalls aufsteigende Marker verwendet, die mittels Tex-Verfahren codiert sind und als Indikator für bestimmte Informationen dienen. Auch bei diesem BLOB können Marker und die dazugehörigen Werte gut identifiziert werden. Die Informationen, die einem Marker zugeordnet werden können, stimmten nicht mit denen aus dem *ctt_data* BLOB überein. Bei beiden BLOBs wird beispielsweise der Marker *0x08* verwendet, dieser steht jedoch bei *ctt_data* für die NutzerID und bei *cht_data* für die KonversationsID. Im Folgenden wird eine Übersicht über die identifizierten Marker und ihre Bedeutung gegeben (siehe Tabelle 4.7). Eine ausführliche Auflistung der Marker sowie ihrer Bedeutung befindet sich im Anhang (siehe Kapitel B.2.3).

Marker	**Bedeutung**
0x08	KonversationsID
0x10	Konversationstyp: Privatchat, Chat, Kanal
0x20	Status: Konversation verlassen / Nutzer blockiert
0x20	EigentümerID
0x2A	Bekannte Teilnehmer
0x30	Erstellungsdatum der Konversation
0x3A	Name der Konversation
0x50, 0x9001	ID der letzten (0x50) und ersten (0x9001) bekannten Nachricht
0xC001	Konversationsart: öffentlich/privat
0xCA01	Link der Konversation
0xE801	Teilnehmerzahl
0xF201	Beschreibung
0xF801	Administratoren der Konversation
0xDA02	Informationen zu den Administratoren, darunter auch Informationen über die Rolle und die Rechte eines aufgeführten Nutzers
0xE202, 0xEA02	URLs des Profilbilds

Tabelle 4.7: Marker des BLOBs *ctt_data* der Tabelle *contacts* der Datenbank *cache.db*

Um die Rollen und Rechte eines Nutzers zu dokumentieren, wird ein einzelnes Byte im BLOB *cht_data* verwendet. Dieses Byte ist dem Marker *0xDA02* zugeordnet. Im Folgenden soll auf die Auswertung des Bytes für Rollen und Rechte genauer eingegangen werden.
Der Wert *0x7F* steht für den Konversationseigentümer und der Wert *0x3F* für einen Super Administrator. Für einen einfachen Administrator gibt es mehrere Bytewerte. Im Gegensatz zu dem Konversationseigentümer und den Super Administratoren können ihm einzelne Rechte zugewiesen werden, die sich auf den Bytewert auswirken. Um die einzelnen Rechte eines Administrators abzulesen, muss das Byte auf Bitebene betrachtet werden. Wie die einzelnen Bits zu interpretieren sind, zeigen Abbildungen 4.7 und 4.8.

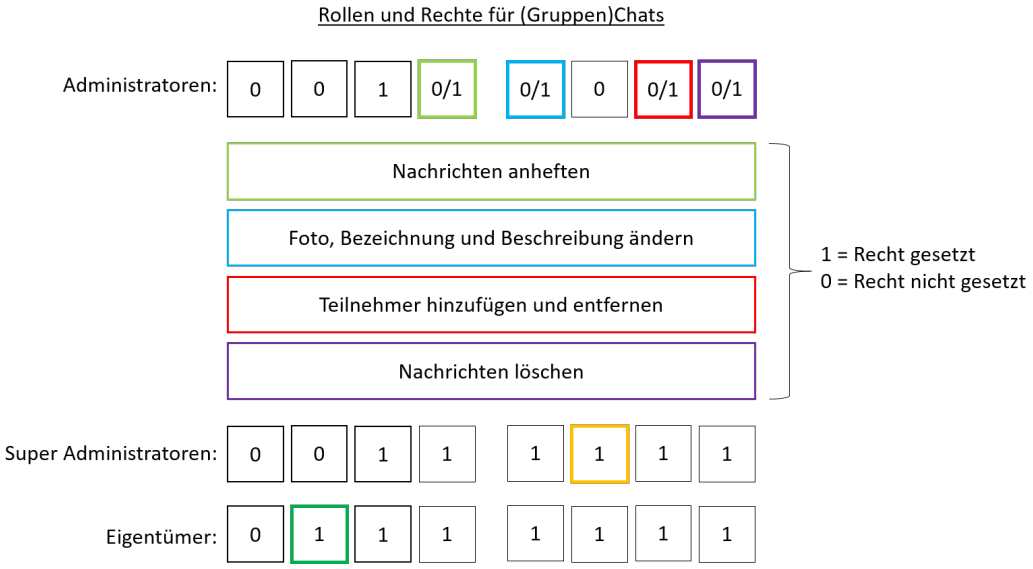


Abbildung 4.7: Interpretation des Bytes für Rollen und Rechte bei Chats

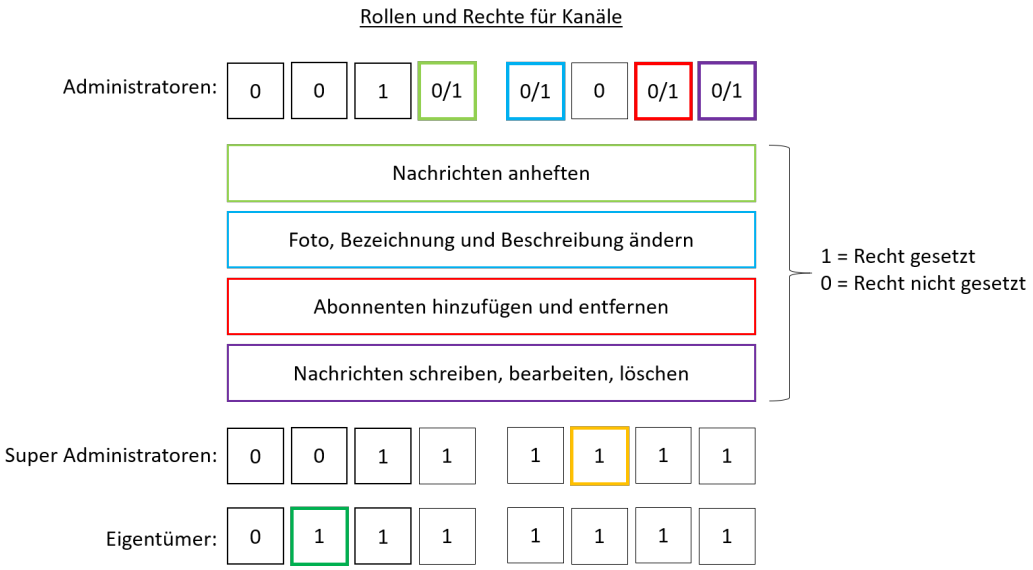


Abbildung 4.8: Interpretation des Bytes für Rollen und Rechte bei Kanälen

Die exemplarische Analyse eins *cht_data* BLOBs anhand der aufgeführten Marker zeigt Abbildung 4.9.

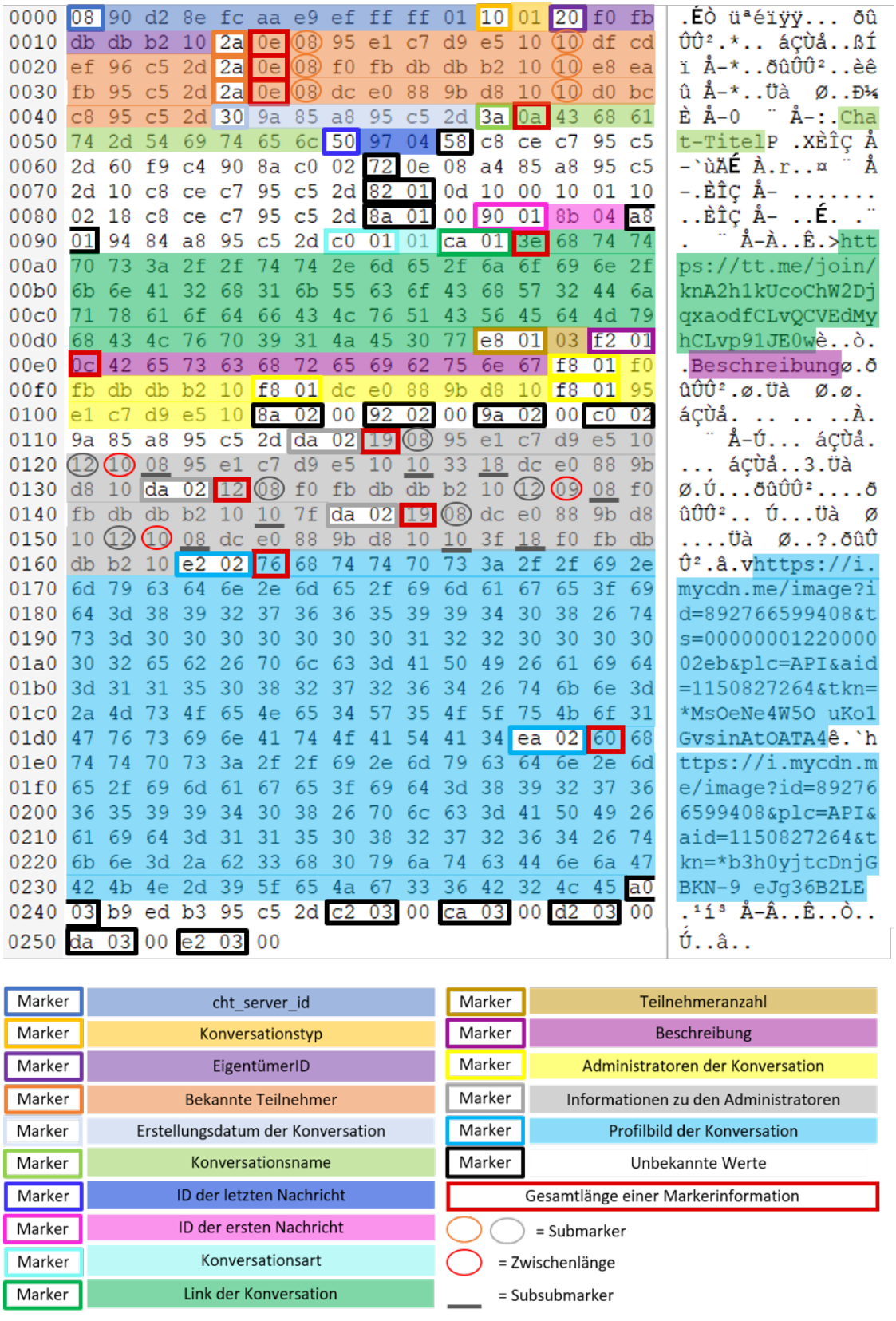


Abbildung 4.9: Exemplarische Auswertung eines *cht_data* BLOBs aus der Tabelle *chats*

Innerhalb des BLOBs lassen sich an verschiedenen Stellen NutzerIDs finden. Diese können mit Hilfe der Tabelle *contacts* verschiedenen Nutzern zugeordnet werden. Auf diesen Prozess wird im Rahmen dieser Auswertung nicht weiter eingegangen.
Bei dem abgebildeten BLOB handelt es sich um den BLOB einer Gruppenkonversation mit der *cht_server_id 0x90d28efcaae9effff01* (ID codiert im Tex-Verfahren, als Dezimalzahl lautet die ID: *-71147483584240*).
Eigentümer der Gruppe ist der Nutzer mit der ID *0xf0fbdbdbb210* („Hannah Wie“). Bei diesem Nutzer handelt es sich um die Accountinhaberin. Ihr sind zusätzlich zu ihr selbst noch zwei weitere Gruppenmitglieder bekannt. Diese sind *0x95e1c7d9e510* („Dana Wie“) und *0xdce0889bd810* („Finja Wie“). Gegründet wurde die Gruppe am 02.08.2019 um 16:09:29 Uhr (im Tex-Verfahren codiert: *0x9a85a895c52d*). Sie besitzt den Namen „Chat-Titel“ und die Beschreibung „Beschreibung“. Die Gruppe ist privat und an einen privaten Einladungslink (*https://tt.me/join/knA2h1kUcoChW2DjqxaodfCLvQCVEdMyhCLvp91JE0w*) gebunden. Zudem verfügt sie über ein Profilbild, welchem die folgenden zwei URLs zugeordnet werden können:

- URL1: *https://i.mycdn.me/image?id=892766599408&ts=0000000122000002eb&plc=API&aid=1150827264&tkn=*MsOeNe4W5O_uKo1GvsinAtOATA4*
- URL2: *https://i.mycdn.me/image?id=892766599408&plc=APIaid=1150827264&tkn=*b3h0yjtcDnjGBKN-9_eJg36B2LE*

Werden die URLs angepasst und in einen Webbrowser eingegeben, ergibt sich das in Abbildung 4.10 dargestellte Profilbild. Dieses basiert auf der zweiten URL, die mit der Endung *&fn=w_1440* modifiziert wurde.

Abbildung 4.10: Profilbild der Konversation „Chat-Titel“

Des Weiteren besitzt die erste bekannte Nachricht der Konversation die ID *0x8404*, die ID der letzten bekannten Nachricht lautet *0x9704*. Beide IDs sind im Tex-Verfahren codiert. Als Dezimalzahl lautet die ID der ersten Nachricht *516* und die ID der letzten Nachricht *535*.
Die Gruppe besitzt drei Mitglieder, welche alle eine Rolle haben, die höher ist als die eines Teilnehmers. Aus diesem Grund werden alle drei Gruppenmitglieder als Administratoren aufgelistet und es sind Informationen zu ihren Rechten und Rollen verfügbar.

Dem Gruppenmitglied „Finja Wie“ wurde ihre Rolle von Nutzer „Dana Wie“ zugewiesen. Das Byte für Rollen und Rechte besitzt bei „Finja Wie“ den Wert *0x33*. Dieser Wert lautet in Binärdarstellung *0011 0011*. Wird er, wie in Abbildung 4.7 beschrieben, interpretiert, so bedeutet dies, dass „Finja Wie“ ein Administrator mit den Rechten „Nachrichten anheften“, „Teilnehmer hinzufügen und entfernen“ und „Nachrichten löschen“ ist. Bei „Dana Wie“ handelt es sich um einen Super Administrator. Ihr Byte für Rollen und Rechte besitzt den Wert *0x3F*. Die Rolle des Super Administratoren wurde ihr von „Hannah Wie“ gegeben. „Hannah Wie“ ist der Gruppeneigentümer. Diese Information ist aufgrund des Markers *0x20* bereits bekannt, kann allerdings mit Hilfe des Markers *0xDA02* und dem Byte für Rollen und Rechte, welches den Wert *0x7F* besitzt, noch einmal bestätigt werden.

Informationen über die ausgetauschten Nachrichten, Medien sowie Anrufe

Informationen über die Nachrichten lassen sich den Tabellen *messages* und *upload_files* der *cache.db* entnehmen. Zudem werden auch Informationen über Anrufe und Videoanrufe in der Tabelle *messages* gespeichert.
Mit Hilfe der *_id* der *messages*-Tabelle und der *upload_message_id* der *upload_files*-Tabelle können beide Tabellen miteinander verknüpft werden. Die *upload_files*-Tabelle ist eine Ergänzung zur *messages*-Tabelle, da hier zusätzliche Informationen über versendete Medien wie Bilder oder Videos gespeichert werden.
Im Folgenden wird zunächst auf Attribute der *messages*-Tabelle eingegangen, die bei allen Nachrichteneinträgen vorhanden und für alle Einträge gleich zu interpretieren sind. Im Anschluss werden die Nachrichteneinträge mit Hilfe des Attributs *msg_media_type* in Nachrichtentypen eingeteilt. Für jeden dieser Nachrichtentypen müssen die verbleibenden Attribute individuell analysiert und im Anschluss miteinander in Verbindung gesetzt werden, um sie erfolgreich zu interpretieren.

***_id* und *msg_server_id*:**

Bei diesen Attributen handelt es sich um die ID einer Nachricht aus der Tabelle *messages*. *_id* ist dabei eine lokale ID innerhalb der *cache.db* aus der sie stammt. *msg_server_id* ist eine globale ID der Nachricht, die auch in anderen *cache.db*-Datenbanken und Datenbanken von iOS-Geräten auftreten kann.

***msg_sender*:**

Das Attribut *msg_sender* der Tabelle *messages* gibt an, welchem Nutzer die betreffende Nachricht zuzuordnen ist. Kanäle stellen allerdings eine Besonderheit dar. Bei diesen kann der Kanaleigentümer einstellen, ob Nachrichten von ihrem Sender unterzeichnet werden sollen. Ist diese Funktion ausgewählt, dann enthält das Attribut *msg_sender*, wie bei Gruppen und Privatchats, die NutzerID des Senders.

Ist diese Funktion nicht aktiviert, so wird für Nutzer, bei denen es sich nicht um den Accountinhaber handelt, keine NutzerID gespeichert. Stattdessen wird bei *msg_sender* der Wert *0* eingetragen. Standardmäßig ist die Funktion „Nachrichten unterzeichnen“ deaktiviert.

***msg_text*:**
Dieses Attribut beschreibt den Text einer Nachricht, der bis auf wenige Ausnahmen (siehe Attribut *msg_inserted_from_link*) von einem TamTam-Nutzer verfasst wurde.

***msg_delivered_status*:**
Mit Hilfe des Attributs *msg_delivered_status* wird der Zustellungs- und Lesestatus einer Nachricht gespeichert. Durch den Wert des Attributs kann festgestellt werden, ob eine Nachricht gesendet, empfangen und gelesen oder ohne Internetverbindung als „Offline-Nachricht“ verschickt wurde. Es ist zu beachten, dass die Interpretation der Attributwerte davon abhängig ist, ob eine Nachricht von dem Accountinhaber stammt, oder ob die Nachricht an ihn gesendet wurde.
Stammt eine Nachricht von dem Accountinhaber, so erhalten gesendete Nachrichten in Chats und Privatchats den Wert *20*. Wurde seine Nachricht empfangen und gelesen, so erhält sie den Wert *30*. Befinden sich mehrere Mitglieder in einem Chat, so erhält die Nachricht den Wert *30*, sobald sie das erste Mal von einem Nutzer gelesen wurde. Ungesendete Nachrichten (Offline-Nachrichten) besitzen den Wert *10*.
Bei Kanälen kann nicht unterschieden werden, ob eine Nachricht des Accountinhabers gesendet oder empfangen und gelesen wurde.
Das Attribut *msg_delivered_status* erhält bei verschickten Nachrichten in Kanälen immer den Wert *20*. Offline-Nachrichten erhalten wie bei Privatchats und Gruppen den Wert *10*.
Wird eine Nachricht von einem anderen Nutzer an den Accountinhaber gesendet, so kann mittels *msg_delivered_status* nicht festgestellt werden, ob er eine Nachricht gelesen hat. Es wurde bisher lediglich der Wert *20* als Attributwert für Nachrichten an den Accountinhaber beobachtet.

***msg_status*:**
Der *msg_status* gibt an, ob eine Nachricht nach dem Verschicken bearbeitet wurde. Im Falle einer Bearbeitung erhält *msg_status* den Wert *20*, andernfalls wird dem Attribut der Wert *0* zugeordnet.

***msg_time*, *msg_update_time* und *msg_local_time*:**
Diese drei Attribute speichern die Zeit, zu der eine Nachricht verschickt beziehungsweise empfangen wurde. Bei allen drei Zeitstempeln handelt es sich um einen Unix-Zeitstempel. *msg_time* und *msg_update_time* weichen nur in Sonderfällen von einander ab. Bei diesen Sonderfällen handelt es sich um Nachrichten,

die ohne Internetverbindung verschickt oder bearbeitet wurden. Auf diese Sonderfälle wird bei der Betrachtung von Textnachrichten noch einmal genauer eingegangen (siehe Textabschnitt *msg_media_type*: *0*). *msg_local_time* wird nur dann ein Wert zugewiesen, wenn die Nachricht von dem Accountinhaber selbst verfasst wurde. Es ist zu beobachten, dass die *msg_local_time* meist wenige Sekunden vor der *msg_time* liegt. Ein Sonderfall sind auch hier Nachrichten, die ohne Internetverbindung verschickt wurden.

***msg_type*:**
Der *msg_type* unterscheidet, ob eine Nachricht in einem Kanal oder einem Privatchat beziehungsweise Chat versandt wurde. Der Attributwert *10* steht für Nachrichten aus Privatchats und Chats, die Werte *30* und *40* stehen für Nachrichten aus Kanälen. Ob ein Kanal den Wert *30* oder *40* erhält, ist davon abhängig, ob die Funktion „Nachrichten unterzeichnen“ aktiviert ist. Kann eine Nachricht einem Administrator zugeordnet werden, so erhält der Kanal den Wert *40*. Ist die Funktion für einen Kanal deaktiviert, so erhalten seine Nachrichten den Wert *30*. Der Attributwert *20* wurde bisher nicht beobachtet.

***msg_chat_id*:**
Diese ID gibt an, zu welcher Konversation eine Nachricht gehört. Mit Hilfe dieser ID können Tabelle *messages* (Attribut: *msg_chat_id*) und *chats* (Attribut: *_id*) miteinander verknüpft werden.

***msg_inserted_from_link*:**
Dieses Attribut wurde noch nicht vollständig untersucht. Ist der Wert des Attributs *1*, so bezieht sich der Text dieser Nachricht (*msg_text*) meist auf die zuvor gesendete Nachricht beziehungsweise auf den zuvor gesendeten Link. Sofern vorhanden, wird der Text „Entwurf Kanal“ als Wert des Attributs *msg_text* von TamTam generiert und bedeutet, dass der Link eines Kanals in eine Konversation gepostet wurde.

***msg_link_type*:**
Der *msg_link_type* beinhaltet Informationen über weitergeleitete Nachrichten, sowie Nachrichten, die mittels „Antworten auf“-Funktion verschickt wurden.
Besitzt das Attribut *msg_link_type* den Wert *1* (Weiterleitung Typ 1), so handelt es sich um eine Antwort auf die Nachricht mit der ID, welche von dem Attribut *msg_link_id* gespeichert wird. Die Nachricht stammt aus der Konversation mit dem Wert des Attributs *msg_link_chat_id* als lokale ID.
Der Wert *2* des Attributs *msg_link_type* weist auf eine weitergeleitete Nachricht hin (Weiterleitung Typ 2). Die ID der Nachricht, welche weitergeleitet wurde, wird von dem Attribut *msg_link_id* gespeichert. Den Namen der Quellkonversation enthält das Attribut *msg_link_chat_name*. Das Attribut *msg_link_chat_id* speichert die globale ID der Konversation, aus der die weitergeleitete Nachricht stammt.

Anmerkung: Zum aktuellen Zeitpunkt ist unbekannt, wie und warum TamTam die IDs der weitergeleiteten Nachrichten generiert, wenn „Weiterleitung Typ 1“ und „Weiterleitung Typ 2“ aufeinander folgen / sich aufeinander beziehen. Diese Unklarheiten beziehen sich lediglich auf die NachrichtenID (Attribut *msg_link_id*), nicht auf eine identifizierte Konversation als Quelle einer weitergeleiteten Nachricht. Verweist „Weiterleitung Typ 2“ auf eine „normale“ Nachricht, so ist diese Anmerkung nicht zu beachten und die ausgelesene NachrichtenID kann der weitergeleiteten Nachricht als Quelle eindeutig zugeordnet werden.

***msg_media_type*:**

TamTam teilt die Nachrichten, die in der Tabelle *messages* gespeichert werden, mit Hilfe des Attributs *msg_media_type* in verschiedenen Typen ein. So kann dokumentiert werden, ob es sich bei einer Nachricht um ein Bild, ein Video, eine Text- oder eine Sprachnachricht handelt. Der folgenden Tabelle kann entnommen werden, wie die verschiedenen Nachrichtentypen zu interpretieren sind.

Attributwert	**Interpretation**
0	Textnachricht / Statusnachricht / Kontakt / Sticker
1	Bildnachricht
2	Sprachnachricht
3	Videonachricht
4	mehrere Dateien (zum Beispiel mehrere Bilder gleichzeitig)
5	geteilte Links
6	bisher nicht beobachtet
7	Datei
8	Anruf
9	Standort

Tabelle 4.8: Interpretation des Attributs *msg_media_type*

Der Typ der Nachricht hat einen Einfluss darauf, welche Informationen abgespeichert werden. Besonders die Attribute *msg_text* und *msg_attaches* sind davon betroffen.
Das Attribut *msg_attaches* speichert als BLOB viele Informationen, zu denen es kein eigenes Attribut in der Tabelle *messages* gibt. Der BLOB folgt dem gleichen Schema wie die BLOBs der Attribute *cht_data* und *ctt_data*.
Das erste Byte im BLOB lautet *0x0A* und stellt einen Marker dar. Im Anschluss an den Marker *0x0A* folgt die Länge der Information (im Tex-Verfahren codiert), die dem Marker zugeordnet ist. Das Ende der Information entspricht dem Ende des BLOBs, weshalb davon ausgegangen werden kann, dass sich in dem BLOB nur ein Marker befindet. Alle weiteren Marker sind Submarker dieses Markers. Die Submarker sind für jeden Nachrichtentypen individuell. Allerdings enthalten alle *msg_attaches*-BLOBs die Submarker *0x08* und *0x6A*.
Der Submarker *0x08* folgt direkt nach der Längenangabe des Markers *0x0A*. Mit Hilfe

des ihm zugehörigen Werts kann differenziert werden, zu welchem Nachrichtentyp der BLOB gehört. So kann ermittelt werden, wie die Werte der Submarker zu interpretieren sind, ohne dass das Attribut *msg_media_type* betrachtet werden muss. Die Bedeutung der Werte des Markers *0x08* zeigt Tabelle 4.9. Es ist zu beachten, dass die Werte des Markers *0x08* und die Werte des Attributs *msg_media_type* in ihrer Bedeutung nicht übereinstimmen.
Der Submarker *0x6A* hat keine feste Position im BLOB *msg_attaches* und folgt teilweise nach dem Marker *0x08*, teilweise handelt es sich allerdings auch um den letzten Submarker innerhalb des BLOBs. Der erste Wert des Markers *0x6A* ist die Länge der Information des Markers. Diese beträgt *0x24*. Nach der Länge der Information folgt die Information selbst. Bei dieser handelt es sich um eine Bytefolge, die für jeden *msg_attaches*-BLOB individuell ist. Wird sie als ASCII-Text betrachtet, so ergibt sich eine Zeichenfolge im Schema *XXXXXXXX-4XXX-XXXXX-XXXXXXXXXXXX*. Dieses Schema entspricht einer UUID ("Universaly Unique Identifier") der Version 4. UUIDs sind eindeutige 128-Bit-Folgen. Bei UUIDs der Version 4 wird die Bitfolge aus Zufallszahlen erzeugt. [52]

Werte des Markers 0x08	**Interpretation**
0x01	Systemnachrichten
0x02	Bildnachrichten
0x03	Videonachrichten
0x04	Sprachnachrichten
0x05	Sticker
0x06	geteilte Links
0x08	Anrufe
0x09	Musikdateien
0x0A	Dateien
0x0B	Kontakte
0x0E	Standorte

Tabelle 4.9: Bedeutung der Werte des Markers *0x08* des BLOBs *msg_attaches* aus der Tabelle *messages* der *cache.db*

Textnachrichten, Kontakte, Sticker, Systemnachrichten (*msg_media_type*: *0*):

Textnachrichten besitzen den *msg_media_type 0*. Der Inhalt von Textnachrichten wird mit Hilfe des Attributs *msg_text* abgespeichert. Wird einer Textnachricht kein Nachrichteninhalt zugeordnet, so handelt es sich um eine Systemnachricht. Systemnachrichten werden durch TamTam generiert und informieren die Nutzer über Veränderungen innerhalb einer Konversation. Dazu zählt beispielsweise, ob ein Nutzer hinzugefügt oder entfernt wurde. Die Informationen einer Systemnachricht werden im BLOB *msg_attaches* gespeichert. Klassische Textnachrichten erhalten keinen Wert für das Attribut *msg_attaches*.

bearbeitete Textnachrichten und Offline-Textnachrichten

Es ist möglich, klassische Textnachrichten zu bearbeiten. Diese Bearbeitung kann mit Hilfe des Attributs *msg_status* festgestellt werden.

Bearbeitete Nachrichten erhalten bei diesem Attribut den Wert *20*. Zudem können bei bearbeiteten Nachrichten *msg_time* und *msg_update_time* voneinander abweichen. *msg_update_time* speichert in diesen Fällen die Uhrzeit, zu der die Textnachricht bearbeitet wurde.
Offline-Textnachrichten werden generiert, wenn Textnachrichten ohne Internetverbindung verschickt werden. Sie besitzen den *msg_delivered_status 10*.
Offline-Nachrichten haben einen großen Einfluss auf die Zeitstempel, welche in der Tabelle *messages* gespeichert werden. Der *msg_update_time* wird bei Offline-Nachrichten der Wert NULL zugewiesen. Die *msg_time* speichert den Zeitpunkt, zu dem zuletzt eine Internetverbindung bestand. *msg_time_local* dokumentiert den Zeitpunkt, an dem die Nachricht vom Accountinhaber abgeschickt wurde. Eine Offline-Nachricht wird von TamTam versendet, sobald eine Internetverbindung zur Verfügung steht. Ist dies der Fall, werden die Zeitstempel der Offline-Nachricht aktualisiert. *msg_time* und *msg_update_time* erhalten den Zeitpunkt, an dem die Offline-Nachricht erfolgreich versendet werden konnte. *msg_time_local* wird nicht verändert.

Systemnachrichten
Systemnachrichten treten vor allem bei Chats auf. Bei Kanälen entfällt ein großer Teil der Systemnachrichten. Bisher wurden lediglich Systemnachrichten nach Erstellung eines Kanals beobachtet. Für das Ändern des Titels oder des Profilbilds wird beispielsweise keine Systemnachricht generiert. Bei Privatchats wurden keinerlei Systemnachrichten beobachtet.
Hauptinformationsquelle für die Systemnachrichten ist der BLOB *msg_attaches*. Die Struktur einer Systemnachricht im BLOB *msg_attaches* zeigt Abbildung B.6 im Anhang. Wie das Ereignis im BLOB zu interpretieren ist, illustriert Abbildung B.7 im Anhang.
Bei dem Ereignis mit dem Wert *0x06* (Änderung des Profilbild der Konversation) bei Chats besteht eine Besonderheit bei der Position der bisher unbekannten Bytefolge *0x5001*. Diese markiert nicht, wie in den anderen Fällen, das Ende des Ereignisses, sondern trennt die zwei URLs des Profilbilds von einander.
Die Information, welcher Nutzer dem Nutzer A zuzuordnen ist, lässt sich *msg_sender* entnehmen. Dieses Attribut speichert bei Systemnachrichten, welcher Nutzer eine Systemnachricht verursacht hat und nicht, wie bisher, welcher Nutzer eine Nachricht verschickt hat.

Kontakte
Versendete Kontakte besitzen ebenfalls den *msg_media_type*-Wert 0. Wie bei den Systemnachrichten werden die Hauptinformationen im BLOB *msg_attaches* gespeichert. Die Struktur eines versendeten Kontakts im BLOB *msg_attaches* zeigt Abbildung B.5 im Anhang. Nicht alle Subsubmarker der Kontaktinformation müssen für einen Kontakt vorhanden sein.

Sticker

Den zuvor vorgestellten Attributen lässt sich unter anderem entnehmen, wann ein Sticker von welchem Nutzer verschickt wurde. Die Informationen, um welchen Sticker es sich dabei handelt, wird im BLOB *msg_attaches* abgespeichert. Da davon auszugehen ist, dass dieser BLOB keine relevanten Nutzerdaten speichert, wird in dieser Arbeit darauf verzichtet, den BLOB vollständig aufzuschlüsseln. Eine Auswahl der Marker eines *msg_attaches*-BLOBs wird in Abbildung B.8 im Anhang dargestellt.

Zu jedem Sticker werden im *msg_attaches*-BLOB drei URLs gespeichert. Diese stimmen mit den URLs überein, die in der Tabelle *stickers* der *cache.db* abgelegt werden (Attribute: *sticker_url*, *sticker_firstUrl*, *sticker_previewUrl*). Aus diesem Grund ist es nicht notwendig, die URLs aus dem BLOB zu extrahieren und es kann darauf verzichtet werden, ihre Marker zu ermitteln.

Um einen Datensatz aus der Tabelle *stickers* mit einem Sticker aus einem Nachrichteneintrag der Tabelle *messages* zu verknüpfen, kann die StickerID verwendet werden. Diese wird in der Tabelle *stickers* mit Hilfe des Attributs *sticker_id* abgespeichert. In der Tabelle *messages* wird die StickerID im BLOB *msg_attaches* als Submarker des Submarkers *0x32* abgelegt (siehe Abbildung B.8 im Anhang). Der Bildinhalt eines Stickers kann mit Hilfe seiner URLs in einem Webbrowser ermittelt werden.

Bilder (*msg_media_type*: *1*):

Bilder können offline und online betrachtet werden, das heißt sie werden auf dem Mobiltelefon gespeichert.

Besitzt ein Bild eine Beschriftung, so wird diese als Wert des Attributs *msg_text* gespeichert. Andernfalls erhält das Attribut keinen Wert.

Die Hauptinformationen über ein Bild befinden sich in dem BLOB des Attributs *msg_attaches* der Tabelle *messages*. Zusätzliche Informationen werden in der Tabelle *upload_files* abgespeichert. Bedingung für diese Zusatzinformationen ist, dass das betreffende Bild von dem Accountinhaber stammt. Wurde ihm ein Bild lediglich von einem anderen TamTam-Nutzer zugeschickt, so enthält *upload_files* keine Informationen über dieses.

Der BLOB *msg_attaches* für Bilder ist in Abbildung B.9 im Anhang illustriert.

Das Vorschaubild (Submarker *0x42*) ist ein PNG. Beginn und Ende des Vorschaubilds wird durch die Header-Footer-Struktur für PNGs (Header: *0x89 50 4e 47*, Footer: *0x49 45 4e 44 ae 42 60 82*) markiert.

Wurde ein Bild von dem Accountinhaber verschickt, so speichert der Wert des Submarkers *0x72* seinen Speicherpfad. Dieser lautet: */storage/emulated/0/Android/data/ru.ok.messages/cache/upload/<Unix-Zeitstempel>*. Der Unix-Zeitstempel, welcher als Dateiname verwendet wird, ist in dem BLOB *msg_attaches* zusätzlich als Wert des Markers *0xB001* abgelegt. Dieser ist im Gegensatz zu dem Dateinamen im Speicherpfad mit dem Tex-Verfahren codiert. Beide Marker fehlen, wenn es sich um eine empfangenes Bild handelt.

Der beschriebene Speicherort ist allerdings nicht die einzige Stelle im Speicher, an der sich ein versendetes Bild befinden kann. Ein weiteres Exemplar lässt sich mit Hilfe des Speicherpfads finden, welcher innerhalb der Tabelle *upload_files* durch das Attribut *upld_path* gespeichert wird. Dieser Pfad ist davon abhängig, wie ein Bild aufgenommen wurde. Bilder können aus der Galerie eines Mobiltelefons stammen oder direkt innerhalb von TamTam aufgenommen werden. Stammt ein Bild aus der Galerie und wurde mit der Kamera-App des Mobiltelefons aufgenommen, so lautet der Wert des Attributs *upld_path*: */storage/emulated/0/DCIM/Camera/<Aufnahmezeitpunkt>.jpg*.
Wird ein Bild direkt über TamTam aufgenommen, so lautet der Pfad: */storage/emulated/0/Pictures/TamTam/IMG_<Aufnahmezeitpunkt>.jpg*.
Als Schema für den Aufnahmezeitpunkt wird jeweils *YYYYMMDD_HHMMSS* verwendet. Ein Bild, welches am 02.08.2019 um 16:14 Uhr in TamTam aufgenommen wurde, heißt beispielsweise: *IMG_20190802_161408.jpg*.
Der Aufnahmezeitpunkt, welcher im Bildnamen abgespeichert wird, muss nicht dem Sendezeitpunkt entsprechen. Dieser wird mit Hilfe des Attributs *msg_time* der Tabelle *messages* abgespeichert.

Sprachnachrichten (*msg_media_type*: *2*):

Sprachnachrichten können offline und online angehört werden. Sie besitzen den Nachrichtentyp (*msg_media_type* 2) und speichern ihre Hauptinformationen mit Hilfe des *msg_attaches* BLOBs. Im Gegensatz zu anderen Medien wie Bildern und Videos werden Sprachnachrichten nicht zusätzlich in der Tabelle *upload_files* aufgeführt.
Die Struktur des BLOBs zeigt Abbildung B.10 im Anhang. Mit Hilfe des Submarkers *0x72* kann der Speicherort einer Sprachnachricht auf dem Mobiltelefon identifiziert werden. Handelt es sich um eine empfangene Sprachnachricht so entfällt die Information über den Speicherpfad sowie der darauf folgende Zeitstempel (Submarker *0x72* und *0x8001*). Ist ein Speicherpfad vorhanden, so lautet er */storage/emulated/0/Android/data/ru.ok.messages/cache/audioCache*. Im Ordner *audioCache* wird eine Sprachnachricht mit dem Namen *audio_<Aufnahmezeitpunkt>.wav* abgespeichert. Der Aufnahmezeitpunkt ist als Unix-Zeitstempel codiert.
Empfangene Sprachnachrichten werden ebenfalls im Ordner *audioCache* mit einem Namen im gleichen Format abgespeichert. Allerdings stellt die Zahlenfolge, welche in dem Dateinamen enthalten ist, keinen Unix-Zeitstempel dar. Stattdessen handelt es sich um die ID der Sprachnachricht (AudioID). Diese wird ebenfalls im *msg_attaches*-BLOB einer Sprachnachricht abgelegt (Submarker *0x2A*, Subsubmarker *0x08*). Aus diesem Grund ist mit ihrer Hilfe eine Zuordnung einer Audiodatei im Speicher des Mobiltelefons zu dem entsprechenden Datenbankeintrag einer empfangenen Sprachnachricht möglich. Da der Wert des Subsubmarkers *0x08* mit dem Tex-Verfahren codiert ist, muss dieser jedoch für eine Zuordnung zunächst decodiert werden. Ein Beispiel für die Namen von gesendeten und emp-

fangenen Sprachnachrichten zeigt Abbildung 4.11.
Eine weitere Möglichkeit, um Sprachnachrichten aus beiden Gesprächsrichtungen anzuhören und herunterzuladen, sind ihre URLs. Diese sind im *msg_attaches*-BLOB enthalten und können ohne eine Anpassung in einen Webbrowser eingegeben werden.

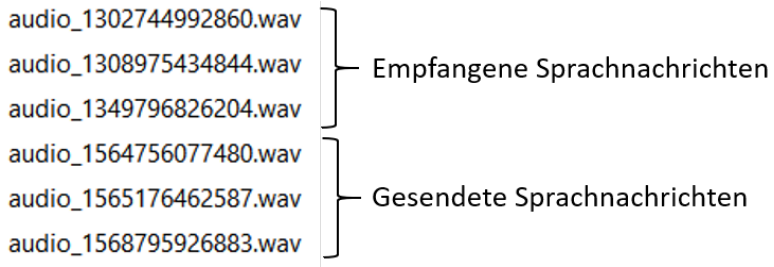


Abbildung 4.11: Namen für Sprachnachrichten aus dem Ordner *audioCache*

Videonachrichten (*msg_media_type*: *3*):

Ein Video, welches der Accountinhaber von einem anderen TamTam-Nutzer erhalten hat, kann offline nicht betrachtet werden. Es ist lediglich ein Vorschaubild des Videos einsehbar. Hat der Accountinhaber ein Video verschickt, so kann es auch ohne Internetverbindung angesehen werden.
Videos können mit und ohne Beschriftung versendet werden. Besitzt ein Video eine Beschriftung, so wird diese mit Hilfe des Attributs *msg_text* gespeichert. Andernfalls erhält dieses Attribut keinen Wert.
Die Eigenschaften der Videos werden zu großen Teilen im BLOB *msg_attaches* abgespeichert. Weitere Informationen können in der Tabelle *upload_files* abgelegt werden, hierfür muss das Video allerdings von dem Mobiltelefon des Accountinhabers stammen. Die Struktur für Videoinformationen im BLOB *msg_attaches* ist in Abbildung B.11 im Anhang dargestellt.
Die im BLOB enthaltene URL gehört zu dem Vorschaubild des Videos. Um dieses Bild einzusehen, muss die URL nicht modifiziert werden. Die URL eines Videos selbst ist in dem dazugehörigen BLOB allerdings nicht gespeichert. Dies ist besonders kritisch, wenn es sich um ein empfangenes Video handelt, da dies offline nicht einsehbar ist und demnach nicht lokal auf dem Mobiltelefon gespeichert wird. Aus diesem Grund kann lediglich mit bestehender Internetverbindung über das Mobiltelefon auf ein solches Video zugegriffen werden.
Das erwähnte Vorschaubild eines Videos kann nicht nur mit Hilfe der URL eingesehen werden. Es wird zusätzlich auch im BLOB selbst als *.png*-Datei gespeichert (Submarker *0x42*). Das Dateiformat *.png* lässt sich anhand der vorhandenen Header-Footer-Struktur für PNGs (Header: *0x89 50 4e 47*, Footer: *0x49 45 4e 44 ae 42 60 82*) erkennen.
Gesendete Videos werden auf dem Mobiltelefon gespeichert und enthalten in ihrem BLOB eine Angabe zu ihrem Speicherpfad (Submarker *0x72*). Dieser Speicherpfad entspricht dem Wert des Attributs *upld_path* der Tabelle *upload_files*.

Versendete Videos können, wie Bilder, aus der Galerie des Mobiltelefons stammen oder aber innerhalb von TamTam aufgenommen worden sein. Die Aufnahmeweise beeinflusst den Speicherpfad und den Namen des Videos.
Mit TamTam aufgenommene Videos werden als *MOV_Aufnahemzeitpunkt.mp4* im Pfad */storage/emulated/0/Movies/TamTam* gespeichert.
Videos, die mit der Kamera-App des Mobiltelefons aufgenommen wurden, sind unter */storage/emulated/0/DCIM/Camera/<Aufnahmezeitpunkt>.mp4* zu finden.
Das Schema für den Aufnahmezeitpunkt ist analog zu dem Schema für Bilder.
Wird ein Video verschickt, welches nicht innerhalb von TamTam aufgenommen wurde, so können Aufnahmezeitpunkt und Sendezeitpunkt beliebig weit auseinander liegen. Der Zeitpunkt zu dem ein Video versendet wurde, kann dem Attribut *msg_time* der Tabelle messages entnommen werden.

geteilte Links (*msg_media_type*: *5*):

Für versendete Links wird ebenfalls ein Nachrichteneintrag in der Tabelle *messages* angelegt. Das Attribut *msg_text* enthält den geteilten Link. Es kann darüber hinaus allerdings auch weiteren Nachrichtentext beinhalten, den der Sender zusätzlich zu dem Link verschickt hat. Der BLOB *msg_attaches* speichert weitere Informationen zu dem gesendeten Link. Es wurde darauf verzichtet den BLOB bis ins letzte Detail zu analysieren, da davon ausgegangen werden kann, dass alle relevanten Daten identifiziert wurden. Die bekannten Marker, Submarker und Subsubmarker sind in der Abbildung B.12 im Anhang dargestellt. Besitzt ein Link ein Vorschaubild, welches innerhalb von TamTam angezeigt wird, so kann ihm kein fester Marker zugeordnet werden. Es wurden bisher die Werte *0x2A* und *0x42* als Identifikator für das Vorschaubild beobachtet. Es ist jedoch möglich, das Vorschaubild auch ohne Kenntnis über seinen Marker zu extrahieren. Zu diesem Zweck kann nach der Header-Footer-Struktur (Header: *0x89 50 4e 47*, Footer: *0x49 45 4e 44 ae 42 60 82*) eines PNGs gesucht werden.

Dateien (*msg_media_type*: 7):

Eine Datei speichert ihre Informationen im BLOB *msg_attaches* sowie, falls die Datei von dem Accountinhaber versandt wurde, in der Tabelle *upload_files*. Der Speicherpfad von empfangenen, heruntergeladenen Dateien kann dem BLOB *msg_attaches* entnommen werden. Er lautet: */storage/emulated/0/Download/TamTam/<dateiname>.**.
Als Datei können Speicherelemente verschickt werden, die selbst einem Nachrichtentypen zugeordnet sind, wie beispielsweise Bilder. Es ist jedoch ebenfalls möglich, Speicherelemente zu verschicken, die keinen eigenen Nachrichtentypen besitzen. Dazu zählen zum Beispiel .pdf-Dateien. Einige der Marker des BLOBs *msg_attaches* sind unabhängig von dem Typen der Datei vorhanden. Ein Teil der Marker ist jedoch spezifisch für bestimmte Dateien. Wie der BLOB aufgebaut ist, zeigt Abbildung B.13 im Anhang.

Sonderfälle bei versendeten Dateien

Es ist möglich, dass bei versendeten Dateien Sonderfälle auftreten. Von diesen wird gesprochen, wenn ein Speicherelement als Datei verschickt werden muss, seinem Dateitypen im BLOB *msg_attaches* jedoch ein eigener Nachrichtentyp zugeordnet wird. Ist dies der Fall, so gehören zu diesem Nachrichtentypen eigene Marker, welche im BLOB *msg_attaches* dem Subsubmarker *0x22* als dateispezifische Marker untergeordnet sind.
Dieser Sonderfall betrifft beispielsweise Musikdateien. Ob es weitere Sonderfälle gibt, ist zum aktuellen Zeitpunkt noch unbekannt.
Einer Musikdatei wird im BLOB *msg_attaches* der Nachrichtentyp *0x09* zugeordnet. Der Aufbau des BLOBs *msg_attaches* für Musikdateien sowie die dateispezifischen Marker werden in Abbildung B.14 im Anhang aufgeführt.
Handelt es sich um eine gesendete Musikdatei, so wird ihr Speicherpfad im BLOB vermerkt. Wurde die Datei von dem Accountinhaber empfangen, so kann er diese herunterladen. Ist dies geschehen, befindet sich die Musikdatei als Download im Order *userdata (ExtX)/Root/media/0/Download/TamTam/*. Anhand des Dateinamens, welcher im BLOB der Musikdatei enthalten ist, kann einem Datensatz aus der Tabelle *messages* das entsprechende Musikstück zugeordnet werden.

Anrufe (*msg_media_type*: *8*):

TamTam ermöglicht Audio- und Videotelefonate zwischen dem Accountinhaber und einer weiteren Person. Gruppenanrufe sind nicht möglich. Jeder Anruf beziehungsweise Anrufversuch wird von TamTam dokumentiert. Dazu generiert TamTam eine Nachricht im Privatchat von Accountinhaber und Gesprächspartner, welche sich in der Tabelle *messages* der *cache.db* finden lässt.
Das Attribut *msg_text* enthält keinen Inhalt. Der Anrufer ist der Nutzer, dessen ID bei *msg_sender* eingetragen ist. *msg_time* und *msg_update_time* speichern den Zeitpunkt an dem ein Telefonat beendet wurde. Weitere Informationen über einen Anruf sind im BLOB *msg_attaches* abgespeichert.
Abbildung B.15 im Anhang stellt die Struktur eines BLOBs für Anrufe dar.
Fand lediglich ein Anrufversuch statt, so fehlt die Gesprächsdauer und ihr Marker. Ob ein Anruf eingehend oder ausgehend ist, lässt sich dem Attribut *msg_sender* entnehmen. Ist der *msg_sender* der Accountinhaber, so handelt es sich um einen ausgehenden Anruf. Entspricht der *msg_sender* dem Gesprächspartner, welcher im BLOB *msg_attaches* gespeichert wird, so war ein Anruf eingehend.

Standorte (*msg_media_type*: *9*):

Standorte besitzen den *msg_media_type*-Wert *9*. Das Attribut *msg_text* verfügt über keinen Inhalt. Zum Speichern des Standorts wird der BLOB *msg_attaches* verwendet. Seine Struktur zeigt Abbildung B.16 im Anhang.
Breiten- und Längengrade sind in Little Endian Schreibweise notiert und nicht mittels Tex-Verfahren codiert. Allerdings müssen sie als Double-Wert interpretiert werden, um die korrekten Koordinaten zu erhalten.

Die wichtigste Information der vorgestellten Mediendateien ist der tatsächliche Dateiinhalt. Um diesen einzusehen, können zum einen die URLs verwendet werden, welche in dem BLOB *msg_attaches* der entsprechenden Mediendatei gespeichert sind. Zum anderen werden die Mediendateien, mit Ausnahme von empfangenen Videos, auch lokal auf dem Mobiltelefon gespeichert und können dort eingesehen werden. Die zweite Art und Weise, Mediendateien einzusehen, ist zu bevorzugen, da die Verwendung der URLs nur möglich ist, solange die Server von TamTam die Dateien online zur Verfügung stellen. Teilweise enthalten die URLs allerdings einen Hinweis darauf, dass diese nach einer gewissen Zeit ablaufen. Dies ist bei Sprachnachrichten der Fall. Wird die URL *https://m.ok.ru/dk?st.cmd=moviePlaybackRedirect&expires=1655196083434&type=2&sig=PcRSMXAywwk&ct=2&clientType=7&id=680019626736* betrachtet, lässt der Bestandteil *expires=1655196083434* darauf schließen, dass die URL ab dem *14.06.2022, 10:41:23 Uhr* nicht mehr verwendet werden kann. Um eine Aussage über den Zeitpunkt treffen zu können, wurde die Zahlenfolge als Unix-Zeitstempel interpretiert.
Das Vorgehen bei der Zuordnung einer Mediendatei im Speicher eines Mobiltelefons zu einem Datenbankeintrag ist von der Gesprächsrichtung (gesendet / empfangen) und dem vorliegenden Medientyp (Bild, Audio, Video, ...) abhängig. Handelt es sich um eine Datei, welche von dem Accountinhaber verschickt wurde, so beinhaltet der BLOB *msg_attaches* eine Information über den Speicherpfad der Datei. Mit der Hilfe dieses Pfades kann die gesuchte Datei im Speicher gefunden werden. Stammt die Datei von einem anderen TamTam-Nutzer und wurde somit vom Accountinhaber empfangen, enthält der BLOB *msg_attaches* keine erkennbare Informationen über den Speicherpfad oder den Namen der Datei. Ob und wie eine Mediendatei dennoch im Speicher gefunden und zugeordnet werden kann, ist vom Medientyp abhängig.
Auf die Verknüpfung von Videos, Sprachnachrichten und Musikdateien zu Datenbankeinträgen wurde bereits bei der Auswertung der jeweiligen Medientypen eingegangen. Bilder, Videovorschaubilder und Sticker werden zusammen in einem Cacheordner gespeichert und auf die gleiche Weise abgelegt. Ihre Zuordnung zu Datenbankeinträgen kann aufgrund ihrer Gemeinsamkeiten wie folgt zusammengefasst werden:

- Bilder, Videovorschaubilder und Sticker liegen im Pfad *userdata (ExtX)/Root/media/0/Android/data/ru.ok.messages/cache/imageCache/fresco/v2.ols100.1/*
- Im Ordner *v2.ols100.1* werden die Medien auf weitere Unterordner aufgeteilt, welche namentlich durchnummeriert sind. Im Rahmen dieser Arbeit wurden alle ganzzahligen Werte von *0* bis *99* beobachtet.
- Die Namen der Mediendateien umfassen 27 Zeichen und sind aus Sonderzeichen, Zahlen sowie Groß- und Kleinbuchstaben von A bis Z zusammengesetzt.
- Die Dateiendung einer Mediendatei lautet *.cnt*.
- Wird der Hexadezimalcode einer Mediendatei betrachtet, so ist ersichtlich, dass es sich um eine Datei im RIFF-Format handelt[5]

[5] Bei dem RIFF-Format (resource interchange file format) handelt es sich um ein Dateiformat von Microsoft und IBM, welches andere Dateiformate enthält. Es wird verwendet, um Multimedia-Daten zu speichern. Zu den gängigsten Formaten, die in einer RIFF-Datei auftreten, gehören AVI und WAV. [53] [54]

Die beschriebene Speicherweise wird nicht ausschließlich von TamTam verwendet. Die App „Facebook Messenger“ nutzt diese ebenfalls.
Bei der Untersuchung des Facebook Messengers in der Version 18.0.0.27.14 konnte von R. Tamma und D. Tindall festgestellt werden, dass dieser sowohl Sprachnachrichten, als auch Bilder und Videos als RIFF-Datei mit *.cnt*-Endung ablegt. [55, S. 217]
Der Cacheordner für Bilder (*/cache/image*) speichert zudem eine Vielzahl an Unterordnern, in die die Bilder und Videos einsortiert werden. Dies stellt eine weitere Gemeinsamkeit mit TamTam auf Android-Geräten dar. [55, S. 217]
Auch S. Tahiri nimmt bei der Auswertung der Facebook Messenger Version 56.0.0.27.64 auf diese Speicherweise Bezug. Sie beschreibt, dass der zuvor erwähnte Cacheordner (*/cache/image*) den Ordner *v2.ols100.1* enthält. Dieser lässt sich bei TamTam auf Android-Geräten ebenfalls finden. Die Ordner */data/data/com.facebook.orca/cache/image/v2.ols100.1* und */data/data/com.facebook.orca/files/image/v2.ols100.1* enthalten ihrer Beobachtung nach *.cnt*-Dateien, welche sich in durchnummerierten Unterordnern befinden. Die Namen der *.cnt*-Dateien bestehen ebenfalls aus 27 Zeichen. [19, S. 181-184]
Die gefundenen Gemeinsamkeiten von TamTam und Facebook Messenger lassen sich höchstwahrscheinlich darauf zurückführen, dass beide Messenger die Bibliothek “Fresco“ verwenden. Diese wurde von Facebook entwickelt und kann dazu verwendet werden, um auf Android-Geräten Bilder für Apps zu verwalten. Die Bibliothek enthält unter anderem Methoden um Bilder zu speichern, zu laden und anzuzeigen. Zudem sind Methoden zum Cachen von Bildern enthalten. Zusätzlich zu TamTam und dem Facebook Messenger wird “Fresco“ von weiteren Apps wie Twitter, Wikipedia, Vimeo oder 9GAG verwendet. [56] [57]
In Zusammenarbeit mit einem erfahrenen digitalen Forensiker des Landeskriminalamts NRW konnte durch eine Analyse des TamTam-Quelltexts sowie der “Fresco“-Methoden aus dem Facebook-Quelltext ermittelt werden, wie die Namen der Cachedateien von TamTam generiert werden. Mit diesem Wissen ist es möglich, einem Nachrichteneintrag aus der *messages*-Tabelle der *cache.db* die dazugehörige Mediendateien aus dem Speicher eines Android-Geräts zuzuordnen. Hierzu wird die URL der Mediendatei benötigt, welche im BLOB *msg_attaches* gespeichert ist. Diese muss zunächst auf die bekannte Weise, durch das Anfügen der Bildgröße, modifiziert werden. Daraufhin wird der SHA-1 Hashwert der URL gebildet und dieser im Anschluss Base64 codiert. Abschließend müssen innerhalb der Zeichenfolge alle Pluszeichen durch Minuszeichen sowie Schrägstiche durch Unterstriche ersetzt werden. Alle Gleichheitszeichen, die sich am Ende der Zeichenkette als Füllzeichen befinden können, müssen entfernt werden.
Wird nun die Dateiendung *.cnt* an die Zeichenfolge angefügt, so erhält man den vollständigen Namen für “Fresco“-Dateien.
Die Bedingung dafür, dass eine Bilddatei im Speicher mit Hilfe dieses Vorgehen gefunden werden kann ist, dass sich die Datei in der entsprechenden Größe im Speicher befinden muss. Neben den bisher vorgestellten Größen für ein Bilddatei (*sqr_192*, *w_1440*) gibt es weitere Formate für Bilder, Videovorschaubilder und Sticker (*w_1280, w_1080, w_1960, w_720, w_600, w_480, w_320, w_240, w_180, sqr_720, sqr_600,*

sqr_480, sqr_320, sqr_128, sqr_96, sqr_64, sqr_32). Eine Datei wird nicht in allen Größen abgespeichert, sodass alle Formatmöglichkeiten überprüft werden müssen, um eine Datei erfolgreich zu finden.
Zur Illustration des Prozesses der Namensgenerierung von “Fresco“-Dateien, folgt hier ein Beispiel:

URL: *https://i.mycdn.me/image?id=892766599408&ts=0000000122000002eb&plc=API&aid=1150827264&tkn=*MsOeNe4W5O_uKo1GvsinAtOATA4*

modifizierte URL: *https://i.mycdn.me/image?id=892766599408&ts=0000000122000002eb&plc=API&aid=1150827264&tkn=*MsOeNe4W5O_uKo1GvsinAtOATA4&fn=sqr_192*

SHA-1-Hash der modifizierten URL: *4a9c75ff6bb2600a4b05b8191623af77ec2ad520*

Base64-codierter Hash: *Spx1/2uyYApLBbgZFiOvd+wq1SA=*

Modifizierter Base64-codierter Hash: *Spx1_2uyYApLBbgZFiOvd-wq1SA*

Dateiname: *Spx1_2uyYApLBbgZFiOvd-wq1SA.cnt*

Die als Beispiel verwendete URL stammt aus dem *cht_data*-BLOB der Konversation „Chat-Titel“. Wird der Ordner *imageCache/fresco/v2.ols100.1* durchsucht, so kann die Datei mit dem ermittelten Namen und somit das Profilbild der Konversation gefunden werden.

4.2.3 Aufbereitungsphase

Die Aufbereitung von TamTam unter Android wurde im Rahmen dieser Arbeit mit einem Pythonskript automatisiert. Das Resultat der Auswertung und Aufbereitung der Messengerdaten wird als HTML-Dokument zur Verfügung gestellt.

Grundsätzlich lässt sich die Auswertung und Aufbereitung im Skript in drei Abschnitte einteilen. Diese entsprechen den Kategorien, welche bereits bei der Durchführung der Experimente und der vorangegangenen Auswertung verwendet wurden (Nutzerinformationen, Konversationen und Nachrichten). Das im Skript verwendete Konzept um die Messengerdaten jeder Kategorie zu untersuchen und die Ergebnisse im Anschluss auszugeben, ist prinzipiell identisch.
Zunächst wird dafür gesorgt, dass auf die benötigten Datenbankinformationen einer Kategorie zugegriffen werden kann. Dazu zählt das Anfordern der zu analysierenden Informationen mit Hilfe eines geeigneten SQL-Aufrufs. Die hierbei verwendeten Aufrufe lauten

- für Kontakte: *SELECT contacts._id, contacts.ctt_data, contacts.ctt_presence, contacts.ctt_server_id, phones.phs_phone FROM contacts left JOIN phones on contacts._id = phones.phs_contact_id*
- für Konversationen: *SELECT chats._id, chats.cht_data, chats.cht_server_id FROM chats*
- für Nachrichten: *SELECT * FROM messages left JOIN upload_files on messages._id = upload_files.upld_message_id ORDER BY messages.msg_chat_id, messages._id*

Die zurückgegebenen Datenbankinformationen werden daraufhin für die Auswertung Zeile für Zeile durchlaufen. Hierzu gehört es, die Attributwerte jedes Datensatzes jeweils einer Variablen im Skript zuzuweisen. So ist die Verarbeitung der Werte innerhalb des Skripts möglich.
Die Auswertung geschieht mittels bedingter Anweisungen, welche, je nach Wert einer Variablen, die entsprechende Interpretationsregel aufrufen. Das abschließende Ergebnis einer Interpretation wird nach Bedarf unterschiedlich abgespeichert. Teilweise wird eine speziell dafür angelegte Variable verwendet. Andernfalls wird die ursprünglichen Variable mit dem interpretierten Wert überschrieben.
Enthält das Attribut *msg_sender* beispielsweise die globale ID eines Nachrichtensenders, so wird diese im Skript, sofern vorhanden, durch den Namen des Nutzers ersetzt (siehe Abbildung 4.12).

```
messageSender = row['msg_sender']#Nachrichtensender
if messageSender == 0:
    messageSender = ""
if messageSender in user_Dez: #Sofern bekannt, dem Messagesender einen Namen zuweisen
    messageSender = user_Dez[messageSender]
```

Anmerkung: Bei „user_Dez“ handelt es sich um ein Dictionary, bei dem der globalen ID (in Dezimal) eines Nutzers der entsprechende Name zugeordnet wird.

Abbildung 4.12: Exemplarische Interpretation des Attributs *msg_sender*

Die Verwendung einer zusätzlichen Variablen zur Speicherung der Interpretationsergebnisse ist sinnvoll, wenn der originale Attributwert für spätere Abfragen beibehalten werden soll. Ebenfalls eignet sich die angelegte Ergebnisvariable dazu, die Auswertung mehrerer Attribute zu kombinieren und zusammenzufassen. Letzteres wird bei der Auswertung der Nachrichten in diesem Skript häufig eingesetzt. Bei Nachrichten, die als Antwort auf eine andere Nachricht verfasst wurden (Antworten-auf-Funktion) kombiniert die Variable *details* zum Beispiel die ID der Nachricht, auf die sich bezogen wird, sowie die ID der Konversation dieser (siehe Abbildung 4.13).

```
msg_link_id = row['msg_link_id'] #ID der Bezugsnachricht
msg_link_chat_id = row['msg_link_chat_id'] #Quelle der Nachricht (Konversation)
msg_link_type = row['msg_link_type'] #Weiterleitungstyp

if msg_link_type == 1: #Auswertung der Antworten-auf-Funktion
    details = "Antwort auf Nachricht "+ str(msg_link_id)+" innerhalb der Konversation "+str(msg_link_chat_id)
```

Abbildung 4.13: Ergebnis der Interpretation mehrerer Attribute bei der Verwendung der Antworten-auf-Funktion.

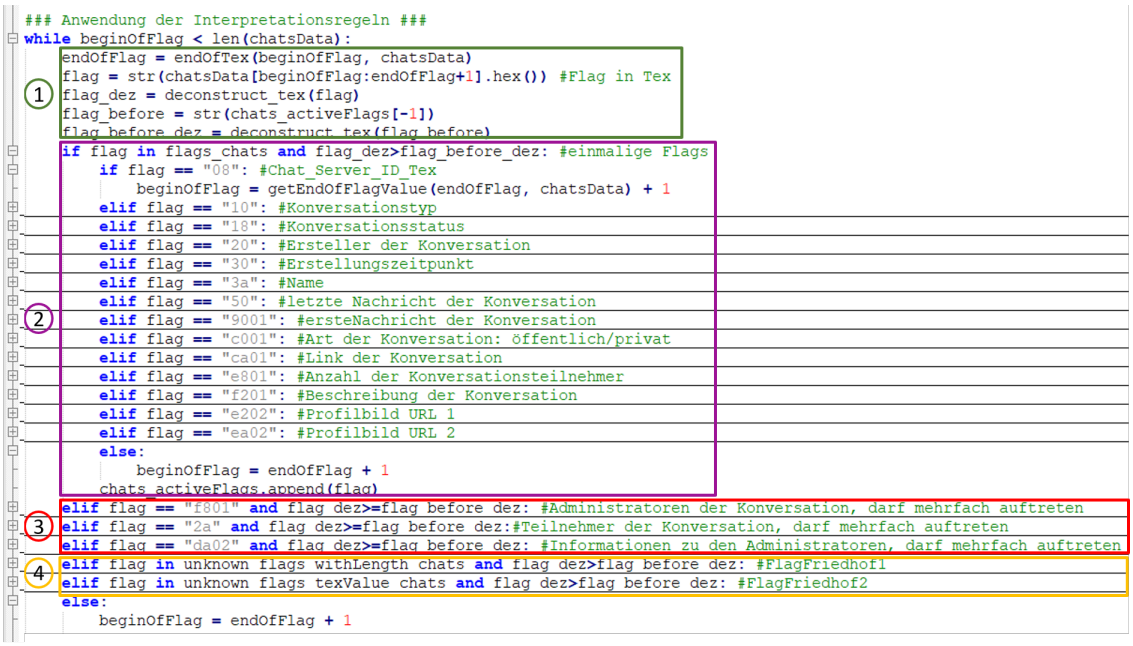
Die besondere Herausforderung bei der Programmierung war die Auswertung der Attribute des Typs BLOB. Die Analyse dieser ist wesentlich komplexer als die Auswertung von Text- oder Integer-Attributen und geht über einfache if/else-Abfragen hinaus.
Die in den BLOBs enthaltenen Marker sind zu kurz, um in einem BLOB einmalig aufzutreten. Zudem werden in den BLOBs keine festen Offsets verwendet. Die Position eines Markers und die seiner Information ist von dem vorherigen Marker abhängig, weshalb Marker und ihre Werte von einem Skript nicht einzeln ausgewertet werden können.
Die Konsequenz daraus ist, dass die verschiedenen Marker bei der Umsetzung in Python miteinander in Verbindung gebracht werden müssen. Hierzu muss das Ende einer Markerinformation zuverlässig berechnet werden, um den Beginn des darauf folgenden Markers zu ermitteln. Zu diesem Zweck wurden die Methoden *endOfTex(startPos, blob)*, *getFlagValue(endOfFlag, blob)* und *getEndOfFlagValue(endOfFlag, blob)* erstellt. Diese Methoden können für alle BLOBs der *cache.db* verwendet werden, da ihnen der BLOB als Parameter übergeben wird.
Die Methode *endOfTex(startPos, blob)* berechnet anhand einer übergebenen Startposition die Endposition einer Bytefolge, welche mit dem Tex-Verfahren codiert wurde. Diese Methode wird von der Methode *getEndOfFlagValue(endOfFlag, blob)* verwendet, damit anhand der Endposition des Markers das Ende seines dazugehörigen Werts berechnet

werden kann. Um den Wert, welcher einem Marker zugeordnet werden kann, effektiv auszulesen, existiert die Methode *getFlagValue(endOfFlag, blob)*.
In den BLOBs der *cache.db* werden allerdings nicht nur Marker, sondern auch Submarker und Subsubmarker verwendet. Damit auch bei diesen die Endposition eines Markerwerts sowie der Markerwert selbst bestimmt werden kann, existieren die Methoden:

- *getSubFlagValue(endOfSubFlag, blob)*
- *getEndOfSubFlagValue(endOfSubFlag, blob)*
- *getSubSubFlagValue(endOfSubSubFlag, blob)*
- *getEndOfSubSubFlagValue(endOfSubSubFlag, blob)*

Diese arbeiten äquivalent zu den Methoden *getFlagValue(endOfFlag, blob)* und *getEndOfFlagValue(endOfFlag, blob)* und wurden lediglich namentlich angepasst, damit jederzeit ersichtlich ist, in welcher Markerebene man sich befindet. So wird dem Betrachter der Überblick im Skript erleichtert.
Bei der Auswertung von Nutzerinformationen und Konversationen werden die jeweiligen BLOBs mit Hilfe einer while-Schleife Byte für Byte bis zu ihrem Ende durchlaufen. Es wird mit dem ersten Byte begonnen, welches als Beginn eines Markers interpretiert wird. Das Ende dieses Markers wird durch die Funktion *endOfTex(startPos, blob)* bestimmt. Im Anschluss wird festgestellt, ob die Auswertung des ermittelten Markers bekannt ist. Hierzu wird im ersten Schritt überprüft, ob der Marker in einer Speicherstruktur (hier: Dictionary oder Liste) mit bekannten Markern enthalten ist. Ist dies der Fall, so ist der Marker bekannt und kommt für eine Auswertung in Frage.
Um Interpretationsfehler zu vermeiden, wird als nächstes ermittelt, ob der bekannte Marker an einer gültigen Stelle im BLOB auftritt. Bei den untersuchten BLOBs konnte festgestellt werden, dass der Wert eines Markers immer größer (Marker tritt lediglich einmal im BLOB auf) oder gleich dem Wert des vorherigen Markers (Marker kann in einem BLOB mehrfach auftreten) ist. Um dies zu überprüfen, muss der Marker zunächst in Dezimal umgerechnet werden, da dieser Tex-codiert ist.
Handelt es sich um einen bekannten Marker an einer gültigen Stelle, so wird der Wert des Markers ausgewertet. Hierzu enthält das Skript if-Abfragen, sodass lediglich die individuellen Auswerteanweisungen ausgeführt werden. Auf die Analyse von Markerwerten wird im weiteren Verlauf dieses Kapitels noch einmal eingegangen. Als potenzieller Beginn des nächsten Markers wird zum Abschluss der Auswertung eines bekannten und gültigen Markers das Byte ausgewählt, welches auf das Ende seines Markerwerts folgt.
Liegt ein unbekannter oder ungültiger Marker vor, so wird das Byte nach dem Ende des Markers als Beginn des neuen Markers interpretiert und der Vorgang wiederholt sich.
Besitzt ein Marker Submarker oder Subsubmarker, so wiederholt sich der beschriebene Aufbau des Skripts. Das bedeutet, es wird innerhalb der Auswertung eines Markers erneut eine while-Schleife durchlaufen, die den Bereich des Markerwerts auf Submarker beziehungsweise den Bereich des Submarkerwerts auf Subsubmarker auswertet.

Um den Ablauf bei der Auswertung besser nachvollziehen zu können, zeigt Abbildung 4.14 eine verkürzte Darstellung der Analyse für Konversationen. Die Auswertung der Nutzerinformationen besitzt den gleichen Ablauf. Anders als bei den Konversationen können bei den Nutzerinformationen allerdings keine Marker mehrfach vorkommen. Zudem findet bei der Auswertung keine zusätzliche Behandlung von unbekannten Markern statt (siehe Abbildung 4.14, 4. Abschnitt).



Anmerkung: Bei dieser Abbildung handelt es sich um eine verkürzte Darstellung der Auswertung eines *cht_data*-BLOBs. Die individuelle Auswertung der Marker und ihrer Werte ist nicht dargestellt. Jede Auswertung eines Markers (if- beziehungsweise elif-Bedingung trifft zu) beinhaltet die Anweisungen *chats_activeFlags.append(flag)* und *beginOfFlag = <Ende des Markerwerts > + 1* (entspricht meist: *getEndOfFlagValue(endOfFlag, chatsData) + 1*). *flags_chats*, *unknown_flags_withLength_chats*, und *unknown_flags_texValue_chats* sind Dictionaries, die die bekannten Marker beinhalten.

1. Definiert den aktuellen und den vorherigen Marker.
2. Überprüft, ob es sich um einen gültigen Marker handelt, der in dem BLOB einmal auftritt und analysiert dessen Wert.
3. Überprüft, ob es sich um einen gültigen Marker handelt, der in dem BLOB mehrfach auftritt und analysiert dessen Wert.
4. Sorgt dafür, dass Marker, deren Bedeutung zwar unbekannt ist, deren Struktur jedoch identifiziert werden konnte, übersprungen werden können.

Abbildung 4.14: Verkürzte Darstellung des Skriptaufbaus zur Auswertung von Konversationen

Die Auswertung von Nachrichten basiert auf dem gleichen Konzept wie die Analyse von Nutzerinformationen und Konversationen. Allerdings verwenden Nachrichten in ihren BLOBs lediglich einen einzigen Marker. Alle weiteren Strukturen sind Sub- und Subsubmarker. Aus diesem Grund wird auf die oberste while-Schleife verzichtet. Stattdessen wird zunächst überprüft, ob der BLOB mit dem Marker *0x0A* beginnt, was der einzige bekannte und gültige Marker ist. Im Anschluss wird der erste Submarker bestimmt. Dieser muss *0x08* lauten, damit der BLOB korrekt ausgewertet werden kann, da dieser angibt, um welchen Medientyp es sich bei der Nachricht handelt. Passend zu dem ermittelten Medientyp wird daraufhin eine Funktion aufgerufen, die den BLOB anhand der individuellen Sub- und Subsubmarker auswertet. Hier wird erneut auf eine while-Schleife und den zuvor beschriebene Ablauf bei der Auswertung von Nutzerinformationen und Konversationen zurückgegriffen.

Das Ergebnis der Auswertung wird als String in der Variable *details* gespeichert. Der beschriebene Abschnitt des Skripts wird durch Abbildung 4.15 illustriert.

```
#Medientyp aus BLOB auslesen und BLOB msg_attaches dementsprechend auswerten
endOfFirstFlag = endOfTex(0, msg_attaches)
flag = str(msg_attaches[0:endOfFirstFlag+1].hex()) #Flag in Tex
if flag == "0a": #Flag für Mediendateien
    beginOfFlagValueFullLength = endOfFirstFlag + 1 #Berechnung der Gesamtlaenge der Information
    endOfFlagValueFullLength = endOfTex(beginOfFlagValueFullLength, msg_attaches)
    attachesLength = deconstruct_tex(msg_attaches[beginOfFlagValueFullLength:endOfFlagValueFullLength+1].hex())
    endOfFlagRegion = endOfFlagValueFullLength+attachesLength
    beginOfFirstSubFlag = endOfFlagValueFullLength + 1
    endOfFirstSubFlag = endOfTex(beginOfFirstSubFlag, msg_attaches)
    subFlag = str(msg_attaches[beginOfFirstSubFlag:endOfFirstSubFlag+1].hex())
    if subFlag == "08":
        beginOfSubFlagValue = endOfFirstSubFlag + 1
        endOfSubFlagValue = endOfTex(beginOfSubFlagValue, msg_attaches)
        mediaTypeBlob = getSubFlagValue(endOfFirstSubFlag, msg_attaches)
        mediaTypeBlob_str = defineMediaType(mediaTypeBlob)
        if mediaTypeBlob == "01":
            details = decodeSystemMsg(endOfSubFlagValue, msg_attaches, conversationType, messageSender)
        elif mediaTypeBlob == "02":
            details = decodePictureMsg(endOfSubFlagValue, msg_attaches, mediaSource)
        elif mediaTypeBlob == "03":
            details = decodeVideoMsg(endOfSubFlagValue, msg_attaches, mediaSource)
        elif mediaTypeBlob == "04":
            details = decodeAudioMsg(endOfSubFlagValue, msg_attaches, mediaSource)
        elif mediaTypeBlob == "05":
            details = decodeStickerMsg(endOfSubFlagValue, msg_attaches)
        elif mediaTypeBlob == "06":
            details = decodeLinkMsg(endOfSubFlagValue, msg_attaches)
        elif mediaTypeBlob == "08":
            details = decodeCallMsg(endOfSubFlagValue, msg_attaches)
        elif mediaTypeBlob == "0a":
            details = decodeFileMsg(mediaTypeBlob, endOfSubFlagValue, msg_attaches,
                                    conversationType, messageSender, mediaSource)
        elif mediaTypeBlob == "0b":
            details = decodeContactMsg(endOfSubFlagValue, endOfFlagRegion, msg_attaches)
        elif mediaTypeBlob == "0e":
            details = decodeLocationMsg(endOfSubFlagValue, endOfFlagRegion, msg_attaches)
        else:
            details = "Fehlgeschlagene Auswertung aufgrund von unbekanntem Medientyp"
```

Abbildung 4.15: Skriptauswertung des BLOBs *msg_attaches*

Bisher wurde der Aufbau der BLOB-Auswertung beschrieben. Es wurde jedoch noch nicht darauf eingegangen, wie genau die Markerwerte anhand der Interpretationsregeln analysiert werden. Es wurde festgestellt, dass, bis auf wenige Ausnahmen, zwei Arten von Markern auftreten (gilt auch für Submarker und Subsubmarker). Dies ist auch der Grund, weshalb das Skript die Marker mit unbekannter Bedeutung überspringen kann (siehe Abbildung 4.14, 4. Abschnitt).
Bei der ersten Art von Markern („einfacher Marker“) wird sowohl der Marker selbst, als auch sein Wert mittels Tex-Verfahren codiert. Die zweite Art von Marker („komplexe Marker“) besitzen Werte, die nicht mittels Tex-Verfahren codiert sind. Aus diesem Grund befindet sich zwischen dem Marker und seinem eigentlichen Wert eine im Tex-Verfahren codierte Angabe über die Länge des Markerwerts.
Lässt sich ein Marker auf eine der genannten Markerarten zurückführen, sind immer die gleichen Basisschritte für Extraktion des Markerwerts notwendig. Je nach Information müssen die extrahierten Werte zudem weiterverarbeitet werden. Dies bedeutet bei komplexen Markern häufig, dass ausgelesene Werte als Text interpretiert werden müssen (zum Beispiel bei Namen, Links oder Beschreibungen von Konversationen). Bei einfachen Markern zählt zur Verarbeitung meist das Decodieren von Tex-Werten oder das Umrechnen eines solchen in ein Datum. Für ersteres steht die Methode *decon-*

struct_tex(string) und für letzteres die Methode *toUnixTimestamp(tex_Wert)* im Skript zur Verfügung. Weitere Verarbeitungen sind von dem Marker abhängig. Die Abbildungen 4.16 und 4.17 illustrieren die Auswertung der Markerarten am Beispiel der Teilnehmeranzahl und der Beschreibung von Konversationen. Die individuellen Markervariablen sind orange hervorgehoben.

```
elif flag == "e801": #Anzahl der Konversationsteilnehmer
    participantsCount = deconstruct_tex(getFlagValue(endOfFlag, chatsData))
    beginOfFlag = getEndOfFlagValue(endOfFlag, chatsData) + 1
```

Abbildung 4.16: Exemplarische Auswertung von einfachen Markern (mit Tex-codiertem Wert)

```
elif flag == "f201": #Beschreibung der Konversation
    beginOfFlagValueLength = endOfFlag + 1 #Berechnung der Laenge der Information
    endOfFlagValueLength = endOfTex(beginOfFlagValueLength, chatsData)
    descriptionLength = deconstruct_tex(chatsData[beginOfFlagValueLength:endOfFlagValueLength+1].hex())
    beginOfFlagValue = endOfFlagValueLength + 1 #Ermitteln der Information
    endOfFlagValue = beginOfFlagValue + descriptionLength - 1
    conversationDescription = chatsData[beginOfFlagValue:endOfFlagValue+1].hex()
    conversationDescription = bytes.fromhex(conversationDescription).decode('utf-8')
    beginOfFlag = endOfFlagValue + 1
```

Abbildung 4.17: Exemplarische Auswertung von komplexen Markern (mit Längenangabe)

Werden nicht Marker, sondern Submarker oder Subsubmarker ausgewertet, so findet eine Anpassung der verwendeten Methoden auf die Methoden für Marker der entsprechenden Ebene statt. Die Methode *getEndOfSubFlagValue(endOfSubFlag, blob)* wird zum Beispiel statt der Methode *getEndOfFlagValue(endOfSubFlag, blob)* verwendet. Die grundsätzliche Vorgehensweise bei der Extraktion und Interpretation der Werte bleibt bestehen. Abbildung 4.18 demonstriert die Auswertung eines komplexen Submarkers.

```
if subFlag == "0a": #Nutzername
    beginOfSubFlagValueLength = endOfSubFlag + 1#Berechnung der Laenge der Information
    endOfSubFlagValueLength = endOfTex(beginOfSubFlagValueLength, contactsData)
    nameLength = deconstruct_tex(contactsData[beginOfSubFlagValueLength:endOfSubFlagValueLength+1].hex())
    beginOfSubFlagValue = endOfSubFlagValueLength + 1 #Ermitteln der Information
    endOfSubFlagValue = beginOfSubFlagValue + nameLength - 1
    userName = contactsData[beginOfSubFlagValue:endOfSubFlagValue+1].hex()
    userName = bytes.fromhex(userName).decode('utf-8')
    beginOfSubFlag = endOfSubFlagValue + 1
```

Abbildung 4.18: Exemplarische Auswertung eines Submarkers (Name eines Nutzers)

Neben der Interpretation der Datenbankattribute, einschließlich der BLOB-Attribute, wurde in dem Skript auch eine Ausgabe der Auswertungsergebnisse umgesetzt. Als Ausgabeformat wurde *.html* gewählt. Innerhalb des Dokuments verteilen sich die Informationen auf drei verschiedene Tabellen (*Nutzerinformationen, Konversationen, ausgetauschte Nachrichten*). Für die Nutzerinformationen und die Konversationsübersicht existiert in dem Ausgabedokument jeweils eine Tabelle. Die ausgetauschten Nachrichten werden auf mehrere Tabellen aufgeteilt. Jede Konversation erhält eine eigene Tabelle in der die dazugehörigen Nachrichten aufgeführt werden.

Je nach Kategorie des Absenders einer Nachricht erhalten die Zeilen der Konversationstabellen unterschiedliche Farben. Nachrichten, die von einem Kanaladministrator stammen, werden in gelb dargestellt. Nachrichten, die von dem Accountinhaber gesendet wurden, besitzen die Farbe grün und empfangene Nachrichten die Farbe blau. Systemnachrichten, die von TamTam generiert wurden, sind grau hinterlegt.
Zu beachten ist, dass derzeit die Integration der gecachten “Fresco“-Dateien in das Skript fehlt. Dies sollte in Zukunft nachgetragen werden. Die theoretischen Grundlagen wurden dazu bereits in Abschnitt 4.2.2 beschrieben. Bis dahin können die URLs der Bilder verwendet werden, um trotzdem auf diese zuzugreifen.

Das fertiggestellte Skript zur automatischen Auswertung und Aufbereitung („aufbereitung_cache_db.py“) sowie die exemplarische Interpretation einer mit Testdaten befüllten *cache.db* (Aufbereitung_TamTam.html) befindet sich auf der CD, welche dieser Arbeit beigefügt ist. Das Skript wurde unter Linux programmiert. Für eine Verwendung unter Windows sind leichte Veränderungen durchzuführen. Diese beziehen sich auf die Pfadangaben zu den Mediendateien, bei denen die Schrägstriche von „/“ nach „\“ umgewandelt werden müssen.
Zur Verwendung des Skripts muss beim Aufruf der Name der zu analysierenden Datenbank als Parameter angegeben werden. Damit das Skript die versendeten und empfangenen Mediendateien zuordnen kann, muss der Ordner *media/0* aus der Datensicherung exportiert werden (Pfad: *userdata (ExtX)/Root/media/0*). Dieser ist im gleichen Verzeichnis wie das Skript zu hinterlegen.
Zu beachten ist bei dem Export dieses Ordners, dass sich der UFED Physical Analyzer hierzu nicht eignet. Bei manchen Musikdateien verändert dieser beim Speichern den Namen in <dateiname>_1.mp3 und erstellt einen <dateiname>.mp3.EMBEDDED Ordner, welcher Bilder enthält, die zu der Musikdatei gehören. Durch die Veränderung des Dateinamens kann das Skript auf diese nicht mehr zugreifen. Um dieses Problem zu vermeiden, wird empfohlen, einen Export des *media/0*-Verzeichnisses mittels X-Ways Forensics durchzuführen, da so keine Veränderungen der Dateinamen stattfinden.
Falls eine Mediendatei trotzdem nicht verfügbar ist, beispielsweise, weil sie sich nicht mehr im Speicher des Mobiltelefons befindet, ist es möglich, die extrahierten URLs in dem Ausgabedokument zu verwenden, welche aus den BLOBs der *cache.db* stammen. Dies gilt auch für die aktuell nicht integrierten “Fresco“-Dateien.

4.3 Vergleich der Android- und iOS-Datenbanken von TamTam

Der Messenger TamTam speichert auf Android- und iOS-Betriebssystemen mit wenigen Ausnahmen die gleichen Informationen. Allerdings weichen die Datenbanken sowohl in ihrer Anzahl, als auch in ihrer Struktur voneinander ab.
Auf iOS-Geräten legt TamTam zwei *app.db* und fünf *manifest.sqlite* Datenbanken ab (siehe Abbildung 4.19). Alle Datenbanken liegen in unterschiedlichen Ordnern und werden nicht, wie auf Android-Geräten, in einem gemeinsamen Ordner abgespeichert.
Aufgrund ihrer Attribute sowie gespeicherter Testdaten ist ersichtlich, dass die *manifest.sqlite*-Datenbanken Informationen über Mediendateien enthalten. Diese Informationen sind bei Android-Geräten in der *cache.db*-Datenbank enthalten.

Name	Reihenanzahl	↓ Größe (Bytes)	Pfad
manifest.sqlite	52	565248	iPhone/Applications/ru.odnoklassniki.messenger/Library/Application Support/okm_storage.images_15/manifest.sqlite
app.db	772	499712	iPhone/Applications/ru.odnoklassniki.messenger/Library/Application Support/database_15.db/app.db
manifest.sqlite	11	73728	iPhone/Applications/ru.odnoklassniki.messenger/Library/Application Support/okm_storage.avatars_15/manifest.sqlite
app.db	18	49152	iPhone/Applications/ru.odnoklassniki.messenger/Library/Application Support/stat_8.db/app.db
manifest.sqlite	9	28672	iPhone/Applications/ru.odnoklassniki.messenger/Library/Application Support/previewCache_15/manifest.sqlite
manifest.sqlite	0	16384	iPhone/Applications/ru.odnoklassniki.messenger/Library/Application Support/user-chat-background/manifest.sqlite
manifest.sqlite	0	16384	iPhone/Applications/ru.odnoklassniki.messenger/Library/Application Support/okm_storage.gifs_15/manifest.sqlite

Abbildung 4.19: Übersicht über die von TamTam generierten Datenbanken auf iOS-Geräten

Die Datenbank, die auf iOS-Geräten die meisten Nutzerinformationen speichert, ist die *app.db*, welche sich im Pfad *iPhone/Applications/ru.odnoklassniki.messenger/Library/Application Support/database_15.db/app.db* befindet. Ein direkter Vergleich mit der Hauptdatenbank von Android-Geräten, der *cache.db*, zeigt, dass sich diese kaum ähneln.
Die Haupttabelle der Datenbank *app.db* ist die Tabelle *database2*. Diese sammelt eine Vielzahl an Informationen mit Hilfe von fünf Attributen. Zum besseren Verständnis illustriert Abbildung 4.20 eine mit Beispieldaten gefüllte *database2*-Tabelle.
Das Attribut *collection* wird dazu verwendet, die Art der Daten, welche in der Tabelle *database2* gespeichert werden, zu spezifizieren. Es kann sich beispielsweise um einen *contact*, einen *chat* oder *messages* handeln. Anhand dieses Eintrags kann ermittelt werden, wie die weiteren Attribute eines Datensatzes zu interpretieren sind. Durch diese Art und Weise, verschiedene Arten von Einträgen in einer Tabelle zu speichern, werden die Hauptinformationen in einer Tabelle gesammelt und nicht, wie bei der *cache.db* bei Android-Geräten, auf mehrere Tabellen verteilt. Allerdings wird diese Tabelle für einen digitalen Forensiker schnell unübersichtlich, da sie sehr viele Datensätze enthält. Zudem wird trotz dieser nicht darauf verzichtet, in der Datenbank weitere Tabellen anzulegen, die zusätzliche Informationen zu speziellen Datensätzen in der *database2* enthalten.

Tabelle: database2

	rowid	collection	key	data	metadata
	Filtern	Filtern	Filtern	Filtern	Filtern
158	269	contacts	563369934320	BLOB	NULL
159	271	contacts_last_seen	563369934320	BLOB	NULL
160	272	chat_stickers	u563369934320	BLOB	NULL
161	273	chat_bookmarks	0-u563369934320	BLOB	NULL
162	274	messages	1567675210459001	BLOB	NULL
163	275	chats	dialog-with-56336...	BLOB	NULL
164	278	contact_settings	OKMUnsentMessa...	BLOB	NULL
165	280	messages	23596486443-102...	BLOB	NULL
166	282	chat_stickers	u577055617173	BLOB	NULL
167	283	chat_bookmarks	0-u577055617173	BLOB	NULL
168	284	messages	1567675290891002	BLOB	NULL
169	285	chats	dialog-with-57705...	BLOB	NULL
170	289	messages	238207566-10273...	BLOB	NULL
171	291	chat_stickers	u573434900572	BLOB	NULL
172	292	chat_bookmarks	0-u573434900572	BLOB	NULL
173	293	messages	1567675392232003	BLOB	NULL
174	294	chats	dialog-with-57343...	BLOB	NULL
175	298	messages	16481583751-102...	BLOB	NULL
176	300	messages	1567675445400004	BLOB	NULL
177	518	chats	local-1567767041...	BLOB	NULL

Abbildung 4.20: Exemplarische *database2*-Tabelle

Eine der wenigen Gemeinsamkeiten der Android-und iOS-Datenbanken ist die Verwendung von BLOBs um Informationen zu speichern. In der Tabelle *database2* wird die Hauptinformation jedes Datensatzes mit Hilfe des *data*-BLOBs gespeichert. In der *cache.db* bei Android-Geräten enthalten die einzelnen Tabellen ebenfalls BLOBs, welche relevante Informationen bündeln.
Allerdings ist der Aufbau dieser BLOBs bei Android- und iOS-Geräten sehr unterschiedlich. Beide enthalten feste Bytefolgen, die als Marker verwendet werden und ankündigen, welche Informationen auf den Marker folgen. Die BLOBs der *cache.db* speichern ihre Informationen jedoch mit Hilfe von Zahlenmarkern, welche mit dem Tex-Verfahren codiert sind. Im Gegensatz dazu verwendet die *database2* Textmarker. Hierbei handelt es sich um Bytefolgen, welche als ASCII-Text für einen digitalen Forensiker interpretierbar sind. So ist es ihm in den meisten Fällen möglich, direkt abzulesen, welche Information einem Marker zuzuordnen ist.
Betrachtet man beispielsweise einen Kontakteintrag, so steht die Bytefolge *0x6465736372697074696f6e*, welche als ASCII-Text interpretiert *description* bedeutet, für die Beschreibung eines Nutzers. Dennoch ist auch bei diesen BLOBs eine Auswer-

tung aufwändig, da festgestellt werden muss, welche Marker ein BLOB enthalten kann und wie seine Informationen zu interpretieren sind.
Um die beschriebenen Unterschiede bei der Struktur der BLOBs zu verdeutlichen, zeigt Abbildung 4.21 einen *data*-BLOB aus der Tabelle *database2* der *app.db* des gesicherten Mobiltelefons „Birthe Wie“. Dieser BLOB enthält Informationen über den Kontakt „Finja Wie“ und ist inhaltlich vergleichbar mit dem BLOB aus Abbildung 4.5, welcher aus der Android-Datenbank *cache.db* stammt.

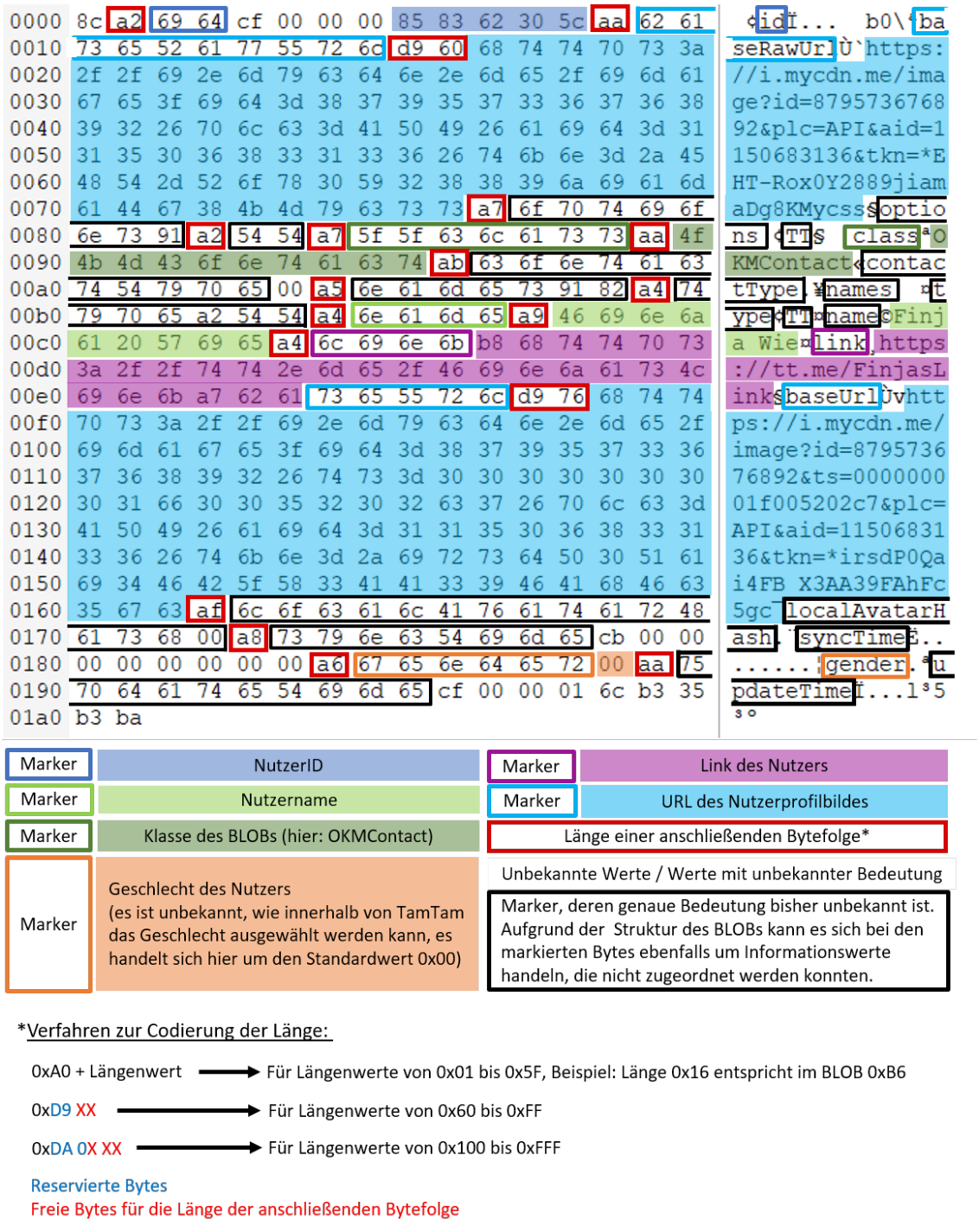


Abbildung 4.21: Auswertung eines *data*-BLOBs der Tabelle *database2* der *app.db* für Kontakte

Eine weitere Gemeinsamkeit der Datenbanken *cache.db* und *app.db* ist die Verwendung der gleichen globalen IDs. Diese IDs beziehen sich auf die Nutzer sowie die Konversationen und Nachrichten. In der iOS-Datenbank werden die IDs als Wert des Attributs *key* der Tabelle *database2* verwendet. Bei der Android-Datenbank *cache.db* werden die globalen IDs als *ctt_server_id*, *cht_server_id* und *msg_server_id* bezeichnet und jeweils in den Tabellen *contacts*, *chats* und *messages* aufgeführt.
Weitere Werte können ebenfalls auf Geräten beider Betriebssysteme gespeichert werden, diese liegen jedoch häufig in einer abweichenden Struktur vor. Hierzu zählt beispielsweise der Text von Nachrichten, welcher sowohl auf Android- als auch iOS-Geräten empfangen wurde. Die Datenbank *cache.db* besitzt auf Android-Geräten für den Nachrichteninhalt ein eigenes Attribut (Tabelle *messages*, Attribut *msg_text*). Auf iOS-Geräten wird der Nachrichtentext innerhalb eines BLOBs mit weiteren Informationen gespeichert (Datenbank *app.db*, Tabelle *database2*, Attribut *data*).
Abschließend kann festgehalten werden, dass ein Vergleich der Messengerdaten zeigt, dass mehr Unterschiede als Gemeinsamkeiten zwischen den Datenbanken auf Android- und iOS-Geräten bestehen. Dies verdeutlicht den Bedarf nach einer getrennten Auswertung von Daten eines Messengers bei verschiedenen Betriebssystemen.

5 Diskussion

Intention dieses Kapitels ist es, das präsentierte Verfahren sowie die Qualität der Informationen, die mit seiner Hilfe gewonnen werden können, zu bewerten. Zudem wird das Vorgehen mit ähnlichen Verfahren aus der Literatur verglichen und ein Ausblick auf die Optimierungsmöglichkeiten für die Auswertung und Aufbereitung von Messengerdaten gegeben.

5.1 Bewertung

Die vorangegangene exemplarische Auswertung der Datenbanken des Messengers TamTam in der Version 2.6.0 auf Android-Geräten hat gezeigt, dass das vorgestellte Verfahren geeignet ist, um eine Vielzahl an Informationen über die Nutzung eines Messengers zu gewinnen. Diese Informationen sind vielfältig und beziehen sich auf den Accountinhaber, seine Kommunikationspartner, sowie geführte Konversationen.
Eine positive Eigenschaft des vorgestellten Verfahrens ist es, dass die Interpretationshypothesen, mit deren Hilfe die Informationen erzeugt werden, stetig anhand von Testdaten kontrolliert und angepasst werden. Auch bei der Auswertung der *cache.db* von TamTam wurden die Hypothesen mehrfach überprüft und in Folge dessen sichergestellt, dass valide Informationen erzeugt werden.
Zum aktuellen Zeitpunkt sind keine Informationen bekannt, welche die finalen Hypothesen widerlegen. Es kann demnach mit hoher Wahrscheinlichkeit davon ausgegangen werden, dass diese korrekt sind und dass das vorgestellte Verfahren funktionstüchtig ist.
Seit der Version 3.2.1.14548 ist in Magnet Axiom (Fa. Magnet Forensics) eine Auswertung und Aufbereitung von TamTam-Daten der Version 2.5.0 integriert. In dieser Arbeit wurde TamTam in der Version 2.6.0 betrachtet, Magnet Axiom interpretiert Daten dieser Version ebenfalls. Für einen Vergleich der Interpretationen wurden die gleichen Daten von TamTam (v.2.6.0) zum einen mittels Magnet Axion und zum anderen mit Hilfe des erstellten Skripts untersucht.
Die Informationen, die die Aufbereitung durch Magnet Axiom zur Verfügung stellt, stimmen mit denen des Skripts überein. Dies kann als Bestätigung gewertet werden, dass das Skript korrekt arbeitet und valide Daten erzeugt. Eine Gegenüberstellung der Interpretationen für die bekannten TamTam-Nutzer sowie für die Konversation „Chat-Titel“ ist im Anhang mittels Screenshots dargestellt (siehe Kapitel B.4).
Ein direkter Vergleich zeigt, dass das Skript eine wesentlich detailliertere Interpretation der Daten erzeugt als Magnet Axiom. Bei Magnet Axiom fehlen viele Informationen, welche das für diese Arbeit entwickelte Skript bereitstellt. Betroffen sind vor allem Nachrichten, die keine einfachen Textnachrichten darstellen wie zum Beispiel Mediennachrichten, Anrufe oder Systemnachrichten. Für diese gibt Magnet Axiom lediglich an, wer

diese Nachricht wann gesendet oder empfangen hat. Der Text einer solchen Nachricht lautet: „Kein Nachrichteninhalt anzuzeigen“.
Aufgrund der Tatsache, dass sich das Verfahren bei einem Messenger mit einer sehr komplexen Datenbank und vielfältigen Funktionen beweisen konnte, ist davon auszugehen, dass andere Messenger ebenfalls mit Hilfe dieses Verfahrens ausgewertet werden können. Im Falle eines Messengers mit weniger komplexen Daten ist zu erwarten, dass die Auswertung weniger Zeit in Anspruch nimmt als die von TamTam, da bei dieser zum Beispiel das aufwändige Analysieren von BLOBs entfällt.
Die Erwartungen an diese Arbeit, dass ein Verfahren vorgestellt wird, welches einem digitalen Forensiker als Leitfaden dienen kann und ihn befähigt die Daten eines beliebigen Messengerdiensts auszuwerten und aufzubereiten wurden auf Grund der angeführten Gesichtspunkte erfüllt.
Es gibt jedoch auch Aspekte in denen das Verfahren kritisiert werden muss. Ein Nachteil ist zum Beispiel, dass dieses lediglich sehr allgemeine Arbeitsschritte beinhaltet. Dies führt dazu, dass vieles dem verantwortlichen digitalen Forensiker und seinem Sachverstand sowie seinen Erfahrungen überlassen ist. Dieser Umstand ist der Tatsache geschuldet, dass die zu untersuchenden Messenger in ihren Funktionen sehr unterschiedlich sind und ihre gespeicherten Informationen, sowie die Elemente, die zur Dokumentation dieser genutzt werden, sehr stark voneinander abweichen können.
Um einem digitalen Forensiker einige Hinweise bei der Auswertung von Messengerdaten zu liefern, wurden die Verfahrensschritte anhand der Auswertung von TamTam in dieser Arbeit in einigen Aspekten konkretisiert. Allerdings ist es nicht gewährleistet, dass ein Experiment, welches sich für TamTam als sinnvoll herausgestellt hat, auch zwangsläufig auf einen anderen Messenger übertragbar ist. In diesem Fall muss der digitale Forensiker selbst entscheiden, welches Vorgehen für seinen Messenger geeignet ist.
Ein weiterer Nachteil des präsentierten Verfahrens sind die Grenzen, denen es unterliegt. Seine Basis ist das Ableiten von Informationen aus kontrolliert erzeugten Testdaten mittels Reverse Engineering. Es ist zwar möglich, eine Vielzahl an Informationen zu gewinnen, manche Erkenntnisse können allerdings erst durch Betrachtung des Messenger Quelltexts oder durch Gemeinsamkeiten zur Speicherweise bei anderen Messengern gewonnen werden. Bei der Auswertung von TamTam wurde diese Grenze bei der Zuordnung von gesendeten und empfangenen Bildern erreicht, welche auf dem Mobiltelefon abgelegt werden. Die Speicherung, welche aus der „Fresco“-Bibliothek stammt, konnte erst mit Hilfe der Parallelen zum Facebook Messenger und der Auswertung des Quelltexts ermittelt werden.
Grundsätzlich kann festgehalten werden, dass die TamTam-Auswertung dieser Arbeit erfolgreich und ausführlich durchgeführt wurde. Es können sämtliche Inhalte der Basisfunktionen, sowie tiefer greifende Informationen zur Verfügung gestellt werden.
Allerdings ist die Auswertung trotz ihrer Ausführlichkeit unvollständig. Funktionen wie das Löschen von Nachrichten und Konversationen, das Anpinnen und der Entwurf einer Nachricht wurden außer acht gelassen und sollten bei Bedarf nachgetragen werden. Dies ist mit dem vorgestellten Verfahren unproblematisch, da es jederzeit das Fortset-

zen einer Auswertung ermöglicht.
Neben der Unvollständigkeit ist ein weiterer Kritikpunkt das Skript, welches zur Auswertung und Aufbereitung programmiert wurde. Da das Hauptaugenmerk dieser Arbeit jedoch nicht auf der Programmierung, sondern auf der Analyse liegt, soll es lediglich zur Demonstration dienen und ist für seine Zwecke ausreichend. Es ist in der Lage, einen großen Datenbestand korrekt auszuwerten und mit Ausnahme der Dateien, welche mit “Fresco“ gecacht wurden, Mediendateien den entsprechenden Nachrichten zuzuordnen. Das theoretische Grundprinzip der Zuordnung von gecachten “Fresco“-Dateien wurde bereits in Kapitel 4.2.2 beschrieben, eine Umsetzung im Skript fehlt derzeit allerdings und sollte nachgetragen werden.
Eine Überarbeitung des Skripts und eine objektorientierte Lösung mit Klassen wie Kontakt, Nachricht oder Konversation könnte dazu beitragen, das Skript übersichtlicher und effektiver zu gestalten. Weiterhin wäre eine Überarbeitung des Ausgabeformats sinnvoll, da die bestehende HTML-Ausgabe aufgrund der Tabellengrößen unübersichtlich ist und schlecht ausgedruckt werden kann. Dies wird jedoch von vielen Personen bei der Untersuchung des Kommunikationsverhaltens präferiert.

5.2 Vergleich mit Verfahren aus der Literatur

In der Vergangenheit wurden bereits verschiedene Messenger für forensische Zwecke untersucht, darunter unter anderem WhatsApp [21], Telegram [23] [58], IMO [22], Kik [59] [60] und LINE [61].
In allen betrachteten Publikationen wird darauf eingegangen, welche Ausgangssituationen geschaffen wurden, um die zu untersuchenden Messengerdaten zu erzeugen. Hierzu zählen die verwendeten Mobiltelefone, deren Betriebssystem und die entsprechende Messengerversion. Im Anschluss wird beschrieben, wie auf die Messengerdaten zugegriffen wird. Letzteres ist davon abhängig, ob Offline- oder Online-Forensik betrieben werden soll.
Bei der Offline-Forensik wird der Speicher der verwendeten Mobiltelefone untersucht, wozu eine Datensicherung mit einer geeigneten Sicherungsmethode durchgeführt wird. Bei Android-Geräten involviert der Zugriff und die Sicherung von Messengerdaten häufig, dass die Geräte zunächst gerootet werden. In einigen Fällen werden keine echten Mobiltelefone ausgelesen, sondern emulierte Geräte verwendet [23] [21]. Der Grundgedanke, dass auf ihre abgespeicherten Elemente zugegriffen wird, bleibt dabei bestehen.
Bei der Online-Forensik wird der erzeugte Netzwerkverkehr bei dem Gebrauch des Messengers untersucht. Da in dieser Arbeit ein Verfahren vorgestellt wird, welches abgespeicherte Messengerdaten analysiert, wird auf die Vorgehensweisen der Online-Forensik nicht eingegangen.
Während die Vorbereitungen, welche für die Auswertung der Messengerdaten getroffen wurden, in allen betrachteten Publikationen enthalten sind, wird auf das Vorgehen bei der Analyse meist recht oberflächlich eingegangen. Im Mittelpunkt der meisten Veröffentlichungen steht vielmehr die Präsentation der Untersuchungsergebnisse.

Häufig wird die Vorgehensweise lediglich als die Durchführung gezielter Experimente mit anschließender Analyse oder aber als Vergleich von Speicherelementen vor und nach durchgeführten Aktionen beschrieben. Auf weitere Details zur Methodik bei der Auswertung oder zu den durchgeführten Aktionen wird nicht eingegangen. Lediglich eine Auflistung der Untersuchungsaspekte wie die Nutzerinformationen, der Austausch von Nachrichten oder die Rekonstruktion gelöschter Kommunikationsinhalte findet in den meisten Fällen statt. [21] [58] [59] [60] [61]
Zwei Paper, bei denen konkreter auf das Verfahren zur Auswertung von Messengerdaten eingegangen wird, sind “Forensic Analysis of Telegram Messenger on Android smartphones“ von Anglano et al. [23] und “Forensics study of IMO call and chat app“ von Sudozai et al. [22]. Bei Anglano et al. [23] wird dieses Verfahren am Beispiel des Messengers Telegram und bei Sudozai et al. [22] exemplarisch am Messengerdienst IMO demonstriert.
Die genannten Verfahren weisen starke Parallelen zu der Vorgehensweise dieser Arbeit auf und enthalten ebenfalls Diagramme mit Anweisungen zur Auswertung, die von digitalen Forensikern verfolgt werden können. Die Verfahren aus den genannten Publikationen sowie das Vorgehen dieser Arbeit verwenden das Reverse Engineering als Basis. Aus diesem Grund entsprechen sich viele ihrer Arbeitsschritte. Der Hauptunterschied der Verfahren ist die Strukturierung des Ablaufs, was zu einer abweichenden Darstellung bei den Diagrammen führt.
Das in dieser Arbeit vorgestellte Verfahren ist in drei Phasen eingeteilt (Vorbereitungsphase, Auswertungsphase und Aufbereitungsphase) und umfasst einen vollständigen Prozess um Messengerdaten für Ermittlungs- und Strafverfahren nutzbar zu machen. Die anderen beiden Verfahren beziehen sich hauptsächlich auf die Auswertungs- beziehungsweise Analysephase. In diese integrieren die Autoren einige Schritte, die in dem Verfahren dieser Arbeit zu der Vorbereitungsphase zählen (Schritte: „Recherche + Funktionen des Messengers testen“, „Übersicht über die vom Messenger erzeugten Daten gewinnen“, „Installation des Messengers“ und „Messengeraccount einrichten und erneute Datensicherung“). Eine Auflistung und Beschreibung der Testgeräte, welche bei Anglano et al. [23] verwendet wurden, findet vor dem ersten Schritt der Methodik statt und wird dieser nicht als Verfahrensschritt zugeordnet. Bei Sudozai et al. [22] werden die Geräte und die Anfertigung ihrer Sicherung nach der Analyse der möglichen Messengerfunktionen und vor der Erwähnung der Experimente zur Auswertung beschrieben. Es findet jedoch weder bei der Schilderung aller Arbeitsschritte, noch innerhalb des Diagramms eine Einordnung der Vorbereitung der Testgeräte in die Methodik statt.
Aufgrund der genannten Arbeitsschritte fehlt die Vorbereitungsphase bei den betrachteten Verfahren nicht vollständig, sie wird lediglich nicht detailliert genug aufgeführt um als eigener Abschnitt zu gelten.
Bei der Aufbereitungsphase ist dies anders. Diese ist weder explizit noch implizit in die Vorgehen integriert und stellt ein Alleinstellungsmerkmal des Verfahrens dieser Arbeit dar.
Grundsätzlich wurde beobachtet, dass die Publikation von Sudozai et al. [22] bei der Beschreibung der Methodik sehr oberflächlich bleibt und lediglich auf Basisschritte ein-

geht. Es wird in dem Paper allerdings auch nicht der Anspruch erhoben, als Leitfaden für die Auswertung von Messengerdaten zu dienen. Aus diesem Grund findet sich innerhalb des Diagramms zum Verfahren auch der Name des Messengerdienstes, welcher analysiert werden soll, statt einer Verallgemeinerung des Messengernamens.
Im Gegensatz zu Sudozai et al. [22] präsentieren Anglano et al. [23] ihr Verfahren nicht nur spezifisch für Telegram, sondern stellen in Aussicht, dass das Verfahren auch für andere Messenger genutzt werden kann. Dennoch ist auch bei dieser Publikation ersichtlich, dass es bei der Erstellung des Papers primär um die Auswertung von Telegram ging. Die Beschreibung des Verfahrens nimmt nur einen geringen Teil der Veröffentlichung ein und die Allgemeingültigkeit wird als Nebensatz erwähnt. Dies ist in dieser Arbeit anders. Es geht primär um die Auseinandersetzung mit Messengerdaten und die Auswertung von TamTam dient lediglich zur Demonstration. Ein positiver Aspekt der Publikation von Anglano et al. [23] ist die Integration der Quelltextanalyse in das Verfahren. Dies ist ein Schritt, welcher in das vorgestellte Verfahren deutlicher hätte aufgenommen werden können. Bisher wird die Quelltextanalyse lediglich im Rahmen der Beschreibung des Verfahrens aufgeführt und wurde nicht in das erstellte Diagramm aufgenommen.
Aufgrund der vielen Gemeinsamkeiten der drei verglichenen Verfahren kann abschließend festgestellt werden, dass sich diese die gleiche Basis teilen. Das Vorgehen dieser Arbeit umfasst allerdings Aspekte, die über die Auswertungsschritte der anderen beiden Verfahren hinausgehen und kann als eine Erweiterung und Verbesserung dieser betrachtet werden. Besonders auf den vollständigen Prozess mit Aufbereitung wird von den Vergleichsverfahren nicht eingegangen.

5.3 Optimierungsmöglichkeiten und Ausblick

Bei der Bewertung des vorgestellten Verfahrens konnte gezeigt werden, dass dieses trotz seiner Nachteile sehr gut dazu geeignet ist, Messengerdaten auszuwerten und aufzubereiten. Allerdings konnte ein grundlegendes Problem bei der Interpretation von Messengerdaten nicht gelöst werden. Die Auswertung und Aufbereitung von Messengerdaten muss für jeden Messengerdienst und seine Betriebssysteme individuell durchgeführt werden und ist dann aufgrund von Versionsupdates möglicherweise schnell überholt. Bei zeitkritischen Ermittlungs- und Strafverfahren ist der Aufwand, der für eine neue Auswertung und Aufbereitung erforderlich ist, möglichst gering zu halten. Aus diesem Grund ist ein potenzieller Gegenstand für zukünftige Forschungen die Entwicklung eines allgemeingültigen Tools, welches die Auswertung und / oder Aufbereitung aller Messenger übernimmt.
Bei der Auswertung der Messengerdaten ist dies aufgrund der Herausforderungen, welche in dieser Arbeit präsentiert wurden (siehe Abschnitt 2.2.1) äußerst komplex und möglicherweise nicht umsetzbar, da die Daten der Messenger zu individuell sind.
Die Aufbereitung hat jedoch sehr großes Potenzial für eine Verallgemeinerung, da diese trotz unterschiedlicher Messengerfunktionen und Eigenschaften gleich abläuft und ähnlich aussehende Ausgaben erzeugt.

Um dieses Vorgehen umzusetzen, müsste gewährleistet werden, dass nach einer Auswertung der Messengerdaten ein Export der entstandenen Informationen in ein festgelegtes „neutrales“ Format stattfindet. Dieses „neutrale“ Format kann dann von einem Universaltool zur Aufbereitung verarbeitet werden. Dies spart nicht nur Zeit und Arbeit, da die Programmierung der Aufbereitung für einen digitalen Forensiker entfällt, es steigert auch die Qualität der Aufbereitung und des daraus resultierende Ausgabedokuments. Wird dieses lediglich ein einziges Mal definiert und programmiert, statt von jedem digitalen Forensiker einzeln, so kann sich in diesem Fall mehr Zeit gelassen werden, um dies umzusetzen und zu gestalten. Es könnte beispielsweise eine Messengeransicht nachgebildet und verschiedene Ausgabeformate wie *HTML*, *XML* oder *PDF* zur Verfügung gestellt werden.
Eine Herausforderung bei der Verwendung eines solchen Universaltools zur Aufbereitung wäre allerdings, dass dieses lediglich vordefinierte Funktionen und Messengerdaten aufbereiten kann. Dies umfasst höchstwahrscheinlich die populären Basisfunktionen aller Messenger wie das Senden von Text-, Sprach- und Mediennachrichten oder das Tätigen von Anrufen, jedoch keine Sonderfunktionen, die manche Messenger bieten. Ein flexibler Umgang mit diesen Sonderfunktionen muss bei der Programmierung dieses Universaltools beachtet und ermöglicht werden.
Die folgende Abbildung illustriert sowohl die herkömmliche Herangehensweise, als auch einen Prozess mit einem Universaltool zur Aufbereitung von Messengerdaten.

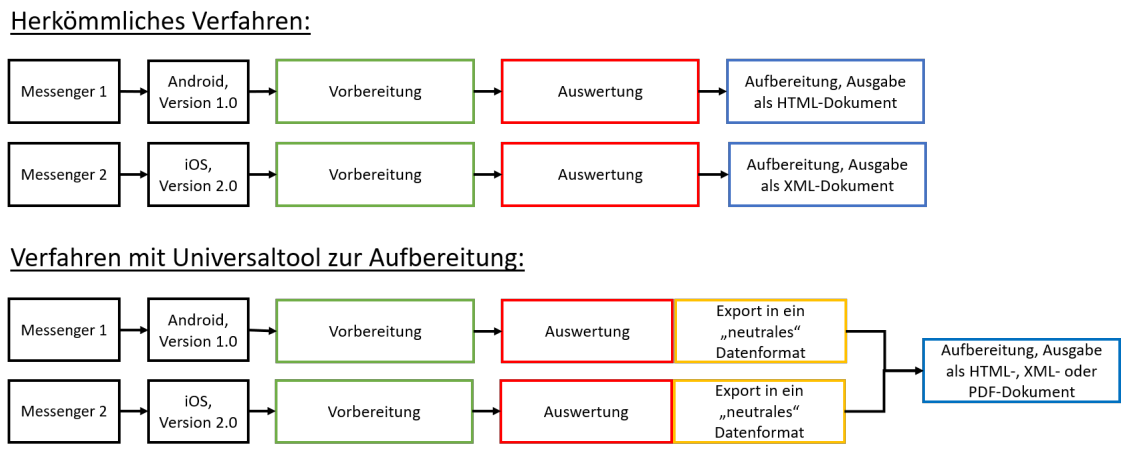


Abbildung 5.1: Gegenüberstellung des Prozessablaufs mit und ohne Universaltool

Auch die Auswertung von TamTam kann in zukünftiger Forschung weiter vorangetrieben werden. Seine Messengerdaten auf Android-Geräten wurden in dieser Arbeit exemplarisch ausgewertet und aufbereitet. Trotz einer ausführlichen Auseinandersetzung mit den Messengerdaten wurden einige Aspekte von TamTam bei der Auswertung nicht betrachtet und sollten ergänzt werden. Dies betrifft vor allem gelöschte Nachrichten und Konversationen. Auch angepinnte Nachrichten oder Nachrichten, die als Entwurf gespeichert werden, wurden bisher nicht untersucht. Zudem bieten Mediennachrichten das Potenzial für weitere Untersuchungen. Es ist bisher nicht bekannt, welche weiteren

Nachrichtentypen existieren. Auch wurden die *msg_attaches*-BLOBs der Mediennachrichten bisher nicht vollständig ausgewertet. Wird die Analyse dieser BLOBs vorangetrieben, ist es möglich, dass dort weitere Informationen gespeichert sind, die für spezielle Fragestellungen relevant sein können.
Des Weiteren wurde bereits bei der vorangegangenen Bewertung darauf eingegangen, dass das Skript zur Aufbereitung von TamTam Potenzial für Optimierungen bietet. Dies umfasst ein Überarbeitung des Ausgabeformats und die Integration von “Fresco“-Dateien.
Neben einer weiteren Auswertung und Verbesserung der Aufbereitung von TamTam auf Android-Geräten sollte eine Analyse von TamTam auf iOS-Geräten stattfinden, um auch diese Daten für Ermittlungs- und Strafverfahren zugänglich zu machen. Die Untersuchungen, welche im Rahmen dieser Arbeit stattfanden, dienten in erster Linie dazu, die TamTam-Daten von Android- und iOS-Geräten miteinander zu vergleichen und war von oberflächlicher Natur.

6 Fazit

In Ermittlungs- und Strafverfahren sind in vielen Fällen die Daten von Messengern auf Mobiltelefonen von Interesse. Die Mobiltelefone können von Tätern, Opfern oder Zeugen stammen und Informationen enthalten, die mit Verbrechen in Verbindung stehen.
Aufgrund ihrer Relevanz wurde sich im Rahmen dieser Arbeit mit der Auswertung und Aufbereitung von Messengerdaten beschäftigt. Bei der Betrachtung der dazu zur Verfügung stehenden Möglichkeiten konnte gezeigt werden, dass digitale Forensiker in der Lage sein müssen, Messengerdaten händisch zu analysieren. Die Darlegung der Herausforderungen bei der Auswertung und Aufbereitung von Messengerdaten machte zudem deutlich, dass ein effizientes Verfahren notwendig ist, um Messengerdaten mit minimalem Arbeits- und Zeitaufwand zu untersuchen. Aus diesem Grund wurde in dieser Arbeit ein Verfahren vorgestellt, welches einem digitalen Forensiker einen Leitfaden bei der Auswertung und Aufbereitung von Messengerdaten bietet.
Eine Demonstration des Verfahrens fand anhand der Daten des Messengers TamTam statt. Angesichts der starken Abweichungen der Speicherelemente auf Android- und iOS-Geräten (Android: TamTam v2.6.0, iOS: TamTam v2.6.2) wurde darauf verzichtet, die Datenbanken von beiden Betriebssystemen zu untersuchen. Der Fokus wurde auf die Untersuchung von TamTam auf Android-Geräten gelegt. Zum Zweck der automatischen Analyse und Aufbereitung wurde ein Pythonskript erstellt, welches die zuvor gewonnenen Erkenntnisse umsetzt. Mit Hilfe dieses Skripts ist es möglich, auch größere Datenbestände zu untersuchen und die erstellten Interpretationsregeln an einem großen Datensatz zu überprüfen. Die Ergebnisse der Untersuchungen beweisen, dass das vorgestellte Verfahren dazu geeignet ist, um Messengerdaten zu interpretieren und dabei verwertbare Resultate zu erzielen.
Kritisieren lässt sich an diesem Verfahren, dass vieles dem verantwortlichen digitalen Forensiker obliegt. Zudem besitzt das Verfahren Grenzen, welche auch bei der exemplarischen Auswertung von TamTam eine Rolle gespielt haben.
Die Daten des Messengers TamTam (v2.6.0) auf Android-Geräten wurden in dieser Arbeit ausführlich, jedoch nicht vollständig untersucht. Eine vollständige Analyse ist aufgrund der vielfältigen und komplexen Funktionen von TamTam lediglich mit einem extrem hohen Arbeitsaufwand möglich. Zudem wird für Ermittlungs- und Strafverfahren häufig lediglich die Analyse von Grundfunktionen, wie das Verschicken oder Empfangen von Text- oder Mediennachrichten sowie das Tätigen von Anrufen, benötigt. Aus diesem Grund sind die Erkenntnisse, die in dieser Arbeit gewonnen wurden, für die meisten Fragestellungen ausreichend. Bezieht sich in Zukunft ein Untersuchungsauftrag auf eine Funktion, welche durch die Auswertung bisher nicht unterstützt wird, so ist es mit Hilfe des vorgestellten Verfahrens jederzeit möglich, diese zu ergänzen.

Aspekte für weitere Forschungen im Bereich der Untersuchung von Messengerdaten könnte der Einsatz eines Universaltools sein, welches zum Beispiel die Aufbereitung von Messengerdaten unabhängig von dem Messenger oder seinem Betriebssystem übernimmt. Des Weiteren wurde in dieser Arbeit TamTam auf iOS-Geräten lediglich oberflächlich betrachtet. Eine Auswertung wurde nicht durchgeführt. Derzeit ist eine Analyse und Aufbereitung einzig durch Magnet Axiom möglich. Da diese allerdings unvollständig ist, können Daten von TamTam auf iOS-Geräten bei Ermittlungs- und Strafverfahren noch immer nicht verwendet werden. Um diesen Umstand zu ändern ist eine händische Auswertung notwendig. Hierzu kann das in dieser Arbeit vorgestellte Verfahren verwendet werden, welches sich bereits bei der Auswertung von TamTam auf Android-Geräten bewiesen hat.

Literaturverzeichnis

[1] Laura Carius. *Neun von zehn Internetnutzern verwenden Messenger*, 02.05.2018. Zu finden unter: https://www.bitkom.org/Presse/Presseinformation/Neun-von-zehn-Internetnutzern-verwenden-Messenger.html (abgerufen am 19.08.2019).

[2] L. Rabe. *Wie häufig nutzen Sie Messenger-Dienste?*, 22.07.2019. Zu finden unter: https://de.statista.com/statistik/daten/studie/1030457/umfrage/haeufigkeit-der-nutzung-von-messenger-diensten-in-deutschland/ (abgerufen am 19.08.2019).

[3] E. J. Chikofsky and J. H. Cross. Reverse engineering and design recovery: a taxonomy. *IEEE Software*, 7(1):13–17, Jan 1990.

[4] Michael Nelson. A survey of reverse engineering and program comprehension. 06 1996.

[5] *Polls: Bringing Choice to Communities*, 22.12.2018. Zu finden unter: https://telegram.org/blog/polls (abgerufen am 20.08.2019).

[6] *Wie erstelle ich eine Umfrage? (Threema)*. Zu finden unter: https://threema.ch/de/faq/poll (abgerufen am 20.08.2019).

[7] *KakaoTalk [Funktionen]*. Zu finden unter: https://www.kakaocorp.com/service/KakaoTalk?lang=en (abgerufen am 20.08.2019).

[8] *KakaoTalk: Free Calls & Text [Google Play]*. Zu finden unter: https://play.google.com/store/apps/details?id=com.kakao.talk\&hl=de (abgerufen am 20.08.2019).

[9] Gandeva Satrya, Philip Daely, and Soo Shin. Android forensics analysis: Private chat on social messenger. 07 2016.

[10] *Wie sicher ist Telegram?* Zu finden unter: https://telegram.org/faq/de\#f-wie-sicher-ist-telegram (abgerufen am 20.08.2019).

[11] *What is a Secret Chat [KakaoTalk]*. Zu finden unter: https://cs.kakao.com/helps?articleId=1073183324&service=8&category=5&device=2&locale=en (abgerufen am 11.10.2019).

[12] *Wie funktioniert die Selbstzerstörung der Nachrichten? [Telegram]*. Zu fin-

den unter: https://telegram.org/faq/de\#f-wie-funktioniert-die-selbstzerstrung-der-nachrichten (abgerufen am 20.08.2019).

[13] *New "Hidden Chat" Feature Released, Enables Sending of Time-Limited Messages*, 22.07.2014. Zu finden unter: http://official-blog.line.me/en/archives/1006361166.html (abgerufen am 20.08.2019).

[14] *LINE Encryption Report*. Zu finden unter: https://linecorp.com/en/security/encryption_report (abgerufen am 20.08.2019).

[15] *Was unterscheidet Threema von anderen populären Messengern mit Verschlüsselung?* Zu finden unter: https://threema.ch/de/faq/crypto_differences (abgerufen am 20.08.2019).

[16] *Ende-zu-Ende-Verschlüsselung [bei WhatsApp]*. Zu finden unter: https://faq.whatsapp.com/de/android/28030015/ (abgerufen am 20.08.2019).

[17] *Messenger-App Wire kurz vorgestellt*. Zu finden unter: https://mobilsicher.de/apps-kurz-vorgestellt/messenger-app-wire-kurz-vorgestellt (abgerufen am 20.08.2019).

[18] *BLOB (binary large object)*. Zu finden unter: https://www.itwissen.info/BLOB-binary-large-object.html (abgerufen am 04.11.2019).

[19] Soufiane Tahiri. *Mastering mobile forensics*. Packt Publishing Ltd, 2016.

[20] *Seitenmanager [Facebook]*. Zu finden unter: https://play.google.com/store/apps/details?id=com.facebook.pages.app&hl=de (abgerufen am 05.11.2019).

[21] Cosimo Anglano. Forensic analysis of whatsapp messenger on android smartphones. *Digital Investigation*, 11, 05 2014.

[22] K. Sudozai, Shahzad Saleem, William Buchanan, Nisar Habib, and Haleemah Zia. Forensics study of imo call and chat app. *Digital Investigation*, 04 2018.

[23] Cosimo Anglano, Massimo Canonico, and Marco Guazzone. Forensic analysis of telegram messenger on android smartphones. *Digital Investigation*, 09 2017.

[24] *UFED Ultimate [UFED Physical Analyzer, Fa. Cellebrite]*. Zu finden unter: https://www.cellebrite.com/en/home/ (abgerufen am 07.07.2019).

[25] *Magnet AXIOM [Fa. Magnet Forensics]*. Zu finden unter: https://

www.magnetforensics.com/magnet-axiom-herunterladen/ (abgerufen am 07.07.2019).

[26] (Vorwort von E. Chikofsky) Eilam, E. *Reversing: Secrets of Reverse Engineering*. Wiley, 2005.

[27] *TamTam is fast and secure messenger available on any platform*. Zu finden unter: https://download.tamtam.chat/en/latest/ (abgerufen am 21.06.2019).

[28] Mikhail Zelensky. *some messenger called tamtam is trying to replace telegram in russia what the heck is it*, 17.04.2018. Zu finden unter: https://meduza.io/en/feature/2018/04/17/some-messenger-called-tamtam-is-trying-to-replace-telegram-in-russia-what-the-heck-is-it (abgerufen am 21.0.2019).

[29] *TamTam is fast and secure messenger available on any platform.* Zu finden unter: https://about.tamtam.chat/en/ (abgerufen am 21.06.2019).

[30] *TamTam Messenger - free chats & video calls [Google Play]*. Zu finden unter: https://play.google.com/store/apps/details?id=ru.ok.messages\&hl=de (abgerufen am 19.06.2019).

[31] *TamTam Messenger [Apple AppStore]*. Zu finden unter: https://apps.apple.com/de/app/tamtam-messenger/id1095345669 (abgerufen am 19.06.2019).

[32] *Instant messaging [Messenger mail.ru Group - TamTam]*. Zu finden unter: https://corp.mail.ru/en/company/messengers/ (abgerufen am 21.06.2019).

[33] *2018 year in review (TamTam*, 31.12.2018. Zu finden unter: https://blog.tamtam.chat/en/2018/12/31/ (abgerufen am 24.06.2019).

[34] *Mail.Ru launches TamTam messenger for Android and iOS*, 25.05.2017. Zu finden unter: https://www.uzdaily.uz/en/post/39526 (abgerufen am 24.06.2019).

[35] Yuri Buyanov. *As we did a new messenger / Blog company Mail.Ru Group / Habrahabr*, 24.07.2017. Zu finden unter: http://www.techort.com/as-we-did-a-new-messenger-blog-company-mail-ru-group-habrahabr/ (abgerufen am 24.06.2019).

[36] *The story of TamTam Bot API [Erwähnung von Yuriy Buyanov]*, 29.03.2019. Zu finden unter: https://blog.tamtam.chat/en/2019/03/29/ (abgerufen am 24.06.2019).

[37] *Forbes ranking features four Russian Internet unicorns*, 25.02.2019.

Zu finden unter: https://www.ewdn.com/2019/02/25/forbes-ranking-features-four-russian-internet-unicorns/ (abgerufen am 21.06.2019).

[38] *Mail.Ru*. Zu finden unter: https://en.wikipedia.org/wiki/Mail.Ru (abgerufen am 21.06.2019).

[39] *Our history (mail.ru Group)*. Zu finden unter: https://corp.mail.ru/en/company/timeline/ (abgerufen am 21.06.2019).

[40] *Our Products [mail.ru group]*. Zu finden unter: https://corp.mail.ru/en/company/portal/ (abgerufen am 21.06.2019).

[41] Briallyn Smith. *Top 8 Russian Social Networks*, 15.08.2016. Zu finden unter: https://www.makeuseof.com/tag/top-8-russian-social-networks-makes-great/ (abgerufen am 21.06.2019).

[42] *Leading active social media platforms in Russia in 2018*. Zu finden unter: https://www.statista.com/statistics/867549/top-active-social-media-platforms-in-russia/ (abgerufen am 21.06.2019).

[43] *Top Sites in Russia*. Zu finden unter: https://www.alexa.com/topsites/countries/RU (abgerufen am 20.06.2019).

[44] *Top Sites in Germany*. Zu finden unter: https://www.alexa.com/topsites/countries/DE (abgerufen am 20.06.2019).

[45] *TamTam Chats and Channels Regulations*. Zu finden unter: https://about.tamtam.chat/en/regulations/ (abgerufen am 22.06.2019).

[46] *Stay connected with your friends in one messenger [TamTam Sign In]*. Zu finden unter: https://tamtam.chat/ (abgerufen am 06.07.2019).

[47] *Informationen zu verifizierten Accounts [Twitter]*. Zu finden unter: https://help.twitter.com/de/managing-your-account/about-twitter-verified-accounts (abgerufen am 24.06.2019).

[48] Martin Maciej. *Instagram: Blauen Haken für Account-Verifizierung bekommen*, 29.08.2018. Zu finden unter: https://www.giga.de/downloads/instagram/tipps/instagram-blauen-haken-fuer-account-verifizierung-bekommen-geht-das/ (abgerufen am 24.06.2019).

[49] *Bestätigungskennzeichen auf Kanälen [Verifizierung bei YouTube]*. Zu finden un-

ter: https://support.google.com/youtube/answer/3046484?hl=de (abgerufen am 24.06.2019).

[50] *Fragen und Antworten [Telegram]*. Zu finden unter: https://telegram.org/faq/de (abgerufen am 27.06.2019).

[51] Pavel Durov. *200,000,000 Monthly Active Users*, 22.03.2018. Zu finden unter: https://telegram.org/blog/200-million (abgerufen am 27.06.2019).

[52] R. Salz M. Mealling, P. Leach. *RFC 4122: A Universally Unique IDentifier (UUID) URN Namespace*, 2005. Zu finden unter: https://www.heise.de/netze/rfc/rfcs/rfc4122.shtml (abgerufen am 14.11.2019).

[53] *Resource Interchange File Format*. Zu finden unter: https://en.wikipedia.org/wiki/Resource_Interchange_File_Format (abgerufen am 12.11.2019).

[54] *RIFF (resource interchange file format)*. Zu finden unter: https://www.itwissen.info/RIFF-resource-interchange-file-format-RIFF-Dateiformat.html (abgerufen am 12.11.2019).

[55] Rohit Tamma and Donnie Tindall. *Learning android forensics*. Packt Publishing Ltd, 2015.

[56] Tyrone Nicholas. *Introducing Fresco: A new image library for Android*, 26.03.2015. Zu finden unter: https://engineering.fb.com/android/introducing-fresco-a-new-image-library-for-android/ (abgerufen am 11.10.2019).

[57] *An Image Management Library [Fresco]*. Zu finden unter: https://frescolib.org/ (abgerufen am 11.10.2019).

[58] Gandeva Satrya, Philip Daely, and Muhammad Arief. Digital forensic analysis of telegram messenger on android devices. pages 1–7, 10 2016.

[59] Olawale Adebayo, Salamatu Sulaiman, Oluwafemi Osho, John Alhassan, and Shafi'i Abdulhamid. Forensic analysis of kik messenger on android devices. 10 2017.

[60] Kenneth Ovens and Gordon Morison. Forensic analysis of kik messenger on ios devices. *Digital Investigation*, 17:40–52, 04 2016.

[61] Vineeta Jain, Divya Sahu, and Deepak Tomar. Evidence gathering of line messenger on iphones. *International Journal of Innovations in Engineering and Management(IJIEM)*, 4:2319–3344, 07 2015.

Anhang A: Rollen und Rechte der Nutzer des Messengers TamTam

A.1 Rollen und Rechte für Chats

Die folgende Auflistung illustriert die Rechte, welche Chatmitgliedern in ihrer Rolle zur Verfügung stehen (können). [45]. Als Chatmitglieder werden TamTam-Nutzer in allen Rollen bezeichnet. Ihre Rechte gelten für alle Nutzer und können nicht aberkannt werden. Die zusätzlichen Rechte eines Teilnehmers weichen insofern von den Rechten eines Chatmitglieds ab, da diese nicht zwingend aktiviert sein müssen.

Alle Chatmitglieder:

- Nachrichten schreiben und empfangen
- auf Nachrichten antworten
- Nachrichten weiterleiten
- Nachrichten als neu markieren
- Nachrichten bearbeiten (innerhalb von 24 Stunden nach dem Versenden der entsprechenden Nachricht)
- Nachrichten für sich selbst löschen
- Nachrichten für alle Chatmitglieder löschen (Chatteilnehmer können dies nur innerhalb von 24 Stunden nach dem Versenden der entsprechenden Nachricht)
- Standort anfordern
- im Chat nach Nachrichten suchen
- Chatverlauf löschen
- Chat verlassen
- Chat melden (dieses Recht gilt für alle Chatmitglieder außer dem Chateigentümer)

zusätzliche Rechte für Teilnehmer:

Folgende Rechte können von dem Chateigentümer für alle Chatteilnehmer einheitlich aktiviert und deaktiviert werden. Standardmäßig sind alle der genannten Rechte, bis auf „Nachrichten anheften“ aktiviert.

- Chatinformationen (Foto, Bezeichnung (Titel), Beschreibung) ändern
- Hinzufügen von Chat Teilnehmern
- Nachrichten anheften

zusätzliche Rechte für Chat Administratoren:

Folgende Rechte können von dem Chateigentümer und von Chat Super Administratoren für jeden Chat Administrator individuell aktiviert und deaktiviert werden. Standardmäßig sind alle der genannten Rechte aktiviert.

- Nachrichten anheften
- Nachrichten für alle Chatmitglieder löschen
- Chatinformationen (Foto, Bezeichnung (Titel), Beschreibung) ändern
- Hinzufügen und Entfernen von Chat Teilnehmern

zusätzliche Rechte für Chat Super Administratoren:

- Nachrichten für alle Chatmitglieder löschen
- Nachrichten anheften
- Chatinformationen (Foto, Bezeichnung (Titel), Beschreibung) ändern
- Hinzufügen und Entfernen von Chat Super Administratoren
- Hinzufügen und Entfernen von Chat Administratoren sowie den einzelnen Administratoren Rechte zuweisen
- Hinzufügen und Entfernen von Chat Teilnehmern

zusätzliche Rechte für Chateigentümer:

- Nachrichten für alle Chatmitglieder löschen
- Nachrichten anheften
- Chatinformationen (Foto, Bezeichnung (Titel), Beschreibung) ändern
- Chat löschen
- Hinzufügen und Entfernen von Chat Super Administratoren
- Hinzufügen und Entfernen von Chat Administratoren sowie den Administratoren Rechte zuweisen
- Hinzufügen und Entfernen von Chat Teilnehmern sowie Teilnehmerrechte festlegen
- auf Chat Einstellungen zugreifen und dort den Chat auf privat beziehungsweise öffentlich stellen sowie den Link für den Chat generieren
- Besitzerrechte übergeben

A.2 Rollen und Rechte für Kanäle

Die konkreten Rechte, welche den Kanalmitgliedern in ihrer Rolle zur Verfügung stehen, werden in der folgenden Auflistung veranschaulicht. Die Rechte eines Kanalmitglieds gelten für alle TamTam-Nutzer und können nicht aberkannt werden. [45]

Alle Kanalmitglieder:

- Die Inhalte des Kanals einsehen
- Im Kanal nach Nachrichten suchen
- Nachrichten weiterleiten
- Den Link des Kanals teilen
- Etwas über den Kanal erzählen: wenn der Kanal privat ist, kann so der Einladungslink an andere TamTam Konversationen verschickt werden
- Benachrichtigungen für neue Nachrichten in einem Kanal ein- und ausschalten

zusätzliche Rechte für Kanalabonnenten und Kanalbesucher:

- Kanal abonnieren und abbestellen

zusätzliche Rechte für Kanal Administratoren:

- Nachrichten als neu markieren
- Kanal abbestellen
- Kanal melden
- Verschiedene Rechte dieser Rolle können von dem Kanaleigentümer und Kanal Super Administratoren für jeden Kanal Administratoren individuell aktiviert und deaktiviert werden. Standardmäßig sind alle Rechte aktiviert. Die Rechte lauten:
  - Nachrichten schreiben, bearbeiten, löschen
  - Nachrichten anheften
  - Foto, Bezeichnung (Titel) und Beschreibung ändern
  - Hinzufügen und Entfernen von Abonnenten

zusätzliche Rechte für Kanal Super Administratoren:

- Nachrichten schreiben, bearbeiten, löschen und anheften
- Auf Nachrichten antworten
- Nachrichten als neu markieren

- Kanal abbestellen
- Kanal melden
- Foto, Bezeichnung (Titel) und Beschreibung ändern
- Hinzufügen und Entfernen von Kanal Super Administratoren
- Hinzufügen und Entfernen von Kanal Administratoren sowie den Administratoren Rechte zuweisen
- Hinzufügen und Entfernen von Abonnenten

zusätzliche Rechte für Kanaleigentümer:

- Nachrichten schreiben, bearbeiten, löschen und anheften
- Auf Nachrichten antworten
- Nachrichten als neu markieren
- Foto, Bezeichnung (Titel) und Beschreibung ändern
- Hinzufügen und Entfernen von Kanal Super Administratoren
- Hinzufügen und Entfernen von Kanal Administratoren sowie den Administratoren Rechte zuweisen
- Hinzufügen und Entfernen von Abonnenten
- Verlauf beziehungsweise Kanal löschen
- auf Kanal Einstellungen zugreifen und dort den Kanal auf privat beziehungsweise öffentlich stellen sowie den Link für den Kanal generieren
- Einstellen, dass Nachrichten unterzeichnet werden sollen (den Vornamen des Administrators beim Absenden der Nachrichten hinzufügen)
- Besitzerrechte übergeben

Anhang B: Auswertung von TamTam auf Android-Geräten

B.1 Experimente zur Auswertung des Messengers TamTam auf Android-Geräten

B.1.1 Nutzerinformationen

Folgende Kontakte wurden im Rahmen der dazugehörigen Experimente untersucht:

Name	Status	Profilbild	Telefonnr.	Beschreibung	Link
Hannah Wie	Account-inhaber	Standard		Ich bin Hannah	
Finja Wie	Kontakt	selbst aufgenommen			https://tt.me/FinjasLink
Dana Wie	Kontakt	Standard			
Lea Wie	Kontakt	Standard	+49 17...		
Birthe Wie	Kontakt	Standard		Ich bin Birthe und habe meine Beschreibung verändert	https://tt.me/BirthesLink
Jule Wie	Kontakt	Standard			

Tabelle B.1: Eigenschaften der untersuchten Nutzer im Rahmen der Experimente für Nutzerinformationen auf Android-Geräten

B.1.2 Informationen über Konversationen

Privatchats

Konversationsname	**Teilnehmer**
Birthe Wie	Birthe Wie, Hannah Wie
Dana Wie	Dana Wie, Hannah Wie
Finja Wie	Finja Wie, Hannah Wie
Lea Wie	Lea Wie, Hannah Wie
Jule Wie	Jule Wie, Hannah Wie

Tabelle B.2: Übersicht der untersuchten Privatchats bei Android-Geräten

Chats

Konversationsname	**Beschreibung**	**Erstellung**	Profilbild
Chatexperiment1		01.07.2019 12:43 Uhr	Standard
Chatexperiment2	Beschreibung_Chatexperiment2	01.07.2019 13:11 Uhr	Standard
Chatexperiment3	Beschreibung_Chatexperiment3	01.07.2019 13:33 Uhr	Standard
Chatexperiment4	Beschreibung_Chatexperiment4	01.07.2019 13:48 Uhr	Standard
Chatexperiment5	Beschreibung_Chatexperiment5	01.07.2019 14:03 Uhr	Standard
Chatexperiment6	Beschreibung_Chatexperiment6	01.07.2019 14:26 Uhr	Standard
Chatexperiment7	Beschreibung_Chatexperiment7	01.07.2019 14:47 Uhr	Standard
Chatexperiment8	Beschreibung_Chatexperiment8	01.07.2019 14:57 Uhr	Standard
Chatexperiment9	Beschreibung_Chatexperiment9	01.07.19 15:21 Uhr	Standard
Chatexperiment10	Beschreibung_Chatexperiment10	01.07.2019 15:29 Uhr	selbst gewählt
Chatexperiment11	Beschreibung_Chatexperiment11	01.07.2019 15:49 Uhr	Standard
Chatexperiment12	Beschreibung_Chatexperiment12	02.07.2019 09:09 Uhr	selbst gewählt
Chatexperiment13	Beschreibung_Chatexperiment13	02.07.2019 14:56 Uhr	Standard

Konversationsname	**Beschreibung**	**Erstellung**	Profilbild
Chatexperiment14	Beschreibung_Chatexperiment14	03.07.2019 09:09 Uhr	selbst gewählt
Chatexperiment15[6]	Beschreibung_Chatexperiment15[7]	03.07.2019 09:20 Uhr	Standard
Chatexperiment16		04.07.2019 09:10 Uhr	Standard
Gruppenchat mit Lea Wie und Hannah Wie		09.07.2019 14:01 Uhr	Standard
Chatexperiment18	Rollen und Rechte	10.07.2019 09:46 Uhr	Standard

Tabelle B.3: Eigenschaften der erstellten Chats für Android-Geräte (1)

Konversationsname	**Art**	**Link**
Chatexperiment1	ohne Link	
Chatexperiment2	privat	https://tt.me/join/Td2RLDSWwz5AMffQMjZiC_SIhPwUdQultaL3lf0G-rk
Chatexperiment3	öffentlich	tt.me/Chatexperiment3_Link
Chatexperiment4	ohne Link	
Chatexperiment5	privat	https://tt.me/join/hdGhU1ybY-vsZre0-JxH-3Gez LNAT3jaql5tCnBLzsI
Chatexperiment6	öffentlich	tt.me/Chatexperiment6_Link
Chatexperiment7	ohne Link	
Chatexperiment8	privat	https://tt.me/join/vr36VL5KVrFMteataUJbTZmlONbH9wlj2h5oYoyFc84
Chatexperiment9	öffentlich	tt.me/Chatexperiment9_Link
Chatexperiment10	ohne Link	
Chatexperiment11	privat	https://tt.me/join/PMxi8Xa4KlwHGgzJ-tXsfe_ 10P4-yHm7tmgPrF_-efU
Chatexperiment12	öffentlich	tt.me/Chatexperiment12_Link
Chatexperiment13	ohne Link	
Chatexperiment14	öffentlich	tt.me/Chatexperiment14_Link
Chatexperiment15	ohne Link	
Chatexperiment16	ohne Link	

[6] Der vollständige Konversationsname lautet: „Chatexperiment15_Das ist ein Chat mit einem Namen in Maximallaenge von angeblich 200 Zeichen..... .....Stopp“

[7] Die vollständige Beschreibung lautet: „Beschreibung_Chatexperiment15 mit Test auf die maximale Laenge der Beschreibung..... ....Stopp“

Konversationsname	**Art**	**Link**
Gruppenchat mit Lea Wie und Hannah Wie	ohne Link	
Chatexperiment18	ohne Link	

Tabelle B.4: Eigenschaften der erstellten Chats für Android-Geräte (2)

Konversationsname	**Accountinhaber**	**Teilnehmer**
Chatexperiment1	Hannah Wie: Ersteller	Finja Wie: Teilnehmer Dana Wie: Teilnehmer
Chatexperiment2	Hannah Wie: Ersteller	Finja Wie: Teilnehmer Dana Wie: Teilnehmer
Chatexperiment3	Hannah Wie: Ersteller	Finja Wie: Teilnehmer Dana Wie: Teilnehmer
Chatexperiment4	Hannah Wie: Super Admin	Finja Wie: Ersteller Dana Wie: Teilnehmer
Chatexperiment5	Hannah Wie: Super Admin	Finja Wie: Ersteller Dana Wie: Teilnehmer
Chatexperiment6	Hannah Wie: Super Admin	Finja Wie: Ersteller Dana Wie: Teilnehmer
Chatexperiment7	Hannah Wie: Admin	Finja Wie: Teilnehmer Dana Wie: Ersteller
Chatexperiment8	Hannah Wie: Admin	Finja Wie: Admin Dana Wie: Ersteller
Chatexperiment9	Hannah Wie: Admin	Finja Wie: Ersteller Dana Wie: Super Admin
Chatexperiment10	Hannah Wie: Teilnehmer	Finja Wie: Ersteller Dana Wie: Super Admin
Chatexperiment11	Hannah Wie: Teilnehmer	Finja Wie: Ersteller Dana Wie: Admin
Chatexperiment12	Hannah Wie: Teilnehmer	Finja Wie: Teilnehmer Dana Wie: Ersteller
Chatexperiment13	Hannah Wie: Ersteller	Dana Wie: Teilnehmer
Chatexperiment14	Hannah Wie: Teilnehmer	Finja Wie: Teilnehmer Dana Wie: Ersteller
Chatexperiment15	Hannah Wie: Teilnehmer	Finja Wie: Ersteller
Chatexperiment16	Hannah Wie: Ersteller	Finja Wie: Admin Dana Wie: Teilnehmer

Konversationsname	**Accountinhaber**	**Teilnehmer**
Gruppenchat mit ...	Hannah Wie: Ersteller	Lea Wie: Teilnehmer
Chatexperiment18	Hannah Wie: Ersteller	Finja Wie: Admin Dana Wie: Admin

Tabelle B.5: Teilnehmer und ihre Rollen in den erstellten Chats für Android-Geräte

Kanäle

Konversationsname	**Beschreibung**	**Erstellung**	Profilbild
Kanalexperiment1	Kanalexperiment1_Beschreibung	11.07.2019 15:05 Uhr	Standard
Kanalexperiment2	Kanalexperiment2_Beschreibung	11.07.2019 15:08 Uhr	Standard
Kanalexperiment3	Kanalexperiment3_Beschreibung	11.07.2019 15:09 Uhr	Standard
Kanalexperiment4	Kanalexperiment4_Beschreibung	11.07.2019 15:16 Uhr	Standard
Kanalexperiment5	Kanalexperiment5_Beschreibung	11.07.2019 15:17 Uhr	Standard
Kanalexperiment6	Kanalexperiment6_Beschreibung	11.07.2019 15:18 Uhr	Standard
Kanalexperiment8	Kanalexperiment8_Beschreibung	22.07.2019 09:29 Uhr	Standard
Kanalexperiment9	Kanalexperiment9_Beschreibung	22.07.2019 09:33 Uhr	Standard
Kanalexperiment10	Kanalexperiment10_Beschreibung	22.07.2019 09:46 Uhr	selbst gewählt
Kanalexperiment11	Kanalexperiment11_Beschreibung	22.07.2019 09:49 Uhr	Standard
Kanalexperiment12	Kanalexperiment12_Beschreibung	22.07.2019 09:50	Standard

Tabelle B.6: Eigenschaften der erstellten Kanäle für Android-Geräte (1)

Konversationsname	**Art**	**Link**
Kanalexperiment1	ohne Link	
Kanalexperiment2	privat	https://tt.me/join/1ToSgQ9ladDi9oredYBwdRV8euoJv1P1CH8P14YDbso
Kanalexperiment3	öffentlich	tt.me/Kanalexperiment3_Link
Kanalexperiment4	ohne Link	

Konversationsname	**Art**	**Link**
Kanalexperiment5	privat	https://tt.me/join/xhZ6Qt-XDEEXfUb1ta6 O8IVwvaBp3wW9uFjOSs5ifbc
Kanalexperiment6	öffentlich	tt.me/Kanalexperiment6_Link
Kanalexperiment8	privat	https://tt.me/join/-zuXUFPA15FZKlzS5s d27MuZm97euZ9Sen-3oQX5Zzo
Kanalexperiment9	öffentlich	tt.me/Kanalexperiment9_Link
Kanalexperiment10	ohne Link	
Kanalexperiment11	privat	https://tt.me/join/z77NtAlvBTmG_KLF t0EYr2BadUHEf4GvvllCGHXSYkg
Kanalexperiment12	öffentlich	tt.me/Kanalexperiment12_Link

Tabelle B.7: Eigenschaften der erstellten Kanäle für Android-Geräte (2)

Konversationsname	**Accountinhaber**	**Teilnehmer**
Kanalexperiment1	Hannah Wie: Ersteller	Finja Wie: Teilnehmer Dana Wie: Teilnehmer
Kanalexperiment2	Hannah Wie: Ersteller	Finja Wie: Teilnehmer Dana Wie: Teilnehmer
Kanalexperiment3	Hannah Wie: Ersteller	Finja Wie: Teilnehmer Dana Wie: Teilnehmer
Kanalexperiment4	Hannah Wie: Super Admin	Finja Wie: Ersteller Dana Wie: Teilnehmer
Kanalexperiment5	Hannah Wie: Super Admin	Finja Wie: Ersteller Dana Wie: Teilnehmer
Kanalexperiment6	Hannah Wie: Super Admin	Finja Wie: Ersteller Dana Wie: Teilnehmer
Kanalexperiment8	Hannah Wie: Admin	Finja Wie: Admin Dana Wie: Ersteller
Kanalexperiment9	Hannah Wie: Abonnent	Finja Wie: Ersteller Dana Wie: Super Admin
Kanalexperiment10	Hannah Wie: Teilnehmer	Finja Wie: Ersteller Dana Wie: Super Admin
Kanalexperiment11	Hannah Wie: Teilnehmer	Finja Wie: Ersteller Dana Wie: Admin
Kanalexperiment12	Hannah Wie: Teilnehmer	Finja Wie: Ersteller Dana Wie: Teilnehmer

Tabelle B.8: Teilnehmer und ihre Rollen in den erstellten Kanälen für Android-Geräte

B.1.3 Informationen über die ausgetauschten Nachrichten sowie Anrufe

Tabelle B.9 zeigt die durchgeführten Experimente für Nachrichten, bei denen Medien ausgetauscht wurden. Tabelle B.10 dokumentiert die Experimente für Anrufe. Abbildung B.1 illustriert die Systemnachrichten bei TamTam. Die Gesprächsrichtung „ausgehend“ bezieht sich auf den Nutzer „Hannah Wie“. „Eingehende“ Nachrichten stammen von „Finja Wie“.

Gesprächsrichtung	**Medientyp**	**Beschreibung**
ausgehend	Bild	Foto aufgenommen mit der TamTam-Kamera
ausgehend	Bild	Foto verschickt aus der Galerie
ausgehend	Bild	Foto aufgenommen mit der TamTam-Kamera + Beschreibung des Bildes
ausgehend	Bild	Foto verschickt aus der Galerie + Beschreibung des Bildes
eingehend	Bild	Foto aufgenommen mit der TamTam-Kamera
eingehend	Bild	Foto verschickt aus der Galerie
eingehend	Bild	Foto aufgenommen mit der TamTam-Kamera + Beschreibung des Bildes
eingehend	Video	Video verschickt aus der Galerie
eingehend	Video	Video verschickt mit der TamTam-Kamera
eingehend	Video	Video verschickt mit der TamTam-Kamera + Beschreibung des Videos
ausgehend	Video	Video verschickt aus der Galerie
ausgehend	Video	Video verschickt mit der TamTam-Kamera
ausgehend	Kontakt	verschickter Kontakt: Dana Wie
eingehend	Kontakt	verschickter Kontakt: Dana Wie
ausgehend	Kontakt	verschickter Kontakt: Lea Wie
ausgehend	Datei	Bilder, MP3- sowie PDF-Dateien und Videos als Datei verschickt
eingehend	Datei	Bilder, MP3- sowie PDF-Dateien und Videos als Datei verschickt
ausgehend	Standort	
eingehend	Standort	
ausgehend	Sprach-nachricht	
eingehend	Sprach-nachricht	
ausgehend	Sticker	
eingehend	Sticker	

Gesprächsrichtung	Medientyp	Beschreibung
ausgehend	Link	
eingehend	Link	

Tabelle B.9: Durchgeführte Experimente für Mediennachrichten

Gesprächsrichtung	Übertragungsart	Gesprächsstatus
ausgehend	Audioanruf	durchgestellt
ausgehend	Audioanruf	abgewiesen
ausgehend	Audioanruf	entgangen
ausgehend	Videoanruf	durchgestellt
ausgehend	Videoanruf	abgewiesen
ausgehend	Videoanruf	entgangen
eingehend	Audioanruf	durchgestellt
eingehend	Audioanruf	abgewiesen
eingehend	Audioanruf	entgangen
eingehend	Videoanruf	durchgestellt
eingehend	Videoanruf	abgewiesen
eingehend	Videoanruf	entgangen

Tabelle B.10: Durchgeführte Experimente für Anrufe

Abbildung B.1: Systemnachrichten in einem TamTam-Chat (hier: Konversation „Chat-Titel“)

B.2 Ergebnisse der Auswertung des Messengers TamTam (v2.6.0) auf Android-Geräten

B.2.1 Die URLs in den BLOBs der Datenbank *cache.db*

- ursprüngliche URL1 (für quadratisches Profilbild):
  *ttps://i.mycdn.me/image?id=892766599408&ts=0000000122000002eb&plc=API&aid=1150827264&tkn=*MsOeNe4W5O_uKo1GvsinAtOATA4*
- angepasste URL1 - quadratisches Bildformat; 192x192 Pixel:
  *ttps://i.mycdn.me/image?id=892766599408&ts=0000000122000002eb&plc=API&aid=1150827264&tkn=*MsOeNe4W5O_uKo1GvsinAtOATA4&fn=sqr_192*
- angepasste URL1 - Bildformat: 747 Pixel hoch, Breite variabel (hier 747 Pixel):
  *ttps://i.mycdn.me/image?id=892766599408&ts=0000000122000002eb&plc=API&aid=1150827264&tkn=*MsOeNe4W5O_uKo1GvsinAtOATA4&fn=w_1440*

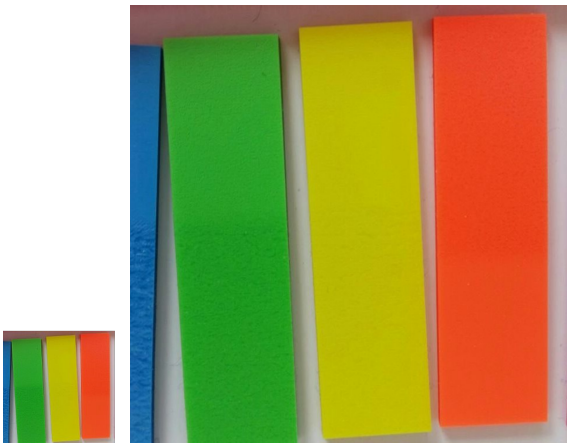


Abbildung B.2: Profilbild der Konversation „Chat-Titel“ (1); links: URL1 (192x192 Pixel), rechts: URL1 (747x747 Pixel)

- ursprüngliche URL2 (für rechteckiges Profilbild):
  *https://i.mycdn.me/image?id=892766599408&plc=API&aid=1150827264&tkn=*b3h0yjtcDnjGBKN-9_eJg36B2LE*
- angepasste URL2 - quadratisches Bildformat; 192x192 Pixel:
  *https://i.mycdn.me/image?id=892766599408&plc=API&aid=1150827264&tkn=*b3h0yjtcDnjGBKN-9_eJg36B2LE&fn=sqr_192*
- angepasste URL2 - Bildformat: 747 Pixel hoch, Breite variabel (hier 1328 Pixel):
  *https://i.mycdn.me/image?id=892766599408&plc=API&aid=1150827264&tkn=*b3h0yjtcDnjGBKN-9_eJg36B2LE&fn=w_1440*

Abbildung B.3: Profilbild der Konversation „Chat-Titel“ (2); links: URL2 (192x192 Pixel), rechts: URL2 (747x1328 Pixel)

B.2.2 Der BLOB Marker *ctt_data*

Folgende Marker des BLOBs *ctt_data* der Tabelle *contacts* der Datenbank *cache.db* konnten mitsamt ihrer Bedeutung identifiziert werden:

Marker: 0x08, Information: NutzerID:
Globale ID des Nutzers, welche mittels Tex-Verfahren codiert ist. Wird sie decodiert, erhält man den Wert des Attributs *ctt_server_id*.

Marker: 0x40, Information: Profilaktualisierung:
Zeitpunkt, an dem das Nutzerprofil das letzte Mal aktualisiert wurde. Der Zeitstempel ist ein Unix-Zeitstempel, welcher mit dem Tex-Verfahren codiert wurde.

Marker: 0x48, Information: Telefonnummer:
Telefonnummer des Nutzers, diese ist mit dem Tex-Verfahren codiert. Wird sie decodiert erhält man den Wert des Attributs *phs_phone* aus Tabelle *phones*.

Marker: 0x50, Information: Kontaktstatus:
Dieser Marker kann dazu verwendet werden, um zu ermitteln, ob ein Nutzer von dem Accountinhaber blockiert wurde. In diesem Fall ist der Wert des Markers *0x01*.

Marker: 0x72, Information: Nutzername:
Name des Nutzers. Die Informationen werden wie folgt abgespeichert: *0x72* (Marker für die gesamte Struktur) + Gesamtlänge der Information (mit Tex-Verfahren codiert) + *0x0A* (Marker für den Nutzernamen) + Länge des Nutzernamens (mit Tex-Verfahren codiert) + Nutzername + *0x10* (Marker) + 1 Byte mit unbekannter Bedeutung. Als Werte wurden *0x02* und *0x04* beobachtet.

Marker: 0x8201, Information: Beschreibung des Nutzers:
Selbstgewählte Beschreibung eines Nutzers. Die Struktur für die Beschreibung lautet: 0x8201 + Länge der Beschreibung (mit Tex-Verfahren codiert) + Beschreibung.

Marker: 0x8A01, Information: selbstgewählter Link des Nutzers:
Selbstgewählter Link eines Nutzers. Die Struktur für die Beschreibung lautet: 0x8A01 + Länge des Links (mit Tex-Verfahren codiert) + Link.

Marker: 0xA201 und 0xAA01, Information: URLs zum Profilbild des Nutzers:
Zwei URLs zum Profilbild des Nutzers. Besitzt der Nutzer kein Profilbild, so wird ihm ein Standardprofilbild zugeordnet. Zu diesem werden im Regelfall keine URLs abgelegt. Dies gilt jedoch nicht für den Accountinhaber, ihm werden auch URLs zugeordnet, wenn er ein Standardprofilbild besitzt.
Mit Hilfe der ersten URL (Marker: 0xA201) kann auf das Profilbild in einem quadratischen Format zugegriffen werden. Die zweite URL (Marker: 0xAA01) ermöglicht den Zugriff das Profilbild in einem rechteckigen Format.
Die Informationen werden wie folgt abgespeichert: Marker + Länge der URL (mit Tex-Verfahren codiert) + URL

B.2.3 Der BLOB *cht_data*

Die folgende Auflistung zeigt die identifizierten Marker von *cht_data*-BLOBs der Tabelle *chats* der Datenbank *cache.db* und ihren Aufbau. Mit Hilfe dieser Auflistung können *cht_data*-BLOBs ausgewertet werden.

Marker: 0x08, Information: KonversationsID:
Der Marker *0x08* ist der Indikator für die ID der Konversation. Sie ist mit dem Tex-Verfahren codiert. Wird sie decodiert und als vorzeichenbehafteter Integer (Zweierkomplement) betrachtet, so entspricht sie dem Wert des Attributs *cht_server_id*.

Marker: 0x10, Information: Konversationstyp:
Der Konversationstyp gibt an, ob es sich bei der Konversation um eine Gruppe oder einen Kanal handelt. Fehlt dieser Marker, so handelt es sich um einen Privatchat zwischen dem Accountinhaber und einer weiteren Person.

Marker: 0x18, Information: Konversationsstatus:
Dieser Marker und sein Wert 0x03 wurden bisher lediglich beobachtet, wenn eine Konversation verlassen wurde. Bei Privatchats bedeutet der Marker und sein Wert, dass der Gesprächspartner blockiert wurde. Im Falle einer solchen Blockierung verschwindet der Privatchat auf dem Mobiltelefon aus den verfügbaren Konversationen. In der Datenbank ist die Konversation weiterhin enthalten, es ist jedoch

auffällig, dass die Zeit der letzten Nutzeraktivität beider Nutzer (Accountinhaber und Gesprächspartner) *0* beträgt.

Marker: 0x20, Information: EigentümerID:

ID des Nutzers, dem dir Konversation gehört. Die NutzerID ist mit dem Tex-Verfahren codiert.

Marker: 0x2A, Information: Bekannte Teilnehmer:

Bekannte Teilnehmer der Konversation. Die Informationen werden wie folgt abgespeichert: *0x2A* + Gesamtlänge der Information (mit Tex-Verfahren codiert) + *0x08* (Submarker für die NutzerID) + NutzerID (mit Tex-Verfahren codiert) + *0x10* (Submarker für den Zeitstempel, der dem Nutzer zugeordnet ist) + Zeitstempel (Unix-Zeitstempel mit Tex-Verfahren codiert). Der Zeitstempel speichert den Zeitpunkt der letzten Aktion des Nutzers innerhalb einer Konversation. Sind sich Accountinhaber und der betreffende Teilnehmer in einer Konversation noch nicht begegnet, wird dem Zeitstempel der Wert *0x00* zugeordnet. Um sich zu begegnen müssen Accountinhaber und Teilnehmer die Konversation nach Hinzufügen des Teilnehmers angeklickt und somit aktualisiert haben.
Welche Teilnehmer mit Hilfe dieses Markers aufgeführt werden, ist vom Typ der Konversation abhängig. Handelt es sich um einen Kanal, so wird lediglich der Accountinhaber als Teilnehmer aufgeführt. Die Mitglieder eines Kanals können mit Hilfe der Datenbank nicht eingesehen werden und werden nur online abgespeichert. Offline hat ein Nutzer keinen Zugriff auf die Teilnehmerliste eines Kanals.

Marker: 0x30, Information: Erstellungsdatum der Konversation:

Zeitpunkt, zu dem eine Konversation erstellt wurde (Unix-Zeitstempel mit Tex-Verfahren codiert). Es wurde beobachtet, dass dieser Zeitstempel fehlt, wenn es sich bei der Konversation um eine öffentliche Gruppe handelt. Bei einem Privatchat wird diesem Marker der Wert 0x00 zugeordnet.

Marker: 0x3A, Information: Name der Konversation:

Name der Konversation, welcher ihr von einem TamTam-Nutzer zugewiesen wurde. Die Struktur für einen Konversationsnamen lautet: *0x3A* + Länge des Namens (mit Tex-Verfahren codiert) + Name. Der Name kann maximal 200 Zeichen lang sein, der höchste Wert für die Länge (mit Tex-Verfahren codiert) lautet demnach: *0xC801*.
Eine Gruppe kann auch ohne einen Namen erstellt werden. Wird einer Gruppe kein individueller Name gegeben, so erhält sie einen, von TamTam generierten, Standardnamen. Dieser setzt sich aus den Namen der Gruppenmitglieder, jedoch ohne den Namen des Accountinhabers, zusammen. Eine Gruppe mit den Nutzern Dana Wie, Hannah Wie und Finja Wie, bei der Hannah Wie der Accountinhaber ist, lautet beispielsweise: „Dana Wie, Finja Wie“. Einem Privatchat kann kein Name zugewiesen werden. Der Name eines Privatchats ist immer gleich dem Namen

des Konversationspartners. Besitzt eine Konversation einen Standardnamen oder kann ihr kein individueller Name zugeordnet werden, so wird in dem BLOB kein Name für diese Konversation gespeichert.

Marker: 0x50 und 0x9001, Information: ID der ersten und letzten Nachricht:
ID der ersten (*0x9001*) und letzten (*0x50*) bekannten Nachricht, die innerhalb dieser Konversation verschickt wurde (mit Tex-Verfahren codiert). Der Inhalt dieser Nachrichten kann mit Hilfe der Tabelle *messages* ermittelt werden.

Marker: 0xC001, Information: Konversationsart:
Gibt an, ob eine Konversation öffentlich oder privat ist. Nach dem Marker folgt der Wert *0x01*, wenn es sich um eine private Konversation handelt. Eine Konversation, die an keinen Link gebunden ist, wird ebenfalls als privat klassifiziert. Der Marker fehlt, wenn es sich um eine öffentliche Konversation handelt.

Marker: 0xCA01, Information: Link der Konversation:
Der Link einer Konversation, sofern sie einen solchen besitzt. Es kann sich um einen privaten oder einen öffentlichen Link handeln. Vor dem Link und nach dem Marker *0xCA01* steht im BLOB die Länge des Links (mit Tex-Verfahren codiert). Ein öffentlicher Link kann von einem TamTam-Nutzer frei gewählt werden und hat aus diesem Grund eine variable Länge. Private Links werden von TamTam generiert. Bisher wurden nur private Links mit der Länge *0x3E* beobachtet. Privatchats kann kein Link zugeordnet werden. Für diese Konversationen fehlt der Marker und sein Wert.

Marker: 0xE801, Information: Teilnehmerzahl:
Anzahl der Teilnehmer dieser Konversation (mit Tex-Verfahren codiert).

Marker: 0xF201, Information: Beschreibung:
Die Beschreibung einer Konversation. Die Struktur für eine Beschreibung lautet: *0xF201* + Länge der Beschreibung (mit Tex-Verfahren codiert) + Beschreibung. Die Beschreibung kann maximal 400 Zeichen lang sein, der höchste Wert für die Länge (mit Tex-Verfahren codiert) lautet demnach: *0x9003*.

Marker: 0xF801, Information: Administratoren der Konversation:
Auflistung der bekannten Konversationsteilnehmer in den Rollen eines Administrators oder höher. Die Auflistung besteht aus dem Marker 0xF801 und der NutzerID des jeweiligen Teilnehmers (mit Tex-Verfahren codiert).
Ob eine Information über die Administratoren einer Konversation vorhanden ist, hängt davon ab, welche Rolle der Accountinhaber besitzt. Ist er lediglich Teilnehmer einer Konversation, so kann er die Administratoren nicht einsehen.

Marker: 0xDA02, Information: Informationen zu den Administratoren:

Dieser Marker ist ein Indikator für verschiedene Informationen über Teilnehmer in den Rollen Administrator und höher. Dazu zählen die Rechte, die einem Nutzer gestattet werden sowie seine zugeordnete Rolle. Bei allen Nutzern, für die es diesen Marker gibt und die nicht der Gruppenersteller sind, wird zudem festgehalten, wer ihnen ihre Rolle und Rechte gegeben hat.

Eine Auskunft über die Administratoren ist, wie bei Marker *0xF801*, nicht immer verfügbar. Ob dieser Marker und seine Informationen in einem *cht_data*-BLOB vorhanden sind, hängt davon ab, welche Rolle der Accountinhaber innehat. Ist er lediglich Teilnehmer einer Konversation, so kann er keine Informationen über die Administratoren einsehen.

Die Struktur dieses Markers und seiner Informationen ist wie folgt aufgebaut:

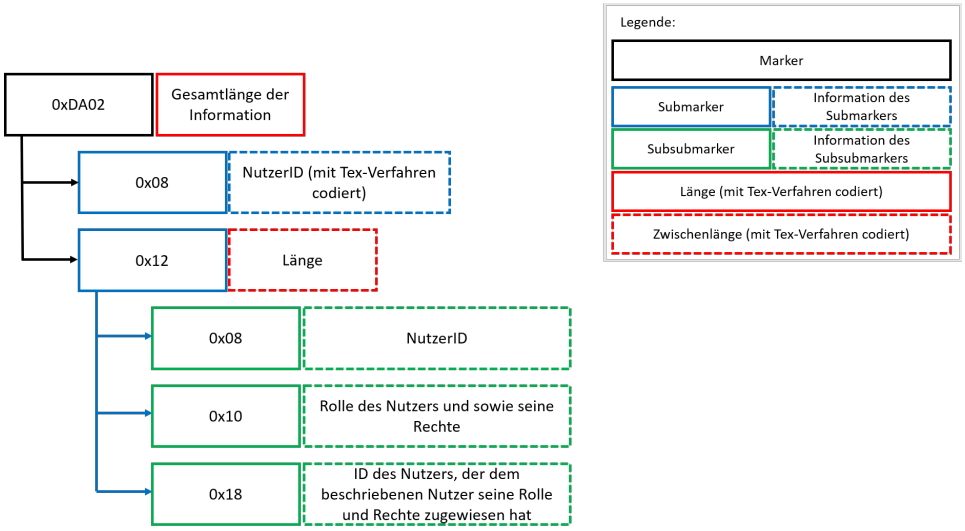


Abbildung B.4: Struktur des Markers 0xDA02 im BLOB *cht_data*

Der Subsubmarker *0x18* und seine Information fehlt, sofern der Marker *0xDA02* den Konversationsgründer beschreibt.

Marker: 0xE202 und 0xEA02, Information: URLs des Profilbilds:

Zwei URLs zum Profilbild der Konversation. Besitzt die Konversation kein individuelles Profilbild, so wird ihr ein Standardprofilbild zugeordnet. Zu diesem werden im BLOB jedoch keine Informationen abgelegt.

Die URLs werden wie folgt abgespeichert: Marker + Länge der URL (mit Tex-Verfahren codiert) + URL. Bei der URL, die dem Marker *0xE202* zugeordnet wird, wurde bisher immer eine Länge von *0x76* Byte beobachtet. Die URL des Markers *0xEA02* verfügte bei allen Untersuchungen über eine Länge von *0x60* Byte.

Privatchats besitzen als Profilbild das Bild des Konversationspartners. Informationen über dieses Profilbild werden in den BLOBs von Privatchats allerdings nicht gespeichert.

B.2.4 Der BLOB *msg_attaches*

Folgende Nachrichtentypen des BLOB *msg_attaches* aus der Tabelle *messages* der Datenbank *cache.db* sind mit ihren individuellen Markern bekannt.

Kontakte, Systemnachrichten, Sticker (*msg_media_type*: 0)

Kontakte

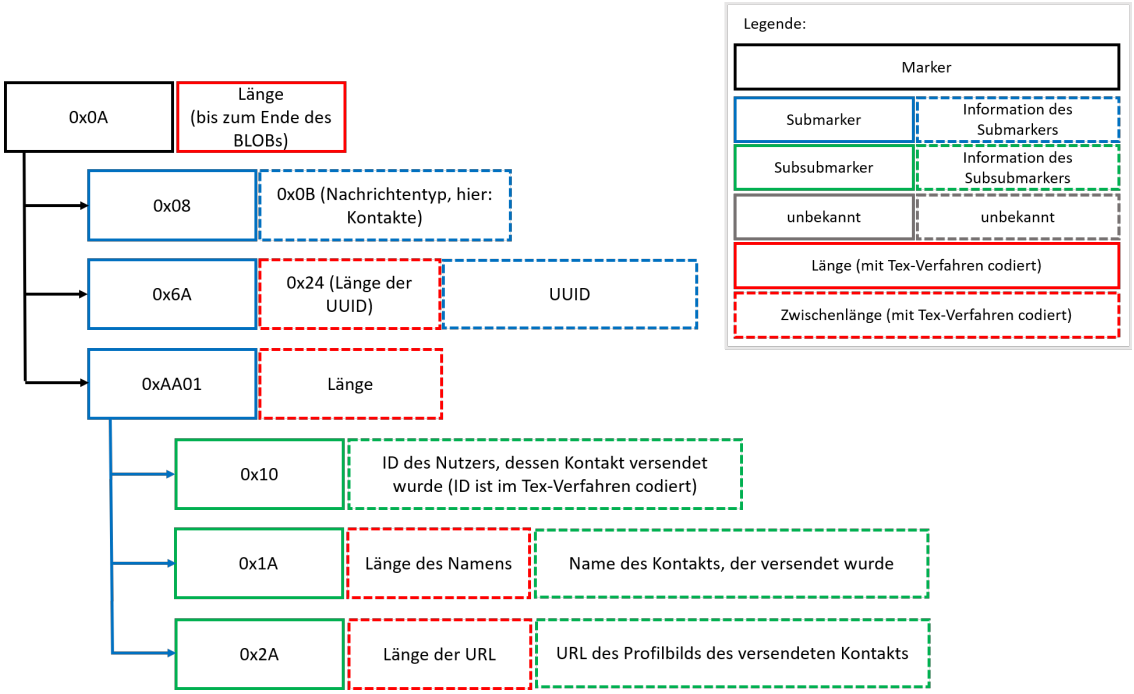


Abbildung B.5: Schema des BLOBs *msg_attaches* bei Kontakten

Systemnachrichten

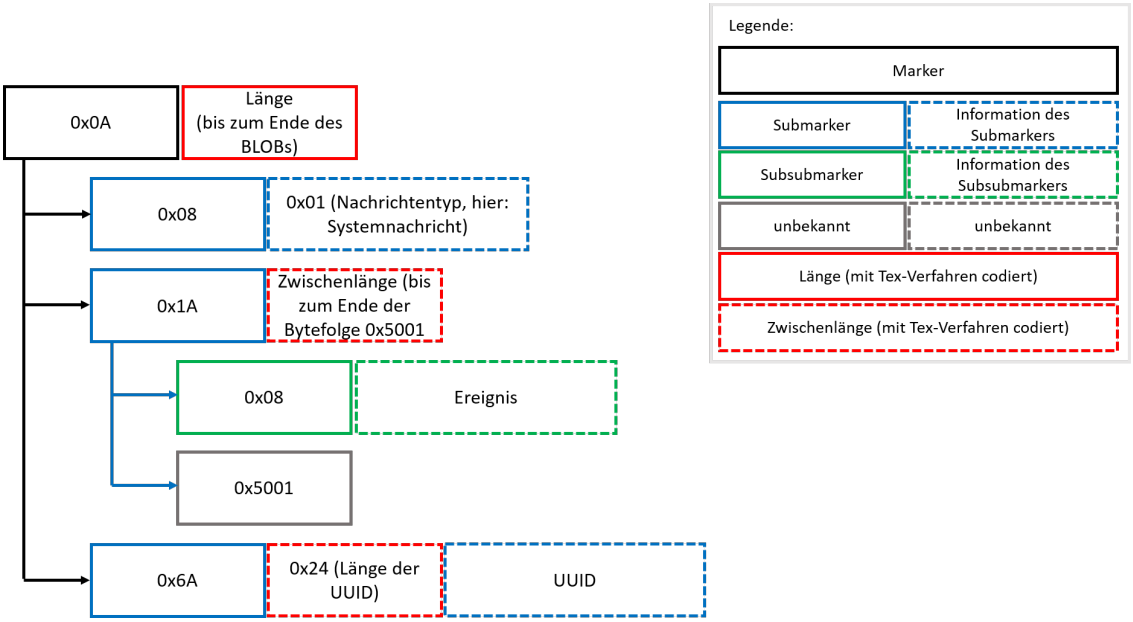


Abbildung B.6: Schema des BLOBs *msg_attaches* bei Systemnachrichten

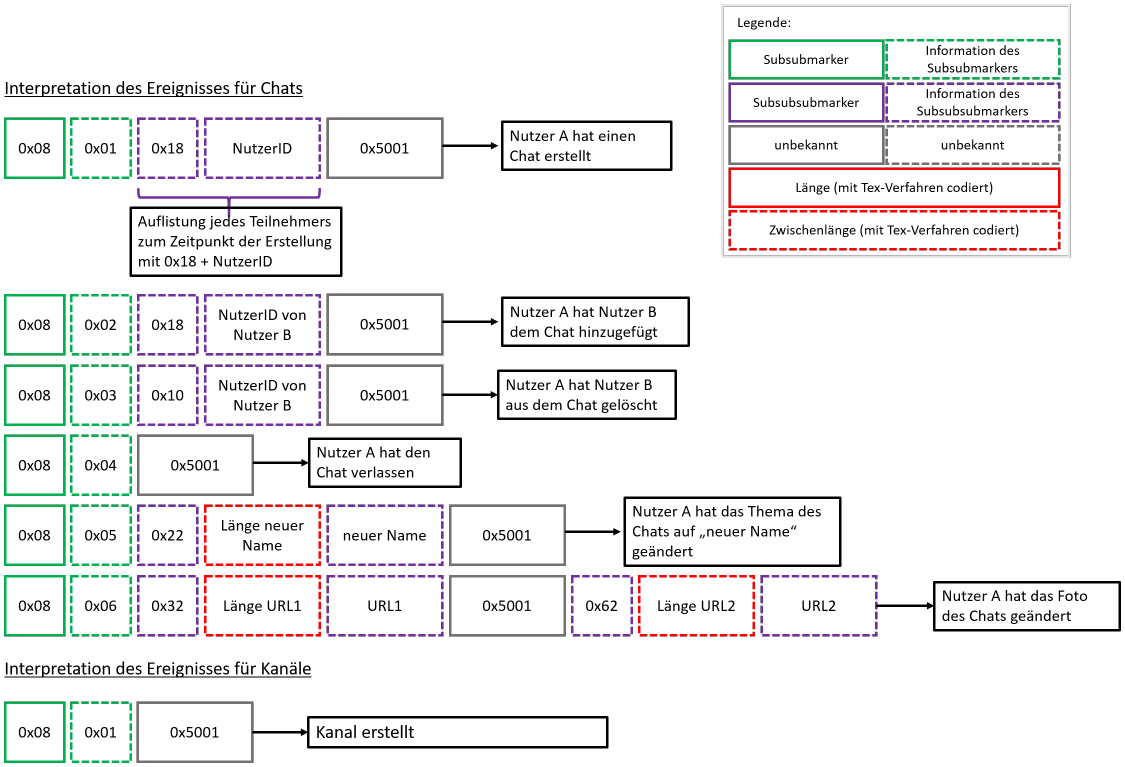


Abbildung B.7: Interpretation des Ereignisses aus dem BLOBs *msg_attaches* für Systemnachrichten

Sticker

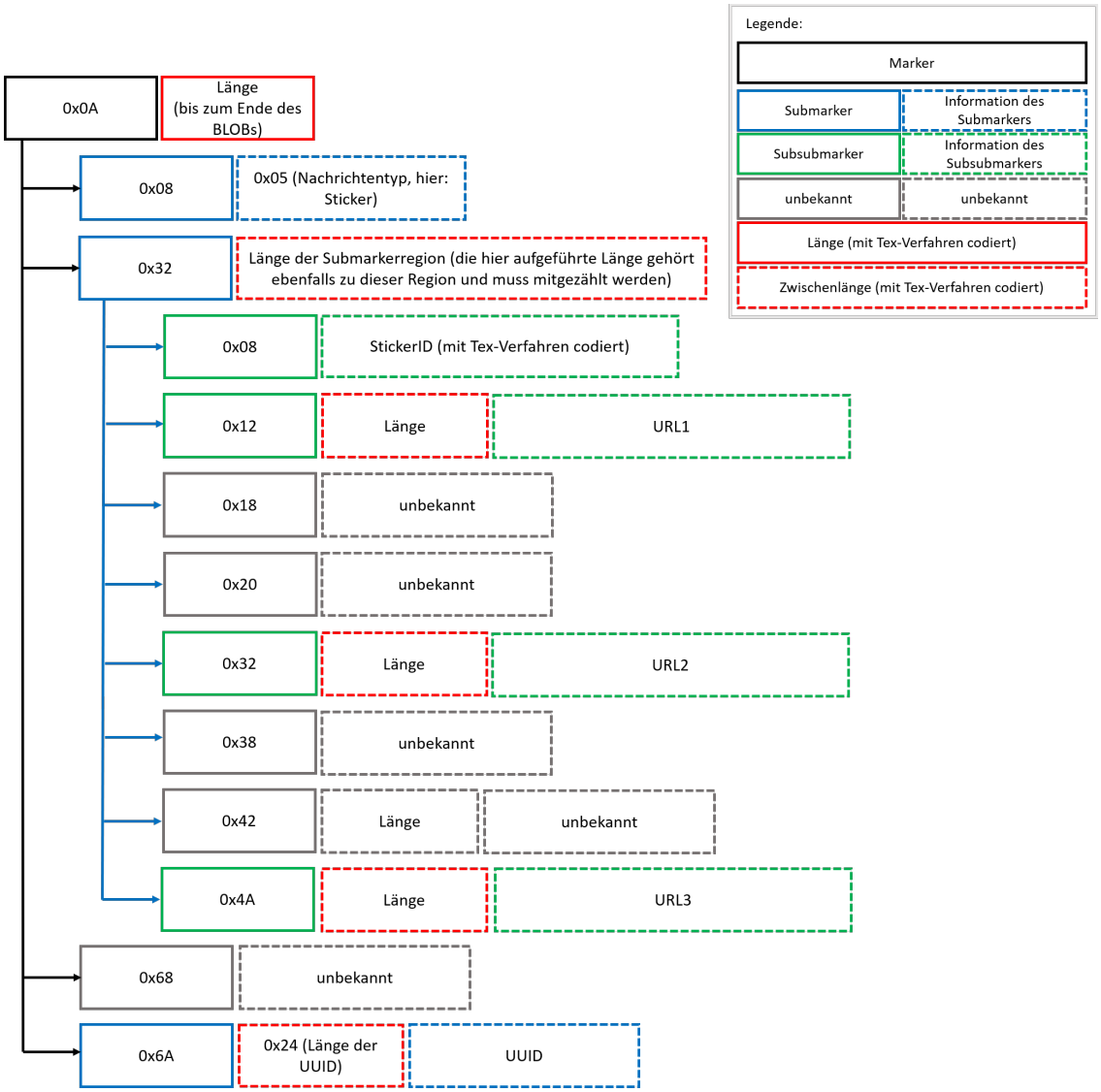


Abbildung B.8: Schema des BLOBs *msg_attaches* bei Stickern

Bilder (*msg_media_type*: 1)

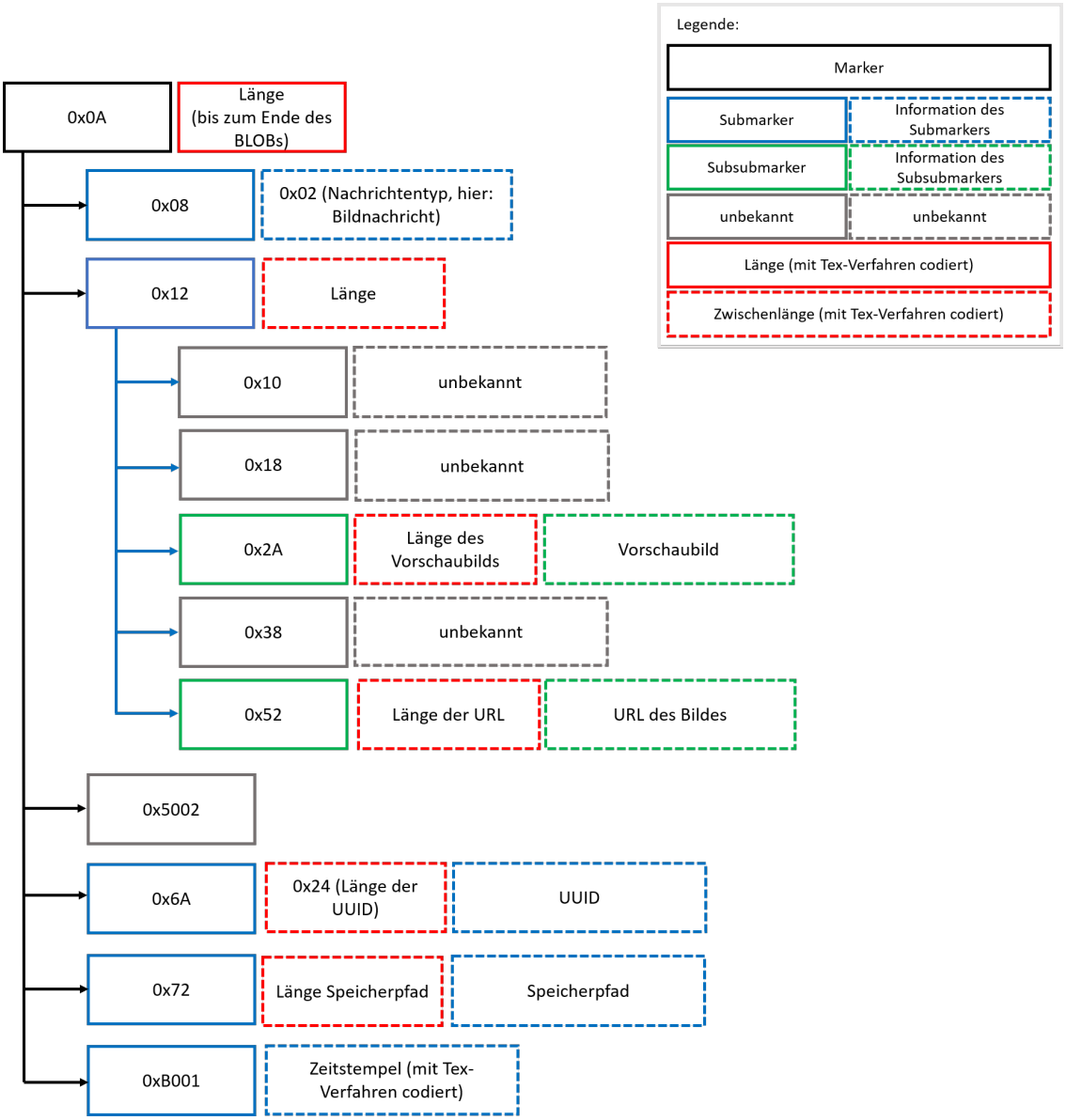


Abbildung B.9: Schema des BLOBs *msg_attaches* bei Bildnachrichten

Sprachnachrichten (*msg_media_type*: 2)

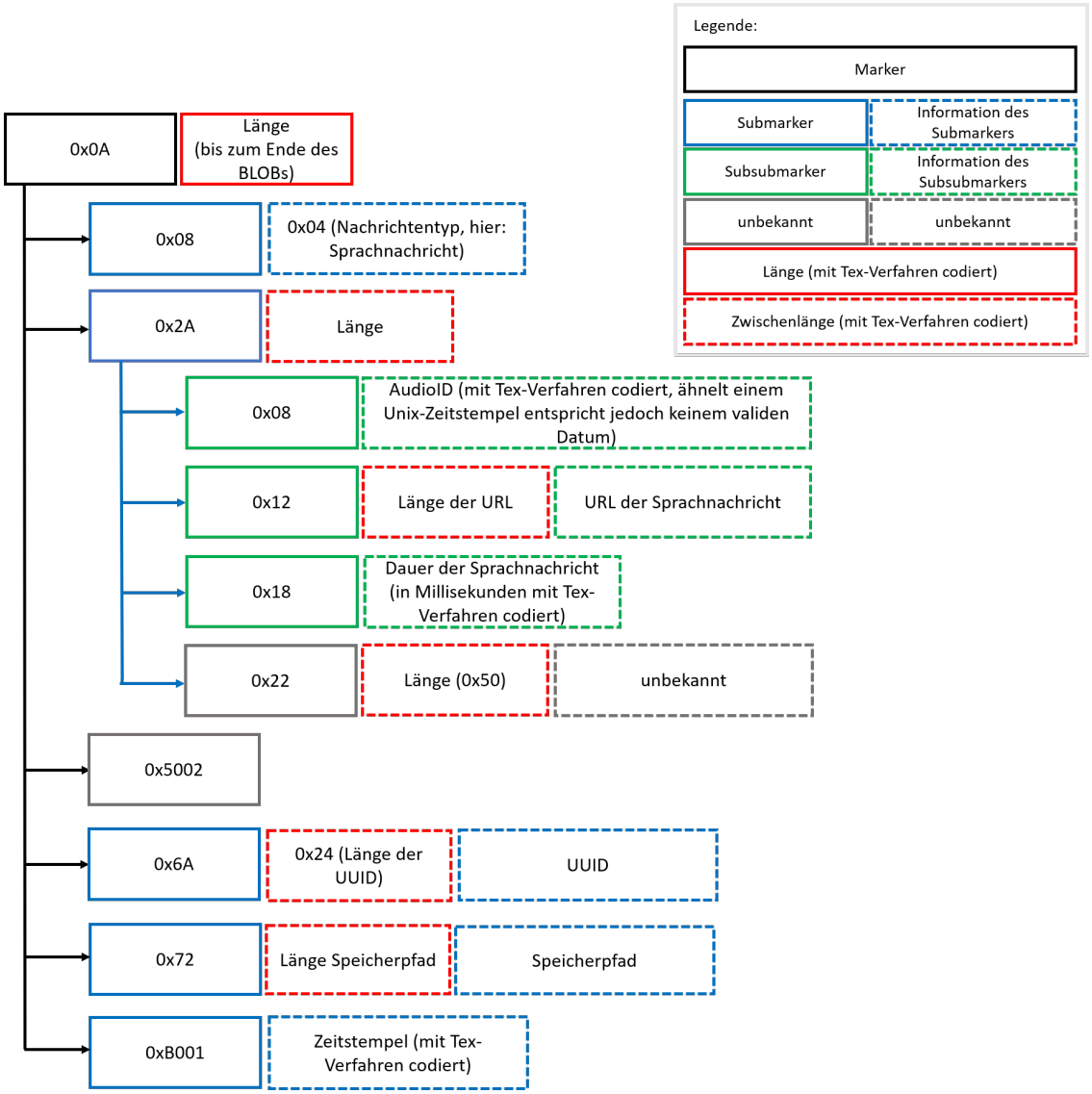


Abbildung B.10: Schema des BLOBs *msg_attaches* bei Sprachnachrichten

Videonachrichten (*msg_media_type*: 3)

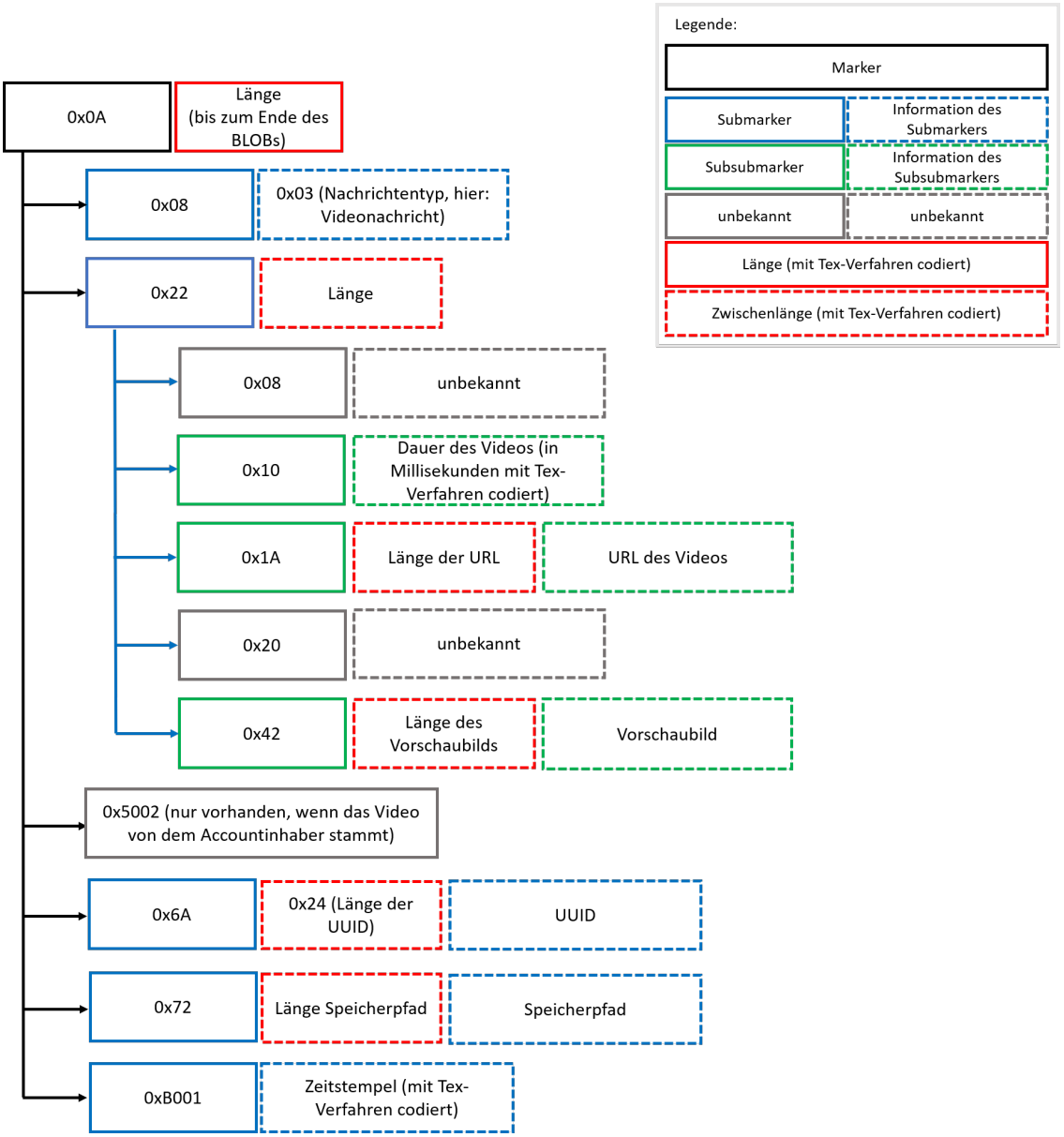


Abbildung B.11: Schema des BLOBs *msg_attaches* bei Videos

geteilte Links (*msg_media_type*: 5)

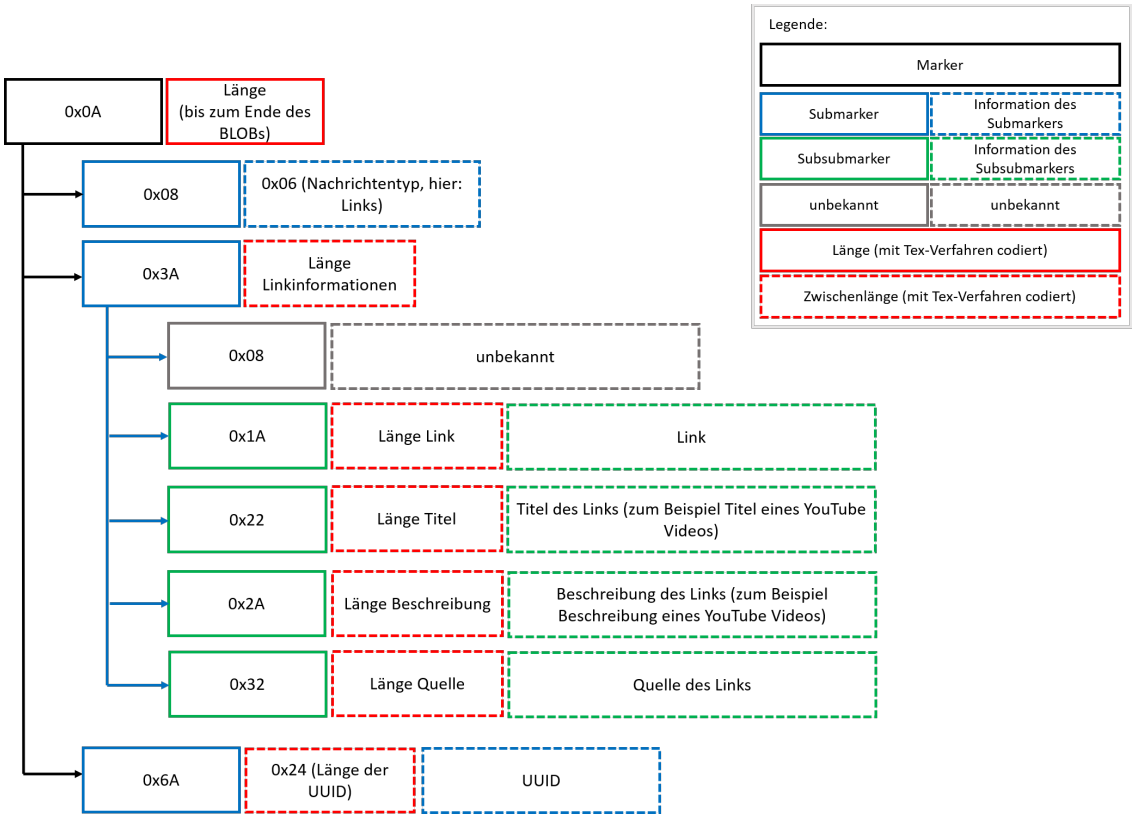


Abbildung B.12: Schema des BLOBs *msg_attaches* bei geteilten Links

Dateien (*msg_media_type*: 7)

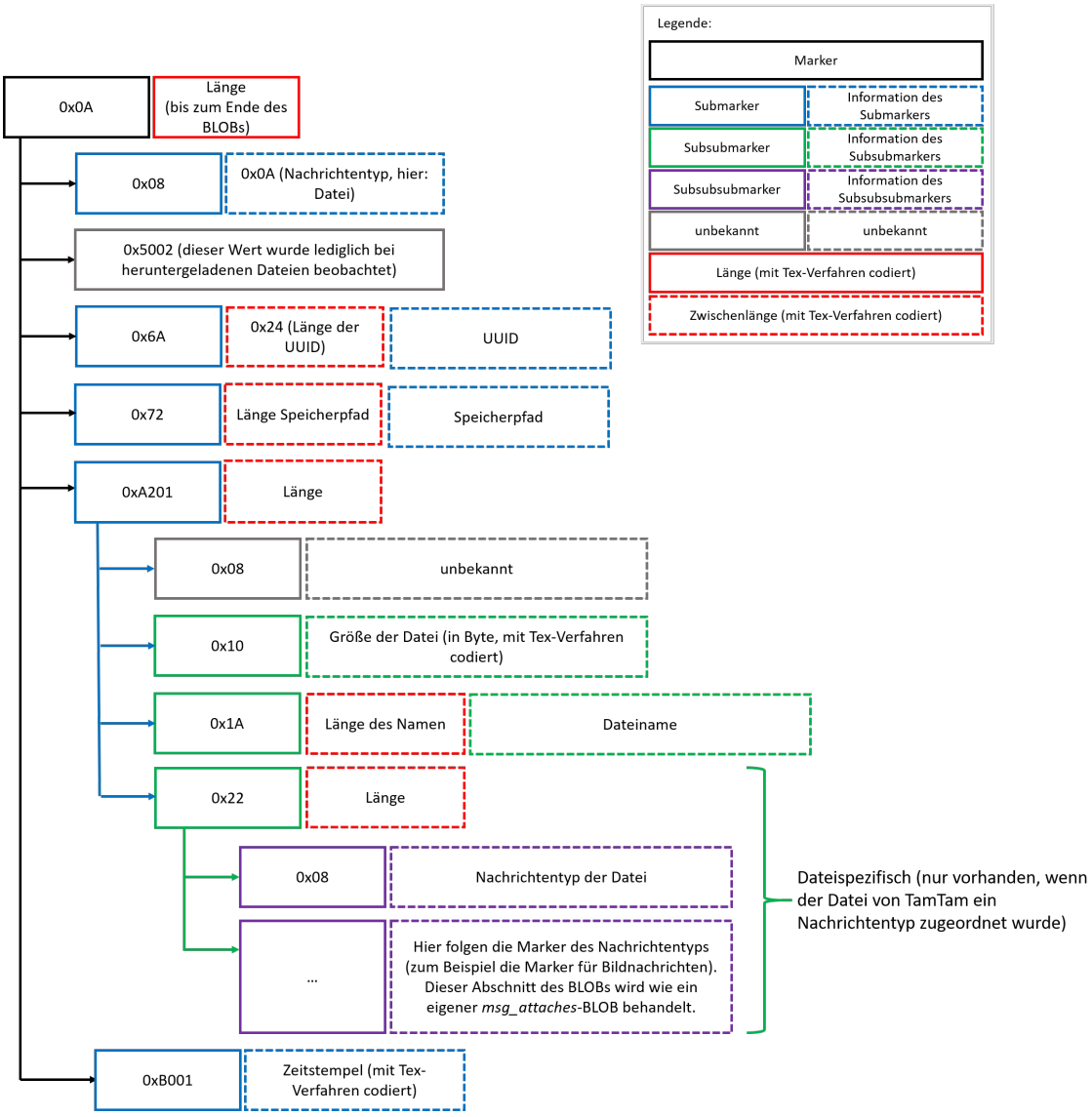


Abbildung B.13: Schema des BLOBs *msg_attaches* bei Dateien

Sonderfälle bei versendeten Dateien: Musikdateien

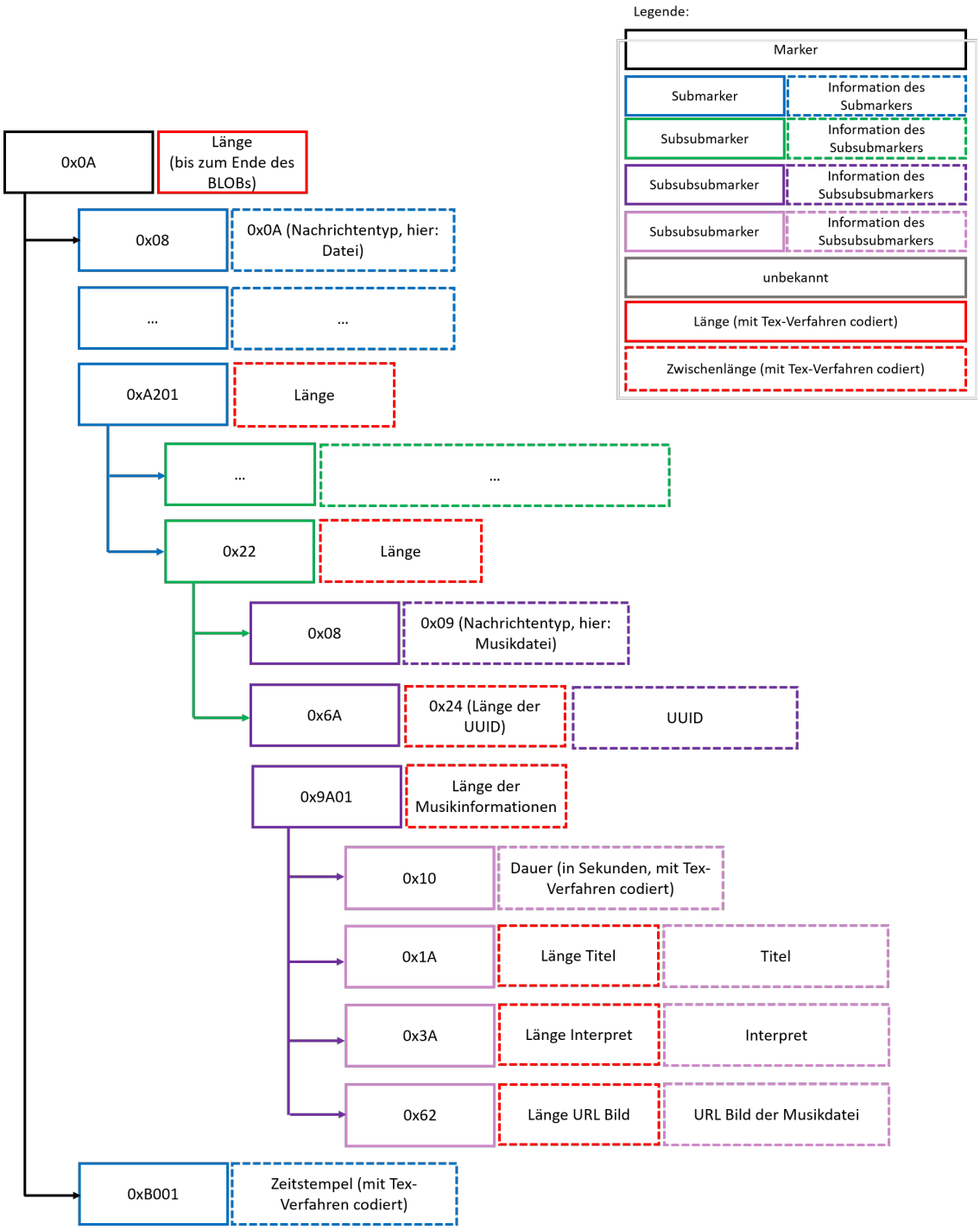


Abbildung B.14: Schema des BLOBs *msg_attaches* bei Musikdateien

Anrufe (*msg_media_type*: 8)

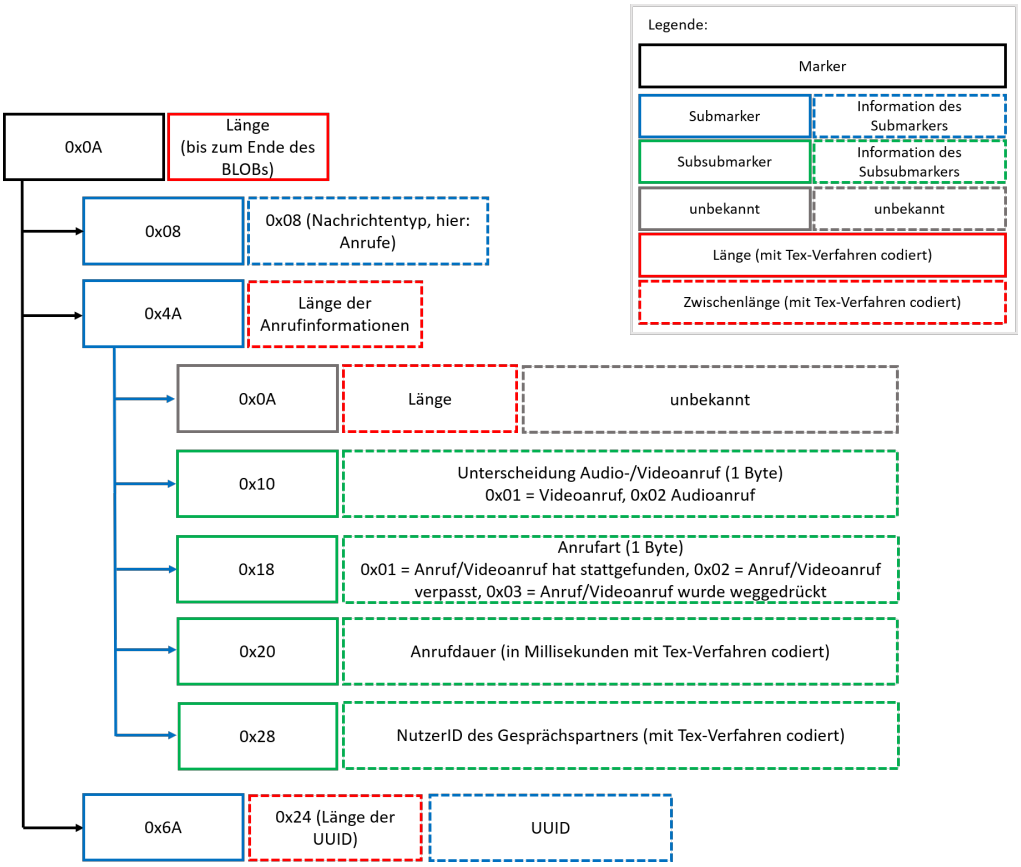


Abbildung B.15: Schema des BLOBs *msg_attaches* bei Anrufen

Standorte (*msg_media_type*: 9)

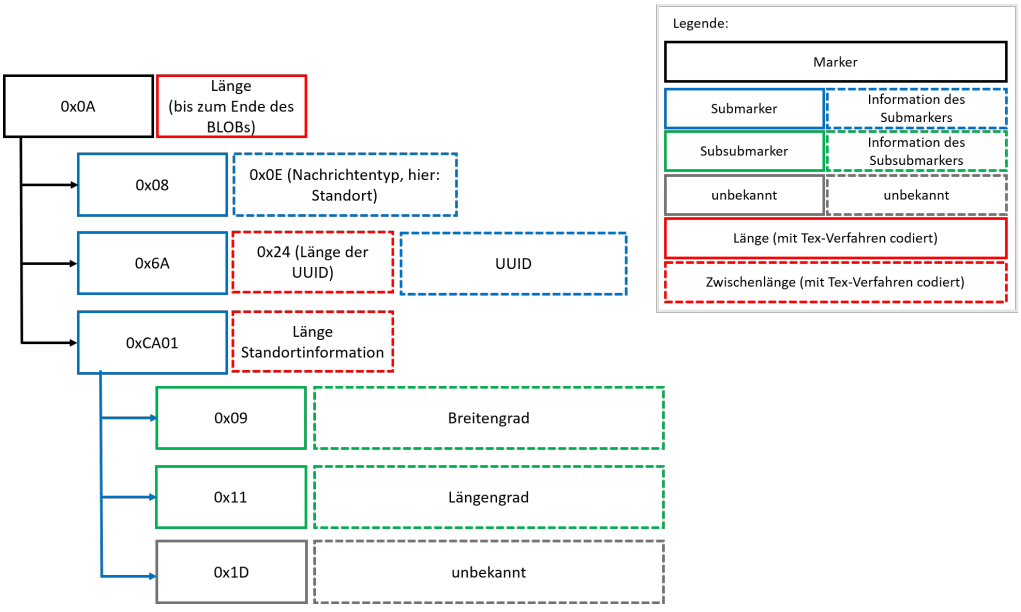


Abbildung B.16: Schema des BLOBs *msg_attaches* bei Standorten

B.3 Ergebnisse der Aufbereitung des Messengers TamTam (v2.6.0) auf Android-Geräten

Der Quelltext des Skripts („aufbereitung_cache_db.py“), welches zur automatischen Auswertung und Aufbereitung für diese Arbeit erstellt wurde, befindet sich auf der beiliegenden CD. Zusätzlich enthält diese CD auch eine exemplarische Interpretation einer *cache.db* (Aufbereitung_TamTam.html). Damit das Skript ausgeführt werden kann, muss diesem bei seinem Aufruf der Name der zu analysierenden Datenbank als Parameter übergeben werden. Zur Zuordnung und Darstellung der Mediendateien sollte sich zudem der exportierte Ordner *media/0* aus dem Pfad *userdata (ExtX)/Root/media/0* im gleichen Ordner wie das Skript befinden.

B.4 Vergleich der Interpretationen mittels Skript sowie durch Magnet Axiom

Die hier dargestellten Screenshots zeigen exemplarisch die Aufbereitungsergebnisse von TamTam auf Android-Geräten in der Version 2.6.0 mit Hilfe des für diese Arbeit erstellten Skripts sowie mittels Magnet Axiom in der Version v3.5.0.15453. Zum einen werden die bekannten Nutzerinformationen und zum anderen die Informationen und Nachrichten der Konversation „Chat-Titel“ präsentiert. Einen Screenshot dieser Konversation von dem Mobiltelefon „Hannah Wie“ zeigt Abbildung B.1.

B.4.1 Nutzerinformationen

Aufbereitung mittels Skript

Nr.	Name (ID_Tex / ID_Dez)	Telefonnummer	Beschreibung	selbstgewählter Link
1 (Account-inhaber)	Hannah Wie (f0fbdbdbb210 / 563369934320)		Ich bin Hannah	
2	Finja Wie (dce0889bd810 / 573434900572)			https://tt.me /FinjasLink
3	Dana Wie (95e1c7d9e510 / 577055617173)			
4	Lea Wie (b4e69ce09611 / 590222930740)	+49 174 [redacted]		
5	Birthe Wie (dbe58ca8e510 / 576951890651)		Ich bin Birthe und habe meine Beschreibung verändert	https://tt.me /BirthesLink
6	Jule Wie (f3f1eea1b410 / 563785480435)			

Nr.	Profilbild	zuletzt online	Profil zuletzt aktualisiert	Details
1	Profilbild 1 Profilbild 2	2019-10-23 13:42:01	2019-09-25 10:25:45	
2	Profilbild 1 Profilbild 2	2019-09-18 12:48:57	2019-09-06 09:38:47	
3	Profilbild 1* Profilbild 2	2019-09-25 11:07:28	2019-09-06 16:16:07	
4		2019-09-13 12:47:59	2019-09-16 11:12:33	
5	Profilbild 1 Profilbild 2	2019-09-13 12:46:23	2019-09-06 09:37:51	
6	Profilbild 1 Profilbild 2	2019-09-25 11:06:05	2019-09-25 11:04:56	Dieser Nutzer wurde von dem Accountinhaber blockiert

*Exemplarisch: URL1 des Profilbilds von Finja Wie: https://i.mycdn.me/image?id=879573676892&ts=000000001f005202c7&plc=API&aid=1150827264&tkn=*JnfxP0pYwnRlu13Fj_JJyvj1qS8&fn=w_1440

Abbildung B.17: Aufbereitung der Nutzerinformationen mit Skript

Aufbereitung mittels Magnet Axiom

Speicherort	Kontakt-ID	Profilname	Website-URL	Über – Info
Table: contacts(_id: 1)	563369934320	Hannah Wie		Ich bin Hannah
Table: contacts(_id: 2)	573434900572	Finja Wie	https://tt.me/FinjasLink	
Table: contacts(_id: 3)	577055617173	Dana Wie		
Table: contacts(_id: 4)	590222930740	Lea Wie		
Table: contacts(_id: 5)	576951890651	Birthe Wie	https://tt.me/BirthesLink	Ich bin Birthe und habe meine Beschreibung verändert
Table: contacts(_id: 6)	563785480435	Jule Wie		

Speicherort	Avatar-URL	Aktualisiert – Datum/Zeit
Table: contacts(_id: 1)	https://i.mycdn.me/image?id=894210779888&ts=0000000014023402b1&plc=API&aid=1150827264&tkn=*coDDhyJ2Wsy95i4Y8xmZ9LNA2MA	23.10.2019 11:42:01
Table: contacts(_id: 2)	https://i.mycdn.me/image?id=879573676892&ts=000000001f005202c7&plc=API&aid=1150827264&tkn=*JnfxP0pYwnRlu13Fj_JJyvj1qS8	18.09.2019 10:48:57
Table: contacts(_id: 3)	https://i.mycdn.me/image?id=891762661781&ts=0000000000000ba04ec&plc=API&aid=1150827264&tkn=*jsL4t06S9s37Cqn5ckQYGNEi58M	25.09.2019 09:07:28
Table: contacts(_id: 4)		13.09.2019 10:47:59
Table: contacts(_id: 5)	https://i.mycdn.me/image?id=885830171355&plc=API&aid=1150827264&tkn=*KAq1IpP0aY5ly1IgXOAq7Lkv2a0	13.09.2019 10:46:23
Table: contacts(_id: 6)	https://i.mycdn.me/image?id=888562023667&plc=API&aid=1150827264&tkn=*KmQp8NcruxnuFKywmFVOuVZeusk	25.09.2019 09:06:05

Abbildung B.18: Aufbereitung der Nutzerinformationen mit Magnet Axiom

B.4.2 Die Konversation „Chat-Titel“ (ID: 53)

Aufbereitung mittels Skript

ID	Name	Typ	Art	Link	Beschreibung	Gründer	Erstellung
53	Chat-Titel	Gruppe	privat	https://tt.me/join/knA2h1kUcoChW2DjqxaodfCLvQCVEdMyhCLvp91JE0w	Beschreibung	Hannah Wie	2019-08-02 16:09:29

ID	Teilnehmer-anzahl	Teilnehmer (letzte Nutzeraktivität)	(Super)Administratoren	Rollen und Rechte von (Super)Administratoren	Profilbild	ID der ersten Nachricht	ID der letzten Nachricht	Details
53	3	**Dana Wie** (2019-08-23 13:50:07) **Hannah Wie** (2019-08-23 13:50:18) **Finja Wie** (2019-08-23 13:53:11)	Hannah Wie Finja Wie Dana Wie	**Dana Wie:** Administrator Die Rechte sind: Nachrichten anheften; Teilnehmer hinzufügen und entfernen; Nachrichten löschen In die Konversation eingeladen durch: Finja Wie **Hannah Wie:** Eigentümer **Finja Wie:** Superadministrator In die Konversation eingeladen durch: Hannah Wie	Profilbild 1 Profilbild 2	523	687	

Abbildung B.19: Übersicht über die Konversation „Chat-Titel“ nach einer Aufbereitung mit Skript

Konversation 53: Chat-Titel (Gruppe) - 3 Teilnehmer, davon bekannt: Dana Wie, Hannah Wie, Finja Wie

ID	Verfasser	Datum	Nachrichtenrichtung	Bearbeitung	Kategorie
523	Hannah Wie	(von TamTam-Server empfangen: 2019-08-02 16:09:29)	**gesendet** (gelesen von mindestens einer Person)		Systemnachricht
524	Hannah Wie	(von TamTam-Server empfangen: 2019-08-02 16:09:46)	**gesendet** (gelesen von mindestens einer Person)		Systemnachricht
525	Hannah Wie	(von TamTam-Server empfangen: 2019-08-02 16:12:31)	**gesendet** (gelesen von mindestens einer Person)		Systemnachricht
685	Hannah Wie	2019-08-23 13:47:09 (von TamTam-Server empfangen: 2019-08-23 13:47:09)	**gesendet** (gelesen von mindestens einer Person)		Textnachricht
686	Finja Wie	2019-08-23 13:48:33	**empfangen**		Textnachricht
687	Dana Wie	2019-08-23 13:50:06	**empfangen**		Textnachricht

ID	Inhalt	Details
523		Gruppe erstellt. Die Teilnehmer zum Zeitpunkt der Erstellung sind: Dana Wie, Finja Wie, Hannah Wie
524		Hannah Wie hat den Namen der Konversation auf 'Chat-Titel' geändert
525		Hannah Wie hat das Profilbild der Konversation auf Profilbild URL 1 Profilbild URL 2 geändert
685	HW-Nachricht001-Chat-Titel	
686	FW-Nachricht002-Chat-Titel	
687	DW-Nachricht003-Chat-Titel	

Abbildung B.20: Nachrichten der Konversation „Chat-Titel“ nach einer Aufbereitung mit Skript

Aufbereitung mittels Magnet Axiom

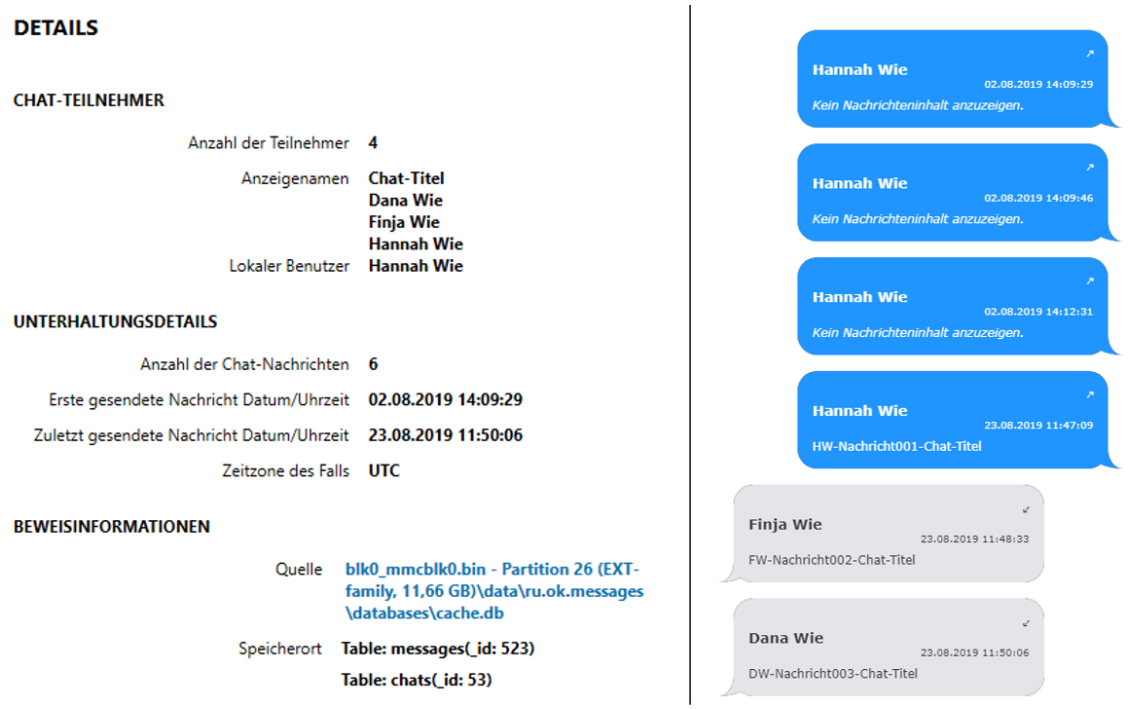


Abbildung B.21: Übersicht über die Konversation „Chat-Titel“ nach einer Aufbereitung mit Magnet Axiom

Speicherort			Absender	Absender-ID	Empfänger	Empfänger-ID
Table: messages(_id: 523), Table: chats(_id: 53)			Hannah Wie	563369934320	Chat-Titel	-71147483584240
Table: messages(_id: 524), Table: chats(_id: 53)			Hannah Wie	563369934320	Chat-Titel	-71147483584240
Table: messages(_id: 525), Table: chats(_id: 53)			Hannah Wie	563369934320	Chat-Titel	-71147483584240
Table: messages(_id: 685), Table: chats(_id: 53)			Hannah Wie	563369934320	Chat-Titel	-71147483584240
Table: messages(_id: 686), Table: chats(_id: 53)			Finja Wie	573434900572	Chat-Titel	-71147483584240
Table: messages(_id: 687), Table: chats(_id: 53)			Dana Wie	577055617173	Chat-Titel	-71147483584240

Speicherort	Nachricht	Nachricht – Zeitstempel – Datum/Zeit	Status	Nach...
Table: messages(_id: 523), Table: chats(_id: 53)		02.08.2019 14:09:29	Sent	Text
Table: messages(_id: 524), Table: chats(_id: 53)		02.08.2019 14:09:46	Sent	Text
Table: messages(_id: 525), Table: chats(_id: 53)		02.08.2019 14:12:31	Sent	Text
Table: messages(_id: 685), Table: chats(_id: 53)	HW-Nachricht001-Chat-Titel	23.08.2019 11:47:09	Sent	Text
Table: messages(_id: 686), Table: chats(_id: 53)	FW-Nachricht002-Chat-Titel	23.08.2019 11:48:33	Received	Text
Table: messages(_id: 687), Table: chats(_id: 53)	DW-Nachricht003-Chat-Titel	23.08.2019 11:50:06	Received	Text

Abbildung B.22: Nachrichten der Konversation „Chat-Titel“ nach einer Aufbereitung mit Magnet Axiom

Erklärung

Hiermit erkläre ich, dass ich meine Arbeit selbstständig verfasst, keine anderen als die angegebenen Quellen und Hilfsmittel benutzt und die Arbeit noch nicht anderweitig für Prüfungszwecke vorgelegt habe.

Stellen, die wörtlich oder sinngemäß aus Quellen entnommen wurden, sind als solche kenntlich gemacht.

Düsseldorf, 15. November 2019