

**Hochschule Wismar**

University of Applied Sciences Technology, Business and Design

Fakultät für Ingenieurwissenschaften



---

Fakultät für Ingenieurwissenschaften

**Master-Thesis**

Automotive Forensik - Forensische Analyse gängiger Car  
Infotainment Systeme

Abschlussarbeit zur Erlangung des Grades eines

**Master of Engineering**

der Hochschule Wismar

Autor:

Jens Knispel

Studiengang:

IT-Sicherheit und Forensik



# **Hinweise zur Veröffentlichung**

An dieser Stelle möchte ich im Rahmen der Veröffentlichung dieser Masterthesis darauf hinweisen, dass sämtliche personenbezogene Daten für die Ergebnis-Präsentation in dieser Arbeit anonymisiert worden sind. Für die in dieser Arbeit dargestellten hersteller- und personenbezogene Daten bedeutet dies, dass Namen, Telefonnummern, Adressen und weitere Daten mit Zunahme eines Zufallsgenerators erstellt wurden. Das betrifft beispielsweise die Daten innerhalb der ermittelten Adressbuch-, der Medien-, der Navi-Datenbank und weitere Daten innerhalb der Analyse und Present Phase. Zusätzlich sind einige Stellen unkenntlich gemacht, sodass auch in diesen Teilen kein Bezug zu Herstellern und Einzelpersonen gemacht werden können.

Dadurch soll der Schutz der personenbezogenen Daten gemäß Artikel 4 Absatz 1 DSGVO von natürlichen Personen durch Anonymisierung gewährleistet werden. Sollten die durch den Zufallsgenerator generierten Daten zufällig auf natürliche Einzelpersonen zutreffen, besteht darin absolut keine Absicht.

Ich behalte mir als Urheber dieser Arbeit das Recht vor, Veränderungen an der Arbeit vorzunehmen.

# Kurzfassung

Im Verlauf der vorliegenden Masterthesis wird im Rahmen der automotiven Forensik eine vollständige logische Auswertung eines PKW Infotainmentsystems erfolgen. Untersucht werden sowohl die im Infotainmentsystem befindlichen Daten als auch das Betriebssystem. Auf Basis dieser Untersuchung werden die erzielten Ergebnisse zusammengefasst und präsentiert. Zusätzlich wird ein Ansatz für eine physikalische forensische Untersuchung erarbeitet und dargestellt.

Im Fokus der Konzeptphase steht ein agiles und iteratives Vorgehen, das eine forschende Herangehensweise unterstützt. Darüber hinaus werden Konzepte für die einzelnen Schritte der forensischen Untersuchung des Datenträgers, der Sicherheitsfunktion „Schreibschutz“ und des physikalischen Ansatzes festgelegt.

Die Analysephase zeichnet sich durch die forensische Untersuchung des Infotainment-systems aus. Abgeschlossen wird die forensische Untersuchung mit der Präsentation der erzielten Ergebnisse. Außerdem wird der Schreibschutz des Dateisystems umgangen und ein Ansatz für zukünftige physikalische forensische Auswertungen entwickelt. Darüber hinaus werden Hinweise auf Themen für mögliche weitere Forschungsarbeiten im Bereich des Betriebssystems Blackberry QNX Neutrino Realtime Operation System (QNX Neutrino RTOS) und der automotiven Forensik gegeben.

Abschließend wird diese Arbeit in kurzer Form zusammengefasst und bewertet.

# **Abstract**

In the course of this work, a complete logical evaluation of a car infotainment system will be performed in the context of automotive forensics. The data contained in the infotainment system will be examined, as well as the operating system itself. Based on this investigation, the obtained results are summarized and presented. In addition, an approach for a physical forensic investigation will be developed and presented.

The focus of the concept phase is an agile and iterative procedure that supports a research-based approach. Moreover, concepts for the individual steps of the forensic examination of the data carrier, the security function „write protection“ and the physical approach are defined.

The analysis phase is characterized by the forensic investigation of the infotainment system. This is concluded with the presentation of the obtained results and data. Also, the write protection of the file system is bypassed, and an approach for future physical forensic evaluations is developed. Furthermore, hints are given on topics for possible further research in the field of the operating system QNX Neutrino RTOS and automotive forensics.

Finally, this thesis is briefly summarized and evaluated.

# Inhaltsverzeichnis

<b>Hinweise zur Veröffentlichung</b>	<b>1</b>
<b>Kurzfassung</b>	<b>2</b>
<b>Abstract</b>	<b>3</b>
<b>Inhaltsverzeichnis</b>	<b>4</b>
<b>1 Einleitung</b>	<b>6</b>
1.1 Motivation . . . . .	6
1.2 Zielsetzung . . . . .	8
1.3 Abgrenzung . . . . .	8
<b>2 Allgemeine Grundlagen</b>	<b>10</b>
2.1 Digitale Forensik . . . . .	10
2.1.1 Grundlagen und Begriffe . . . . .	11
2.1.2 Anforderungen und Ziele einer IT-forensischen Untersuchung . .	12
2.1.3 Arten der IT Forensik . . . . .	13
2.2 Automotive Forensik - Das vernetzte Fahrzeug . . . . .	14
2.2.1 Fahrzeugsensorik . . . . .	16
2.2.2 Infotainment Systeme . . . . .	18
2.3 Blackberry QNX Neutrino Realtime Operation System . . . . .	19
2.4 SQLite Datenbanken . . . . .	21
2.4.1 Architektur . . . . .	23
2.4.2 Besondere Merkmale . . . . .	23
2.5 Rekonstruktion von SQLite Datenbanken . . . . .	24
2.6 Open Source Software . . . . .	28
<b>3 Konzeption</b>	<b>31</b>
3.1 Forschungsumgebung . . . . .	32
3.1.1 Forensik Maschine . . . . .	32
3.1.2 Virtuelle Maschine (VM) . . . . .	32
3.2 Vorgehensmodelle der IT-Forensik . . . . .	33
3.2.1 Auswahl eines geeigneten Vorgehensmodells der IT Forensik . .	33
3.3 Vorgehensweise der forensischen Untersuchung . . . . .	41
3.3.1 Secure Phase . . . . .	41
3.3.2 Analyse Phase . . . . .	42
3.3.3 Present Phase . . . . .	44

3.4 Softwareauswahl . . . . .	45
3.4.1 Autopsy . . . . .	45
3.4.2 The Sleuth Kit . . . . .	46
3.4.3 bring2lite . . . . .	47
3.4.4 bulk_extractor . . . . .	47
3.4.5 Binwalk . . . . .	48
3.4.6 QEMU . . . . .	49
3.5 Security Ausblick . . . . .	49
3.6 Physikalischer Analyseansatz . . . . .	51
3.7 Vergleichskonzept Linux vs QNX . . . . .	53
<b>4 Analyse</b>	<b>55</b>
4.1 Secure Phase . . . . .	56
4.2 Analyse Phase . . . . .	57
4.2.1 Automatisierte Analyse . . . . .	57
4.2.2 Manueller Analyse Prozess . . . . .	59
4.3 Erzwingen des Schreibzugriffs auf QNX Neutrino RTOS . . . . .	74
4.3.1 Vorbereitung des Images . . . . .	75
4.3.2 VM - QNX System . . . . .	77
4.3.3 Manipulation des Images . . . . .	79
4.4 Weiterführende Analyse Phase . . . . .	81
4.5 Present Phase . . . . .	85
4.6 Virtualisierung . . . . .	93
4.7 Physikalischer Analyseansatz . . . . .	97
4.8 Vergleich Standard-Distribution vs QNX Neutrino RTOS . . . . .	105
4.9 Gefahrenanalyse und Sicherheitsbetrachtung . . . . .	108
<b>5 Fazit &amp; Ausblick</b>	<b>111</b>
5.1 Fazit . . . . .	111
5.2 Ausblick . . . . .	113
<b>Anhang</b>	<b>115</b>
<b>Quellenverzeichnis</b>	<b>124</b>
<b>Abbildungsverzeichnis</b>	<b>130</b>
<b>Tabellenverzeichnis</b>	<b>133</b>
<b>Abkürzungsverzeichnis</b>	<b>134</b>
<b>Thesen</b>	<b>136</b>

# 1 Einleitung

Innerhalb dieses Kapitels werden die Rahmenbedingungen und der thematische Umfang der vorliegenden Masterthesis definiert.

Begonnen wird mit einer Erläuterung zur Motivation für das vorliegende Thema. Anschließend werden die Ziele dieser Arbeit festgelegt und dokumentiert. In dem letzten Abschnitt dieses Kapitels werden Abgrenzungen festgehalten, die den thematischen Umfang dieser Arbeit aufzeigen. Es werden nicht behandelte Umfänge spezifiziert.

## 1.1 Motivation

Der PKW von heute ist schon längst nicht mehr mit dem Fahrzeug von vor 10 Jahren vergleichbar. Nicht nur die Elektromobilität bringt neue Herausforderungen und Fortschritt mit sich. Die Automobilhersteller wollen die Fahrt mit dem Auto zu einem großen Erlebnis mit viel Komfort gestalten. Diesbezüglich wird von dem Kunden ein sehr hohes Maß an Digitalisierung und Vernetzung erwartet.

Somit soll das Fahrzeug, ohne den Schlüssel einzustecken, starten und die kommende Fahrstrecke unter Einbeziehung von Staudaten vom Navigationssystem berechnet werden. Darüber hinaus sollen Telefonate vom Mobiltelefon auf das Freisprechsystem des Fahrzeugs übertragen werden und die favorisierte Musik über den Streaming-Dienst unterhalten. Auch das Abrufen der E-Mails, der Wetterdaten und die Parkplatzsuche in der Stadt soll vollkommen automatisiert über das Infotainmentsystem funktionieren. Dies sind nur einige Beispiele für die innovativen Möglichkeiten durch die Vernetzung des Fahrzeugs mit dem Internet oder dem Smartphone des Fahrers.

In den meisten aktuellen Fahrzeugen sind Bluetooth, Wireless Lan und ein Telemodul mit einer SIM-Karte verbaut. Dadurch steht das Fahrzeug 24 Stunden 7 Tage in der Woche mit der Hersteller-Cloud in Verbindung. Das zeigt, dass die heutigen Akteure nicht nur auf den Hersteller und den Besitzer des Fahrzeugs zu reduzieren sind, sondern auch Streaming-Dienste, soziale Netzwerke und viele weitere digitale Dienste und Akteure hinzugekommen sind.

Jedoch entstehen nicht nur Daten durch die Mensch-Maschine Interaktion. Auch die unzähligen Fahrzeugsensoren generieren viele wertvolle Daten. Dazu gehören zum Beispiel der Zeitpunkt des letzten Motorstarts, Spritverbrauch, durchschnittliches und maximales Drehmoment des Motors, Drehzahlen, Position des Gaspedals und viele weitere

## **1.1. MOTIVATION**

---

Merkmale. Diese fahrzeugbezogenen Daten lassen sich bei ausreichender Anzahl sehr genau einem Fahrer zuordnen, was diese Daten zu personenbeziehbare Daten macht. Aus diesen Daten lassen sich von dem Hersteller, Vollzugsbehörden und Versicherungen wertvolle Erkenntnisse ziehen.

Viele Menschen interagieren täglich mit dem Infotainmentsystem ihres Fahrzeugs, die dadurch anfallenden Daten sind für die forensische Untersuchung ähnlich interessant, wie die Daten des Smartphones selbst. Das bedeutet, dass durch die forensische Untersuchung eines Personenkraftwagens sehr viele Daten und Erkenntnisse über Personen gesammelt werden können. Der große Datenumfang aus den unterschiedlichen Quellen und deren Auswertungsmöglichkeiten legen eine spannende und attraktive Forschungsarbeit nahe. Zum heutigen Zeitpunkt sind alleine mit den Blackberry Systemen mehr als 150 Millionen Fahrzeuge ausgestattet. Dies zeigt die enorme Bedeutung, die durch forensische Untersuchungen von Blackberrys Infotainmentsystemen erreicht werden kann.

Die neuen Technologien der Kommunikation zwischen Fahrzeug und Hersteller und der Mensch-Maschine Interaktion mittels WLAN- und Bluetooth-Verbindungen schaffen nicht nur Komfort, sondern bringen auch Risiken und Herausforderungen. Die auf dem Fahrzeug befindlichen personenbezogenen, personenbeziehbaren und fahrzeugbezogenen Daten können in einem hohen Maße sensibel sein und müssen deswegen geschützt werden.

Aus diesem Grund werden im letzten Kapitel dieser Masterthesis mögliche Angriffsvektoren auf Grundlage der Analyseergebnisse behandelt. Dabei werden aktuell existierende und bekannte Risiken, aber auch zukünftig denkbare Angriffsvektoren betrachtet. Neben den Angriffsvektoren von außen können auch durch die vielen neuen digitalen Akteure Risiken entstehen. Darüber hinaus wird aus verschiedenen Blickwinkeln beleuchtet, welche Möglichkeiten die Analyse der Daten eines PKWs für einzelne Akteure mit sich bringen kann und inwiefern diese Daten in verschiedenen Situationen genutzt werden können.

Die Daten aus dem Fahrzeug vermitteln ähnlich viele Informationen über Personen wie ein Smartphone. Das macht die automotive Forensik, also forensische Untersuchung von Fahrzeugen und deren Infotainmentsystemen, ebenso unabdingbar und wichtig wie die Smartphone-Forensik.

Die forensische Untersuchung wird zeigen, wie stark die Daten geschützt werden, welche Daten auslesbar sind und wie ein Zugriff von Extern verhindert werden soll.

## 1.2 Zielsetzung

In der vorliegenden Masterthesis mit dem Titel „Automotive Forensik – Forensische Analyse gängiger Car Infotainment Systeme“ soll eine forensische Untersuchung gängiger Car Infotainment Systeme durchgeführt werden. In Bezug auf die Forensik sind bisher keine Forschungsarbeiten öffentlich zugänglich, weshalb keine Informationen zu diesem Themenkomplex verfügbar sind.

Aus diesem Grund soll der Fokus dieser Forschungsarbeit nicht nur auf den eigentlichen Daten innerhalb des Infotainmentsystems, sondern auch auf dem Betriebssystem selbst liegen. Einerseits wird untersucht, welche Daten in welchen Formaten auf dem Infotainmentsystem vorliegen und andererseits der Aufbau des Betriebssystems. Besonderheiten des Betriebssystems sind herauszustellen und die Funktionsweisen zu erläutern. Bei den analysierten Daten wird der Fokus auf personenbezogene und personenbeziehbare Daten gelegt. Weiterhin soll analysiert werden, inwiefern Daten miteinander verknüpfen werden können, sodass aus diesen Daten ein Mehrwert generiert werden kann.

Aufgrund der Gegebenheiten des Betriebssystems QNX Neutrino RTOS ist nur ein logischer und kein physischer Zugriff auf die Daten des Infotainmentsystems möglich. Deswegen soll geprüft werden, ob SQLite Datenbanken aus dem nicht-allozierten Bereich durch Carving wieder hergestellt werden können. Darüber hinaus soll versucht werden, beschädigte SQLite Datenbanken aus dem Rohdaten zu rekonstruieren. Zusätzlich soll geprüft werden, ob ein physikalischer Analyseansatz ermöglicht werden kann.

Die dritte Teilaufgabe beschäftigt sich mit dem Einsatz des QNX Neutrino RTOS. Im Kontext der forensischen Analyse soll untersucht werden, welche Vor- und Nachteile durch das Einsetzen des Betriebssystems entstehen. Anschließend soll anhand der Ergebnisse ein Vergleich zwischen dem Betriebssystem QNX Neutrino RTOS und den Linux-Standard Distributionen, wie beispielsweise Debian oder Arch, durchgeführt werden. Abschließend soll evaluiert werden, ob es gewinnbringend für den Forensiker sein würde, ein QNX Neutrino RTOS für seine Arbeit bereitzuhalten.

## 1.3 Abgrenzung

Diese Masterthesis stellt eine Forschungsarbeit dar, die die Grenzen des aktuellen Standes der Technik im Kontext der Forensik und des Echtzeit Betriebssystems QNX Neutrino auslotet.

Neben der Untersuchung des Betriebssystems und der im Infotainmentsystem gespeicherten Daten finden gängige Praktiken und Techniken der Forensik Anwendung. Durch die

### **1.3. ABGRENZUNG**

---

damit im Zusammenhang stehenden Versuche werden einige Grenzen bei der forensischen Untersuchung des QNX Neutrino RTOS aufgezeigt.

Ziel dieser Arbeit ist es nicht sämtliche Software und Techniken zu entwickeln, die diese Probleme abschließend lösen. Viel mehr werden verschiedene Ansätze dokumentiert und auf ihre jeweilige Funktionsweise hin geprüft. Aus diesem Grund finden sich in einigen Kapiteln Lösungsvorschläge zu zukünftigen Herausforderungen, ohne diese vollständig zu erproben und auszuformulieren.

## 2 Allgemeine Grundlagen

Das Kapitel „Allgemeine Grundlagen“ unterteilt sich in die folgenden sechs Unterkapitel

- Digitale Forensik
- Automotive Forensik - Das vernetzte Fahrzeug
- QNX Neutrino Realtime Operation System (RTOS)
- SQLite-Datenbanken
- Datenbank-Rekonstruktion
- Open Source Software

und erläutert die Grundlagen für diese Arbeit.

Im ersten Abschnitt werden Begriffe der digitalen Forensik und verschiedene Vorgehensmodelle dieses Handlungsfeldes aufgezeigt. Im Anschluss gibt der zweite Abschnitt eine Übersicht über die Bedeutung der automotiven Forensik und verschiedenen Komponenten im Personenkraftwagen (PKW), die für eine forensische Untersuchung zentral sind. Es werden im Kontext des heutigen Standes der Technik Komponenten beleuchtet, die mit dem Fahrer, dem Beifahrer oder mit dem Fahrzeugherrsteller kommunizieren.

Im Weiteren werden das Betriebssystem QNX Neutrino RTOS, sowie Herausstellungsmerkmale des Betriebssystems betrachtet. Anschließend wird das grundlegende Verständnis für SQLite-Datenbanken, der Datenbank-Rekonstruktionen und das Wiederherstellen von gelöschten Datenbankeinträgen hergeleitet und auf die technischen Verfahrensweisen und Spezifikationen eingegangen. Abschließend wird der Begriff **Open Source Software** (OSS) definiert und dessen Voraussetzungen dargestellt.

### 2.1 Digitale Forensik

Folgend werden die grundlegenden Begriffe der IT-Forensik erläutert und Zusammenhänge beschrieben. Als Grundlage für die im Kapitel 3.2 beschriebenen Konzepte wird die CERT-Taxonomie der digitalen Forensik vorgestellt. Darüber hinaus werden die Ziele und Anforderungen an eine IT-forensische Untersuchung beleuchtet. Abschließend wird auf die Unterschiede der möglichen Arten einer IT-forensischen Untersuchung eingegangen.

### 2.1.1 Grundlagen und Begriffe

Der Begriff Forensik leitet sich aus dem Lateinischen „*forensis*“ ab und bedeutet „*dem Marktplatz zugehörig*“. Dies ist darauf zurückzuführen, dass unter anderem Gerichtsverhandlungen, Strafvollzug und weitere juristische Verhandlungen im antiken Rom öffentlich auf den Marktplätzen abgehalten wurden. Nach heutiger Auffassung beschreibt die *Forensik* Aufgabengebiete, die sich mit der Untersuchung von kriminellen Handlungen beschäftigt [1].

Die IT Forensik, digitale Forensik oder auch Computer Forensik, ist ein Teilgebiet der Forensik und beschäftigt sich mit der methodischen Aufklärung von Straftaten in Bezug auf IT Systeme und Cyberkriminalität. Kriminalistische Vorfälle werden identifiziert, analysiert, Beweise gesammelt und gerichtsverwertbar dokumentiert. Wodurch Tathergänge rekonstruiert, Annahmen falsifiziert oder verifiziert werden können, um mögliche Täter zu ermitteln [1].

„Computer Forensik“ und „digitale Forensik“ sind Synonyme für den Begriff der „IT Forensik“ und werden innerhalb dieser Arbeit ebenso verwendet [2]. Die vorangegangene Definition wird von dem Bundesministerium für Sicherheit in der Informationstechnologie (BSI) durch, die „... *Einbeziehung der Möglichkeiten der strategischen Vorbereitung insbesondere aus der Sicht des Anlagenbetreibers eines IT-Systems*“ [3], ergänzt. Diese Erweiterung hat einen positiven Einfluss auf die Ergebnisqualität der forensischen Untersuchung, weil präventive Maßnahmen ergriffen sowie akzeptierte Vorgehensweisen und Techniken angewendet werden können, weit bevor eine Sicherheitsverletzung (Vorfall) auftritt [3].

Für die erfolgreiche Durchführung einer Untersuchung eines Vorfalles muss dieser vorerst klar definiert sein. Die CERT-Taxonomie bildet dabei die systematische Beschreibung und eine einheitliche Definition zur Klassifizierung von Vorfällen [2].

Grundsätzlich wird der Vorfall in drei Bereiche unterteilt: Vorfall, Angriff und Ereignis.

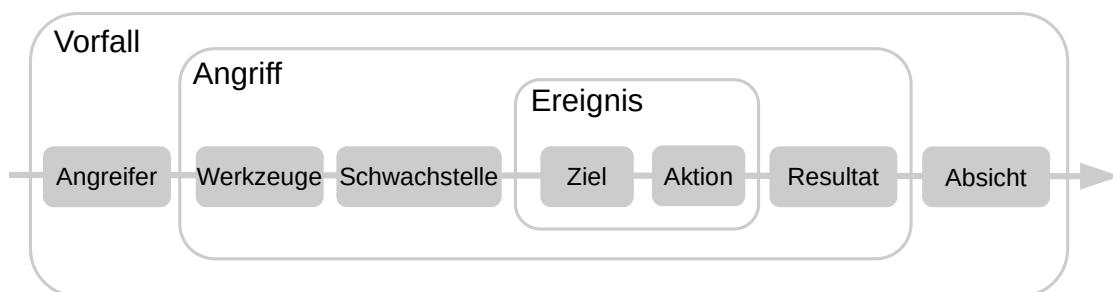


Bild 1: Visualisierung der CERT-Taxonomie [2]

## 2.1. DIGITALE FORENSIK

---

Dem **Bild 1** ist zu entnehmen, dass für einen IT-forensischen Vorfall ein Angreifer und eine Absicht vorliegen müssen. Der Angriff bedingt neben der vom Angreifer forcierten Schwachstelle im IT-System auch die verwendeten Werkzeuge, die zu dem Erreichen des Resultats führen. Dabei adressiert die CERT-Taxonomie eine vorsätzliche Handlung, welche als Ereignis im Mittelpunkt steht. Diese bildet sich aus der Aktion und dem Ziel, das der Angreifer verfolgt. Die CERT-Taxonomie dient ebenso als Grundlage der Dokumentation, wie sie zugleich den möglichen Angriffsverlauf darstellt [2, 3].

### 2.1.2 Anforderungen und Ziele einer IT-forensischen Untersuchung

#### Ziele

Ziel einer jeder forensischen Untersuchung ist es, Spuren aufzusuchen und Thesen an Hand von Beweisen zu untermauern oder zu widerlegen. Von besonderer Bedeutung für diesen Prozess ist die Verwendung allgemein anerkannter Methoden und Vorgehensweisen für die forensische Untersuchung. Dafür muss ermittelt werden, was innerhalb des Netzwerks beziehungsweise (bzw.) des IT Systems geschehen ist [3].

Anschließend soll geklärt werden, wo der Angriff passiert ist. Dies kann sich auf einzelne Geräte als auch auf eine gesamte IT Infrastruktur beziehen. Weiterhin soll belegt werden, wann der Vorfall erfolgte, mit welchen Werkzeugen dieser durchgeführt wurde und welche Schwachstellen im System ausgenutzt wurden [2, 3].

Im Kontext der Strafverfolgung kommt die Frage nach dem Täter hinzu, die ebenfalls durch die forensische Untersuchung beantwortet werden soll. Darüber hinaus müssen die Beweise juristisch verwertbar dokumentiert und das Schadensausmaß festgestellt werden [3].

#### Anforderungen

Aus den oben genannten Zielen lassen sich die in **Tabelle 1** dargestellten Anforderungen an die IT-forensische Untersuchung ableiten:

Anforderung	Erläuterung
Akzeptanz	Die angewendeten Analyseschritte und -methoden müssen allgemein akzeptiert sein [4].
Glaubwürdigkeit	Die Funktionalität und Stabilität der verwendeten Methoden muss gegeben und im Zweifelsfall nachgewiesen werden können [4].
Wiederholbarkeit	Ein Dritter muss den gesamten Untersuchungsprozess wiederholen können und dabei dieselben Resultate erlangen [4].

Fortsetzung auf der nächsten Seite

## 2.1. DIGITALE FORENSIK

---

Fortsetzung der vorherigen Seite	
Anforderung	Erläuterung
Integrität	Die Spuren dürfen nicht unbemerkt manipuliert werden. Zu jedem Zeitpunkt der forensischen Untersuchung gilt, dass die Integrität der digitalen Spuren bewiesen werden kann [3].
Ursache & Auswirkung	Es müssen logisch nachvollziehbare Verbindungen zwischen Beweisen, Ereignissen und Personen bestehen [4].
Dokumentation	Der gesamte Untersuchungsprozess muss lückenlos, juristisch verwertbar und jeden einzelnen Prozessschritt beinhaltend dokumentiert werden [4].

Tabelle 1: Anforderungen der forensischen Untersuchung

Besondere Berücksichtigung gilt der Integrität der Spuren, der Authentizität der erhobenen Daten und der Vorgehensweise des Ermittlers. Die lückenlose Dokumentation muss für außenstehende Personen reproduzierbar und nachvollziehbar sein. Zudem muss eine Manipulation der Ergebnisse oder auch der digitalen Spuren vom ersten bis zum letzten Prozessschritt der Analyse nachvollziehbar sein [3].

Besonders empfohlen werden Open Source Werkzeuge zur Durchführung der Analyse, da deren Funktionen zusätzlich am Quellcode geprüft und sichergestellt werden können [4].

### 2.1.3 Arten der IT Forensik

Grundsätzlich werden in der IT Forensik die Untersuchungsformen Live-Response von der Post-Mortem Analyse unterschieden.

Eine Untersuchung im laufenden Betrieb wird Live-Response Analyse genannt. Diese Form der Forensik erlaubt die Sicherung von flüchtigen und temporären Daten, weshalb der Sicherungsreihenfolge der Daten ein besonderer Stellenwert zu kommt. Hierbei sollte die Sicherungsreihenfolge immer anhand des Grades der Veränderung durch die Sicherung selbst und der Halbwertszeit der Daten entschieden werden. Flüchtige Daten sind Daten, welche nur zur Laufzeit des Systems existieren und bei dem Herunterfahren unwiderruflich gelöscht werden. Das trifft beispielsweise auf CPU-Register, CPU-Cache, Routingtabellen, den Arbeitsspeicher oder auch Kernel Statistiken zu [3, 4].

Temporäre Daten hingegen sind zwar dauerhaft existent, können jedoch nur zur Laufzeit ausgelesen werden, ein Beispiel dafür wären geöffnete, echtzeitverschlüsselte Krypto-

Container<sup>1</sup> [3]. Diese Form der Analyse setzt voraus, dass das System nach dem Vorfall nicht schon mehrfach neugestartet wurde, sowie besonders interessante flüchtige Daten enthält (zum Beispiel Passwörter im Arbeitsspeicher, o.Ä.) oder das System kritische Anwendungen bereitstellt, die nicht heruntergefahren werden dürfen. Zusätzlich sollte der Forensiker auf den Start von Programmen und das Ausführen von Kommandos innerhalb des Systems verzichten. Damit wird verhindert, dass sich einerseits der Systemzustand verändert und andererseits der Start von Programmen, die der Angreifer manipuliert haben könnte, aktiviert wird [4].

Das Gegenteil von der Live-Response Untersuchung ist die Post-Mortem Analyse. Mit Hilfe dieser Untersuchungsart werden hauptsächlich persistente Daten und Spuren erfasst und analysiert. Dadurch bietet diese Form der Analyse den Vorteil nicht ständig in Gefahr zu laufen, dass zu untersuchende System zu verändern oder flüchtige Daten (Beweise) zu verlieren. Ein weiterer Vorteil ist, dass diese nicht zeitkritisch ist und von mehreren Forensikern parallel abgearbeitet werden kann [6].

Besonderes Augenmerk muss bei der Post-Mortem Analyse auf die Datensicherung gelegt werden, denn Daten, die nicht gesichert werden, sind verloren. Ebenso darf auch bei der Datensicherung keine Veränderung am Quellsystem entstehen [4].

Der erste Schritt der Post-Mortem Analyse ist die Bildung und Dokumentation von Hashsummen über die Quelldateien. Anschließend wird ein Abbild, eine bitgenaue Kopie des Quellspeichermediums erstellt und dessen Hashsumme berechnet. Darauf folgt der Vergleich der beiden Hashsummen, die sich exakt gleichen müssen. Sind die Kopie und das Original identisch, kann mit der Analyse der Images begonnen werden [6].

## 2.2 Automotive Forensik - Das vernetzte Fahrzeug

Die „Automotive Forensik“, „KFZ-Forensik“<sup>2</sup> oder auch „Vehicle Forensic“ genannt, ist ein Teilgebiet der IT-Forensik. Dieses Teilgebiet beschäftigt sich mit der Spurensuche und Wiederherstellung von digitalen Beweisen und Daten, die von Steuergeräten, Fahrzeug-Netzwerken und der Interaktion Mensch Maschine erzeugt werden [7].

Das Fachgebiet der KFZ-Forensik ist sehr komplex und setzt Fachwissen über die Funktionsweise und den Aufbau von Steuergeräten, Fahrerassistenzsystemen, Telematik-

---

<sup>1</sup>Ein Krypto-Container ist eine Datei, die ein verschlüsseltes virtuelles Laufwerk enthält. Im ausgehängten Zustand ist der gesamte Container als einzige Datei dargestellt und kann eben so behandelt werden. Wird dieser eingebunden, ist der Krypto-Container ein virtuelles Laufwerk bzw. Verzeichnis, dass sich in der Handhabung nicht von anderen unterscheidet [5].

<sup>2</sup>Kraftfahrzeug (KFZ)

Modulen und der Sicherheits-Funktionen und -Methoden der eingesetzten Betriebssysteme voraus.

Von einem „vernetzten Fahrzeug“ wird gesprochen, wenn eine informationstechnologische Verbindung mit diesem hergestellt werden kann. Das können WLAN- und Bluetooth-Verbindungen mit dem Smartphone oder auch Verbindungen über die im Fahrzeug verbaute SIM<sup>3</sup>-Karte sein [8].

Im direkten Zusammenhang mit dem Begriff des vernetzten Fahrzeugs steht die Thematik „Car-IT“, zu Deutsch „Fahrzeug-IT“. Diese Thematik ist in der Wissenschaft noch sehr jung und aus diesem Grund gibt es keine festen Definitionen. Ein erster Versuch den Begriff der Fahrzeug-IT zu definieren, ist dem Online-Artikel von Peter Rademacher zu entnehmen. Diese lautet wie folgt: „Car-IT betrachtet alle Informationsflüsse, die in das Fahrzeug hinein-, aus dem Fahrzeug heraus- oder im Fahrzeug selbst fließen. Mit dem Ziel, das Fahrzeug beziehungsweise den Fahrer als direkten Informationsempfänger/-lieferanten in erweiterte Geschäftsprozesse und Geschäftsmodelle zu integrieren, unabhängig von Zeitpunkt und Standort des Fahrzeugs.“ [9]

In Bezug auf die Informationsflüsse gibt es einige neuartige Kommunikationsmodelle und Akteure im Zusammenhang mit dem Fahrzeug und den Daten. Vor der Vernetzung von Fahrzeugen spielten, nach dem Verkauf bzw. Kauf eines Fahrzeugs, die folgenden Akteure eine zentrale Rolle: Hersteller, Händler und Fahrer. Neu entstandene Kommunikationsmodelle sind die Car2Car- (C2C), Car2Infrastructure- (C2I), Car2Home- (C2H) und die Car2Enterprise- (C2E) Kommunikation. In **Abbildung 2** sind die neuen Kommunikationsmodelle, sowie die damit verbundenen Systeme und deutlich gewachsene Anzahl der Akteure dargestellt [8].

Studien und Prognosen zeigen, dass sowohl der Marktanteil von vernetzten Fahrzeugen und Services weltweit, als auch der Anteil von Neufahrzeugen mit Internetzugang stark angestiegen sind und weiterhin stark ansteigen werden. Durch die reine Menge und die Aussagekraft dieser Daten werden Begehrlichkeiten auf Seiten von Vollzugsbehörden, Versicherungsgesellschaften und Herstellern geschaffen. Darüber hinaus sind auch Unternehmen mit den Geschäftsfeldern der Big Data Analysen, der Vermarktung von zielgerichteter Werbung und vieler weiterer Geschäftsfelder an diesen Daten interessiert [10, 11].

---

<sup>3</sup>Subscriber Identity Module (SIM)

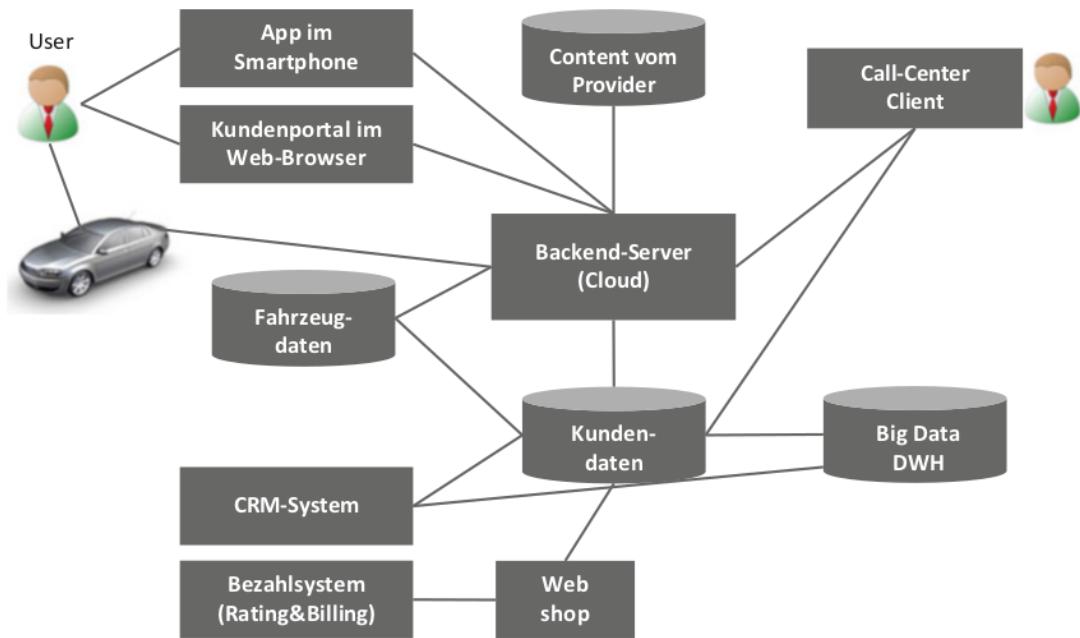


Bild 2: IT Architektur Fahrzeug [8]

Grundsätzlich können die Daten in zwei Kategorien aufgeteilt werden. Dies sind zum einen hauptsächlich personifizierbare Daten generiert durch das Fahrzeugsystem und dessen Sensoren und zum anderen personenbezogene Daten erzeugt durch Interaktion zwischen dem Menschen und den Systemen des Fahrzeugs. Diese werden in den folgenden zwei Abschnitten untersucht und der Mehrwert durch diese Daten im Kontext der Forensik dargestellt.

### 2.2.1 Fahrzeugsensorik

Aus Sicht der Elektronik gliedert sich das Fahrzeug in vier Funktionsbereiche: Antriebsstrang, Chassis, Innenraum und Telematik. Dabei übernimmt die Telematik-Domäne hauptsächlich die Vernetzung von Multimedia- und Infotainmentanwendungen. Die Innenraum-Domäne besitzt einen vielfältigen Aufgabenbereich, wohingegen der Antriebsstrang und die Chassis-Domäne primär Echtzeitanwendungen fokussiert. Für die verschiedenen Einsatzgebiete stehen unterschiedliche Datenbussysteme<sup>4</sup> zur Verfügung, die dem jeweiligen Einsatzgebiet angepasst verbaut werden. Die Hauptkenngroßen spielen hierbei die Datenübertragungsrate und die Echtzeitfähigkeit.

Eine zentrale Rolle bei der Vernetzung der unterschiedlichen Datenbussysteme (Netzwer-

<sup>4</sup>Der Datenbus bzw. Bus ist das Transportmedium, über dem die Kommunikation der Netzwerk-Clients (Sensoren, Aktoren, Steuergeräte) abläuft [12].

ke) übernehmen Gateways<sup>5</sup>, die eine Kommunikation zwischen den einzelnen Netzwerken ermöglichen [12].

In dem **Bild 3** ist eine schematische Darstellung von vernetzten Steuergeräten aufgezeigt.

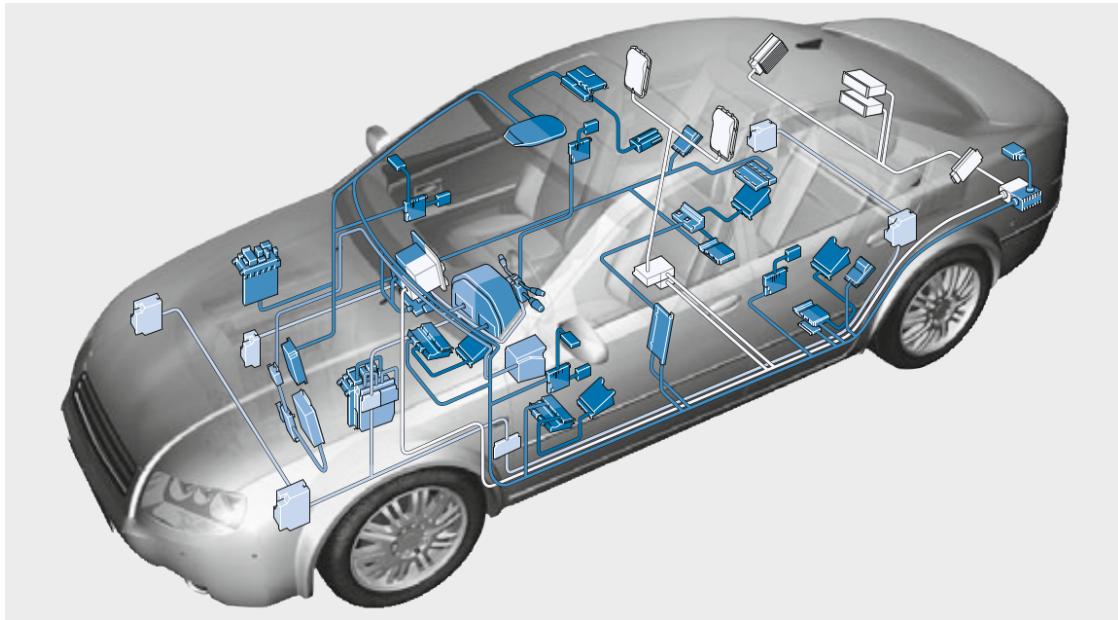


Bild 3: Fahrzeugsensorik [12]

Der Sensor besitzt die Aufgabe analoge Signale aufzunehmen, diese aufzubereiten und in der Regel in digitale Signale umzuwandeln. Die ausgehenden Signale werden anschließend über den Datenbus an das jeweilige Steuergerät gesendet. Die Steuergeräte verarbeiten diese Signale anhand derer die Steuerung und Regelung der einzelnen Prozesse im Fahrzeug durchgeführt werden [12]. Schon 2015 wurden in Premium-Fahrzeugen circa 100 Steuergeräte verbaut [13].

Das Telematik-Modul ist ebenso wie das Diagnose-Modul an das zentrale Gateway angeschlossen. Die meisten der dort zusammengeführten Informationen werden über das Telematik-Modul an den Backend-Server des Herstellers versendet. Zum Versenden der Daten wird die fest verbaute SIM-Karte mit dauerhaftem Zugriff auf das Internet genutzt. Ebenfalls können die meisten dieser Informationen über den Diagnose-Stecker oder über das Mitschneiden des Netzwerkverkehrs auf dem Bussystem abgegriffen werden [12].

Bei einer forensischen Untersuchung werden diese Daten in der Regel über den Diagnose-Stecker abgegriffen. Jedoch können nicht alle Daten auf diese Art und Weise sichergestellt werden, was in einigen Fällen dafür sorgt, dass Steuergeräte ausgebaut und deren Chips

---

<sup>5</sup>Ein Gateway fungiert als Übersetzer, der die eingehenden Daten des einen Netzwerks in das jeweilige Format des weiteren daran angebundenen Netzwerks übersetzt [12].

mit Hilfe der Chip-Off Methode entfernt und untersucht werden müssen. Diese Vorgehensweise wird in der Regel nur verwendet, wenn Memory-Speicher ausgelesen werden, deren flüchtige Daten auf anderem Wege nicht abgreifbar sind [12].

Das Paper mit dem Titel „Human Behavior Characterization for Driving Style Recognition in Vehicle System“ ist in Zusammenarbeit mehrerer Universitäten aus Italien und Indien erarbeitet worden. Dabei wurden 51 Merkmale der Fahrzeugsensorik eines Kia Soul untersucht, die über den Diagnose-Stecker auslesbar sind. Über den Ansatz des maschinellen Lernens und mit Hilfe von zehn Testfahrern wurde ein Algorithmus entwickelt, der diese fahrzeugbezogenen Daten personifiziert und zu 99% dem entsprechenden Fahrer zuordnet. Das macht die fahrzeugbezogenen Daten zu personenbeziehbare Daten. Einige der untersuchten Merkmale sind das Drehmoment, der Spritverbrauch, Drehzahl, Geschwindigkeiten und viele weitere. Generell gilt, um so mehr Merkmale und Testfahrer untersucht werden, desto höher ist die Trefferquote. Diese Art von Daten sind für Sachverständige von Verkehrsunfällen, Versicherungen und den Hersteller von besonderem Interesse [14].

Eine vollständige Liste der untersuchten Merkmale ist in der Literatur [14] aufgeführt.

### 2.2.2 Infotainment Systeme

Wo früher noch das Radio zur Unterhaltung und für Staumeldungen verbaut wurde, ist heute ein Infotainmentsystem platziert. Das Wort „Infotainment“ setzt sich aus den Wörtern „Information“ und „Unterhaltung“ (Entertainment) zusammen. Das Infotainmentsystem ist ein sogenanntes **Human Machine Interface (HMI)** bzw. die OnBoard-Unit und bildet damit die zentrale Einheit für die Kommunikation zwischen Fahrzeug und Fahrzeuginsassen. Die Aufgaben des Infotainmentsystems können in vier Kategorien eingeteilt werden und sind in **Bild 4** dargestellt.

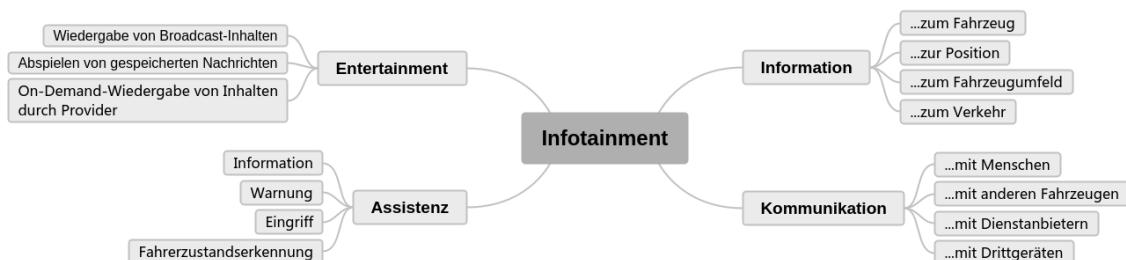


Bild 4: Aufgaben eines Infotainmentsystems [15]

Fahrerassistenzsysteme sollen den Fahrer bei seinen Aufgaben entlasten und für mehr Sicherheit und Komfort sorgen [16]. Die Fahrerassistenzsysteme melden ihre Informatio-

nen und Warnungen an das Infotainmentsystem, dieses verarbeitet die Daten und stellt dem Fahrer diese anschließend auf dem Monitor bereit. Systeme dieser Art sind zum Beispiel die Geschwindigkeits- und Reifendrucküberwachung oder auch die Verkehrsschilderkennung [16]. Fahrerassistenzsysteme wie zum Beispiel der Notbremsassistent oder der Spurhalte- und Abstandshalteassistent können darüber hinaus auch aktiv eingreifen und das Fahrzeug steuern, bremsen und im Zweifelsfall auch abschalten [16]. Im Fachjargon wird bei dieser Art Daten von „fahrzeugbezogenen“ Daten gesprochen [17].

Die für die Auswertung der personenbezogenen Daten einflussreichsten Faktoren sind die Aufgaben der Kommunikation und der Bereitstellung von Informationen. Eine der Kernaufgaben im Kontext der Information ist das Navigationssystem. Das zeigt den kürzesten Weg, den schnellsten Weg, Gefahrenstellen und Staus an. Über eine Verbindung mit Bluetooth oder WLAN kann das Smartphone mit dem Fahrzeug vernetzt werden. Anwendungsfälle dafür sind zum Beispiel die Nutzung von Streaming-Diensten, das Abspielen von Videos, Telefonie, Messenger und der Versand von E-Mails [18].

Über verschiedene Smartphone Applikationen der Hersteller kann der Fahrer weitere Dienste beziehen. Diese beinhalten beispielsweise Stauinformationen in Echtzeit, Streaming-Dienste, Live-Aufnahmen der eigenen Fahrten, Remote-Dienste zur Überwachung des Fahrzeugs, ein Online-Fahrtenbuch, ein Concierge Service oder Empfehlungen für den nächsten freien Parkplatz [18].

Darüber hinaus können bekannte Social Media Anbieter und Hersteller ihre Inhalte über die ständige Verbindung des Fahrzeugs mit der Hersteller-Cloud auf das Infotainmentssystem übertragen. Dazu gehören Facebook, Google, Twitter und viele weitere bekannte Dienste. Zusätzlich bieten einige Hersteller Call Center Dienste und Sprachassistenten, die über die Cloud erreichbar sind, an. Weiterhin können auf diesem Wege auch aktuelle Wetterdaten und Nachrichten abgerufen werden [18].

## **2.3 Blackberry QNX Neutrino Realtime Operation System**

Das Blackberry QNX Neutrino RTOS ist ein Linux-ähnliches Betriebssystem der Firma Blackberry QNX. Insgesamt wird Blackberry QNX in mehr als 100 Millionen eingebetteten Geräten (embedded Devices) eingesetzt. Mehr als 45 Automobilhersteller setzen auf Blackberry QNX. Zum heutigen Zeitpunkt sind mehr als 150 Millionen Fahrzeuge mit Blackberry Systemen ausgestattet [19].

Zur Übersicht des Einflusses vom QNX Neutrino RTOS in Bezug auf Infotainmentsysteme wird in dem **Bild 5** die weltweite Marktaufteilung der Betriebssysteme für Infotainmentsysteme dargestellt.

## 2.3. BLACKBERRY QNX NEUTRINO REALTIME OPERATION SYSTEM

---

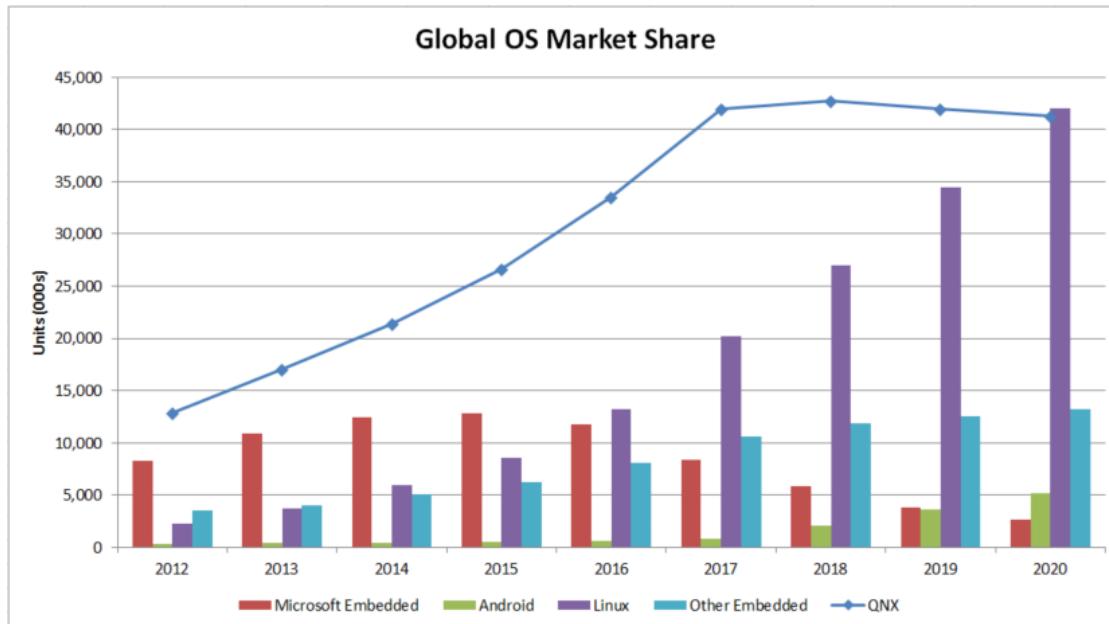


Bild 5: Globale Marktaufteilung von Betriebssystemen der Infotainmentsysteme [20]

Darüber hinaus werden die Geschäftsfelder der Automobilindustrie, Luftfahrt, Militär, Lebens-kritische Systeme in der Medizin, kritische Infrastrukturen und Robotik durch ihre Software abgedeckt. Blackberry QNX ist der Marktführer für sicherheits-zertifizierte und zuverlässige Software für die Automobilindustrie. Heutige Kerngeschäfte der Firma beinhalten Infotainmentsysteme, Fahrerassistenzsysteme und autonome Fahrsysteme. Aktuell ist die größte Sparte von QNX in der Automobilindustrie innerhalb der Telematik-Betriebssysteme und der Infotainment Software. Dabei liegt die Kernkompetenz in der Sicherheit für eingebettete Systeme, ihrer Betriebssysteme und der angebotenen Middle-ware.

Zukünftig will sich Blackberry QNX auf Hypervisoren<sup>6</sup> und Plattformen für funktionale Sicherheit konzentrieren. So bietet Blackberry unter anderem mit ihrem „Hypervisor 2.0“ den ersten sicherheits-zertifizierten Hypervisor auf dem Weltmarkt, sowie viele weitere sicherheitszertifizierte Produkte an [20].

Das QNX Neutrino RTOS ist ein Echtzeitbetriebssystem, das auf einer Mikrokern-Architektur basiert und für sämtliche Prozessorarchitekturen, wie beispielsweise die ARM- und Intel-Architekturen bereitsteht. Ein Echtzeitbetriebssystem definiert sich durch die Fähigkeit mehrere Ereignisse gleichzeitig zu verarbeiten und sicherzustellen,

<sup>6</sup>Ein Hypervisor, auch Virtual-Maschine-Monitor (VMM) genannt, ist eine abstrahierende Schicht zwischen der physischen Hardware und der virtuellen Betriebsumgebung. Mit Hilfe des Hypervisors können Hardware Ressourcen auf verschiedene Virtuelle Maschinen (VM) aufgeteilt, gesteuert und dessen Prozesse überwacht werden [21].

dass innerhalb vorhersehbarer Grenzen auf diese Ereignisse reagiert wird [22].

Darüber hinaus baut das Betriebssystem auf einen mehrschichtigen Sicherheitsansatz auf. Das ermöglicht eine feingranulare Steuerung und Implementierung von Sicherheitsprotokollen. Ferner können beispielsweise verschlüsselte und selbstverifizierende Dateisysteme, AES-256-Verschlüsselung und Sicherheitsprofile implementiert werden. Die Systemaktivitäten können protokolliert und ein Heap-, Stack- und Speicherschutz implementiert werden [23].

Diese und weitere Sicherheits-Features sind auf der Internetpräsenz und den Dokumentationen zum QNX Neutrino RTOS ausführlich beschrieben und können an diesen Stellen weitergehend recherchiert werden.

## 2.4 SQLite Datenbanken

SQLite ist eine prozessintegrierte Bibliothek, die eine in sich geschlossene SQL-Datenbank-Engine ohne Server implementiert. SQLite ist eine OSS und steht unter keiner Lizenz, sondern ist ein Allgemeingut. Diese kann für jeden Zweck, ob kommerziell oder privat, frei verwendet werden [24].

Weiterhin wird SQLite durch die folgenden Charakteristiken definiert:

**Serverlos** Es wird kein separater SQL-Server benötigt. Die C-Bibliothek besitzt sämtliche Funktionalitäten und hat direkten Zugriff auf die Dateien [25].

**Konfigurationslos** Weil kein Server eingesetzt wird, muss keine Konfiguration vorgenommen werden [26].

**Portabilität** Die gesamte Datenbank Instanz ist in einer einzigen Datei gespeichert. SQLite kann auf allen gängigen Desktop-Plattformen, eingebetteten Plattformen und Architekturen kompiliert werden. Dazu zählen Linux, Windows, Mac, QNX, Palm OS, VxWorks Plattformen, sowie 16-, 32- und 64-Bit Architekturen [25, 26].

**Kompaktheit** SQLite besteht aus insgesamt zwei Teilen, der C-Bibliothek und einer Header-Datei. In der Standardausführung umfasst SQLite circa ein Megabyte. Durch das Entfernen von Funktionsumfängen kann die Größe bis auf 69 Kilobyte reduziert werden, sodass es zum Beispiel auch auf Smartcards (Chipkarten) Verwendung finden kann.

Durch den Einsatz von Datensätzen mit variabler Länge wird nur so viel Speicherplatz alloziert, wie auch benötigt wird. Ebenso kompakt sind die benötigten Ressourcen zur Laufzeit. Es werden zur Laufzeit nur einige wenige Megabytes an freien Speicherplatz benötigt [25, 26].

**Zuverlässigkeit** Der gesamte Quellcode von SQLite beläuft sich auf circa 30.000 Zeilen. Zu der SQLite Bibliothek wird eine Test-Suite (Testumgebung) mit ausgeliefert. Diese ist sehr umfassend und beinhaltet über zehn Millionen Unit- und Query-Tests. SQLite durchläuft vor jedem Release circa 2,5 Milliarden Einzeltests, damit wird sichergestellt, dass die Anzahl an Fehlern sehr gering und die Robustheit der Software sehr hoch gehalten wird [25, 26].

**Einfachheit** Die Dokumentationen zu SQLite sind sehr umfangreich. Zusätzlich ist der Quellcode modular aufgebaut und die Kommentierung des Quellcodes ausführlich. Die Open Source Community hat zusätzlich sehr viele Schnittstellen zu verschiedenen Programmiersprachen erstellt, die eine Implementierung in die jeweilige Applikation stark vereinfacht [26].

**Flexibilität** Zur Flexibilität von SQL tragen einige Faktoren bei. Im Kontext der eingebetteten Datenbank wird durch den architektonischen Aufbau von SQLite das Beste aus zwei Welten bereitgestellt: die Performance und Flexibilität eines relationalen Datenbank Front-Ends, wie auch die Kompaktheit und Einfachheit eines B-Baum<sup>7</sup> Back-Ends. Zusätzlich müssen keine Server konfiguriert oder Netzwerk- und Verbindungsprobleme erwartet werden. Weiterhin gibt es durch die Plattformunabhängigkeit keine Limitierung in der Wahl der Plattform oder der Architektur. Zuletzt gibt es keine Lizenz, die beachtet werden muss, wodurch gerade im kommerziellen Umfeld keine Kosten für Lizenzgebühren anfallen [25, 26].

Laut Literatur „sqlite.org“ ist die SQLite Datenbank-Engine die weltweit am weitesten verbreitete und genutzte Datenbank-Engine. SQLite Datenbanken finden sich in jedem Android-, iPhone-, Mac- und Windows10-Gerät wieder. Weiterhin sind diese in Firefox, Safari, Chrome und den meisten fahrzeugbezogenen Multimediasystemen implementiert. Dies sind nur einige von vielen Millionen Anwendungen und Geräten [28]. Auch Applikationen wie zum Beispiel WhatsApp und sämtliche andere Messenger-Dienste, E-Mail Applikationen und viele weitere setzen auf SQLite [29].

Daraus ableitend werden viele personenbezogene Daten in verschiedenen SQLite Datenbanken gespeichert. Diese sind im Kontext der digitalen Forensik von besonderem Interesse und der Umgang mit diesen essenziell [30].

---

<sup>7</sup>Der B-Baum beschreibt in der Informatik eine Daten- oder Indexstruktur, die ihren Einsatz in Datenbanken und Dateisystemen findet. Es handelt sich um einen vollständig balancierten Baum. Dieser kann in einem Knoten mehrere Elemente sortiert speichern [27].

### 2.4.1 Architektur

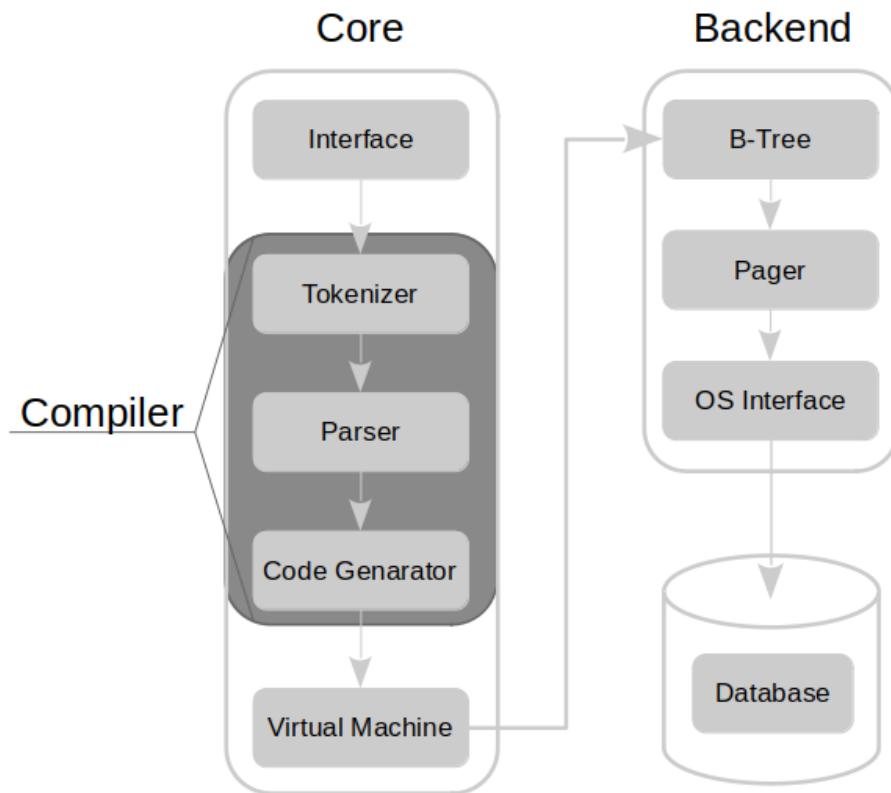


Bild 6: Schematische Darstellung der SQLite Architektur [26]

In Bild 6 ist der architektonische Aufbau der Software dargestellt. Der modulare Aufbau der Architektur gliedert sich in drei Subsysteme: Compiler, Core und das Back-End. Diese beinhalten insgesamt acht separate Module. Der Core bildet mit dem ersten Modul die Schnittstelle zu dem Anwender, sowie den Compiler und die VM. Der Compiler kompiliert die Anfrage, die VM führt diese aus und das Back-End bildet die Schnittstelle zum Betriebssystem und handhabt die Speicherung [26].

### 2.4.2 Besondere Merkmale

Im Gegensatz zu den meisten anderen Datenbanksystemen verwendet SQLite eine dynamische Typen Systematik und keine statischen Typen für Relationen (Datenbank-Tabelle). Das bedeutet, dass jeder Attributwert (Wert), unabhängig von seinem Typen (String, Integer, Float) in ein Attribut (Spalte) geschrieben werden kann. Dieses Verhalten ist von vielen Skriptsprachen, wie beispielweise von Javascript, bekannt und in SQLite adaptiert

[25, 26].

Eine weitere Besonderheit ist die Möglichkeit mehrere Datenbanken gleichzeitig zu bearbeiten. Dadurch können Abfragen über mehrere Datenbanken gestellt, sowie Manipulationen mehrerer Datenbanken zur selben Zeit durchgeführt werden. Darüber hinaus ist es möglich SQLite Datenbanken vollständig im Memory-Speicher zu erstellen, sodass diese die gesamte Lebenszeit über im Cache existieren. Die Verarbeitungskapazität der Datenbanken ist sehr hoch und werden hauptsächlich für temporäre Daten verwendet [25, 26].

## 2.5 Rekonstruktion von SQLite Datenbanken

Es existieren einige Forschungsansätze zu dem Thema der Rekonstruktion von gelöschten oder beschädigten SQLite Datenbanken. Ein Vergleich der folgenden drei Forschungsansätze [29, 30, 31] zeigt, dass die Basis der Rekonstruktion ein struktur-basierter Ansatz ist. Das bedeutet, dass das Know-How über den strukturellen Aufbau einer SQLite Datenbank genutzt wird, um diese wiederherzustellen [29, 30, 31].

Unterschiede in den Forschungen liegen lediglich in der Motivation zur Rekonstruktion, wie auch in der Nutzung der durch SQLite erzeugten Dateien. So fokussiert sich die Forschung [30] zur Rekonstruktion von Datensätzen lediglich auf die B-Baum-Seiten der Datenbank-Datei (\*.db). Darüber hinaus beziehen die Forschungen von Christian Meng, Harald Baier[29] und Yao Liu et al.[31] das *WAL* bzw. das *rollback journal* mit in das Rekonstruktionsverfahren ein. Dies ist zum einen in der Effizienzsteigerung der Rekonstruktion und zum anderen darin begründet, dass sich auch in diesen Dateien gelöschte Datensätze befinden, die für die forensische Untersuchung von Bedeutung sein können [29, 31].

Die Forschung von Christian Meng und Harald Baier [29] liefert nicht nur die theoretische Verfahrensweise der Rekonstruktion, sondern darüber hinaus auch das aktuell erfolgreichste Werkzeug zur Rekonstruktion von Datenbanken [29]. Infolge dessen wird die Methodik der struktur-basierten Rekonstruktion anhand des Werkzeugs *bring2lite* erläutert. In dem **Bild 7** ist die schematische Darstellung der Arbeitsweise eines Rekonstruktions-Tools für Datenbanken dargestellt.

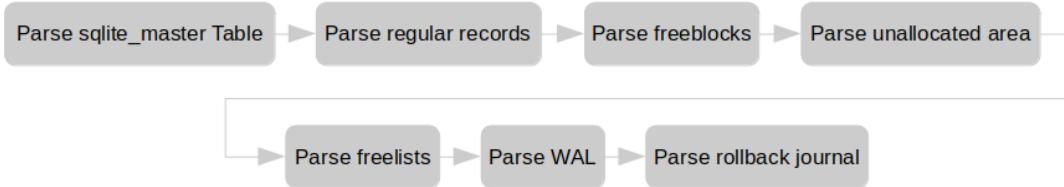


Bild 7: Schematische Darstellung der Arbeitsweise bring2lite [32]

Der Gesamtprozess der Rekonstruktion von SQLite Datenbanken teilt sich in sieben Prozessschritten bzw. Einzelprozessen auf. Die sieben Prozessschritte können wiederum in vier Bereiche aufgeteilt werden.

Der erste Bereich beinhaltet den ersten Prozessschritt und kann als Informationssammlung der Standardparameter und Speicherbereiche der SQLite-Datenbank betrachtet werden. Im zweiten Schritt werden die vorhandenen Datensätze erfasst; dieser bildet somit den zweiten Bereich der Erfassung allozierter Datensätze. Die Verarbeitung der gelöschten Daten erfolgt in den Einzelprozessen drei bis fünf und bildet damit den dritten Bereich, die Rekonstruktion der gelöschten Datensätze. Der vierte Bereich verarbeitet die zusätzlich zur eigentlichen Datenbankdatei erstellte Datei. Das sogenannte **Write-Ahead Log (WAL)** und das *rollback journal* können ebenfalls gelöschte Datensätze enthalten und geben Aufschluss auf die Historie der verarbeiteten Transaktionen innerhalb der SQLite-Datenbank [32].

Im ersten Teilprozess wird die *SQLite\_master* Tabelle verarbeitet, dadurch werden Erkenntnisse über das Format und die Datentypen der einzelnen Tabellen und damit zugleich die Struktur der gelöschten Daten erlangt. Diese Systemtabelle ist Teil der *Kopfseite* (engl. header page), die die erste Seite einer jeden SQLite Datenbankdatei bildet. In dem Header der *header page* werden die Informationen über Basisparameter der SQLite Datenbank, die Versionsnummer, Seitengröße und Metavariablen gespeichert. Die *SQLite\_master* Tabelle ist eine Systemtabelle, die im B-Baum Format organisiert ist. Diese beinhaltet die Schemata der Datenbanken und den jeweilig zugehörigen Wurzelknoten des B-Baumes. Der B-Baum speichert die eigentlichen Datensätze [29, 30]. Dies ist in dem **Bild 8** visualisiert.

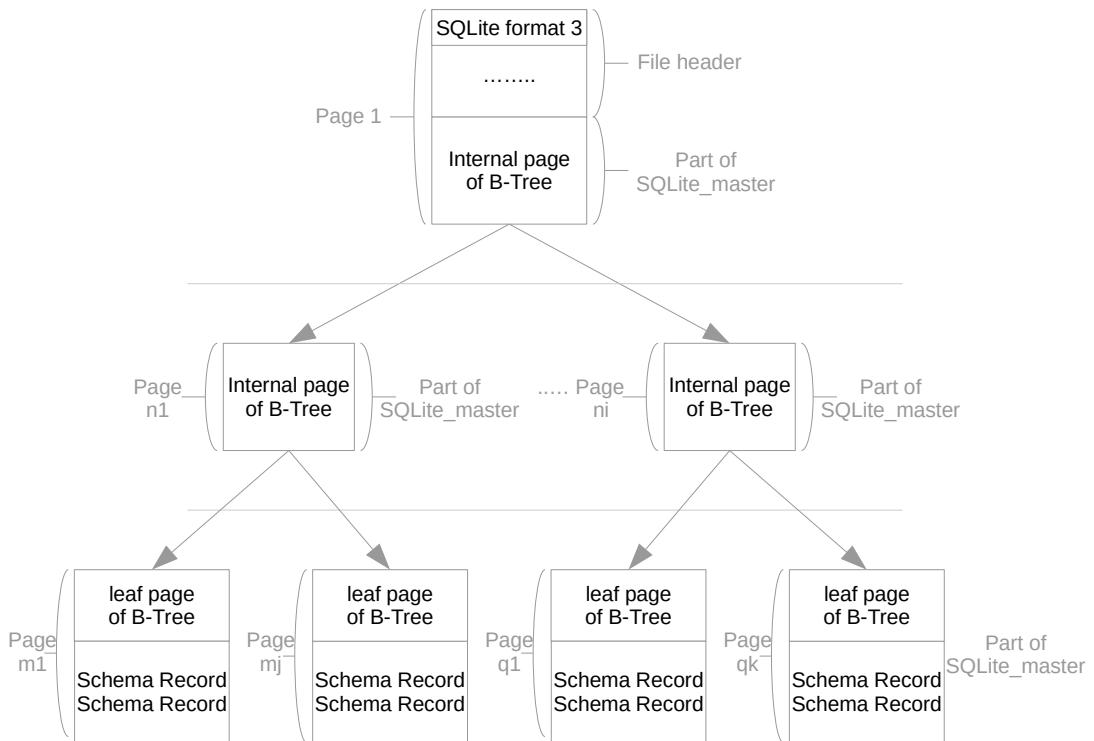


Bild 8: Schematischer Aufbau SQLite Datenbank [30]

Die Erfassung der regulären Datensätze erfolgt im zweiten Prozessschritt. Anschließend beginnt die eigentliche Rekonstruktion gelöschter Datensätze. Dafür wird im dritten Prozessschritt damit begonnen die *Freiblöcke* (englisch (engl.) Freeblocks) zu verarbeiten. Die Freiblöcke stellen einzelne gelöschte Tupel (Datenbankeinträge) dar. Bei dem gelöschten Tupel werden die ersten 4 Bytes überschrieben, diese geben die Größe der eigentlichen Daten (engl. Payload) und die Reihennummer an. Durch das Überschreiben dieser Informationen ist es SQLite nicht mehr möglich, diese als reguläre Tupel zu verarbeiten. Dadurch, dass die Freiblöcke als verkettete Liste organisiert sind und die ersten 2 Bytes der Datenbank die Adresse des jeweils nächsten Freiblocks angeben, ist es möglich sämtliche Freiblöcke zu identifizieren und zu extrahieren [29, 30].

Der vierte Teilprozess verarbeitet den nicht allozierten Bereich. Der nicht allozierte Bereich liegt zwischen dem Header der B-Baum-Seite und dem Start des Datenbereichs. Dies resultiert aus der Funktionsweise von SQLite. Denn Datensätze, die sich am Anfang einer Datensektion befinden und gelöscht werden, bleiben vollständig erhalten. Lediglich der Zeiger wird manipuliert und auf den Start der „neuen“ aktiven Daten adressiert. Für die Rekonstruktion der nicht allozierten Bereiche muss dieser extrahiert und anschließend auf die exakte Größe der Daten reduziert werden [29, 30].

Abschließend werden die sogenannten Freilisten (engl. Freelists) extrahiert. Diese werden

immer dann von SQLite erzeugt, wenn eine ganze B-Baum-Seite gelöscht wird und beinhaltet die Zeiger der gelöschten B-Baum-Seiten. Denn auch die B-Baum-Seiten werden nicht überschrieben [29, 30].

Nachdem die Datenbankdatei vollständig untersucht und gelöschte Daten extrahiert wurden, wird das WAL bzw. das Rollback-Journal untersucht. Das Rollback-Journal ist eine temporäre von SQLite erzeugte Datei, die eine Kopie des ursprünglichen unveränderten Datenbankinhalts darstellt. Im Falle eines Absturzes, einer Fehlfunktion oder des Rollbacks kann der im Rollback-Journal enthaltene Originalinhalt der Datenbankdatei wieder zurück in die eigentliche Datenbankdatei gespielt werden. Dadurch wird die Datenbankdatei wieder in ihren Ursprungszustand zurückversetzt [33, 34].

<u>Size</u>	<u>Description</u>
1 sector (usually 512 bytes)	Database header – only first 28 bytes used
4 bytes	Page number in main DB
Page size	Page data
4 bytes	Checksum
4 bytes	Page number in main DB
Page size	Page data
4 bytes	Checksum
4 bytes	Page number in main DB
Page size	Page data
4 bytes	Checksum

Bild 9: Schematische Darstellung SQLite Rollback-Journal [33]

Der grundlegende Aufbau ist in **Bild 9** dargestellt. Die Rollback-Journal-Datei besteht aus einem Header mit der Größe eines Dateisektors, dieser beträgt in der Regel 512 Byte. Darauf folgen für jede Seite der Datenbank jeweils vier Byte für die logische Seitennummer, angehangen daran der unkomprimierte Dateninhalt der Seite und zuletzt noch einmal vier Byte für die Checksumme. Erkennbar ist die Journal-Datei an der Dateiendung *.db-journal* [33, 34].

Im Gegensatz zum Rollback-Journal werden in der ebenfalls temporären WAL-Datei lediglich die Veränderungen an der Datenbankdatei gespeichert. Die WAL-Datei ist mit der Version 3.10.10 eingeführt worden und bietet einige Vorteile gegenüber dem Rollback-Journal. So ist beispielsweise die Geschwindigkeit des WALs in den meisten Szenarien höher und erlaubt dabei gleichzeitige Lese- und Schreiboperationen. Der schematische Aufbau dieser Datei wird in **Bild 10** gezeigt. Der Header der WAL-Datei ist 32 Bytes groß, darauf folgen für jede Änderung (Frame) die gespeichert wird, ein Header und separat die Daten der jeweiligen Änderung. Dabei ist der Header der Änderung mit 16 Bytes und der Dateninhalt mit 1024 Bytes fest definiert [31, 34]. Auch die WAL-Datei kann an der Dateiendung *.db-wal* identifiziert werden.

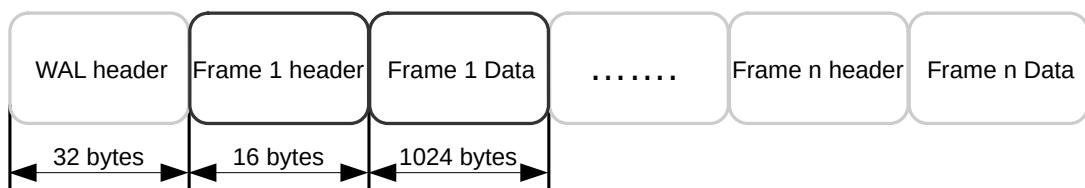


Bild 10: Schematische Darstellung SQLite WAL [31, 34]

## 2.6 Open Source Software

Der englische Begriff OSS kann sinngemäß in *quelloffene Software* übersetzt werden. Entstanden ist dieser Begriff mit der 1998 von Eric S. Raymond und Bruce Perens gegründeten **Open Source Initiative (OSI)** [35]. Diese Initiative vergibt, nach dem Bestehen des standardisierten Lizenz-Review-Prozesses, bis heute das allgemein anerkannte Open Source Zertifizierungssiegel. Innerhalb des Lizenz-Review-Prozesses wird geprüft, ob die Software den von der OSI definierten Anforderungskatalog erfüllt [36].

Der Anforderungskatalog beinhaltet insgesamt zehn Kriterien, die eine Software erfüllen muss, um eine Open-Source-Lizenz zu bekommen. Die Kriterien wurden von den Debian Free Software Guidelines (DFSG) abgeleitet und auf die eigenen Bedürfnisse angepasst [37].

In der folgenden **Tabelle 2** werden die Kriterien die eine Open-Source-Software erfüllen muss in entsprechender sinngemäßer Übersetzung dargestellt. Im Anhang in der **Tabelle 10** in Kapitel 5.2 sind die 10 Open-Source-Software Kriterien der DFSG im originalen Wortlaut angefügt.

### Nr. Freie Übersetzung

---

#### 1 Freie Weitergabe

Die Lizenz darf keine Partei in ihrem Recht einschränken, die Software als Teil eines Software-Paketes zu verkaufen oder zu verschenken. Für einen derartigen Verkauf darf der Verkäufer für die Lizenz keine Lizenz- oder sonstige Gebühren veranschlagen.

#### 2 Quellcode

Das Programm muss den Quellcode beinhalten. Die Verbreitung des Programms muss in Form des Quellcodes und auch in kompilierter Form erlaubt sein. Sollte der Quellcode nicht direkt enthalten sein, so muss dieser zum Selbstkostenpreis, in publizierter Art und Weise, angeboten werden. Vorzugsweise wird der kostenlose Download aus dem Internet favorisiert. Absichtlich missverständlich geschriebener oder verschleierter Quellcode ist unzulässig. Darüber hinaus muss der Quellcode in der für einen Programmierer bevorzugten Form angeboten werden. Zwischenformen des Codes, die beispielweise ein Präprozessor oder ein Übersetzer ausgeben, sind nicht erlaubt.

#### 3 Abgeleitete Arbeiten

Die Lizenz muss Modifikationen der Software und Derivate der Software, sogenannte „Forks“, erlauben. Weiterhin müssen derartige Programme unter den gleichen Lizenzbedingungen, wie die Originalsoftware, vertrieben werden können.

#### 4 Integrität des Quellcodes des Autors

Die Weitergabe von Quellcodes in modifizierter Form darf nur von der Lizenz eingeschränkt werden, wenn diese vorsieht, dass sogenannte „Patch Files“ mitgegeben werden dürfen. *Patch Files* sind Dateien die Einfluss auf den Programmcode haben und zum Zeitpunkt der Kompilierung den Quellcode verändern. Die Lizenz muss die Verbreitung von Software, die aus modifiziertem Quellcode gebaut wurde, ausdrücklich erlauben. Darüber hinaus kann die Lizenz verlangen, dass Forks einen anderen Namen oder eine andere Versionsnummer als die Originalsoftware tragen müssen [38].

#### 5 Keine Diskriminierung gegen Personen oder Gruppen

Die Lizenz darf keine Person oder Personengruppe benachteiligen [38].

#### 6 Keine Einschränkung von Arbeits- und Forschungsgebieten

Die Lizenz darf das Einsatzgebiet des Programmes nicht einschränken. So darf der kommerzielle Einsatz oder der Einsatz in Forschungsgebieten, wie beispielsweise der Genforschung, nicht ausgeschlossen werden.

---

Fortsetzung auf der nächsten Seite

Fortsetzung der vorherigen Seite

Nr.	Freie Übersetzung
7	<b>Verbreitung der Lizenz</b> Die mit dem Programm verbundenen Rechte müssen auf alle Parteien übergehen, die die Software erhalten, ohne dass eine zusätzliche Lizenz ausgeübt werden muss [38].
8	<b>Die Lizenz darf nicht produktspezifisch sein</b> Die Rechte des Programmes dürfen nicht in Abhängigkeit zu einem bestimmten Software-Paket stehen. Wird das Programm aus dem Software-Paket extrahiert und unter Einhaltung der Lizenzbedingungen verwendet oder weitergegeben, sollen die Parteien die gleichen Rechte haben, die auch in Verbindung mit der Originalsoftware gewährt werden.
9	<b>Die Lizenz darf keine andere Software einschränken</b> Die Lizenz darf keine Einschränkung für andere Software enthalten, die mit der lizenzierten Software verbreitet wird. Beispielsweise darf die Lizenz nicht darauf bestehen, dass alle anderen Programme, die auf dem gleichen Medium verbreitet werden, Open-Source-Software sein müssen [38].
10	<b>Die Lizenz muss technologienutral sein</b> Die Lizenz darf keine bestimmte Technologie oder Art der Schnittstelle fordern oder auf dieser basieren.

Tabelle 2: 10 Kriterien von Open Source Software - Original Wortlaut

Zusammenfassend kann Open-Source-Software als eine Software bezeichnet werden, die unter einer Open-Source-Lizenz von der OSI betrieben wird. Weiterhin ist sie frei verfügbar, manipulierbar, darf in manipulierter Form weitergegeben werden und ist nicht zwangsläufig kostenlos [39].

# 3 Konzeption

Das Kapitel der „Konzeption“ unterteilt sich in die folgenden sieben Unterkapitel:

- Forschungsumgebung
- Vorgehensmodelle der IT-Forensik
- Vorgehensweise der forensischen Untersuchung
- Softwareauswahl
- Security Ausblick
- Physikalischer Analyseansatz
- Vergleichskonzept Linux Desktop-Distribution vs QNX Neutrino RTOS

Dieses Kapitel beinhaltet die Herangehensweise zur Entwicklung eines zielgerichteten Konzeptes zur Untersuchung eines Infotainment-Images eines PKWs. Die Grundlage des Konzeptes ist eine forschende Herangehensweise, auf dessen Basis anschließend eine Analyse sowie eine erweiterte Sicherheitsbetrachtung erfolgen kann. Dafür wird im ersten Abschnitt die verwendete Forschungsumgebung, die Forensik Maschine und die VM beinhaltend, vorgestellt.

Die digitale Forensik und deren Vorgehensweisen sind streng methodisch und bilden in der Regel ein vollumfängliches Konzept ab, das mehr als den eigentlichen Untersuchungsprozess beinhaltet. Deshalb werden die vier meistverwendeten Vorgehensweisen kurz vorgestellt, dessen Charakteristika beleuchtet und anschließend miteinander verglichen. Daraus soll ein Konzept entstehen, das den forschenden Ansatz unterstützt, dabei jedoch die Ziele einer forensischen Untersuchung klar im Mittelpunkt belässt. Nachdem das Konzept für die forensische Untersuchung des Infotainment-Images beschrieben ist, wird eine Softwareauswahl für die verschiedenen Aspekte der Analyse getroffen.

Im Anschluss daran findet die Konzeptentwicklung für eine sicherheitstechnische Betrachtung des QNX Neutrino RTOS und möglicher Aushebelung von Sicherheitsfunktionen und Schutzmaßnahmen statt. Daran anknüpfend werden Möglichkeiten abgesteckt, ein solches Betriebssystem auf Basis eines physikalischen Analyseansatzes zu untersuchen.

Einen Abschluss findet das Kapitel in der Entwicklung von Kriterien und der Auswahl geeigneter Methoden, um das QNX Neutrino RTOS mit freien Linux Desktop-Distribution zu vergleichen und zu bewerten.

## 3.1 Forschungsumgebung

In diesem Abschnitt wird der Versuchsaufbau dargestellt. Für die forensische Untersuchung eines Speicherabilds muss ein Computer zur Verfügung stehen, der spezielle Software für die digitale Forensik bereithält. Es wird ein für die Forensik optimiertes Betriebssystem verwendet, dass zur Optimierung der Prozessorgeschwindigkeit nativ installiert ist.

Für weitere Versuche an dem Speicherabbild, wie beispielweise das Aushebeln der read-only Berechtigung, wird eine VM benötigt, die im weiteren Verlauf des Abschnitts vorgestellt wird.

### 3.1.1 Forensik Maschine

Zur Durchführung der forensischen Untersuchung des Infotainmentsystems wird ein Industrie-Notebook der Marke *Dell* Modell E6220 verwendet.

Technische Spezifikation:

**Marke** DELL

**Modell** E6220

**CPU** Intel i5 2540M 2.6 GHz 3MB Cache

**Grafikkarte** Intel HD Graphics 3000

**Arbeitsspeicher** 4GB DDR3 SDRAM

**Festplatte** Samsung SSD 860 EVO 250GB

**Betriebssystem** Ubuntu 18.04 Final Release HPM Live

### 3.1.2 Virtuelle Maschine (VM)

Die verwendete Virtuelle Maschine wird von dem Hersteller des QNX Betriebssystems *Blackberry* bezogen. Es handelt sich um eine VM für die Software *VMWare Workstation* in dem Format .vmx, die mit den folgenden Eigenschaften konfiguriert ist:

**Name** QNX-SDP-x86\_64

**Arbeitsspeicher** 1 GB (Empfohlen: mindestens 256 MB)

**Prozessoren** 2

**Festplatte (IDE)** 65 MB

**Festplatte 2 (IDE)** 3 GB

**Betriebssystem** QNX Realtime Operating System (RTOS) 64 Bit

Dies sind die unveränderten Konfigurationsparameter des Herstellers.

### 3.2 Vorgehensmodelle der IT-Forensik

In der Wissenschaft existieren viele verschiedene Vorgehensmodelle der IT-Forensik. Internationale Akzeptanz und anerkannte Methoden werden unter anderem den folgenden Vorgehensmodellen zugesprochen [40]:

- S(ecure)-A(nalyse)-P(resent)-Modell
- Kent, Chevalier, Grance, Dang Modell
- Casey
- BSI Modell

Im folgenden Unterkapitel erfolgt die strategische Planung der Auswahl eines geeigneten Vorgehensmodells in Bezug auf die vorliegende Arbeit. Im ersten Schritt werden die Prozesse der einzelnen Vorgehensmodelle visualisiert und deren prinzipiellen Abläufe dargestellt.

Darauf folgt die Einordnung der einzelnen Vorgehensmodelle in die drei Phasen „Beweissicherung“, „Analyse“ und „Präsentation“ des Secure-Analyse-Present (SAP)-Modells. Anschließend werden die vier Vorgehensmodelle auf ihre Stärken und Schwächen hin analysiert.

Im letzten Teilkapitel dieses Abschnitts werden die Ergebnisse der Analyse evaluiert. Es werden Gemeinsamkeiten und Unterschiede herausgearbeitet. Auf Basis der Analyse und der Evaluierung wird ein Vorgehen für die weitere Arbeit festgelegt.

#### 3.2.1 Auswahl eines geeigneten Vorgehensmodells der IT Forensik

Die Basis für die Einordnung der verschiedenen Vorgehensmodelle bildet das SAP-Modell. Der Name dieses Vorgehens bildet sich aus den drei Phasen des Modells „S(ecure)“, „A(nalyse)“ und „P(resent)“. Es ist das am meisten genutzte, abstrahierte und verallgemeinerte der vier Vorgehensmodelle. Es werden keine detaillierten Erläuterungen über die einzelnen Prozessschritte und verwendeten Methoden vorgegeben [2].

### 3.2. VORGEHENSMODELLE DER IT-FORENSIK

Das „Casey“-Modell hingegen besitzt mit zwölf einzelnen Phasen den höchsten Detailierungsgrad. Weiterhin folgt es einem Wasserfall-Verlauf, sodass Iterationen einzelner Phasen nicht vorgesehen sind. Durch diese Eigenschaft folgt das Vorgehen einen festen und statischen Ablauf, der Iterationen nicht vorsieht [2].

Das Vorgehensmodell nach dem BSI verfolgt den entgegengesetzten Ansatz. Dieses Vorgehensmodell besitzt viele Eigenschaften moderner und agiler Projektmanagement- und Softwareentwicklungs-Modelle. Jede einzelne der Phasen ist iterativ durchführbar und kann bei neuen Erkenntnissen innerhalb der forensischen Untersuchung beliebig häufig wiederholt werden. Eine Besonderheit liegt in der phasen- und prozessübergreifenden Dokumentation. Diese wird begleitend durch die gesamte forensische Untersuchung geführt und sukzessive erweitert [2].

Die Phasen des Kent, Chevalier, Grance, Dang Modells ähneln dem SAP-Modell sehr stark. Lediglich die Phase „Analyse“ des SAP-Modells wird in die zwei Phasen „Examination“ und „Analyse“ unterteilt [40].

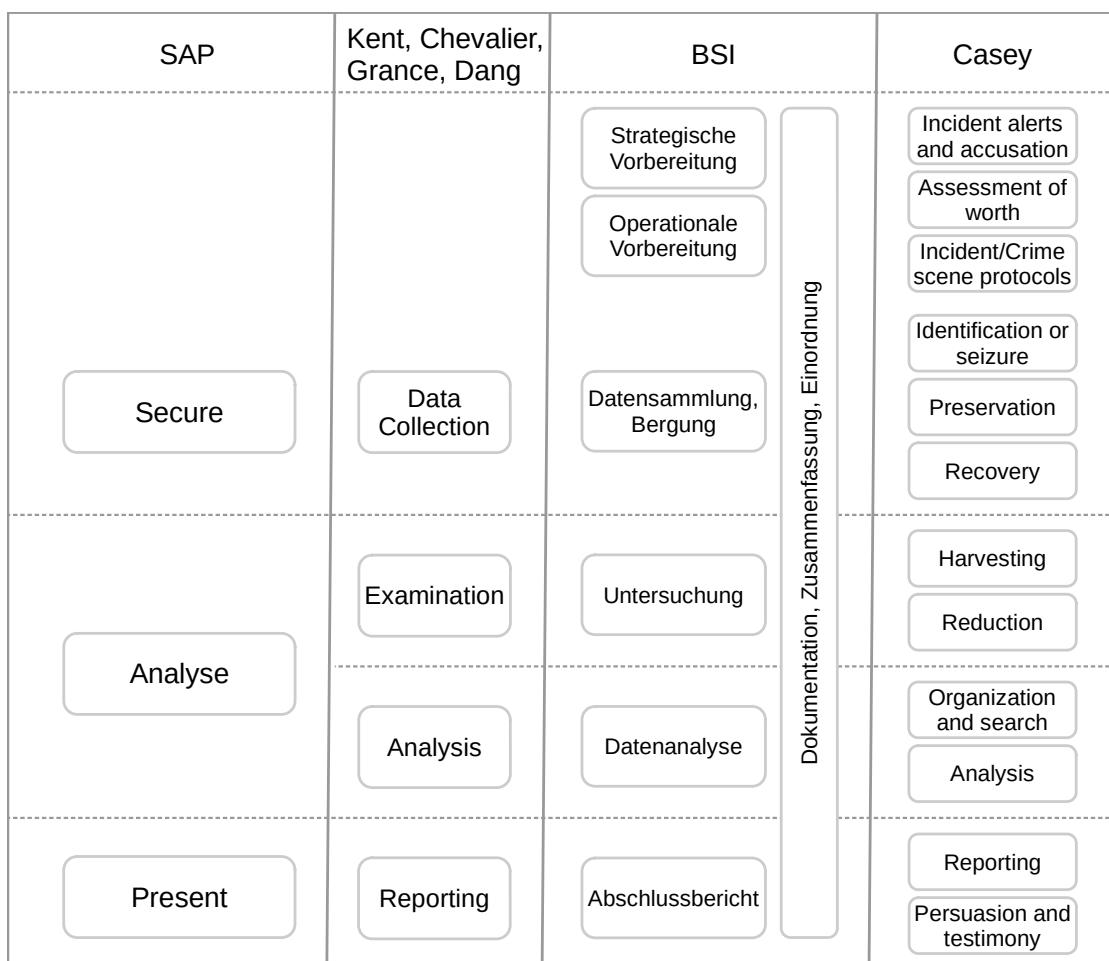


Bild 11: Abstrahierte Darstellung prozessualer Abläufe der Vorgehensmodelle

## 3.2. VORGEHENSMODELLE DER IT-FORENSIK

---

Das vorangegangene **Bild 11** visualisiert die prozessualen Abläufe und zeigt die Einordnung der verschiedenen Phasen.

### SWOT-Analyse

Die Methodik der „SWOT-Analyse“ stammt aus dem Bereich der strategischen Unternehmensanalyse / -ausrichtung. Jedoch kann diese Methodik für viele andere Anwendungsbereiche adaptiert und einen sinnvollen Einsatz finden. Mit Hilfe der „SWOT-Analyse“ können die Stärken (Strengths) und Schwächen (Weaknesses), sowie die damit verbundenen Möglichkeiten (Opportunities) und Risiken (Threats) funktional und transparent einander gegenüber gestellt werden [41].

Diese Methode wird in Vorbereitung auf die Nutzwertanalyse zur Auswahl eines geeigneten Vorgehensmodells für die forensische Untersuchung erstellt.

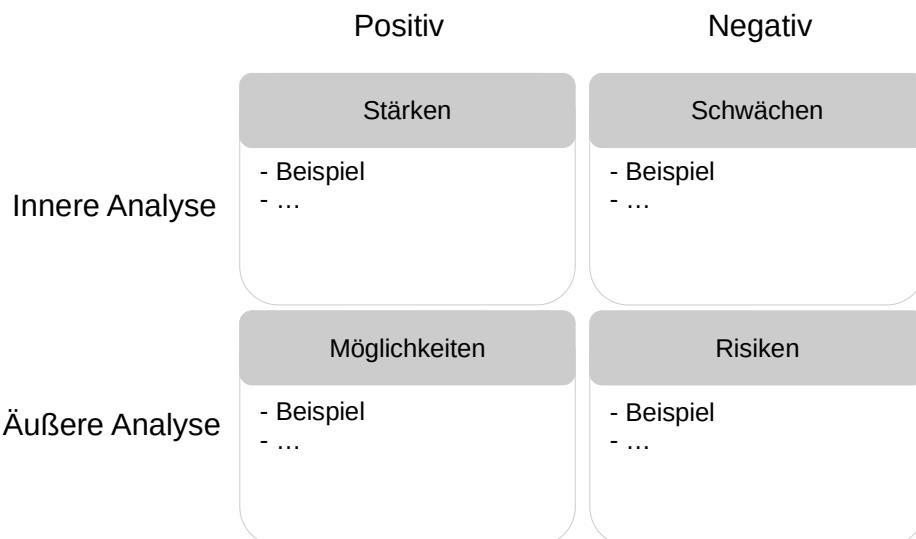


Bild 12: Schematische Darstellung der SWOT-Analyse

Wie in **Bild 12** beispielhaft dargestellt, wird innerhalb der Methodik hauptsächlich zwischen der internen und der externen Analyse unterschieden. Die Analyse der Stärken und Schwächen zählt zu der internen Analyse, wohingegen die Auslotung der Möglichkeiten und Risiken zur externen Analyse gehören [41].

Diese Unterscheidung ist zur Unternehmensanalyse oder deren Strategien sinnvoll. Im Kontext dieser Arbeit findet diese Unterscheidung hingegen keine Beachtung und wird lediglich der Vollständigkeit halber dargestellt. Ferner wird im Sinne dieser Arbeit der linke Bereich der SWOT-Analyse (Stärken, Möglichkeiten) als positiv und der rechte Bereich (Schwächen, Risiken) als negativ betrachtet.

### 3.2. VORGEHENSMODELLE DER IT-FORENSIK

Folgend werden die vier Vorgehensmodelle mit der Methodik der SWOT-Analyse untersucht, bevor anschließend Gemeinsamkeiten und Unterschiede aufgezeigt und bewertet werden.

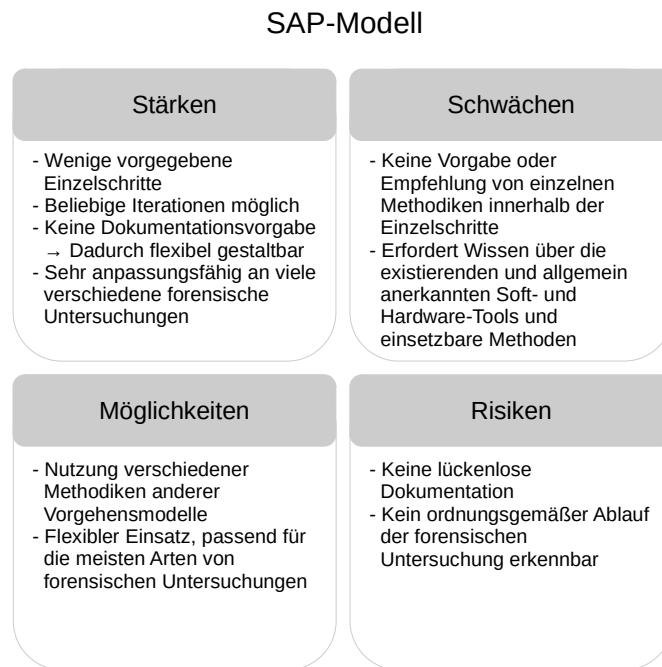


Bild 13: SWOT-Analyse des SAP-Modells

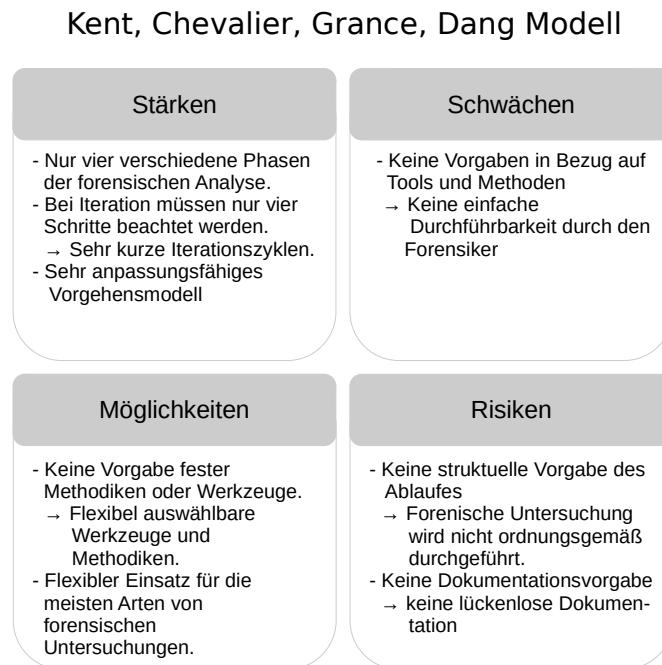


Bild 14: SWOT-Analyse des Kent, Chevalier, Grance, Dang Modells

### 3.2. VORGEHENSMODELLE DER IT-FORENSIK

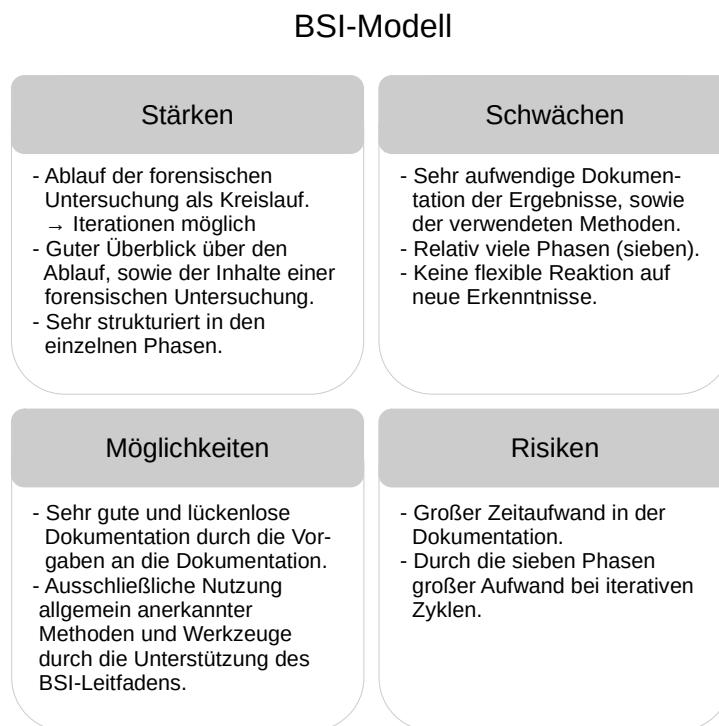


Bild 15: SWOT-Analyse des BSI-Modells

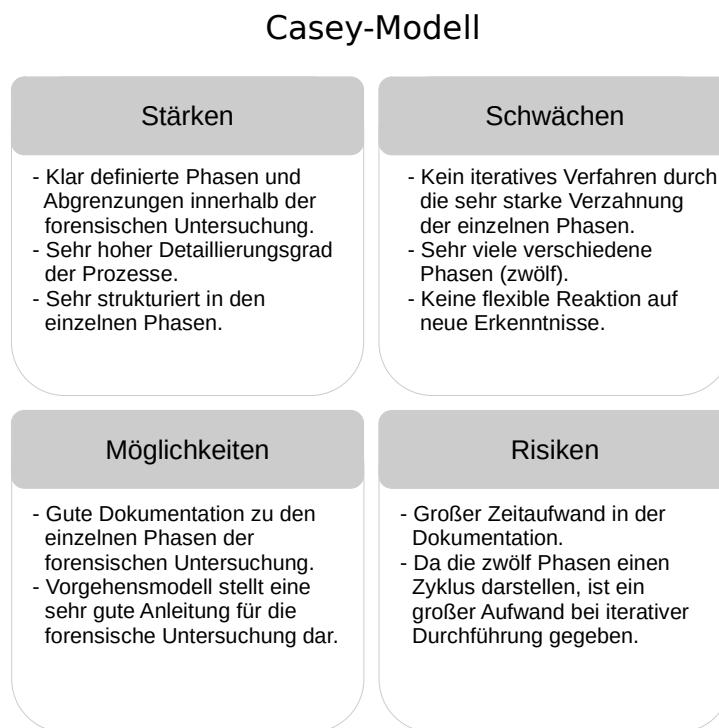


Bild 16: SWOT-Analyse des Casey-Modells

#### Evaluierung

In diesem Abschnitt werden die vier SWOT-Analysen ausgewertet. Dafür wird eine Nutzwertanalyse durchgeführt und die gewonnenen Ergebnisse interpretiert und bewertet. Die Nutzerwertanalyse stellt eine wissenschaftliche Methode zur systematischen Beurteilung verschiedener Lösungsvarianten dar. Dadurch wird die Entscheidungsfindung fundiert untermauert und der Entscheidungsprozess unterstützt.

Für eine Nutzwertanalyse werden verschiedene Bewertungskriterien erarbeitet und ihrer Priorität nach gewichtet. Anschließend werden die Kriterien auf die Lösungsvarianten angewendet und bewertet. Der Lösungsvorschlag mit den meisten Punkten, unter Beachtung der Gewichtung, in der Gesamtbewertung sollte bevorzugt werden [42].

#### Bewertungsschema der Nutzwertanalyse

Folgend wird das Bewertungsschema der Nutzwertanalyse erläutert. Dieses ist allgemeingültig für sämtliche Nutzwertanalysen innerhalb dieser Arbeit.

Die Bewertungskriterien sind der Aufgabenstellung angepasst gewählt. Sie sollen dazu beitragen, eine möglichst objektive Bewertung der verschiedenen Vorgehensmodelle zu treffen. Die Schwierigkeit dabei ist, dass es keine objektiv messbaren Kriterien gibt, sodass auf subjektive Einschätzungen und Erfahrungen zurückgegriffen werden muss.

Somit ist das Ziel dieser Bewertung möglichst objektiv betrachtbare Kriterien zu wählen, um diese abschließend zu bewerten.

Bewertung	Bedeutung	
1 Punkt	sehr gering	<i>invertiert (sehr hoch)</i>
2 - 3 Punkte	gering	<i>invertiert (hoch)</i>
4 - 6 Punkte	mittel	<i>invertiert (mittel)</i>
7 - 8 Punkte	hoch	<i>invertiert (gering)</i>
9 - 10 Punkte	sehr hoch	<i>invertiert (sehr gering)</i>

Tabelle 3: Bewertungsschema der Nutzwertanalyse

Für die Bewertung der einzelnen Kriterien wird das Punktesystem der **Tabelle 3** verwendet. Im Hinblick auf die Bewertungskriterien der Komplexität und des Dokumentationsaufwands muss die Tabelle invertiert dargestellt werden. Im Falle der Komplexität bedeutet dies, dass eine niedrige Bewertung mit einer großen Komplexität einhergeht. Das Gleiche gilt für die Bewertung des hohen Dokumentationsaufwandes.

#### **Bewertungskriterien und dessen Gewichtung**

Für die möglichst objektive Bewertung der vier Vorgehensmodelle werden die folgenden Bewertungskriterien herangezogen.

Besondere Bedeutung wird der Iterationsfähigkeit zugesprochen. Die Begründung für diese Entscheidung liegt in dem Fokus der forensischen Untersuchung innerhalb der vorliegenden Arbeit. Die forensische Untersuchung soll einen forschenden Charakter besitzen und sich bewusst von einer gerichtsverwertbaren Untersuchung unterscheiden. Es kann davon ausgegangen werden, dass einige Schritte der forensischen Untersuchung oftmals wiederholt werden. Dadurch kann auf neue Erkenntnisse schnell reagiert werden und jeweilige Schritte effizient und ohne großen Überhang wiederholt werden. Aus diesem Grund erhält die Iterationsfähigkeit eine Gewichtung von 40%.

Die Anpassungsfähigkeit spielt ebenfalls eine wichtige Rolle, dies bezieht sich auf die Einzelschritte des Vorgehensmodells. Es muss gewährleistet sein, dass nicht jeder einzelne Prozessschritt in der vorgegebenen Reihenfolge, innerhalb jeder Iterationsschleife, durchgeführt werden muss. Dementsprechend wird diesem Bewertungskriterium eine Gewichtung von 20% zugesprochen.

Ein weiteres Kriterium, das für die Auswahl eines geeigneten Vorgehensmodells eine besondere Bedeutung einnimmt, ist die Dokumentation. Denn der Dokumentationsaufwand bei einer forensischen Untersuchung ist nicht zu vernachlässigen. Im Fokus der forensischen Analyse liegt der forschende Charakter und keine gerichtsverwertbare Analyse. Dadurch ist der Anspruch an die Ausführlichkeit der Dokumentation deutlich geringer als in einem Gerichtsverfahren. Die Dokumentation muss nicht vollumfänglich lückenlos und gerichtsverwertbar sein, sondern lediglich die Forschungsergebnisse präsentieren und in der Gesamtheit den Ansprüchen an eine Mastarbeit genügen. Deswegen wird dem Bewertungskriterium der Dokumentation eine Gewichtung von 20% zugeteilt.

Die Komplexität des Vorgehensmodells korreliert sehr stark mit der benötigten Zeit zur Durchführung der forensischen Analyse. Ein Beispiel hierfür wäre eine starke Verzahnung der einzelnen Prozessschritte, die bei neuen Erkenntnissen zu einem höheren Wiederholungsaufwand führen würden. Deshalb wird der Komplexität eine Gewichtung von 10% zugeteilt.

Das letzte gewählte Bewertungskriterium ist die Methoden- und Werkzeugwahl. Damit eine möglichst flexible und ausführliche Untersuchung durchgeführt werden kann, müssen eine Vielzahl von allgemein anerkannten Methoden und Werkzeugen zur Auswahl stehen. Diese können strategisch ausgewählt und wenn möglich angepasst werden. Dadurch kann eine gezielte und effiziente Untersuchung sichergestellt werden. Auch dieses Kriterium wird mit einer Gewichtung von 10% in die Bewertung mit einbezogen.

### Bewertung der Nutzwertanalyse

Kriterium	Gewichtung	BSI		SAP	
		Punkte	Bewertung	Punkte	Bewertung
Iterationsfähigkeit	40 %	10	4,0	10	4
Anpassungsfähigkeit	20 %	5	1,0	9	1,8
Dokumentationsaufwand	20 %	4	0,8	7	1,4
Komplexität	10 %	4	0,4	9	0,9
Methodenauswahl	10 %	8	0,8	10	1
<b>Ergebnis</b>	<b>100 %</b>		<b>7,0</b>		<b>9,1</b>

Tabelle 4: Ausschnitt: Bewertungsmatrix Nutzwertanalyse zu den Vorgehensmodellen

In der **Tabelle 4** ist ein Ausschnitt der Bewertungsmatrix dargestellt. Die vollständige Bewertungsmatrix ist im Anhang in Kapitel 10 dargestellt.

Das beste Ergebnis hat das SAP-Vorgehensmodell erzielt. Das SAP-Modell erzielt 9,1 von möglichen 10 Punkten. Einen großen Einfluss auf dieses hohe Ergebnis hat die Iterationsfähigkeit, sowie die Anpassungsfähigkeit und die Methodenauswahl. Durch die Einfachheit des Modells und die geringen Vorgaben ist dieses Modell am flexibelsten anpassbar. Weiterhin besticht es durch eine sehr gute Iterationsfähigkeit, die kurze Iterationszyklen und effiziente Reaktionen auf neue Erkenntnisse erlaubt. Weiterhin spricht die geringe Komplexität für dieses Vorgehensmodell.

Sehr ähnlich zu dem SAP-Modell ist das Modell nach Kent, Chevalier, Grance, Dang. Da dieses Modell dem SAP Modell gegenüber keine weiteren positiven Eigenschaften besitzt und dieses lediglich um einen Prozessschritt erweitert, wird das SAP-Modell dem Modell nach Kent, Chevalier, Grance, Dang vorgezogen.

Das Casey-Modell erzielt mit Abstand das schlechteste Ergebnis mit 2,2 Punkten von 10 möglichen Punkten. Dies liegt an dem starren Ablauf (Wasserfall-Modell) und den stark ineinander verzahnten Einzelschritten. Darüber hinaus ist es nicht dafür ausgelegt einzelne Prozessschritte iterativ zu wiederholen, sodass es dem forschenden Charakter dieser Arbeit nicht gerecht werden kann.

Das BSI-Modell besticht durch die Iterationsfähigkeit. Die Prozessschritte sind kreisförmig angeordnet und somit auf Iterationen ausgelegt. Weiterhin wird der BSI-Leitfaden positiv gewertet, in diesem sind sämtliche Prozessschritte detailliert beschrieben und zum größten Teil mit Beispielen versehen. Sehr aufwändig und für diese Arbeit

### **3.3. VORGEHENSWEISE DER FORENSISCHEN UNTERSUCHUNG**

---

nicht notwendig ist der große Dokumentationsaufwand. Für die lückenlose und gerichtsfeste Dokumentation muss jeder Prozessschritt sehr ausführlich dokumentiert werden. Dies würde den Umfang innerhalb dieser Arbeit sprengen und liegt ebenfalls außerhalb der Erwartungen.

Damit fällt die Entscheidung auf das SAP-Modell. Das wird auf die forensische Untersuchung des Infotainment-Images in dieser Arbeit angewendet.

## **3.3 Vorgehensweise der forensischen Untersuchung**

Nach der Wahl eines geeigneten Vorgehensmodells muss die Vorgehensweise in den einzelnen Phasen konzipiert werden. Begonnen wird mit der Secure Phase, die die Datensammlung und -sicherstellung beinhaltet.

Darauf aufbauend wird das Konzept für die Analyse Phase erstellt. Dieses unterteilt sich hauptsächlich in die Bereiche „Automatisierte forensische Untersuchung“, „Manuelle forensische Untersuchung“ und „SQLite Datenbank-Rekonstruktion“.

Abschließend wird das Konzept der Present Phase ausgearbeitet. Diese dient der übersichtlichen und einfachen Präsentation der Ergebnisse aus den Analysen und Untersuchungen.

### **3.3.1 Secure Phase**

Die Secure Phase beinhaltet die Sicherung der für die forensische Untersuchung wertvollen Daten. Dies bedeutet für die vorliegende Arbeit die Sicherung des Infotainment-Images. Ursprünglich sollte noch ein weiteres aktuelles Image aus dem Jahr 2020 eines anderen Herstellers bereitgestellt werden. Aufgrund der aktuellen Pandemie ist der Hersteller nicht mehr interessiert.

Grundlegend sind diese Art Images nur sehr schwer zugänglich. Im Internet stehen leider keine Images von Infotainmentsystemen zur Verfügung. Somit wird ein anonymisiertes bereitgestelltes Image untersucht.

Die allgemeine Signifikanz des Ergebnisses wird nicht davon beeinträchtigt. Denn wie in Kapitel 2.3 beschrieben, wird das QNX Neutrino RTOS von mehr als 40 Herstellern in vielen Millionen aktuellen PKWs eingesetzt.

Die Vorgehensweise der Secure Phase wird in **Bild 17** visualisiert.

### 3.3. VORGEHENSWEISE DER FORENSISCHEN UNTERSUCHUNG

---

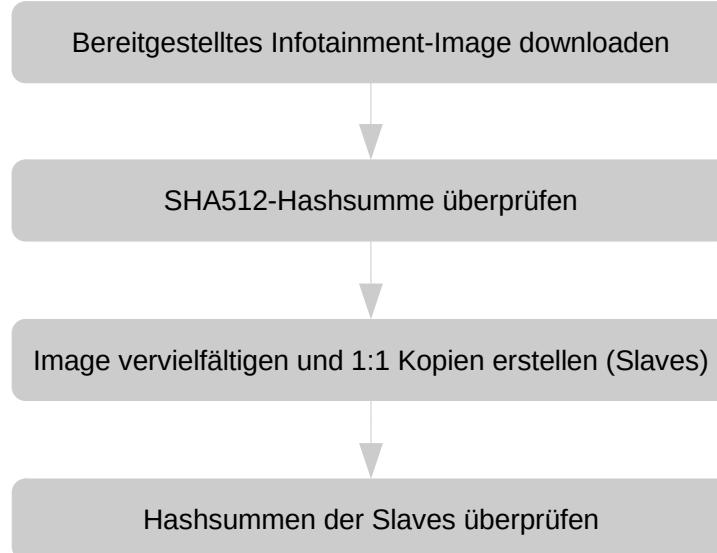


Bild 17: Vorgehensweise der Secure Phase

#### 3.3.2 Analyse Phase

Innerhalb dieses Kapitels wird die Vorgehensweise innerhalb der Analyse Phase ausgearbeitet. Unterschieden wird zwischen der automatisierten und der manuellen forensischen Untersuchung. Die automatisierte forensische Untersuchung ist der erste Ansatz, indem ein grober Überblick über das Image und die darin befindlichen Daten generiert werden sollen.

Im nächsten Schritt wird darauf aufbauend die manuelle forensische Untersuchung durchgeführt. Diese stellt somit die Detailanalyse dar.

In dem letzten Abschnitt der Analyse Phase wird versucht, soweit vorhanden, beschädigte SQLite Datenbanken zu rekonstruieren. Dies dient der vollumfänglichen Analyse des Infotainment-Images.

##### Automatisierte forensische Untersuchung

Im ersten Schritt wird *Autopsy* verwendet. Dies soll einen ersten Überblick über das Betriebssystem QNX Neutrino RTOS und den Inhalt des Images geben.

Jedoch zeigen die Spezifikationen, dass die Dateiformate QNX4 und QNX6 nicht von *Autopsy* unterstützt werden und damit aller Voraussicht nach auch nicht sinnvoll untersucht werden können [43]. Gleiches gilt für die Windows-basierten Forensik-Programme *AccessData FTK*, *EnCase*, *AXIOM*, *X-Ways Forensics* und *Nuix* [44, 45].

### **3.3. VORGEHENSWEISE DER FORENSISCHEN UNTERSUCHUNG**

---

Aus diesem Grund muss der Analyseprozess für eine ganzheitliche forensische Untersuchung manuell durchgeführt werden. Trotzdem soll ein Versuch gestartet werden, eine forensische Untersuchung mittels *Autopsy* durchzuführen. Dadurch soll ermittelt werden, welcher Datengehalt auch ohne eine offizielle Unterstützung des Dateiformats generiert wird. Die Ergebnisse der forensischen Untersuchung mittels *Autopsy* werden in dem Kapitel 4.2.1 veranschaulicht.

Definierte These: Ohne eine offizielle Unterstützung von QNX kann *Autopsy* keinen sinnvollen Datengehalt zur forensischen Untersuchung beitragen.

#### **Manuelle forensische Untersuchung**

Eine forensische Untersuchung eines Computersystems ist eine sehr komplexe Aufgabe. Um diese Aufgabe souverän durchzuführen, sind einige Dinge zu beachten.

Zu jedem Zeitpunkt der forensischen Untersuchung steht die Integrität der zu untersuchenden Daten im Vordergrund. Die forensische Analyse muss neben den extrahierten Daten und den erzielten Ergebnissen zeigen, dass die Daten weder manipuliert noch fälschlicherweise unvollständig sind. Aus diesem Grund wird ständig darauf geachtet, dass mit den Slaves (1:1 Kopien) und nicht mit dem Master (Original) gearbeitet wird.

Für die Wiederholbarkeit der einzelnen Prozessschritte innerhalb der manuellen Analyse werden bei jedem Einsatz von Software die jeweilige verwendete Version sowie die abgesetzten Befehle und Erläuterungen zu diesen dokumentiert. Da dies eine Postmortem-Analyse ist, sind keine flüchtigen Daten vorhanden, die unter besonderer Beachtung zuerst untersucht werden müssten. Aus diesem Grund wird die Reihenfolge der systematischen manuellen forensischen Untersuchung des Infotainment-Images, wie in dem **Unified Modeling Language (UML)** Aktivitätsdiagramm visualisiert **Bild 18**, festgelegt.

Im ersten Schritt soll ein Eindruck über den Umfang des Infotainment-Images erzeugt werden. Das bedeutet, dass zuerst das Partitions-Layout und die enthaltenen Dateisysteme festgestellt werden sollen. Anschließend werden die einzelnen Partitionen untersucht und relevante Dateien aus dem Infotainment-Image extrahiert und analysiert.

Nachdem sämtliche Partitionen untersucht und analysiert sind, wird der nicht-allozierte Bereich des Images untersucht. Die in diesem Bereich befindlichen Daten sollen ebenfalls extrahiert und analysiert werden. Oftmals können einige Dateien aus dem nicht-allozierten Bereich eines Datenträgers nicht direkt untersucht werden, da sie sich nicht öffnen lassen. Speziell wird hierbei auf die in diesem Bereich befindlichen SQLite Datenbanken geachtet, weil vermutet wird, dass in diesen weitere personenbezogene oder personenbeziehbare Daten liegen könnten.

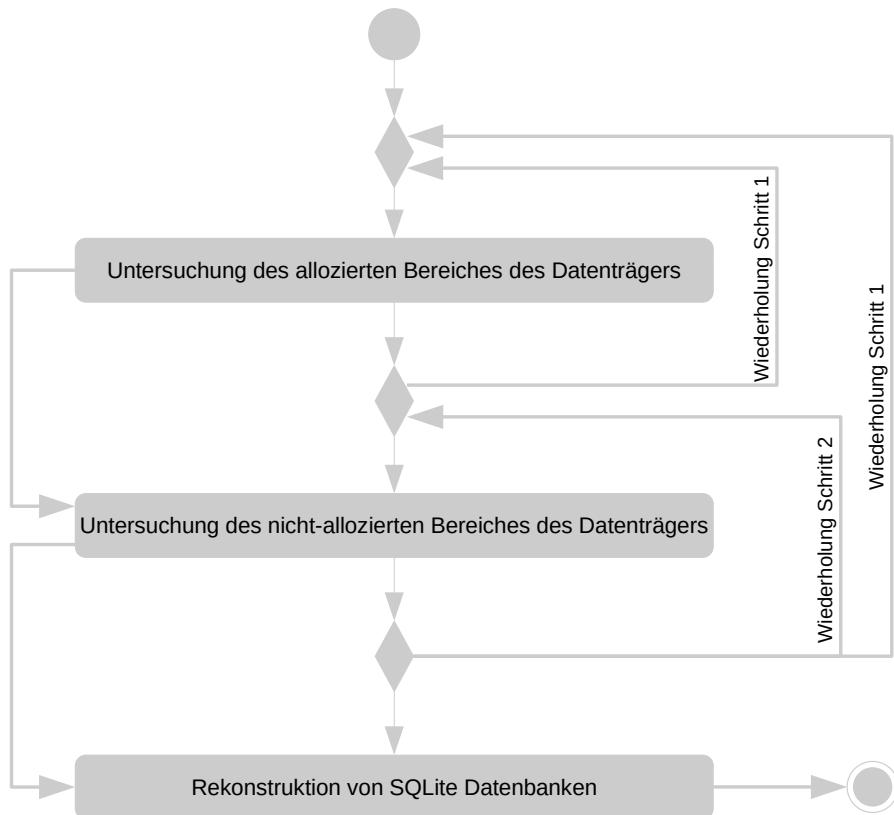


Bild 18: Vorgehensweise der manuellen forensischen Untersuchung

Sollten beschädigte oder unvollständige SQLite Datenbanken aufgefunden werden, so wird im letzten Schritt der manuellen forensischen Untersuchung versucht diese SQLite Datenbanken zu rekonstruieren.

Wenn im Laufe der Analyse auf Basis gewonnener Ergebnisse festgestellt wird, dass einzelne Prozessschritte wiederholt werden müssen oder durch andere Techniken erweitert werden können, so ist dies zu jedem Zeitpunkt der Analyse möglich.

#### 3.3.3 Present Phase

Innerhalb dieser Phase der forensischen Untersuchung werden die Ergebnisse, die in der Analyse Phase analysiert und untersucht werden, zusammengefasst und visuell aufgearbeitet.

Eine einfache und zusammenhängende Präsentation der Ergebnisse ist von besonderer Bedeutung. Die gewonnenen Informationen müssen in sich schlüssig sein und einen guten Überblick über das untersuchte Infotainment-Image ermöglichen. Das Ziel ist die Ergebnisse in einer Form darzustellen, dass es auch einem Laien ermöglicht wird, die Zusam-

### **3.4. SOFTWAREAUSWAHL**

---

menhänge und Ergebnisse zu verstehen. Im Gegensatz zu einer gerichtlichen forensischen Untersuchung sollen nicht nur die aufgefundenen personenbezogenen und personenbeziehbaren Daten dargestellt werden, sondern auch ein Überblick über das eingesetzte Infotainmentsystem, das installierte Betriebssystem und dessen Funktionsweise im Allgemeinen gegeben werden.

Im Anschluss an die Present Phase soll das Infotainmentsystem virtualisiert werden. Die Virtualisierung soll das System im laufenden Betrieb zeigen. Dadurch soll ein besserer Eindruck davon gewonnen werden, wie die in der forensischen Untersuchung analysierten und ausgewerteten Daten durch das Infotainmentsystem aufbereitet und dargestellt werden. Weiterhin sollen die verschiedenen Applikationen, die auf dem Infotainmentsystem zur Verfügung stehen, präsentiert werden.

## **3.4 Softwareauswahl**

Dieses Unterkapitel vermittelt einen Überblick über die verwendete Open-Source-Software, sowie Erläuterungen zu den einzelnen Programmen. Aufgrund der großen Funktionsumfänge der einzelnen Programme werden nur die für diese Arbeit notwendigen Funktionen weitergehend erläutert. Der restliche Funktionsumfang kann mit Hilfe der Quellenangaben recherchiert werden.

### **3.4.1 Autopsy**

Primär wird eine forensische Untersuchung automatisiert bzw. teilautomatisiert mit einer Forensik-Software durchgeführt. Dafür gibt es einige verschiedene Programme auf Basis von Mac, Windows und Linux am Markt.

Eine Marktübersicht zeigt, dass einige kommerzielle, sowie nicht-kommerzielle Software-Programme zur Verfügung stehen. Die bekanntesten kommerziellen und nicht-kommerziellen Forensik-Programme für das Betriebssystem Windows sind *AccessData FTK*, *EnCase*, *AXIOM*, *X-Ways Forensics* und *Nuix*.

Auf der Basis von Unix Betriebssystemen wird hauptsächlich auf das Open-Source-Programm *Autopsy* zurückgegriffen. Da die Forensik-Maschine mit einer installierten Linux-Forensik-Distribution ausgestattet ist, wird der Einsatz von *Autopsy* favorisiert. Dadurch könnte auf den Einsatz einer Windows 10 VM verzichtet werden.

*Autopsy* ist eine Forensik-Plattform und die grafische Schnittstelle zu dem *The Sleuth Kit* zur automatisierten forensischen Untersuchung von Computersystemen und Speichermedien. Die Software hat einen modularen Aufbau und kann durch Dritt-Parteien mit neu-

### 3.4. SOFTWAREAUSWAHL

---

en Funktionsumfängen erweitert werden. Die kommerzielle und nicht-kommerzielle Verwendung von Autopsy ist vollständig kostenlos. Autopsy bietet die selben Kernfunktionen und darüber hinaus weitere essenzielle Funktionen, die andere kommerzielle Forensik-Programme nicht anbieten. Verwendung findet *Autopsy* bei Strafverfolgern, im Militär, in Unternehmen und auch bei Privatpersonen [46].

Durch das intuitive Design, die grafische Benutzerschnittstelle und dem digitalen Assistenten „wizard“ sollen auch nicht-technische Benutzer effektiv mit der Software arbeiten können. Die von *Autopsy* eingesetzte Technik für die forensische Untersuchung basiert auf der Technik des *File Carvings*. Das bedeutet, dass die Rohdaten der Dateien ohne Beachtung der Metadaten analysiert und anhand der Signaturen und der *Magic Numbers* identifiziert werden. Eine möglichst schnelle Analyse wird durch die Verarbeitung der Hintergrundprozesse auf mehreren Prozessorkernen und deren parallelen Abarbeitung erreicht [47].

#### 3.4.2 The Sleuth Kit

Für den manuellen Untersuchungsprozess des Infotainment-Images wird das *The Sleuth Kit* verwendet. Dies ist einerseits darin begründet, dass es auf Unix/Linux basiert und andererseits sämtliche für die forensische Untersuchung benötigte Software beinhaltet. Darüber hinaus ist es allgemein anerkannt und besitzt einen offenen Quellcode (Open Source).

Das **The Sleuth Kit** (TSK) ist eine Open-Source-Software zur forensischen Analyse von Computer Systemen und Speichermedien. In die deutsche Sprache übersetzt bedeutet der Name *Das Schnüffelstück*. Hauptbestandteile der Software sind eine C-Bibliothek sowie eine Sammlung von Unix und Windows basierten Kommandozeilen-Programmen. Grob können die verschiedenen Kommandozeilen-Programme in Dateisystem Tools und Volume-System Tools unterteilt werden [43].

Die Volume-System Tools stellen Programme zur Verfügung, mit deren Hilfe sich Layouts von Festplatten und anderen Medienspeichern untersuchen und extrahieren lassen. Unterstützte Formate sind beispielweise BSD-, DOS-, GPT- und Mac-Partitionen [43].

Die Dateisystem Tools bieten die Möglichkeit einer nicht-intrusiven Untersuchung des Dateisystems. Das jeweilige Dateisystem kann ohne das Betriebssystem verarbeitet werden, wodurch der sichtbare *allocated* und auch der gelöschte, beziehungsweise versteckte, *unallocated* Bereich des zu untersuchenden Dateisystems analysiert werden kann. Dabei werden sämtliche historische und aktuelle Dateisystem-Formate, wie beispielsweise NTFS, FAT/ExFAT, Ext2, Ext3 und Ext4 unterstützt. Durch die aktive Weiterentwicklung

der einzelnen Kommandozeilen-Programme kommen immer wieder neue Formate hinzu [43].

#### 3.4.3 bring2lite

Das Kommandozeilen-Programm *bring2lite* ist ein digitales Werkzeug für die Datenbank-Rekonstruktion<sup>1</sup> von SQLite Datenbanken<sup>2</sup>. Die Entwickler von *bring2lite* geben eine Wiederherstellungsrate von 53% an. Verglichen mit acht weiteren Tools dieser Art, hat es die höchste Wiederherstellungsrate erreicht [29]. Aus diesem Grund wird es für die forensische Untersuchung innerhalb dieser Arbeit verwendet und folgend vorgestellt.

Die Open Source Software ist im Rahmen einer Masterthesis von Christian Meng entwickelt und publiziert worden [32]. Bring2lite basiert auf der Programmiersprache Python und nutzt einen strukturellen Ansatz zur Wiederherstellung gelöschter Daten einer Datenbank. Charakterisiert wird dieser Ansatz über Wiederherstellungsalgorithmen, die den allgemeinen strukturellen Aufbau von SQLite Datenbanken und deren Funktionsweise nutzen, um gelöschte Datensätze wiederherzustellen [29].

#### 3.4.4 bulk\_extractor

Der *bulk\_extractor* ist ein kommandozeilen-basiertes Carving<sup>3</sup> und Feature-Extraktions-Tool. Im Gegensatz zu der üblichen forensischen Praxis analysiert und extrahiert dieses Programm keine Dateien, sondern betrachtet die Daten in „Masse“. Mit dem Ansatz der strombasierten Massendatenanalyse wird das gesamte Medium ohne Berücksichtigung des Dateisystems oder der Dateisystem-Strukturen des Mediums untersucht. In Kombination mit den Funktionen von *Multithreading*<sup>4</sup> und *Parallelization*<sup>5</sup> ist eine performante Verarbeitung gewährleistet [50].

Das Programm hat verschiedene Funktionen, wobei die wichtigsten Funktionen im Folgenden dargestellt sind:

---

<sup>1</sup>Für Erläuterungen zu dem Thema SQLite Datenbanken siehe Abschnitt 2.4 SQLite Datenbanken

<sup>2</sup>Für Erläuterungen zu dem Thema Datenbank-Rekonstruktion siehe Abschnitt 2.5 Rekonstruktion von SQLite Datenbanken

<sup>3</sup>Das *Carving* ist eine forensische Vorgehensweise, bei der primär die nicht-allozierten Bereiche des Datenträgers auf „gelöschte Dateien“ untersucht werden [6].

<sup>4</sup>Multithreading bezeichnet eine Technik, die es ermöglicht, mehrere Threads (Abarbeitungsläufe) annähernd gleichzeitig auf einem einzelnen Prozessorkern zu verarbeiten [48].

<sup>5</sup>Parallelisierung ist die simultane Verarbeitung von Programmteilen, bei der die Reihenfolge der Verarbeitung keinen Einfluss auf das Resultat hat [49].

#### Identifikation und Extraktion von Merkmalen

Das Programm findet Informationen, wie beispielsweise Email-Adressen, Web-Adressen, Telefonnummern, Kreditkartennummern und viele weitere derartige Informationen. Darüber hinaus werden Merkmale von Datei-Typen erkannt und somit auch PNG, JPEG, Word, Datenbanken oder ähnliche Dateien identifiziert. Eine Besonderheit dabei ist, dass der *bulk\_extractor* dabei komprimierte, unvollständige und gelöschte Daten verarbeitet. Selbst verschlüsselte RAR-Dateien können erkannt und extrahiert werden [51].

#### Erstellung von Wörterlisten

Das Programm listet jedes gefundene Wort unabhängig, ob es in komprimierten Archiven oder aus dem nicht-allozierten Bereich des Mediums stammt, in einer sogenannten Wörterliste auf [51].

#### Erstellung von Histogrammen

Ein weiteres Feature ist die Erstellung von Histogrammen zu den am meisten verwendeten Informationen. Dies gibt dem Ermittler Hinweise zu dem Nutzerverhalten und eine quantitative Aussage zu den verschiedenen Merkmalen [51].

Auszeichnend durch diese und weitere Funktionen findet sich der Einsatz dieser Software im frühen Stadium der Untersuchung wieder. Einerseits entsteht ein Überblick über die Informationen und Inhalte zu den vordefinierten Merkmalen. Andererseits werden Daten aus den nicht-allozierten Bereichen des Mediums extrahiert, wodurch der Einsatz im späteren Verlauf ebenso sinnvoll sein kann. Anhand der Informationen kann eine weiterführende detaillierte Untersuchung und Analyse der Dateien auf dem Medium durchgeführt werden.

### 3.4.5 Binwalk

Das Open Source Extraktions-Tool *Binwalk* wurde 2010 von Craig Heffner entwickelt und unter der MIT<sup>6</sup>-Lizenz publiziert [52].

Das Programm ist ein ganzheitliches Werkzeug für die Analyse, Reverse-Engineering und die Extraktion von Firmwareabbildern. Es untersucht Binärabbilder auf ausführbaren Code und eingebettete Dateien. Insbesondere wird es genutzt, um Firmware-Images in Binärdateien aufzufinden und zu extrahieren [53].

Durch die Implementierung der *libmagic* Bibliothek können die „Magic Numbers“<sup>7</sup> von

---

<sup>6</sup>Massachusetts Institute of Technology (MIT)

<sup>7</sup>Als *Magische Zahlen* wird die Anfangskennung im Dateikopf (Header) bezeichnet. Dies ist ein einzigartiger Bytecode, der das Dateiformat kennzeichnet [54].

Dateien erkannt und somit der jeweilige Dateityp (MIME Type) bestimmt werden. Weiterhin enthält *Binwalk* eine eigene benutzerdefinierte Datei die verschiedene optimierte „magische Signaturen“ enthält. In der Datei sind Signaturen enthalten, die besonders häufig in Firmware-Abbildern vorkommen. Mit Hilfe dieser verbesserten Signaturen-Datei können beispielweise Dateisysteme, Linux-Kernel, Firmware-Header oder auch komprimierte Dateiarchive identifiziert werden. Darüber hinaus kann *Binwalk* auch dazu verwendet werden Boot-Informationen zu identifizieren. Auch unvollständige Gesamt- oder Partitionsimages können mit *Binwalk* analysiert werden [55, 56].

Aufgrund der verbesserten Signaturen-Datei wird *Binwalk* innerhalb dieser Arbeit verwendet. Mit Hilfe der verbesserten Signaturen-Datei sollen die einzelnen Dateisysteme des Infotainment-Images identifiziert werden.

#### 3.4.6 QEMU

Für die Virtualisierung des Infotainmentsystems wird der generische und quelloffene Maschinенemulator und Virtualisierungssoftware *QEMU* ausgewählt. Mit der Verwendung von *QEMU* wird in den zwei Modi als Maschinенemulator oder als Virtualisierung unterschieden.

Wird *QEMU* als Maschinенemulator verwendet, können Programme und Betriebssysteme auf einer anderen Maschine ausgeführt werden, als für die diese eigentlich gemacht wurden. Somit kann beispielsweise ein Programm oder Betriebssystem, dass für ein Board mit einem ARM-Prozessor entwickelt ist, auf dem eigenen Intel-basierten Computer ausgeführt werden [57].

Für die Virtualisierung auf einem Linux-Host wird neben *QEMU* der Xen-Hypervisor oder das KVM-Kernelmodul benötigt. *QEMU* führt den Gastcode direkt auf der Host-CPU aus. Das hat zur Folge, dass nahezu native Leistung erreicht werden kann. *QEMU* kann in Verbindung mit dem KVM-Kernelmodul x86-, Server- und Embedded-PowerPC-, 64-Bit-POWER-, S390-, 32-Bit- und 64-Bit-ARM-Gäste virtualisieren [57].

## 3.5 Security Ausblick

Dieser Abschnitt behandelt die Sicherheitsmechanismen von dem Betriebssystem QNX Neutrino RTOS. Blackberry bewirbt eine Sicherheitsfunktion sehr stark und gezielt. Dabei handelt es sich um einen Schreibschutz, sodass die Dateisysteme nur im Lesemodus (Read-Only) eingehängt werden können.

### 3.5. SECURITY AUSBLICK

---

Ziel ist es einen Ansatz zu entwickeln, mit dessen Hilfe eben diese Sicherheitsfunktion umgangen werden kann und somit ein Schreibzugriff auf das System ermöglicht wird. Zur Beweisführung sollen Dateien erstellt, manipuliert und gelöscht werden. Ist dies möglich, so gilt der Schreibschutz als gebrochen und der Ansatz als erfolgreich.

Zur Vorbereitung muss das Infotainment-Image vorerst in ein Rohformat konvertiert werden. Hierfür wird das Infotainment-Image in das RAW-Format konvertiert. Anschließend soll von dem im Rohformat vorliegendem System eine Bit-genaue Kopie auf einen externen Datenträger erfolgen. Hierbei ist zu beachten, dass der externe Datenträger mindestens genauso groß oder größer als das Image im Rohformat selbst ist.

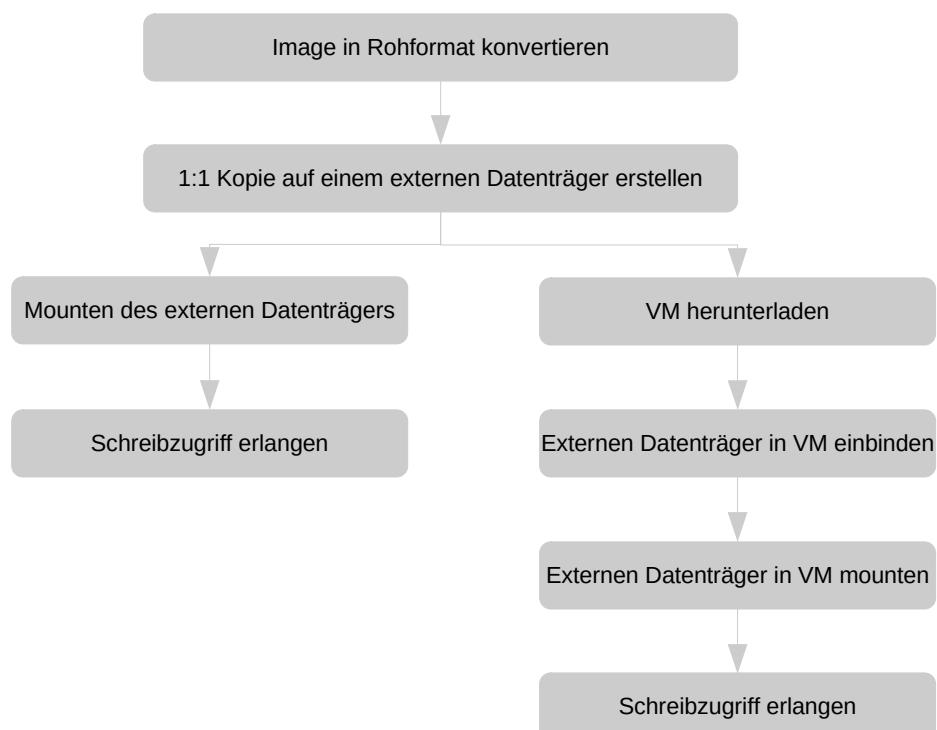


Bild 19: Lösungsansätze Schreibzugriff

Daraufhin werden zwei verschiedene Ansätze untersucht. Im ersten Ansatz soll versucht werden, den externen Datenträger direkt in die Forensik-Maschine einzuhängen. Ist dieser Ansatz nicht zielführend oder kann kein Schreibzugriff erlangt werden, dann wird der zweite komplexere Ansatz gewählt.

Für den zweiten Ansatz braucht es eine VM vom QNX Neutrino RTOS, bereitgestellt von Blackberry. Diese soll auf der Forensik-Maschine gestartet werden. Anschließend soll der externe Datenträger in die VM eingebunden und eingehängt werden.

Auch in diesem Ansatz gilt es nun den Schreibschutz zu überlisten. Zur Beweisführung

soll ein Verzeichnis auf einer der Partitionen angelegt werden, innerhalb dessen eine Datei erstellt, beschrieben und anschließend wieder gelöscht werden soll. Die letzte Operation beinhaltet die Löschung des erstellten Verzeichnisses. Kann dies umgesetzt werden, gilt der Schreibschutz als gebrochen und der Ansatz als erfolgreicher Angriff.

In dem **Bild 19** werden die beiden Ansätze visualisiert.

## 3.6 Physikalischer Analyseansatz

Für die physikalische Auswertung gibt es mehrere Ansätze die angewendet werden können. Eine Möglichkeit wäre eine Linux-Distribution mit QNX Bibliotheken, Treibern und Programmen auszustatten, sodass eine weitestgehende Unterstützung von QNX ermöglicht wird. Weil einige dieser Bibliotheken, Treiber und Programme proprietär sind, wird es schwierig die eigene Forensik Distribution für eine vollumfängliche Unterstützung von QNX zu erweitern.

Es sind zwar einige quelloffene und frei zugängliche Treiber und Programme zur Unterstützung von QNX im Internet verfügbar, jedoch wird die Nutzung dieser zwangsläufig dazu führen, dass ständig nach neuen Lösungen gesucht und sehr viel probiert werden muss. Schlussendlich werden einige Projekte eingestellt oder für neuere Funktionen nicht entwickelt, wodurch dieser Ansatz zu ständigen Herausforderungen und Wartungsarbeiten des Systems führen würde. Aus diesem Grund wird dieser Ansatz nicht weiter verfolgt.

Daraus folgt, dass zwangsläufig eine native Installation von QNX durchgeführt werden muss. Für die native Installation stehen zurzeit vier sinnvolle Herangehensweisen bereit.

Der zweite mögliche und am einfachsten umzusetzende Ansatz besteht daraus, die von QNX bereitgestellte Installations-CD in Form einer ISO-Datei auf eine CD oder DVD zu brennen. Danach kann sie im CD Laufwerk gestartet werden. Anschließend wird die Installation mithilfe des Installations-Assistenten durchgeführt. Als Nachteil ist das Alter -elf Jahre- der darauf bereitgestellten Version 6.5. des QNX Neutrino RTOS zu bewerten. Ältere Versionen werden möglicherweise von modernen Computern nicht mehr unterstützt und deshalb nicht gestartet. Trotzdem wird dieser Ansatz als erstes verfolgt, da er zum einen sehr einfach umsetzbar ist und zum anderen für das vorliegende Infotainment-Image ausreicht. Denn das vorliegende Infotainment-Image verfügt über ein installiertes QNX Neutrino RTOS in der Version 6.5.

Für die beiden anderen Ansätze muss ein QNX Board Support Package (BSP) heruntergeladen und die für das Betriebssystem notwendigen Bibliotheken, Treiber und Programme

### 3.6. PHYSIKALISCHER ANALYSEANSATZ

selbst kompiliert werden. Dadurch kann anschließend ein eigenes QNX Neutrino RTOS-Image erstellt werden. Auch hierbei stehen zwei Möglichkeiten zur Verfügung: Im QNX Software Center werden zwei BSPs angeboten, die es ermöglichen könnten, eine native Installation durchzuführen, ohne dass ein spezielles Board für QNX beschafft werden muss.

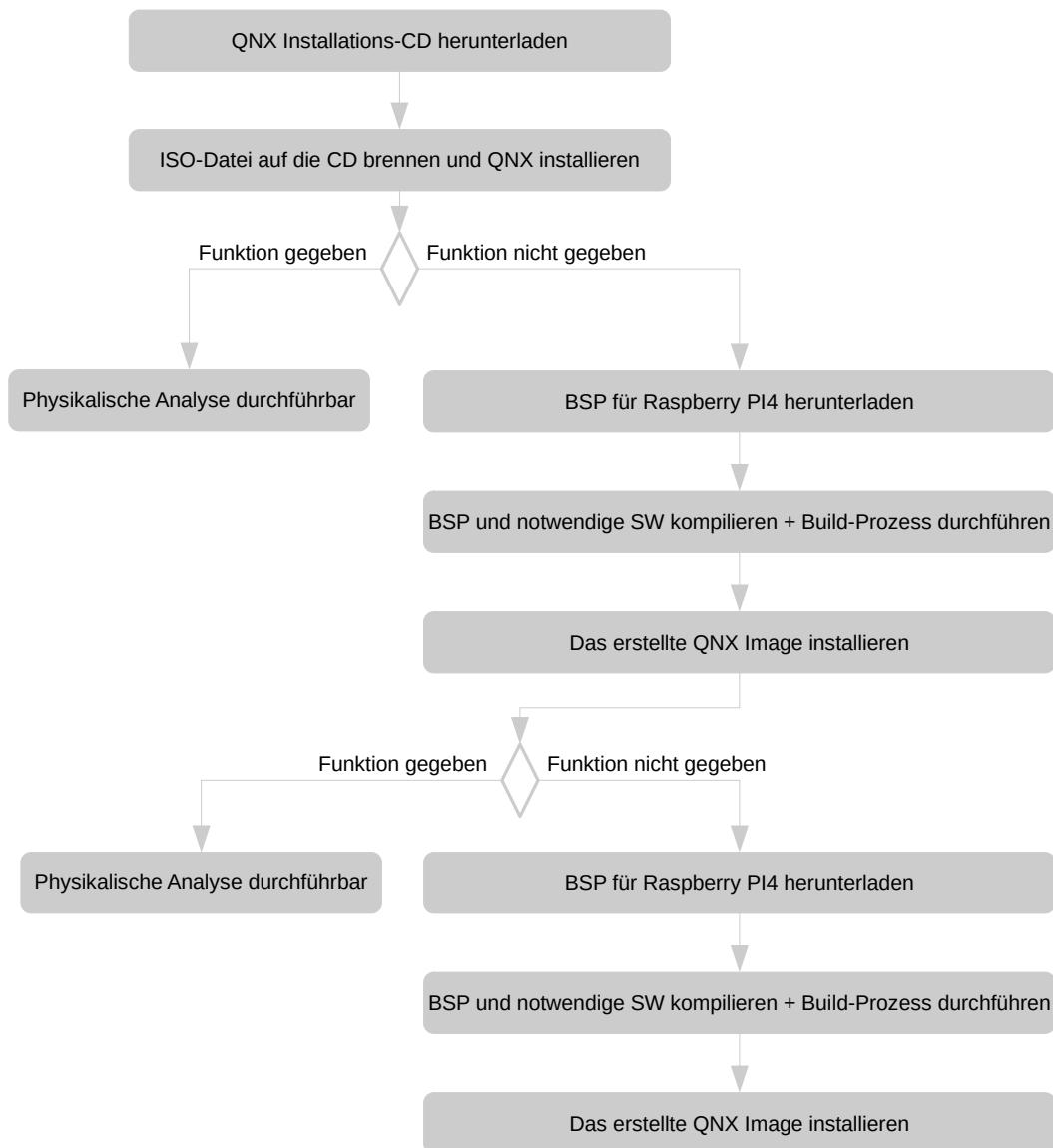


Bild 20: L\u00f6sungsans\u00e4tze physikalische Auswertung

Das erste BSP, das einen sinnvollen Einsatz finden kann, ist ein generisches BSP für x64-Architekturen. Dieses wird offiziell von sechs CPUs unterstützt. Dadurch ist die Kompatibilität auf wenige Computer reduziert und es müsste für die Forensik-Maschine darauf geachtet werden, dass eine der unterstützten CPUs verbaut ist. Da das QNX Neutrino RTOS

### **3.7. VERGLEICHSKONZEPT LINUX VS QNX**

---

für Ein-Kern CPUs entwickelt wurde, werden trotz passender Architektur und CPU Kompatibilitätsprobleme erwartet.

Um diese Kompatibilitätsprobleme zu umgehen, gibt es ein weiteres BSP für den *Raspberry 4B*. Die Verwendung dieses BSPs würde sicherstellen, dass es auf vielen Geräten funktioniert, da der Raspberry sehr weit verbreitet ist und immer die gleiche Hardware besitzt. Zudem ist dieser sehr mobil und kostengünstig. Bei diesem Ansatz liegt das Problem jedoch in dem BSP selbst. Das ist aktuell noch mit dem Zusatz *experimental* markiert, wodurch unerwartete Probleme während des Kompilierens oder im laufenden Betrieb entstehen können.

Wie aus den vorangegangen Erklärungen ersichtlich, wird es nicht ganz trivial eine native Installation von QNX durchzuführen. Für den physikalischen Ansatz innerhalb dieser Arbeit wird deswegen die in der Grafik 20 dargestellte Vorgehensweise festgelegt.

Im ersten Ansatz wird die von QNX bereitgestellte ISO-Datei heruntergeladen und auf eine CD oder einen USB-Stick gebrannt. Anschließend wird die Installations-CD oder der USB-Stick in den Computer eingesteckt und versucht darüber zu booten. Sollte dies gelingen, wird das QNX Neutrino RTOS in der Version 6.5 nativ auf die Festplatte des Computers installiert. Startet die Installations-CD nicht, dann wird dieser Ansatz verworfen und das BSP für den Raspberry PI 4B heruntergeladen.

Anschließend werden die notwendigen Treiber und Hilfsprogramme heruntergeladen und zusammen mit dem BSP kompiliert. Daraufhin kann das QNX-Image gebaut und auf den Datenträger des Raspberry PIs installiert werden. Wenn dieser komplikationslos startet, wird der Raspberry PI zukünftig für die physikalische forensische Untersuchung von QNX-Images verwendet.

Sollte auch dieser Ansatz fehlschlagen, wird auf das generische BSP für x64-Architekturen zurückgegriffen.

## **3.7 Vergleichskonzept Linux vs QNX**

In diesem Abschnitt wird ein Konzept entwickelt und eine Methodenauswahl getroffen, um das Betriebssystem QNX Neutrino RTOS im Kontext der Forensik näher zu beleuchten. Es gilt Unterschiede und Herausstellungsmerkmale gegenüber Linux-Standarddistributionen für den Desktop, wie beispielsweise Debian, Ubuntu oder Arch, herauszufiltern.

Im Vordergrund dieses Vergleichskonzepts und der anschließenden Bewertung steht die IT-Forensik. Das bedeutet, dass gezielt nach Aspekten bezogen auf die Arbeit des Foren-

### ***3.7. VERGLEICHSKONZEPT LINUX VS QNX***

---

sikers gesucht werden soll. Einander gegenübergestellt werden die positiven sowie negativen Aspekte des Betriebssystems. Auf Basis dieser Pro-Contra-Liste soll eine Nutzwertanalyse erstellt werden. Diese bildet die Grundlage, auf der anschließend die Bewertung erfolgt.

Ziel ist die Bewertung der Nützlichkeit, des Leistungsspektrums und möglicher Schwächen für den Anwender in der Forensik. Für die Auswahl der verschiedenen Bewertungskriterien müssen die Spezifikationen der verschiedenen Betriebssysteme verglichen werden. Abschließend wird eine Empfehlung für oder gegen den Einsatz eines QNX Neutrino RTOS ausgesprochen.

# 4 Analyse

Das Kapitel der „Analyse“ unterteilt sich in die folgenden acht Unterkapitel:

- Secure Phase
- Analyse Phase
- Erzwingen des Schreibzugriffs auf QNX Neutrino RTOS
- Weiterführende Analyse Phase
- Present Phase
- Virtualisierung
- Physikalischer Analyseansatz
- Vergleich von Linux Desktop-Distribution vs QNX Neutrino RTOS

In diesem Kapitel wird die Untersuchung des Infotainment-Images auf Basis der zuvor entwickelten Konzepte durchgeführt. Das erste Unterkapitel bildet hierbei die Secure Phase, die das methodische Sicherstellen der für die forensische Untersuchung wertvollen Daten beschreibt. Darauf folgt die Analyse Phase, in der die sichergestellten Daten untersucht, Spuren verfolgt und abschließend analysiert werden.

Im Anschluss daran erfolgt der Versuch der möglichen Aushebelung der Schreibschutz-Sicherheitsfunktion. Weiterhin wird das explizite Vorgehen hierbei dokumentiert und ein Beweis zum erfolgreichen Aushebeln angeführt. Auf Basis des neu erlangten Wissens kann die Untersuchung des Infotainmentsystem bzw. des Infotainment-Images fortgeführt werden und weitere Erkenntnisse dokumentiert und bewertet werden.

Daran anknüpfend werden in der Present Phase sämtliche Erkenntnisse und die erzielten Ergebnisse grafisch aufbereitet und leicht verständlich dargestellt. Darüber hinaus wird die Funktionsweise des Betriebssystems QNX Neutrino RTOS erläutert sowie eine Gesamtübersicht über das untersuchte Infotainmentsystem aufgezeigt.

Für eine Präsentation der untersuchten Daten im System wie auch der Darstellung installierter Applikationen soll das Infotainmentsystem virtualisiert werden. Anschließend wird das Konzept des physikalischen Ansatzes umgesetzt, um aufzuzeigen, wie zukünftig nicht nur eine logische, sondern auch eine physikalische Auswertung des QNX Neutrino RTOS ablaufen kann.

Den Abschluss dieses Kapitel bildet der Vergleich von Linux Desktop-Distribution und dem Betriebssystem QNX Neutrino RTOS. Im Mittelpunkt dieses Unterkapitels steht die Beantwortung der Frage inwiefern sich das QNX Neutrino RTOS von anderen Linux Distributionen unterscheidet und ob der Einsatz in der digitalen Forensik einen sinnvollen Platz finden kann.

### 4.1 Secure Phase

Für die folgenden Einsätze von Software wird zukünftig vor der ersten Nutzung die jeweilige Version ermittelt. Dies dient der Wiederholbarkeit der einzelnen Schritte und der Nachvollziehbarkeit bei Anomalien aufgrund der Versionierung.

Für die Übertragung ist das Infotainment-Image in das Format `.tar.bz2` konvertiert und komprimiert worden. Zur Überprüfung der Integrität wird die folgende SHA512-Hashsumme vorgegeben:

```
249b00e041ad6a89659bce9c40d3cb74a69a2e04e81d753550b54baba0ac6c5e91233  
e848e20c9ce1788e092e4a2fe205e8f44a0cb1224d896ff23a908cbe3c5
```

Die Überprüfung dient dazu, dass während der Dateiübertragung keine absichtlichen oder unabsichtlichen Änderungen an dem Image vorgenommen werden.

#### Überprüfung der Integrität

Mit dem Kommandozeilen-Befehl **sha512sum** wird die SHA512-Hashsumme ermittelt. Im ersten Schritt wird die Version des Kommandozeilen-Programms ermittelt und in **Bild 21** dargestellt. Dies geschieht mit dem folgenden Befehl: `sha512sum --version`

```
root@forensics:/home/user/master/image# sha512sum --version  
sha512sum (GNU coreutils) 8.30  
Copyright (C) 2018 Free Software Foundation, Inc.  
License GPLv3+: GNU GPL version 3 or later <https://gnu.org/licenses/gpl.html>.  
This is free software: you are free to change and redistribute it.  
There is NO WARRANTY, to the extent permitted by law.  
  
Written by Ulrich Drepper, Scott Miller, and David Madore.
```

Bild 21: Version Kommandozeilen-Programm sha512sum

Anschließend wird die Hashsumme, wie in **Bild 22** aufgezeigt, mit dem Befehl `sha512sum InfotainmentImage.tar.bz2` ermittelt.

```
root@forensics:/home/user/master/image# sha512sum InfotainmentImage.tar.bz2  
249b00e041ad6a89659bce9c40d3cb74a69a2e04e81d753550b54baba0ac6c5e91233e848e20c9  
ce1788e092e4a2fe205e8f44a0cb1224d896ff23a908cbe3c5 InfotainmentImage.tar.bz2
```

Bild 22: SHA512-Hashsumme des Infotainment-Images

## 4.2. ANALYSE PHASE

---

Die Hashsumme der Vorgabe und der eigenen Ermittlung sind exakt gleich. Damit kann bestätigt werden, dass das Infotainment-Image nach der Übertragung im Original vorliegt und keine Änderungen durchgeführt worden sind.

Das Original muss anschließend vervielfältigt werden, da das Original nicht verändert werden darf. Aus diesem Grund werden 1:1 Kopien erstellt, diese werden *Slaves* genannt. Auch diese werden mittels Hashsumme auf ihre Gleichheit geprüft.

```
root@forensics:/home/user/master/image# sha512sum InfotainmentImage_Slave01.tar.bz2  
249b00e041ad6a89659bce9c40d3cb74a69a2e04e81d753550b54bab0ac6c5e91233e848e20c9ce1788e09  
2e4a2fe205e8f44a0cb1224d896ff23a908cbe3c5  InfotainmentImage_Slave01.tar.bz2
```

Bild 23: SHA512-Hashsumme eines Slaves

Auch die im Bild 23 dargestellte Hashsumme des Slaves stimmt mit der Hashsumme des Originals überein. Damit kann die forensische Untersuchung des Infotainment-Images beginnen.

## 4.2 Analyse Phase

Zu Beginn der Analyse Phase wird die Datei des komprimierten Archives *InfotainmentImage\_Slave01.tar.bz2* in den Ordner /home/user/master/image entpackt. Dies wird mit dem Befehl tar -xf archive.tar.bz2 -C /home/user/master/image über die Kommandozeile erreicht.

Anschließend werden die extrahierten Dateien in *Infotainment.E0?* umbenannt.

### 4.2.1 Automatisierte Analyse

In dem Kapitel 3.4.1 wird schon darauf hingewiesen, dass möglicherweise keine aussagekräftige forensische Untersuchung mittels Autopsy durchgeführt werden kann. Zur Prüfung der Auswertungsmöglichkeiten mittels Autopsy werden, bis auf das *iOS Analyze* Modul, sämtliche Module aktiviert. Somit soll der gesamte Funktionsumfang von Autopsy genutzt werden, um größtmöglichen Output zu generieren. Es wird nach allen verfügbaren Artefakten gesucht.

## 4.2. ANALYSE PHASE

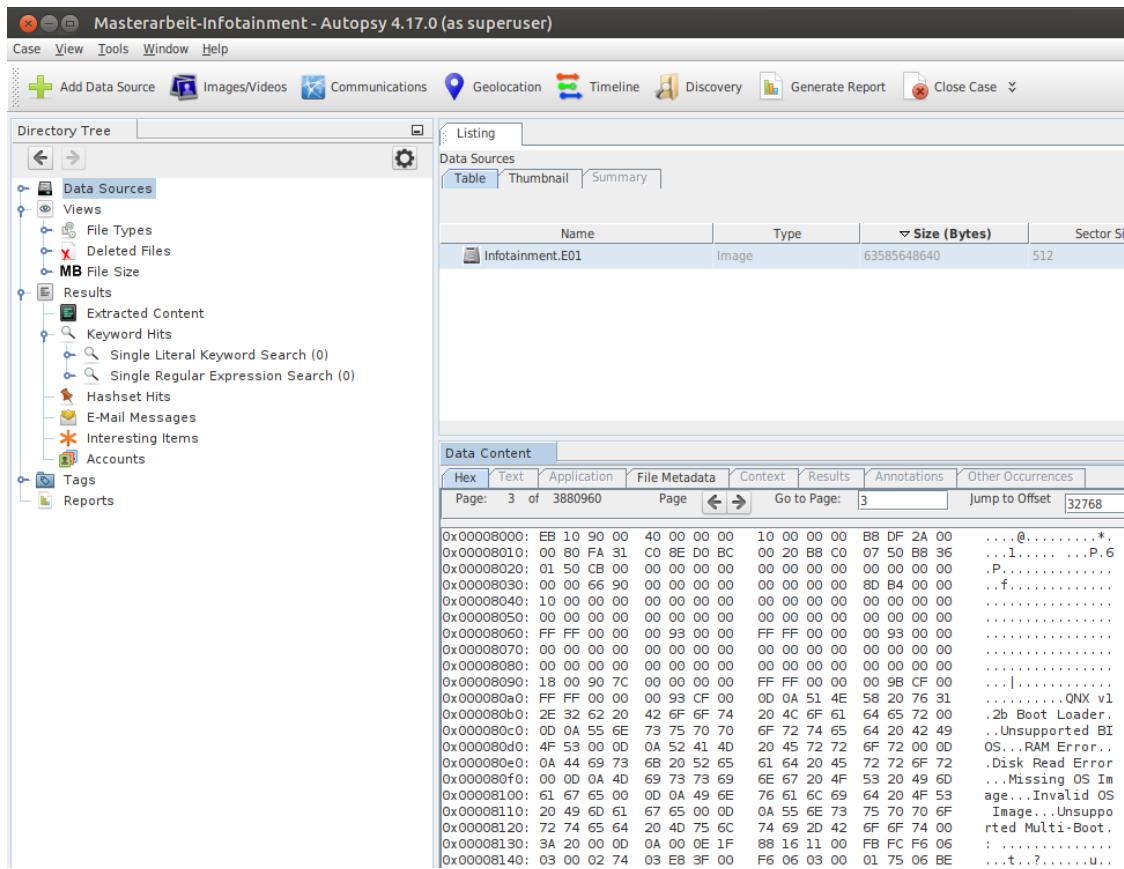


Bild 24: Bildausschnitt der forensische Untersuchung mittels Autopsy

**Bild 24** zeigt einen Ausschnitt der grafischen Oberfläche von Autopsy. Weil Autopsy weder das Betriebssystem QNX Neutrino RTOS noch deren Dateisystem-Formate unterstützt, kann wie erwartet keine sinnvolle Auswertung durchgeführt werden. Zu den verschiedenen Artefakten werden keine Daten gefunden. Die Fehlermeldung von Autopsy ist im integrierten Hex-Editor zu finden. Dieser befindet sich im Bereich des *Data Contents* in der unteren Hälfte auf der rechten Seite des Bildausschnitts.

Die Fehlermeldung lautet: „*QNX v1 2b Boot Loader. Unsupported BIOS. RAM Error. Disk Read Error. Missing OS Image. Invalid OS Image. Unsupported Multi-Boot.*

Auch die Fehlermeldung untermauert die vorher definierte These, die besagt, dass ohne eine offizielle Unterstützung von Autopsy auch kein sinnvoller Datengehalt generiert werden kann. Aus diesem Grund wird die forensische Untersuchung mittels Autopsy an dieser Stelle abgebrochen. Jedoch wird im späteren Verlauf im Kapitel 4.7 nochmals auf die Software Autopsy zurückgegriffen.

### 4.2.2 Manueller Analyse Prozess

Für die manuelle logische Auswertung wird das im Expert Witness Format vorliegende Infotainment-Image in das .dd-Format (Rohformat) konvertiert und in das Verzeichnis /ewf geschrieben.

**Bild 25** zeigt den dafür notwendigen xmount (Cross-Mount) Befehl. Eine Überprüfung der erfolgreichen Konvertierung ist ebenfalls dem **Bild 25** entnehmbar. Das Infotainment-Image beinhaltet insgesamt 14 Partitionen, von denen ausschließlich die jeweilige Größe bekannt ist.

```
root@forensics:/home/user/master/image# xmount --in ewf Infotainment.E0* --cache /tmp/info.ovl --out raw /ewf
root@forensics:/home/user/master/image# fdisk -lu /ewf/Infotainment.dd
Disk /ewf/Infotainment.dd: 59,22 GiB, 63585648640 bytes, 124190720 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x00000000

Device      Boot   Start     End   Sectors  Size Type
/ewf/Infotainment.dd1          64 2809855 2809792 1,3G b1 unknown
/ewf/Infotainment.dd2    2809856 114712575 111902720 53,4G 5 Extended
/ewf/Infotainment.dd3  114712576 124174335 9461760 4,5G b3 unknown
/ewf/Infotainment.dd5  2809920 4087807 1277888 624M b2 unknown
/ewf/Infotainment.dd6  4087872 9330687 5242816 2,5G b2 unknown
/ewf/Infotainment.dd7  9330752 17719295 8388544 4G b2 unknown
/ewf/Infotainment.dd8  17719360 18243583 524224 256M b2 unknown
/ewf/Infotainment.dd9  18243648 20340735 2097088 1024M b2 unknown
/ewf/Infotainment.dd10 20340800 22437887 2097088 1024M b2 unknown
/ewf/Infotainment.dd11 22437952 26632191 4194240 2G b2 unknown
/ewf/Infotainment.dd12 26632256 28991487 2359232 1,1G b2 unknown
/ewf/Infotainment.dd13 28991552 93741055 64749504 30,9G b2 unknown
/ewf/Infotainment.dd14 93741120 114712575 20971456 10G b2 unknown

Partition table entries are not in disk order.
```

Bild 25: Konvertierung des Infotainment-Images in Rohformat

Mit dem Befehl cat proc/filesystems kann überprüft werden, welche Dateisysteme automatisch erkannt werden. Die Ausgabe zeigt, dass keine spezifischen QNX Dateisysteme bekannt sind. Eine weitere Prüfung auf einer aktuellen Kali-Distribution zeigt ebenfalls keine QNX spezifischen Dateisysteme.

Weil für die 14 vorhandenen Partitionen kein Dateisystem erkannt wurde, ist kein gezieltes Einbinden der Partitionen möglich. Aus diesem Grund werden im nächsten Schritt alle 14 Partitionen in logische Devices eingebunden. Dies ist im **Bild 26** dargestellt. Das Einbinden der logischen Devices wird mit dem Befehl ls -la | /dev/loop4\* auf die Richtigkeit hin überprüft.

## 4.2. ANALYSE PHASE

---

```
root@forensics:/home/user/master/image# losetup --partscan --find --show /ewf/Infotainment.dd  
/dev/loop4  
root@forensics:/home/user/master/image# ls -la /dev/loop4*  
brw-rw---- 1 root disk 7, 4 Dez 8 20:14 /dev/loop4  
brw-rw---- 1 root disk 259, 0 Dez 8 20:14 /dev/loop4p1  
brw-rw---- 1 root disk 259, 8 Dez 8 20:14 /dev/loop4p10  
brw-rw---- 1 root disk 259, 9 Dez 8 20:14 /dev/loop4p11  
brw-rw---- 1 root disk 259, 10 Dez 8 20:14 /dev/loop4p12  
brw-rw---- 1 root disk 259, 11 Dez 8 20:14 /dev/loop4p13  
brw-rw---- 1 root disk 259, 12 Dez 8 20:14 /dev/loop4p14  
brw-rw---- 1 root disk 259, 1 Dez 8 20:14 /dev/loop4p2  
brw-rw---- 1 root disk 259, 2 Dez 8 20:14 /dev/loop4p3  
brw-rw---- 1 root disk 259, 3 Dez 8 20:14 /dev/loop4p5  
brw-rw---- 1 root disk 259, 4 Dez 8 20:14 /dev/loop4p6  
brw-rw---- 1 root disk 259, 5 Dez 8 20:14 /dev/loop4p7  
brw-rw---- 1 root disk 259, 6 Dez 8 20:14 /dev/loop4p8  
brw-rw---- 1 root disk 259, 7 Dez 8 20:14 /dev/loop4p9
```

Bild 26: Einbinden der Partitionen in ein logisches Device

Da sehr viele Hersteller der Infotainment-Systeme auf das Betriebssystem QNX Neutrino RTOS setzen, wird das Infotainment-Image auf Dateisysteme des QNX Neutrino RTOS Betriebssystems hin untersucht. Dafür wird das Programm *Binwalk* in der Version 2.2.0 eingesetzt. Speziell wird nach QNX Boot-Sektoren gesucht, die Hinweise auf die Typen der Dateisysteme geben sollen. Weiterhin ist bekannt, dass QNX4 und QNX6 Dateisysteme existieren, also wird das Image nacheinander auf beide Dateisysteme hin untersucht.

Die erste Untersuchung mit *Binwalk* auf QNX6 Boot-Sektoren ergibt keine Ergebnisse. Aufgrund dessen wird folgend nach QNX4-Boot Sektoren gesucht.

```
root@forensics:/home/user/master/image# binwalk /ewf/Infotainment.dd | egrep -i 'QNX4 Boot'  
32768      0x8000      QNX4 Boot Block  
2549469367 0x97F5D0B7  QNX4 Boot Block
```

Bild 27: Untersuchung des Rohimages mittels Binwalk

Das **Bild 27** zeigt das Ergebnis der Analyse mit dem Programm *Binwalk*. *Binwalk* konnte zwei QNX4 Partitionen in den Stellen 32768 und 2549469367 auffinden.

Im **Bild 25** ist die Sektorgröße mit 512 bytes pro Sektor angegeben. Diese Information wird genutzt, um mögliche Kandidaten für Partitionen mit dem Dateisystem QNX4 zu berechnen. Für die Berechnung des Startsektors werden die durch *Binwalk* aufgefundenen Stellen durch die Anzahl der Bytes pro Sektor geteilt.

Stelle 32768 / 512 Bytes = 64 Sektor

Stelle 2549469367 / 512 Bytes = 4979432,36 Sektor

Ein Vergleich mit dem **Bild 25** zeigt, dass die erste Partition an der Stelle 64 beginnt. Somit kann davon ausgegangen werden, dass die erste Partition das Dateisystem QNX4 besitzt. Der zweite Wert kann keiner Partition zugeordnet werden, die anderen Partitionen können ebenfalls noch nicht zugeordnet werden.

## 4.2. ANALYSE PHASE

---

Da dieses Ergebnis nicht sehr zufriedenstellend ist, wird eine ältere Version der Software *Binwalk* installiert und die Untersuchung des Rohimages wiederholt. Oftmals liefern die verschiedenen Versionen andere Ergebnisse, da die Software ständig weiterentwickelt wird. Weil der Fokus von den Entwicklern von *Binwalk* nicht auf dem QNX Neutrino RTOS liegt, können auch durchaus mit älteren Versionen mehr Ergebnisse erzeugt werden.

Als Erstes wird das Release mit der Version 2.0.0 heruntergeladen und installiert. Mit dieser Version werden keine QNX-Attribute gefunden. Für den nächsten Versuch wird das Release mit der Version 2.1.1 auf der Github-Seite heruntergeladen und installiert. Jedoch liefert auch diese Version keine weiteren Ergebnisse. Aus diesem Grund wird auf Basis der ersten Ergebnisse von *Binwalk* in der Version 2.2.0 weiter gearbeitet.

Für das Mounten der aufgefundenen QNX4-Partitionen wird ein Verzeichnis *infotainment* im root-Verzeichnis angelegt. Weiterhin wird in dem *infotainment*-Verzeichnis für jede Partition ein eigenes Verzeichnis angelegt.

```
root@forensics:/home/user/master/image# mount -t qnx4 /dev/loop4p1 /infotainment/1
mount: /infotainment/1: wrong fs type, bad option, bad superblock on /dev/loop4p1,
      missing codepage or helper program, or other error.
```

Bild 28: Fehlversuch beim Einhängen als QNX4 Partitionen

**Bild 28** zeigt den Versuch die erste Partition mit dem Parameter -t qnx4 einzuhängen. Dabei gibt der Parameter -t qnx4 das Dateisystem an, das eingehängt werden soll. Jedoch schlägt das Einhängen fehl. Ein Versuch die Partition mit dem vorgegebenen Dateisystem QNX6 einzuhängen funktioniert. Der vollständige Befehl lautet: mount -t qnx6 /dev/loop4p1 /infotainment/1 Das gleiche Vorgehen wird für die zweite Partition probiert und schlägt fehl. Die Fehlermeldung ist im **Bild 29** dargestellt.

```
root@forensics:/home/user/master/image# mount -t qnx6 /dev/loop4p2 /infotainment/2
mount: /infotainment/2: wrong fs type, bad option, bad superblock on /dev/loop4p2,
      missing codepage or helper program, or other error.
```

Bild 29: Erfolgreiches Einhängen als QNX6 Partitionen

Durch Probieren aller 14 Partitionen wird festgestellt, dass sich die Partitionen 1, 10 und 12 als QNX6-Partitionen einhängen lassen. Die übrigen Partitions-Verzeichnisse im *infotainment*-Verzeichnis werden wieder entfernt.

Eingehängt werden die drei QNX6 Partitionen mit den folgenden Befehlen:

```
mount -t qnx6 /dev/loop4p1 /infotainment/1
mount -t qnx6 /dev/loop4p10 /infotainment/10
mount -t qnx6 /dev/loop4p12 /infotainment/12
```

## 4.2. ANALYSE PHASE

---

Nach dem sämtliche verfügbaren Partitionen eingehängt sind, kann die Datenextraktion und -auswertung beginnen. Jedoch wird vor der eigentlichen Untersuchung der einzelnen Partitionen und dessen Daten eine weitere Analyse des Images mit *Binwalk* durchgeführt. Dadurch soll ein erster Eindruck über das vorliegende System gewonnen und mögliche wichtige Hinweise für die folgende Analyse aufgedeckt werden.

Für eine allgemeine Analyse des Images wird der folgende Befehl abgesetzt:  
binwalk /ewf/Infotainment.dd.

Diese Analyse ohne gezielte Suchparameter zeigt, dass es sich bei dem System um eine 32 Bit ARM Architektur handelt. Darüber hinaus können über die Ausgaben der UNIX-Pfadangaben einige Verzeichnisse, die einer handelsüblichen Linux-Distribution gleichen, aufgefunden werden. Bekannt sind das Root-Verzeichnis / und die darin befindlichen Verzeichnisse *opt*, *usr*, *dev*, *etc*, *var* und *home*.

Über das /home-Verzeichnis können mindestens fünf verschiedene Benutzer auf dem System identifiziert werden. Dies sind die Benutzer *user*, *builder*, *jenkins*, *tpadlikar* und *nbfw01*.

Weiterhin weisen einige Datei- und Verzeichnisnamen darauf hin, dass das verwendete Betriebssystem ein QNX Neutrino RTOS 6.5.0 ist. Außerdem weisen einige Dateinamen darauf hin, dass dieses Infotainmentsystem in einem PKW des Herstellers Skoda verbaut ist.

Im nächsten Schritt werden die einzelnen Partitionen und die darauf befindlichen Daten extrahiert und analysiert.

### Datenextraktion

Im ersten Schritt werden die drei Partitionen nach *db-Dateien* durchsucht, da dort die meisten personenbezogenen oder personenbeziehbaren Daten liegen.

**Bild 30** zeigt die mit dem *file* Befehl in der Version 5.38 aufgefundenen SQLite Datenbanken.

```
root@forensics:/infotainment# find . -type f -iname "*.db"
./12/predictiveNav/PNav1.db
./10/addressbook.db
./10/picturestore.db
./10/truffles.0.db
./1/eso/strokemw.db
```

Bild 30: Gefundene SQLite Datenbanken

## 4.2. ANALYSE PHASE

---

Die aufgefundenen Datenbanken werden mit dem `cp` Befehl in das Verzeichnis `/home/user/master/results` kopiert und im späteren Verlauf analysiert.

Im Anschluss werden jetzt die einzelnen Partitionen detailliert untersucht. Im **Bild 31** ist ein Überblick über die erste Partition dargestellt.

```
root@forensics:/infotainment# ls -la /infotainment/1
total 56
drwxrwxr-x 1 root root 1024 Aug  2 2018 .
drwxr-xr-x 5 root root 4096 Dez  9 15:39 ..
drwxrwxrwx 1 root root 1024 Jun 25 2018 armic
drwx----- 1 root root 1024 Aug  2 2018 .boot
-rwxrwxrwx 1 root root 644 Jun 25 2018 build_parameters.txt
-rwxrwxrwx 1 root root 397 Jun 25 2018 detailed_changelists_info.txt
drwxrwxrwx 1 root root 1024 Jun 25 2018 eso
-rwxrwxrwx 1 root root 8943 Jun 25 2018 EsoTraces.esd
-rwxrwxrwx 1 root root 11283 Jun 25 2018 EsoTracesSec.esd
lrwxrwxrwx 1 root root 22 Aug  2 2018 gemib -> /mnt/app/gemib.factory
drwxrwxrwx 1 root root 1024 Jun 25 2018 gemib.factory
drwxrwxrwx 1 root root 1024 Jun 25 2018 hmi
drwxrwxrwx 1 root root 1024 Jun 25 2018 img_restore
-rwxrwxrwx 1 root root 34 Jun 25 2018 img_ver.txt
drwxrwxrwx 1 root root 5120 Jun 25 2018 navigation
-rwxrwxrwx 1 root root 9 Jun 25 2018 p4changelist.txt
-rwxrwxrwx 1 root root 8825 Jun 25 2018 p4client.txt
drwxrwxrwx 1 root root 1024 Jun 25 2018 root
drwxrwxrwx 1 root root 1024 Jun 25 2018 speech
lrwxrwxrwx 1 root root 27 Aug  2 2018 streetview -> /mnt/app/streetview.factory
drwxrwxrwx 1 root root 1024 Jun 25 2018 streetview.factory
-rwxrwxrwx 1 root root 750 Jun 25 2018 target.properties
drwxrwxrwx 1 root root 1024 Jun 25 2018 var
-rwxrwxrwx 1 root root 1587 Jun 25 2018 version_info.txt
```

Bild 31: Vorschau der Dateien und Verzeichnisse auf der Partition 1

Innerhalb der ersten Partition werden ein Verzeichnis *img\_restore* und ein Verzeichnis mit dem Namen *FwImage* sichergestellt. In dem Verzeichnis *FwImage* befinden sich zwei Binary-Dateien, die der Namensgebung nach zu urteilen für die SD-Karte und das W-Lan Modul zuständig sind. Das Verzeichnis *img\_restore* hingegen enthält eine Image-Datei mit dem Namen *persist.img* und ein *main\_stage2.ifs.lzo* Archiv. Hierbei handelt es sich um ein komprimiertes Dateiarchiv, das auf der Datenkomprimierungsbibliothek LZO (Lempel-Ziv-Oberhume) basiert. Das *persist.img* und das Dateiarchiv *main\_stage2.ifs.lzo* werden in dem Kapitel 4.5 ausführlich behandelt.

Darüber hinaus sind in dem Verzeichnis *hmi* sämtliche Dateien, Skripte und Programme hinterlegt, um das Infotainmentsystem und dessen Komponenten hoch und herunter zu fahren. Zusätzlich werden noch einige Treiber und sehr viele Konfigurationsdateien für sämtliche Dienste und Programme ermittelt. Für das dynamische Logging der Programmausführungen sind für sämtliche Module Tracing-Programme hinterlegt.

Mit dem Befehl `find . -type f -iname "*.cfg" | wc -l` konnten insgesamt 341 Konfigurationsdateien für den Browser, die Telefonfunktion, die Navigation, die Sprach-

## 4.2. ANALYSE PHASE

---

ausgabe und für Google Streetview aufgefunden werden. Weiterhin beinhaltet die Partition einige Standard-Klingeltöne im .wav-Format.

Die restlichen Dateien sind Text-Dateien und kompilierte Bibliotheken im .so-Format (Shared Object). Diese Bibliotheken sind vergleichbar mit den .DLL-Dateien in der Windows-Welt.

Eine der Text-Dateien besitzt einen interessanten Namen *version\_info.txt* und wird für die im späteren Verlauf folgende Analyse sichergestellt.

Der Inhalt der zweiten Partition (Partition 10) ist im **Bild 32** abgebildet.

```
root@forensics:/infotainment# ls -la /infotainment/10
total 2302
drwxrwxr-x 1 root root    4096 Jan  1 1970 .
drwxr-xr-x 5 root root    4096 Dez  9 15:39 ..
-rw-r--r-- 1 root root   91136 Jan  1 1970 addressbook.db
-rw-r--r-- 1 root root   32768 Jan  1 1970 addressbook.db-shm
-rw-r--r-- 1 root root 1048032 Jan  1 1970 addressbook.db-wal
drwx----- 1 root root   4096 Aug  2 2018 .boot
drwx----- 1 root root   4096 Jan  1 1970 pics
-rw-r--r-- 1 root root   24576 Jan  1 1970 picturestore.db
-rw-r--r-- 1 root root   32768 Jan  1 1970 picturestore.db-shm
-rw-r--r-- 1 root root 1048032 Jan  1 1970 picturestore.db-wal
-rw-r--r-- 1 root root    1024 Jan  1 1970 truffles.0.db
-rw-r--r-- 1 root root   32768 Jan  1 1970 truffles.0.db-shm
-rw-r--r-- 1 root root  29376 Jan  1 1970 truffles.0.db-wal
```

Bild 32: Vorschau der Dateien und Verzeichnisse auf der Partition 10

Diese Partition beinhaltet die drei von den vier sichergestellten Datenbanken. Die restlichen Verzeichnisse innerhalb dieser Partition beinhalten keine Dateien. Mit Hilfe des *DB Browser for SQLite* werden die sichergestellten Datenbanken dieser Partitionen nachfolgend analysiert. Der installierte *DB Browser for SQLite* liegt in der Version 3.11.2 vor.

Die erste zu untersuchende Datenbank ist die *addressbook.db*. Im **Bild 33** ist der grundlegende Aufbau der Datenbank dargestellt. Dem Bild ist zu entnehmen, dass die Datenbank insgesamt aus sechs Relationen besteht. Darüber hinaus sind Indices und Schlüssel zur Verknüpfung der Relationen untereinander gesetzt.

## 4.2. ANALYSE PHASE

---

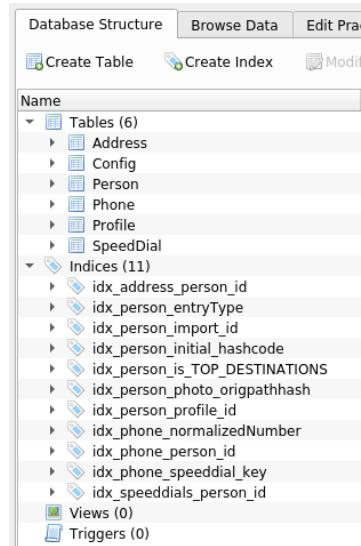


Bild 33: Datenbank und Datenstruktur der *Adressbuch* Datenbank

In der Relation *Profile* wird der Gerätename und eine Geräte-ID des Telefons gespeichert. **Bild 34** zeigt den Datenbankeintrag des Telefons. Diesem kann entnommen werden, dass ein iPhone mit dem Infotainmentsystem verbunden war. Weiterhin hat der iPhone-Besitzer sich mit dem Namen *Anton* auf dem Telefon registriert.

Table: <b>Profile</b>					
<a href="#">id</a>	<a href="#">name</a>	<a href="#">device_identifier</a>	<a href="#">hashcode_SIM</a>	<a href="#">hashcode_ME</a>	<a href="#">_person_home</a>
1 0	PUBLIC	NULL	NULL	NULL	0
2 1	iPhone von Anton	70:70:0D:96:25:DC	NULL	NULL	0

Bild 34: Profile Relation der *Adressbuch* Datenbank

Aus der Struktur der zweiten Datenbank ist zu erkennen, dass diese aus insgesamt elf Relationen besteht. Darüber hinaus wurden Indices angelegt, die Relationen selbst beinhaltet keine Tupel. Somit ist diese Datenbank vollständig angelegt, jedoch nicht mit Inhalten gefüllt. Die Übersicht über die Struktur der Datenbank ist im **Bild 35** dargestellt.

## 4.2. ANALYSE PHASE

---

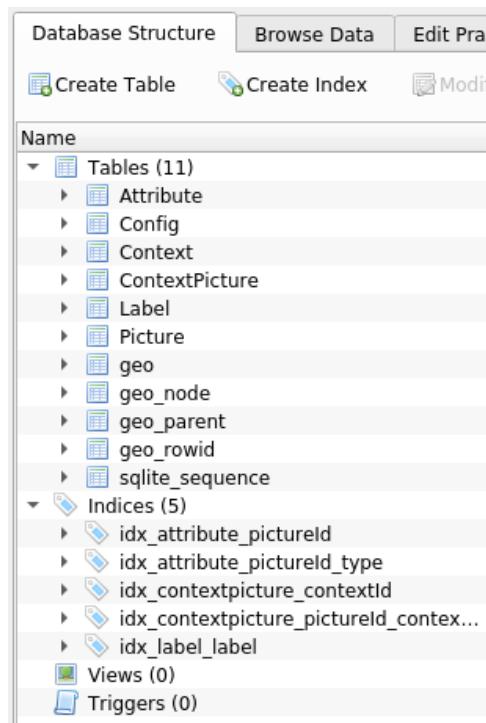


Bild 35: Datenbank und Datenstruktur der *Bilderspeicher* Datenbank

Die dritte und letzte Datenbank dieser Partition ist vollständig leer. Es sind keine Relationen angelegt und damit auch keine Inhalte vorhanden. Einen Ausschnitt der Ansicht in dem *DB Browser for SQLite* wird im **Bild 36** visualisiert.

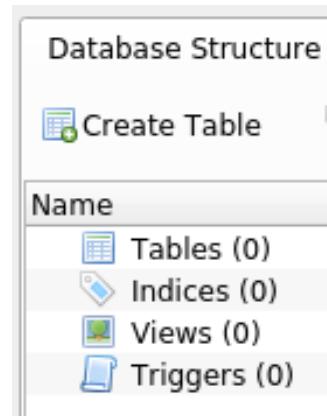


Bild 36: Datenbank und Datenstruktur der *Truffles* Datenbank

Eine detaillierte Untersuchung der dritten Partition zeigt, dass auf dieser, abgesehen von der schon sichergestellten Navi-Datenbank, nur einige Standard-Bilder und Icons für die Navigations-Software vorhanden sind. Einen Überblick über die Dateien und Verzeichnisse auf der Partition 12 gibt das **Bild 37**.

## 4.2. ANALYSE PHASE

---

```
root@forensics:/infotainment# ls -la /infotainment/12
total 12
drwxrwxr-x 1 root root 1024 Jan  1 1970 .
drwxr-xr-x 5 root root 4096 Dez  9 15:39 ..
drwx----- 1 root root 1024 Aug  2 2018 .boot
drwx----- 1 root root 1024 Jan  1 1970 cache
drwxrwxrwx 1 root root 1024 Jul 30 2018 Parrot
drwx----- 1 root root 1024 Jan  1 1970 predictiveNav
drw-rw-rw- 1 root root 1024 Jan  1 1970 rcc
drwxrwxrwx 1 root root 1024 Jul 30 2018 rhmi
drwx----- 1 root root 1024 Jan  1 1970 userdata
```

Bild 37: Vorschau der Dateien und Verzeichnisse auf der Partition 12

Folglich wird die SQLite Datenbank des Navigationssystems untersucht. Ein Blick auf den strukturellen Aufbau der Datenbank zeigt, dass diese aus fünf Relationen besteht. **Bild 38** zeigt die Struktur der SQLite-Datenbank.

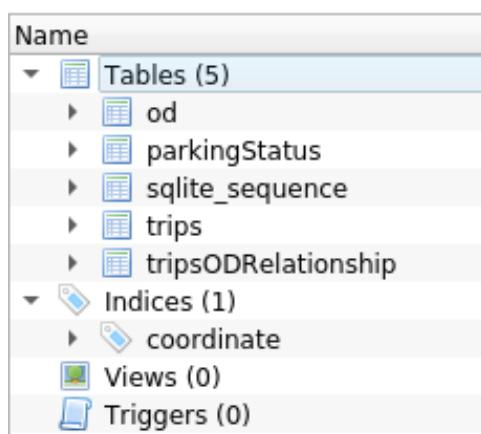


Bild 38: Datenbank und Datenstruktur der *Navigations* Datenbank

Das folgende **Bild 39** stellt einen Ausschnitt aus der Relation *Trips* dar. Anscheinend zeichnet das Navigationssystem verschiedene Punkte eines Trips auf. Dazugehörig werden die Werte der Latitude, Longitude, dem zugehörigen UNIX-Zeitstempel erfasst und eine einmalige Trip-Id festgehalten.

## 4.2. ANALYSE PHASE

---

Table: trips

	id	latitude	longitude	unixtime	seqNo	tripId	gxHash	gyHash	forceSave	isStreetResolved
1	1	52.538869	13.611147	1557403848	0	1	4323	2642	0	0
2	2	52.538404	13.609855	1557403888	1	1	4323	2642	1	1
3	3	52.537632	13.609742	1557403901	2	1	4323	2642	1	1
4	4	52.537771	13.608694	1557403940	3	1	4323	2642	1	1
5	5	52.539553	13.608508	1557403962	4	1	4323	2642	1	1
6	6	52.540892	13.619361	1557403999	5	1	4323	2642	1	1
7	7	52.536596	13.620718	1557404048	6	1	4323	2642	1	1
8	8	52.533339	13.619675	1557404071	7	1	4323	2642	1	1
9	9	52.531537	13.617958	1557404135	8	1	4323	2642	1	1
10	10	52.531423	13.615147	1557404168	9	1	4324	2642	1	1

Bild 39: Ansicht der *Trips* Relation im *DB Browser for SQLite*

In der folgenden **Tabelle 5** wird eine Teilübersicht über die untersuchten Partitionen dargestellt. Dies dient einer besseren Struktur und Übersicht über das Partitions-Layout des Infotainmentsystems.

Benennung	Einsatz	Beschreibung
Partition 1	Systempartition	Betriebssystem, Images, Systemprogramme, Dienste, Bibliotheken und Konfigurationsdateien
Partition 10	Infotainment	Adressbuch, Bilder und Truffles-Datenbank <sup>1</sup>
Partition 12	Navigationssystem	Datenbank für Navigationssystem (Standorte, gefahrene Strecken etc.)

Tabelle 5: Analyseergebnis der Partitionen und dessen Einsatzzweck mit Linux-Boardmittel

Eine weitere Aufbereitung der Daten ist in dem Kapitel 4.5 dargestellt.

Allgemein zeigt die Analyse der vorhandenen Partitionen, dass die meisten Verzeichnisse auf den Partitionen leer sind. Explizit zu erwähnen ist, dass die Verzeichnisse mit der Bezeichnung *userDaten* ohne Inhalt sind. Denn diese werden in der Regel dafür verwendet personenbezogene oder personenbeziehbare Daten abzulegen.

Auch versteckte Dateien sind nicht vorhanden, diese würden mit dem Befehl `ls -la` angezeigt werden. Denn der zusätzliche Parameter `-a` bewirkt, dass auch versteckte Dateien angezeigt werden.

<sup>1</sup>Truffles ist eine Entwicklungsumgebung, ein Test-Framework und eine Asset-Pipeline für Blockchains unter Verwendung der Ethereum Virtual Machine (EVM) [58].

### Nicht-allozierter Bereich

In diesem Abschnitt wird der nicht-allozierte (unallocated) Bereich des Infotainment-Images untersucht. Für die Untersuchung des nicht-allozierten Bereiches wird das Kommandozeilen-Programm *bulk extractor* verwendet.

Dafür muss das Infotainment-Image wieder in das .dd-Format konvertiert werden. Dies wird wieder mit dem Befehl `xmount --in ewf Infotainment.E0* --cache /tmp/info.ovl --out raw /ewf` gemacht.

Anschließend werden die loop-Devices wieder mit dem Befehl `losetup --partscan --find --show /ewf/Infotainment.dd` automatisch erstellt.

Mit dem Befehl `bulk_extractor /dev/loop2 -o /home/user/master/results/outputBulkExtractor/` wird die Untersuchung mittels *bulk\_extractor* gestartet. Wichtig dabei ist die Angabe des Output-Verzeichnisses über den Parameter *-o*.

Das folgende Bild 40 zeigt den erfolgreichen Durchlauf des Programms. Darüber hinaus werden die Durchlaufzeit, das verarbeitete Datenvolumen, die Anzahl der gefundenen E-Mail Merkmale und Weiteres dargestellt.

```
16:51:02 Offset 68685MB (99.95%) Done in 0:00:00 at 16:51:02
All data are read; waiting for threads to finish...
Time elapsed waiting for 8 threads to finish:
    (timeout in 60 min.)
All Threads Finished!
Producer time spent waiting: 594.581 sec.
Average consumer time spent waiting: 103.8 sec.
*****
** bulk_extractor is probably CPU bound. **
** Run on a computer with more cores **
** to get better performance.      **
*****
MD5 of Disk Image: c797acea8ffa4a3084598673f3840727
Phase 2. Shutting down scanners
Phase 3. Creating Histograms
Elapsed time: 1030.87 sec.
Total MB processed: 68719
Overall performance: 66.6615 MBytes/sec (8.33269 MBytes/sec/thread)
Total email features found: 85727
```

Bild 40: Screenshot des erfolgreichen Durchlaufs von *bulk\_extractor*

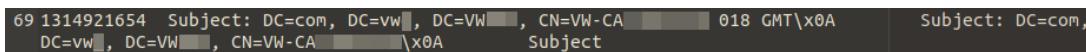
Der *bulk\_extractor* erstellt zu jedem Merkmal eine Text-Datei mit den Funden, sowie ein dazugehöriges Histogramm in Form einer weiteren Text-Datei. Bei den aufgefundenen Domains ist auffällig, dass verschiedene Domains von Google Maps, wie auch Verbindungen zu den Webseiten [www.volkswagen.de](http://www.volkswagen.de) und [www.volkswagen.com](http://www.volkswagen.com) nachvollzogen werden können. Die extrahierten MAC-Adressen, GPS-Daten, Exif-Daten, HTTP Logs

## 4.2. ANALYSE PHASE

---

und IP-Adressen zeigen keine verwertbaren Spuren oder sind vollständig leer. Innerhalb der extrahierten E-Mail-Adressen sind hauptsächlich die E-Mail-Adressen von Software-Herstellern oder Linux-Entwicklern aufgeführt. Jedoch sind auch die E-Mail-Adressen von zwei internen und einem externen Mitarbeiter der Volkswagen AG extrahiert worden.

Die Analyse der E-Mail Header weist zusätzlich darauf hin, dass Kontakt zu der Volkswagen AG bestand. Im **Bild 41** ist der extrahierte E-Mail-Header dargestellt.



69 1314921654 Subject: DC=com, DC=vw, DC=VW, CN=VW-CA 018 GMT\x0A Subject: DC=com, DC=vw, DC=VW, CN=VW-CA \x0A Subject: DC=com, DC=vw, DC=VW, CN=VW-CA \x0A Subject:

Bild 41: Auszug aus den extrahierten E-Mail Headern nach RFC822 Standard

Der veranschaulichte E-Mail-Header zeigt ein SSL-Zertifikat, das extrahiert werden konnte. In der Regel werden diese Zertifikate für verschlüsselte Kommunikation über das LDAPS<sup>2</sup> Protokoll bzw. zur Smart Card Authentifizierung verwendet. LDAPS wird hauptsächlich für Dienste von Drittanbietern oder nicht domänen-verbundenen Systemen verwendet, die einen sicheren Weg zur Abfrage des Domänencontrollers benötigen. Oftmals wird die verschlüsselte Kommunikation auch vom Domänencontroller oder von dem Active Directory gefordert. Mit LDAPS können diese Systeme von einer verschlüsselten Kommunikation profitieren, auch wenn sie nicht direkt mit der Domäne verbunden sind [59].

Das Kürzel *CN* steht für *Common Name* und stellt den Eintrag des Servernamens im Betreff des SSL-Zertifikates dar. Auch die Angaben des *Domain Controllers (DV)* sprechen für eine Verbindung mit Volkswagen-Servern.

Weiterhin konnten einige deutsche Mobil- und Festnetz Telefonnummern extrahiert werden. Ein Abgleich dieser Telefonnummern mit den Telefonnummern aus der extrahierten Adressbuch-Datenbank zeigt, dass diese übereinstimmen. Das lässt darauf schließen, dass sich im nicht-allozierten Bereich des Infotainment-Images eine Version der Adressbuch-Datenbank befinden muss.

Die Datei *sqlite\_carved.txt* zeigt, dass einige SQLite Datenbanken aufgefunden und extrahiert wurden. Zuletzt werden die extrahierten SQLite Datenbanken untersucht, diese befinden sich in einem von dem Programm *bulk\_extractor* angelegtem Verzeichnis *sqlite\_carved/000*. Es konnten insgesamt 95 SQLite Datenbanken aus dem nicht-allozierten Bereich extrahiert werden.

Die Analyse der extrahierten SQLite Datenbanken zeigt, dass die meisten SQLite Datenbanken Konfigurationen für das Navigations- und Multimediasystem beinhalten. Weitere beinhalten die Länder- und Städtenamen Europas in der Landessprache, sowie auch

---

<sup>2</sup>Secure Lightweight Directory Access Protocol (LDAPS)

## 4.2. ANALYSE PHASE

---

im Deutschen. **Bild 42** präsentiert einen Ausschnitt aus der SQLite Datenbank für die Städtenamen Ungarns.

	<a href="#">id</a>	<a href="#">word</a>	<a href="#">original_word</a>	<a href="#">datasets</a>	<a href="#">type</a>	<a href="#">first_char</a>	<a href="#">nitespace_cou</a>	<a href="#">conflict_lang</a>	<a href="#">lang</a>
1	1	budapest	Budapest		2	b	0	NULL	*
2	2	budapest	Budapest	BLOB	0	b	0	NULL	*
3	3	bp	Bp	BLOB	0	b	0	NULL	hu_HU
4	4	debrecen	Debrecen	BLOB	0	d	0	NULL	*
5	5	miskolc	Miskolc	BLOB	0	m	0	NULL	*
6	6	szeged	Szeged	BLOB	0	s	0	NULL	*
7	7	pecs	Pécs	BLOB	0	p	0	NULL	*
8	8	gyor	Györ	BLOB	0	g	0	NULL	*
9	9	nyiregyhaza	Nyíregyháza	BLOB	0	n	0	NULL	*
10	10	kecskemet	Kecskemét	BLOB	0	k	0	NULL	*
11	11	szekesfehervar	Székesfehérvár	BLOB	0	s	0	NULL	*

Bild 42: Auszug aus Städtenamen Datenbank für das Navigationssystem

Darüber hinaus ist eine exakte Kopie der Navi-Datenbank *PNav1.db* sowie des Adressbuches *addressbook.db* beschrieben in Abschnitt 4.2.2 extrahiert worden.

Zusätzlich zu den bisher bekannten SQLite-Datenbanken ist eine Multimedia Datenbank extrahiert worden die Musik-Titel, -Alben, -Interpreten, -Genres und weitere Attribute umfasst. Ein Ausschnitt dieser ist im **Bild 43** dargestellt.

	<a href="#">genre_id</a>	<a href="#">genre</a>	<a href="#">cover_id</a>
1	1	filterCriteria.unknownGenre	112
2	2	Pop	104
3	3	Alternative Musik	3
4	4	Dance	140
5	5	Electronic, Pop/Rock/Stage & Screen/Musicfire.in	9
6	6	Disco	19
7	7	Rock	119
8	8	House	116
9	9	Other	136
10	10	www.djwitek.org	0
11	11	Blues	110
12	12	Electronica & Dance	46
13	13	Unbekannt	0
14	14	Hip-Hop	81
15	15	Electronica	86
16	16	Soundtrack	103
17	17	www.djwitek.prv.pl	0
18	18	Top40	0
19	19	pop	132
20	20	New Age	141

Bild 43: Auszug aus der Multimedia Sqlite Datenbank

## 4.2. ANALYSE PHASE

---

Dies sind weitere personenbeziehbare Daten und Spuren die sichergestellt werden. Weitere 17 SQLite Datenbanken konnten nicht untersucht werden, da diese beschädigt sind. Die beschädigten SQLite Datenbanken werden in dem nächsten Abschnitt versucht zu rekonstruieren und zu analysieren.

### SQLite Datenbank-Rekonstruktion

Für die Rekonstruktion der beschädigten SQLite Datenbanken wird die Software *bring2lite* eingesetzt. Im ersten Schritt wird für die Installation das aktuelle *master* Repository (Commit: e876bf2 letztes Update August 2019) auf Github geklont und die Software anschließend installiert. Die Installation wird mit Warnungen und Fehlern durchgeführt.

Damit die volle Funktionsfähigkeit des Programmes garantiert werden kann, sind einige Anpassungen im Programmcode notwendig. Nach der Behebung der Fehler läuft die Installation fehlerlos ab.

Eine Anleitung zur Bedienung der Software ist ebenfalls auf Github verfügbar. Mithilfe dieser werden im Folgenden möglichst viele der 17 beschädigten SQLite3 Datenbanken rekonstruiert.

Mit dem Befehl `python3 main.py --filename /path/to/database/file --out /path/to/output/folder` wird die *.db-Datei* der Sqlite Datenbank rekonstruiert.

Für die Rekonstruktion der WAL-Datei (Write-Ahead Log) wird der folgende Befehl verwendet:

```
python3 main.py --wal /path/to/wal/file --out /path/to/output/folder
```

Unter der Verwendung dieser beiden Kommandozeilen-Befehle werden die beschädigten Datenbanken, soweit möglich, rekonstruiert und analysiert.

Insgesamt können vier der 17 beschädigten SQLite-Datenbanken auf diese Art und Weise wiederhergestellt werden. Die WAL-Dateien, die aus dem nicht allozierten Bereich des Datenträgers extrahiert wurden, sind alle komplett leer. Dadurch ist eine Rekonstruktion der Write-Ahead Logs nicht möglich.

Für die rekonstruierten Datenbanken werden von der Software jeweils zwei Verzeichnisse für die rekonstruierten Dateien angelegt. In dem Verzeichnis „regular-page-parsing“ werden die einzelnen wiederhergestellten B-Seiten der Datenbank mit der Bezeichnung *X-page.log* abgespeichert. Das X steht dabei für die jeweilige Seitennummer. In dem zweiten Verzeichnis *schemas* wird das vorgefundene Datenbankschema abgelegt.

Die **Bilder** 44a und 44b zeigen die zwei möglichen Ergebnisse bei der Wiederherstellung der B-Seiten.

## 4.2. ANALYSE PHASE

---

```
TEXT,TEXT,  
NULL,-  
3,b'\x1e\xb9\xf1\x0e\x84\x0b\x00  
NULL,-  
1,b'\x19sp\x04\x90\xcd*\x00#\x00  
NULL,1,b'/-  
\x045\x04\x90\xcd*\x00\x0e\x00\x  
NULL,-  
1,b'\xaeyp\x00\xc1\x0b\x00\x00\x  
NULL,-  
1,b'\xeay\x8a\xd1\x00\x06\x9a\x02  
NULL,-  
2,b'uP\xa5\x00\xd7\x91N\x00\x12\x  
NULL,-  
1,b'fH\x0b\x04\x8dX\x04\x00\x0e\x  
NULL,-  
1,b'm\xda\xdc\x03A\x04~\x00\x14\x  
++++++
```

(a) Vollständig

```
TEXT,TEXT,  
NULL,-  
3,b'\x1e\xb9\xf1\x0e\x84\x0b\x00  
NULL,-  
1,b'\x19sp\x04\x90\xcd*\x00#\x00  
NULL,1,b'/-  
\x045\x04\x90\xcd*\x00\x0e\x00\x  
NULL,-  
1,b'\xaeyp\x00\xc1\x0b\x00\x00\x  
NULL,-  
1,b'\xeay\x8a\xd1\x00\x06\x9a\x02  
NULL,-  
2,b'uP\xa5\x00\xd7\x91N\x00\x12\x  
NULL,-  
1,b'fH\x0b\x04\x8dX\x04\x00\x0e\x  
NULL,-  
1,b'm\xda\xdc\x03A\x04~\x00\x14\x  
++++++
```

(b) Unvollständig mit Bytecode

Bild 44: Mögliche Wiederherstellungsresultate

Der linken Abbildung sind die einzelnen wiederhergestellten Tupel mit vollständig wiederhergestellten und lesbaren Attributwerten entnehmbar. Diese können ohne weitere Bearbeitung ausgewertet und analysiert werden.

Drei der vier wiederhergestellten Datenbanken beinhalten Daten für das Navigationssystem. In diesen sind die verschiedenen Städte der Länder Belarus, Kasachstan und Spanien mit den zugehörigen Angaben der Latitude und Longitude gespeichert. Die vierte wiederhergestellte Datenbank beinhaltet diverse Radiosender für alle europäischen Länder.

Die rechte Abbildung zeigt den Fall, dass die Attributwerte der Tupel nicht korrekt ge-parst und wiederhergestellt werden konnten. Dieses Phänomen ist nur bei Attributen des Typs „TEXT“ beobachtbar. Anstatt des lesbaren Textes aus Buchstaben wird lediglich der Python Bytecode dargestellt.

Damit dieser Bytecode in ein lesbares Format überführt werden kann, wird ein kleines Python-Skript *pythonByteCodeToText.py* geschrieben. Der Programmcode des Skripts ist im Anhang 10 aufgeführt. Das Skript übersetzt den Python Bytecode zuerst in Dezimalzahlen des ASCII-Codes. Im nächsten Schritt wird der ASCII-Code in lesbarem Text überführt.

Das Ergebnis und die zwei Umwandlungen sind dem **Bild 45** zu entnehmen.

#### **4.3. ERZWINGEN DES SCHREIBZUGRIFFS AUF QNX NEUTRINO RTOS**

---

```
user@forensics:~$ python3 pythonByteCodeToText.py

Initial list: [234, 138, 209, 0, 6, 154, 2, 0, 7, 0, 0, 0, 0, 97, 0, 0, 0, 115, 0,
0, 0, 114, 0, 97, 0, 115, 0, 97, 0, 97, 0, 107, 0, 117, 0, 114, 0, 107, 0, 117,
0, 1, 0]

Resultant string: èÑasrasaakurku
```

Bild 45: Ergebnis des Übersetzer-Skripts von Python Bytecode in Textform

Wie in dem **Bild 45** entnehmbar, ergibt der Text, der aus der Konvertierung von Python Bytecode in Text resultiert, keinen Sinn. Der Grund dafür, dass einige Text-Attributwerte nicht korrekt geparsst werden können, kann viele verschiedene Ursachen haben. Somit liegt die Vermutung für das gezeigte Beispiel nahe, dass bei dem Parsen ein Fehler aufgetreten ist. Der Fehler könnte in dem vorliegenden Format liegen, dass von dem Programm *bring2lite* nicht korrekt ausgewertet werden konnte.

Die Rekonstruktion von SQLite-Datenbanken ist ein sehr umfassendes und komplexes Thema. Es gibt noch viele weitere Ansätze SQLite Datenbanken wiederherzustellen. Eine Möglichkeit wäre eine Software zu wählen, die einen anderen Ansatz als die hier verwendete Software *bring2lite* verwendet.

Darüber hinaus gibt es auch die Möglichkeit manuell mit einem Hexadezimal-Editor die einzelnen Flags der SQLite Datenbank Header zu setzen und diese damit wieder lesbar zu machen. Auch diesen Ansatz könnte man in einer weiteren Forschungsarbeit mit dem Kernthema der Wiederherstellung von SQLite Datenbanken weiter ausführen.

### **4.3 Erzwingen des Schreibzugriffs auf QNX Neutrino RTOS**

In diesem Unterkapitel wird das Konzept aus Kapitel 3.5 zum Brechen des Schreibschutzes umgesetzt. Im ersten Schritt wird ein externer Datenträger mit dem Infotainment-Image im RAW-Format beschrieben. Anschließend werden die beschriebenen Ansätze durchgeführt und auf ihre Möglichkeiten hin untersucht.

Abschließend soll zur Beweisführung der Richtigkeit der Ansätze zum Brechen des Schreibschutzes gezeigt werden, dass Schreib-Operationen vollständig durchgeführt und Daten auf dem Infotainment-Image manipuliert werden können.

### 4.3.1 Vorbereitung des Images

In diesem Unterkapitel werden die vorbereitenden Maßnahmen zum Umgehen des Schreibschutzes ausgeführt. Wie in Kapitel 3.5 erläutert, muss das Image vorerst in ein Rohformat konvertiert werden.

Anschließend wird eine Bit-genaue Kopie erstellt und auf einen externen Datenträger geschrieben. Dafür wird das in dem Expert Witness (.E0) Dateiformat vorliegende Infotainment-Image in das RAW-Format konvertiert. Für diesen Einsatz wird das Kommandozeilen-Programm *xmount* (Crossmount) aus dem TSK verwendet.

Die installierte Version des Programms wird mit *xmount --version* abgefragt und im **Bild 46** dargestellt.

```
root@forensics:/home/user/master/image# xmount --version
xmount v0.7.6 Copyright (c) 2008-2018 by Gillen Daniel <gillen.dan@pinguin.lu>

compile timestamp: Sep  2 2018 12:00:15
gcc version: 8.2.0
loaded input libraries:
- libxmount_input_aff.so supporting "aff"
- libxmount_input_aewf.so supporting "aewf"
- libxmount_input_ewf.so supporting "ewf"
- libxmount_input_aaff.so supporting "aaff"
- libxmount_input_raw.so supporting "raw", "dd"
loaded morphing libraries:
- libxmount_morphing_raid.so supporting "raid0"
- libxmount_morphing_combine.so supporting "combine"
- libxmount_morphing_unallocated.so supporting "unallocated"
```

Bild 46: Version *xmount*

Anschließend wird das Infotainment-Image in das RAW-Format konvertiert. Hierfür wird im root-Verzeichnis ein Ordner *ewf* angelegt.

Die Konvertierung wird mit dem Befehl *xmount -in ewf Infotainment.E0? -cache /tmp/info.ovl -out raw /ewf* durchgeführt. Überprüft wird die Konvertierung mit dem Partitionstabellen-Editor *fdisk*.

```
root@forensics:/home/user/master/image# fdisk -V
fdisk from util-linux 2.34
```

Bild 47: Version *fdisk*

Dem **Bild 47** ist die verwendete Version von *fdisk* zu entnehmen.

Zur Überprüfung wird der Befehl: *fdisk -lu /ewf/Infotainment.dd* angewendet.

#### 4.3. ERZWINGEN DES SCHREIBZUGRIFFS AUF QNX NEUTRINO RTOS

---

```
root@forensics:/home/user/master/image# fdisk -lu /ewf/Infotainment.dd
Disk /ewf/Infotainment.dd: 59,22 GiB, 63585648640 bytes, 124190720 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x00000000

Device            Boot   Start     End   Sectors  Size Id Type
/ewf/Infotainment.dd1          64    2809855  2809792  1,3G b1 unknown
/ewf/Infotainment.dd2        2809856 114712575 111902720 53,4G 5 Extended
/ewf/Infotainment.dd3      114712576 124174335  9461760  4,5G b3 unknown
/ewf/Infotainment.dd5      2809920   4087807 1277888 624M b2 unknown
/ewf/Infotainment.dd6      4087872   9330687 5242816  2,5G b2 unknown
/ewf/Infotainment.dd7      9330752  17719295 8388544   4G b2 unknown
/ewf/Infotainment.dd8      17719360 18243583 524224 256M b2 unknown
/ewf/Infotainment.dd9      18243648 20340735 2097088 1024M b2 unknown
/ewf/Infotainment.dd10     20340800 22437887 2097088 1024M b2 unknown
/ewf/Infotainment.dd11     22437952 26632191 4194240   2G b2 unknown
/ewf/Infotainment.dd12     26632256 28991487 2359232  1,1G b2 unknown
/ewf/Infotainment.dd13     28991552 93741055 64749504 30,9G b2 unknown
/ewf/Infotainment.dd14     93741120 114712575 20971456  10G b2 unknown

Partition table entries are not in disk order.
```

Bild 48: Überprüfung der Konvertierung in Rohformat

**Bild 48** zeigt die fehlerfreie Konvertierung des Infotainment-Images und dessen Partitionstabelle.

Im nächsten Schritt wird der externe Datenträger eingehängt und mit dem Kommandozeilen-Programm *dd* (*disk dump*) formatiert. Dadurch wird sichergestellt, dass keine alten Daten mehr auf dem Datenträger vorhanden sind.

Die verwendete Version von des Kommandozeilen-Programms *dd* wird im **Bild 49** gezeigt.

```
root@forensics:/home/user/master/image# dd --version
dd (coreutils) 8.30
Copyright (C) 2018 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <https://gnu.org/licenses/gpl.html>.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Written by Paul Rubin, David MacKenzie, and Stuart Kemp.
```

Bild 49: Version des Befehlszeilen-Programms *dd*

Der folgende Befehl löscht den externen Datenträger durch Überschreiben mit Nullen: *dd bs=1M status=progress if=/dev/zero of=/dev/sdb*

Der Parameter *bs* gibt die Blockgröße in Bytes an, 1M steht hierbei für 1048576 Byte (1024 \* 1024 Byte). Mit dem Parameter *status=progress* wird der Fortschritt visuell angezeigt. Der weitere Parameter *if* steht für Input-File und *of* für Output-File [60].

Nach dem erfolgreichen Überschreiben des externen Datenträgers wird die Bit-genaue Kopie erstellt. Hierfür wird das Rohimage mit dem Befehl `dd if=/ewf/Infotainment.dd of=/dev/sdb` Bit-genaug auf den externen Datenträger geschrieben. Sobald dieser Vorgang abgeschlossen ist, ist auch die Phase der Vorbereitung abgeschlossen und es kann mit dem Brechen des Schreibschutzes begonnen werden.

#### 4.3.2 VM - QNX System

Die erste Variante des Konzepts in Kapitel 3.5 beschreibt das direkte Einbinden des externen Datenträgers in die Forensik-Maschine. Der folgende Befehl dient dem korrekten Einhängen einer einzelnen QNX-Partitions des externen Datenträgers:

```
mount -t qnx6 -o sync=none /dev/sdb1 /mnt/usb/
```

```
root@forensics:/home/user/master/image# mount -t qnx6 -o sync=none /dev/sdb1 /mnt/usb/
mount: /mnt/usb: wrong fs type, bad option, bad superblock on /dev/sdb1, missing codepage
or helper program, or other error.
```

Bild 50: Fehlerhaftes Einhängen des externen Datenträgers in die Forensik-Maschine

**Bild 50** zeigt die Fehlermeldung, die bei dem Versuch der Ausführung des Befehls auftritt. Die Fehlermeldung weist darauf hin, dass die übergebenen Parameter falsch sind oder nicht unterstützt werden. Ohne die Zusatzparameter kann die Partition zwar mit dem Befehl `mount -t qnx6 /dev/sdb1 /mnt/usb/` eingehängt werden, jedoch wird diese ohne den Zusatzparameter schreibgeschützt eingehängt. Das folgende **Bild 51** zeigt die Fehlermeldung bei dem Versuch eine Datei zu erstellen.

```
root@forensics:/mnt/usb# touch testFile
touch: cannot touch 'testFile': Read-only file system
```

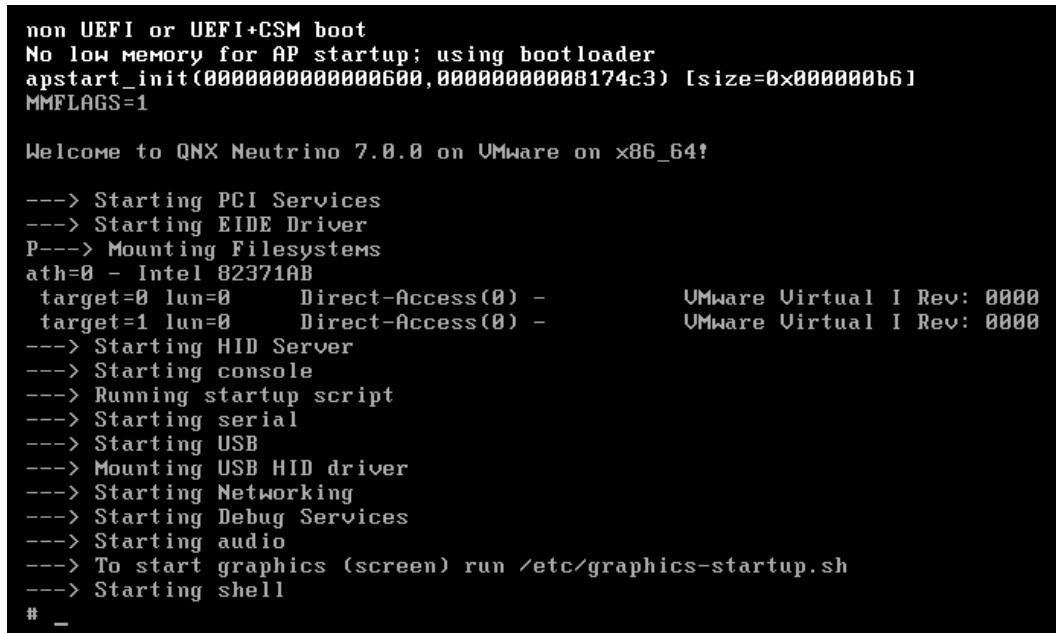
Bild 51: Fehlermeldung Datenträger in Read-Only Modus

Aus diesem Grund wird die erste Variante erst einmal nicht weiter verfolgt und die zweite Variante des Konzepts geprüft.

Für die Umsetzung der zweiten Variante des Konzepts zur Brechung des Schreibschutzes muss eine VM mit einem installierten QNX Neutrino RTOS beschafft werden. Dafür muss eine Forschungslizenz bei Blackberry beantragt werden. Mithilfe dieser Lizenz kann anschließend das *Software Center* von QNX und die passende VM heruntergeladen werden.

Die VM liegt in dem Format `.vmx` vor. Zum Starten der VM wird die Software *VMWare Workstations 15 Player* verwendet. Nach dem Öffnen der VM wird diese mit den Standardparametern eingebunden.

Im **Bild 52** ist die gestartete VM mit dem Betriebssystem QNX Neutrino RTOS dargestellt.



```

non UEFI or UEFI+CSM boot
No low memory for AP startup; using bootloader
apstart_init(000000000000600,0000000008174c3) [size=0x000000b6]
MMFLAGS=1

Welcome to QNX Neutrino 7.0.0 on VMware on x86_64!

---> Starting PCI Services
---> Starting EIDE Driver
P---> Mounting Filesystems
ath=0 - Intel 82371AB
    target=0 lun=0      Direct-Access(0) -
    target=1 lun=0      Direct-Access(0) -          VMware Virtual I Rev: 0000
    VMware Virtual I Rev: 0000
---> Starting HID Server
---> Starting console
---> Running startup script
---> Starting serial
---> Starting USB
---> Mounting USB HID driver
---> Starting Networking
---> Starting Debug Services
---> Starting audio
---> To start graphics (screen) run /etc/graphics-startup.sh
---> Starting shell
# -

```

Bild 52: Geladene QNX Neutrino RTOS VM

Weil das Tastatur-Layout innerhalb der VM auf dem US-Format basiert, wird als Erstes ein SSH-Benutzer angelegt, damit eine Verbindung via SSH ermöglicht wird und mit dem eigenen Tastatur-Layout gearbeitet werden kann.

Für die Einrichtung eines Benutzers werden administrative Rechte benötigt. Der Login als root wird mit dem Benutzernamen *root* und dem Passwort *root* durchgeführt.

Der SSH-Benutzer wird mit dem Befehl *adduser -M sss* angelegt. Der Benutzername wird auf *sss* und das Passwort mit dem Befehl *passwd sss* auf *test* festgelegt. Mit dem Befehl *usermod -aG sshd sss* wird der SSH-Benutzer der sshd-Gruppe hinzugefügt. Anschließend wird der externe Datenträger an die Forensik-Maschine angeschlossen und die VM neugestartet.

Als Erstes wird der externe Datenträger in die VM eingebunden, dies geschieht über die grafische Oberfläche des VMWare Players. Unten rechts in der Ecke ist ein USB-Stick dargestellt, mit einem Doppelklick wird dieser eingebunden. Anschließend erfolgt der Login als root-Benutzer.

Im nächsten Schritt wird der SSH-Daemon mit dem Befehl */usr/sbin/sshd* gestartet. Für den Login via SSH wird die lokale IP-Adresse der VM benötigt. Hierfür wird innerhalb der VM der Befehl *ifconfig* abgesetzt und die IP-Adresse in dem Netzwerkadapter vx0 abgelesen. Daraufhin kann in den Terminal der Forensik-Maschine gewechselt und der

Login via SSH in die VM erfolgen. Für den Login in die VM wird der folgende Befehl abgesetzt: `ssh sss@192.168.188.23`.

Ein direktes Mounten des externen Datenträgers ist nicht möglich. Vorerst müssen zwei Treiber gestartet werden.

Der erste Treiber ist der `io-usb-otg` Treiber, dieser startet den USB-Stack und die USB-Treiber auf einem PCI-basierten System. Weiterhin wird der Befehl über den Parameter `-d uhci` spezifiziert. Dieser Zusatzparameter ermöglicht das gezielte Starten des Treibers für Universal Host Controller Interface (UHCI) USB-Controller. Der vollständige Befehl zum Starten des Treibers lautet: `io-usb-otg -d uhci` [61].

Anschließend muss der Treiber für die USB-Massenspeicher-Schnittstelle gestartet werden. Dies geschieht über den Befehl `devb-umass` [62]. Hierbei ist die Reihenfolge exakt einzuhalten, denn der Treiber für die USB-Massenspeicher-Schnittstelle setzt den USB-Treiber `io-usb-otg` voraus.

Schließlich kann der externe Datenträger eingehängt werden. Eingehängt wird der externe Datenträger mit dem Befehl `mount -e -t qnx6 /dev/hd2`

Zuletzt muss noch die ausgewählte Partition eingehängt werden. Die Partition wird mit dem Befehl `mount -t qnx6 -o sync=none /dev/hd2t178.2 /mnt/fs` in den vorher erstellten Ordner `fs` eingehängt. Der Parameter `-t qnx6` gibt den Typen des einzuhängenden Dateisystems vor. Die zusätzliche Option `-o sync=none` unterbindet die synchrone Aktualisierung aller Operationen auf dem externen Datenträger. Die Option `sync=none` sorgt dafür, dass die Partition nicht-schreibgeschützt eingehängt wird. Im Bild 53 ist die eingehängte Partition und dessen Inhalt dargestellt.

```
# mount -t qnx6 -o sync=none /dev/hd2t178.2 /mnt/fs
# ls -la /mnt/fs
total 270
drwxrwxr-x  5 root      root          32768 Jan  05 19:14 .
drwxr-xr-x  6 root      root          4096 Jan 24 18:01 ..
drwx-----  2 root      root          32768 Aug  02  2018 .boot
-rw-rw-rw-  1 root      root          616 Aug 23  2017 Update.txt
-rwx-w--w-  1 root      root          98 Jul 30  2018 Update.txt.checksum
-rwx-w--w-  1 root      root          32 Jul 30  2018 Update.txt.fileinfo
drwxrwxrwx  2 root      root          32768 Jul 30  2018 config
drwxrwxrwx  2 root      root          32768 Jul 30  2018 database
-rwx-w--w-  1 root      root          99 Jul 30  2018 database.checksum
-rwx-w--w-  1 root      root          32 Jul 30  2018 database.fileinfo
```

Bild 53: Vorschau der Dateien und Verzeichnisse auf der Partition `hd2t178.2`

### 4.3.3 Manipulation des Images

Zur Beweisführung sollen verschiedene Schreib-Operationen auf der eingehängten Partition durchgeführt werden. Die Vorgehensweise ist dabei wie im Konzept 3.5 beschrieben.

Wie im **Bild 54** dargestellt, wird ein Test-Verzeichnis mit dem Namen *beweisDir* erstellt.

```
# mkdir beweisDir
# ls -la
total 334
drwxrwxr-x  6 root      root      32768 Jan 27 00:02 .
drwxr-xr-x  6 root      root      4096 Jan 27 00:01 ..
drwx-----  2 root      root      32768 Aug  02  2018 .boot
-rw-rw-rw-  1 root      root      616 Aug 23  2017 Update.txt
-rwx-w--w-  1 root      root      98 Jul 30  2018 Update.txt.checksum
-rwx-w--w-  1 root      root      32 Jul 30  2018 Update.txt.fileinfo
drwxr-xr-x  2 root      root      32768 Jan 27 00:02 beweisDir
drwxrwxrwx  2 root      root      32768 Jul 30  2018 config
drwxrwxrwx  2 root      root      32768 Jul 30  2018 database
-rwx-w--w-  1 root      root      99 Jul 30  2018 database.checksum
-rwx-w--w-  1 root      root      32 Jul 30  2018 database.fileinfo
```

Bild 54: Erstellter Ordner auf der nicht schreibgeschützten Partition

Anschließend wird eine Test-Datei erstellt und mit einem Beispiel-Text befüllt. Die Datei hat den Namen *testFile*. Abschließend wird die Datei wieder gelöscht. **Bild 55** zeigt die zuvor beschriebenen Operationen des Erstellens, Bearbeitens und Löschens einer Beispiel-Datei.

```
# ls -la
total 129
drwxr-xr-x  2 root      root      32768 Jan 27 00:07 .
drwxrwxr-x  6 root      root      32768 Jan 27 00:02 ..
-rw-r--r--  1 root      root      34 Jan 27 00:11 testFile
# cat testFile
Der Schreibschutz ist aufgehoben.
# rm testFile
# ls -la
total 128
drwxr-xr-x  2 root      root      32768 Jan 27 00:12 .
drwxrwxr-x  6 root      root      32768 Jan 27 00:02 ..
```

Bild 55: Datei erstellen, beschreiben und löschen

Somit ist bewiesen, dass der Sicherheitsmechanismus des Schreibschutzes überwunden werden kann. Auf diese Art und Weise könnte Schadcode oder auch Spionage-Software in das Infotainmentsystem impliziert werden. Das könnte genutzt werden, um Personen zu überwachen, deren Daten abzugreifen oder auch um das Infotainmentsystem zu zerstören.

Besonders kritisch ist es bei Leih- oder Leasing-Fahrzeugen zu sehen, denn diese werden in der Regel von vielen verschiedenen Personen genutzt. Ebenso gilt dies für Firmenflotten im Zusammenhang mit Industrie-Spionage. Es ist vorstellbar das mit Hilfe von Schadcode bzw. Spionage-Software Daten wie Whatsapp-Nachrichten, E-Mails, Kontaktbücher, SMS und Anrufhistorien an einen Angreifer übermittelt werden können. Die Datenübertragung könnte über die W-LAN, Bluetooth oder die bestehende Internetverbindung realisiert werden.

Geübte Personen können das Infotainmentsystem mit nur wenigen Handgriffen ausbauen und den USB-Stick oder die SD-Karte, auf der die Infotainmentsystem-Software installiert ist, manipulieren.

## 4.4 Weiterführende Analyse Phase

Mit den in dem Kapitel 4.4 gewonnenen Erkenntnissen und der erarbeiteten Möglichkeit die verschiedenen Partitionen in der VM zu mounten, können nun insgesamt 12 Partitionen des Datenträgers analysiert werden.

Im **Bild 56** ist der in die QNX Neutrino RTOS VM eingehängte externe Datenträger dargestellt. Innerhalb der VM wird dieser mit der Bezeichnung „hd2“ aufgeführt. Die einzelnen Partitionen sind mit dem Zusatz „t“ und einer Zahl benannt. Die Benennung weiterer Partitionen des selben Dateisystems werden durch „.“ und eine Zahl von 1 bis 9 erweitert.

```
$ ls -la | grep "hd2t*"
brw----- 1 root      root      3,   2 Jan 26 23:25 hd2
brw----- 1 root      root      1,  12 Jan 26 23:25 hd2t177
brw----- 1 root      root      1,  14 Jan 26 23:25 hd2t178
brw----- 1 root      root      1,  15 Jan 26 23:25 hd2t178.1
brw----- 1 root      root      1,  16 Jan 26 23:25 hd2t178.2
brw----- 1 root      root      1,  17 Jan 26 23:25 hd2t178.3
brw----- 1 root      root      1,  18 Jan 26 23:25 hd2t178.4
brw----- 1 root      root      1,  19 Jan 26 23:25 hd2t178.5
brw----- 1 root      root      1,  20 Jan 26 23:25 hd2t178.6
brw----- 1 root      root      1,  21 Jan 26 23:25 hd2t178.7
brw----- 1 root      root      1,  22 Jan 26 23:25 hd2t178.8
brw----- 1 root      root      1,  23 Jan 26 23:25 hd2t178.9
brw----- 1 root      root      1,  13 Jan 26 23:25 hd2t179
```

Bild 56: Übersicht der eingehängten Partitionen

Ein Vergleich der **Bilder 56** und **48** zeigt, dass von den zuvor 14 Partitionen nur noch 12 Partitionen angezeigt werden. Der Grund dafür ist bisher unklar. Eine Überprüfung und Wiederholung des Erstellens einer Bit-genauen Kopie des Rohimages auf den externen Datenträger zeigt das selbe Ergebnis.

Im Folgenden werden die 12 Partitionen nacheinander eingehängt und untersucht. Übersichten über die Verzeichnisse und Dateien auf den jeweiligen Partitionen sind in dem Anhang 10 aufgeführt.

### hd2t177

Die Partition *hd2t177* ist die schon zuvor untersuchte Partition 1.

### **hd2t178**

Als Nächstes wird die Partition *hd2t178* untersucht. Diese beinhaltet insgesamt drei Verzeichnisse „RSDB“, „tst“ und „boot“.

In dem Verzeichnis „RSDB“ ist eine von Volkswagen erstellte SQLite Datenbank für die europäischen Radiosender abgespeichert, diese trägt den Namen „VW\_STL\_DB“. Ein Abgleich der Radiosender Datenbank, die in dem Kapitel 5 wiederhergestellt wurde, zeigt das diese übereinstimmen und den gleichen Inhalt besitzen.

Weiterhin ist in dem Verzeichnis eine Text-Datei mit dem Namen *update.txt*. Diese zeigt lediglich das Datum des letztens Updates der Radiosender Datenbank. Das versteckte Verzeichnis „boot“ sowie das Verzeichnis „tst“ sind leer.

### **hd2t178.1**

Die Partition *hd2t178.1* beinhaltet insgesamt 5 Verzeichnisse. Innerhalb dieser Verzeichnisse liegen viele Konfigurationsdateien und -datenbanken für die „Text To Speech“ Sprachausgabe in sämtlichen europäischen Sprachen.

Weiterhin sind auch die verschiedenen wählbaren Stimmen für das Navigationssystem in sämtlichen Sprachen auf dieser Partition abgelegt. Von besonderer Bedeutung sind die Datenbanken „data.adb.plugin.data“ und „data.media.plugin.data.db“.

Die Datenbank „data.adb.plugin.data“ beinhaltet das Adressbuch, das zuvor schon auf der Partitionen 10 und im nicht-allozierten Bereich des Images extrahiert wurde. Die Media-Datenbank „data.media.plugin.data.db“, ist auch schon durch die Extraktion im nicht-allozierten Bereich des Images bekannt.

### **hd2t178.2**

Die nächste Partition *hd2t178.2* besitzt ebenfalls ein leeres Verzeichnis „boot“. Darüber hinaus werden noch eine „update.txt“ Text-Datei und zwei weitere Verzeichnisse extrahiert. Das Verzeichnis *config* beinhaltet eine Konfigurationsdatei für Gracenote im JASON-Format. In dem Verzeichnis *database* liegen verschiedene Gracenote-Datenbanken.

### **hd2t178.3**

Auf der Partition *hd2t178.3* befinden sich insgesamt drei Media-Datenbanken und dessen versteckte Abbilder. Die erste Media-Datenbank *mediadb\_2* hat keine Einträge, durch die vorhanden Relationen lässt sich jedoch erkennen, dass dies eine Standard-Datenbank für Musik ist.

#### **4.4. WEITERFÜHRENDE ANALYSE PHASE**

---

Die zweite Media-Datenbank *mediadb\_4* hat ebenfalls keine Einträge, jedoch lässt auch diese durch ihren Aufbau darauf schließen, dass es sich hierbei um eine Datenbank für Podcasts und Hörbücher handelt. Die letzte der drei vorhandenen Media-Datenbanken ist die gleiche Datenbank, die bereits in dem nicht-allozierten Bereich extrahiert wurde.

##### **hd2t178.4**

Auf der Partition *hd2t178.4* befinden sich insgesamt drei Verzeichnisse. Ein verstecktes „boot“ Verzeichnis, sowie zwei weitere Verzeichnisse „tmp“ und „persistent“. Das „boot“- und das „tmp“-Verzeichnis besitzen keinen Inhalt. Im Gegensatz dazu befinden sich in dem Verzeichnis „persistent“ insgesamt 46 .PNG-Bilddateien. Diese Bilddateien sind Albumcover, auf die in der Media-Datenbank *mediadb\_4* verwiesen wird.

##### **hd2t178.5**

Die Partition *hd2t178.5* gleicht dem Inhalt der zuvor untersuchten Partition 10. Es kann sich dabei aber nicht um genau die gleiche Partition handeln, denn auf der Partition 10 ist das Adressbuch gefüllt mit Namen und Telefonnummern. Auf der Partition *hd2t178.5* sind alle Datenbanken ohne Einträge und Verzeichnisse ohne Inhalt.

##### **hd2t178.6**

Die nächste zu untersuchende Partition ist die Partition *hd2t178.6*. Auf dieser liegen lediglich ein verstecktes Verzeichnis „boot“ und ein Verzeichnis mit der Benennung „navigation“. Das Verzeichnis „navigation“ beinhaltet zwei .blob-Dateien mit den Namen „SDKQuickStartPersistence.blob“ und „SDKVZODatabasePersistence.blob“.

##### **hd2t178.7**

Der Inhalt der Partition *hd2t178.7* gleicht dem Inhalt der zuvor untersuchten Partition 12. Somit könnte dies eine exakte Kopie derer oder sie selbst sein, dass kann nicht abschließend geklärt werden.

##### **hd2t178.8**

Sämtliche weitere Datenbanken für die Koordinaten und Namen von den europäischen Ländern, Städten, Landmarken für Point of Interest (POIs), weitere Konfigurationsdateien sowie 2D- und 3D-Modelle für die Karten des Navigationssystems sind auf der Partition *hd2t178.8* abgespeichert.

##### **hd2t178.9**

Auf der Partition *hd2t178.9* ist lediglich ein verstecktes Verzeichnis „boot“, dass ebenfalls keine Daten beinhaltet.

#### 4.4. WEITERFÜHRENDE ANALYSE PHASE

---

##### **hd2t179**

Die letzte zu untersuchende Partition ist die Partition *hd2t179*. Diese umfasst insgesamt vier Verzeichnisse mit den Bezeichnungen „boot“, „app“, „system“ und „tmp“.

Auch in diesem Falle ist das leere Verzeichnis „boot“. In dem Verzeichnis „app“ befindet sich ein weiteres verstecktes und leeres Verzeichnis mit der Benennung „pers“. Das Verzeichnis „system“ besitzt zwei leere Unterverzeichnisse mit den Bezeichnungen „core“ und „logs“. Auch das letzte Verzeichnis „tmp“ besitzt ein leeres Unterverzeichnis mit der Benennung „sdis“.

Die nachfolgende **Tabelle 6** zeigt eine Teilübersicht über die in diesem Kapitel untersuchten und analysierten Partitionen.

<b>Benennung</b>	<b>Einsatz</b>	<b>Beschreibung</b>
hd2t177	Systempartition	Betriebssystem, Images, Systemprogramme, Dienste, Bibliotheken und Konfigurationsdateien
hd2t178	Infotainment	Radiosender
hd2t178.1	Navigationssystem	Sprachausgabe, Mediendateien und personenbezogene Daten (Adressbuch)
hd2t178.2	Infotainment	Konfigurations-Datenbanken und Konfigurationsdateien für Gracenote
hd2t178.3	Infotainment	Datenbanken für Musik, Podcasts und Hörbücher
hd2t178.4	Infotainment	Bilddateien der Albumcover
hd2t178.5	Infotainment	Adressbuch, Bilder und Truffles-Datenbank
hd2t178.6	Navigationssystem	Systemdateien
hd2t178.7	Navigationssystem	Datenbank für Navigationssystem (Standorte, gefahrene Strecken etc.)
hd2t178.8	Navigationssystem	Konfigurationsdateien, Kartenmaterial und Datenbanken für europäische Länder- und Städtenamen
hd2t178.9	Unklar	Nur ein verstecktes und leeres „boot“ Verzeichnis
hd2t179	Systempartition	Verzeichnisse für Kernel, Applikationen und temporäre Dateien

Tabelle 6: Analyseergebnis der Partitionen und dessen Einsatzzweck mit QNX-VM

In dem folgenden Kapitel 4.5 wird eine Gesamtübersicht über alle analysierten Partitionen erstellt. Dies dient einem strukturierten Überblick über die verschiedenen Partitionen, wie auch dessen Einsatzzweck. Zusätzlich wird erörtert, warum die Menge der Partitionen zwischen den Systemen inkonsistent ist.

## 4.5 Present Phase

### Systemüberblick

In dem Bild 57 ist der Inhalt der extrahierten Datei *version\_info.txt* dargestellt. Der Datei können einige grundlegende Informationen über das Infotainmentsystem und die verwendete Software entnommen werden.

```
root@forensics:/infotainment/1# cat version_info.txt
Product = MIB2MAIN
Branch = MIB2CLU8_SEAT
Up to CL = 10800
FirstTierSupplier = MIB2MAIN
Project = project_mib2_skoda_MIB2MAIN
OEM = SK
Region = ER
Variant = G1
Media-Stack = EVO

Framework = 5.49.12.SR3 MIB2MAIN I199 CI68
JavaVM = j9_1.16+ObjectSwapper_1.02+XIPLoader_1.10+MCFObjectReader_1.01C+JavaCrypto_1.05+Socks5Support_1.00
DSI = 2_67.1
GraphicsServices = MIB2CLU8 199.1
Splashscreens = 65.0 MIB2MAIN
rdiserver = 122
Navigation = CLU8_16291MIB2MAIN_B17.11.1.80227
NavigationMapStyles:
    - ASIA = 10.0.1097
    - AUS = 10.0.1097
    - EU = 10.0.1104
    - IL = 10.0.1097
    - INDIA = 10.0.1097
    - MEAST = 10.0.1097
    - MSA = 10.0.1097
    - MSA2 = 10.0.1097
    - NEAST = 10.0.1097
    - ZA = 10.0.1097
LocationAccessor = 0.9.15
Speech = 129.1650.1-MIB2MAIN
SpeechResources = 129.1706.1
sseProc = 1.4.3
HMI = H29.319.79
VoiceEncoderApp = v2.1.287_FW5.49.12.SR3_TC17.45.05A
GoogleEarth = v2.1.1222_FW5.49.12.SR3_TC17.45.05A_v7_Bclu8
RenderDataProvider = 24
Streetview = v2.11.29_FW5.49.12.SR3_TC17.45.05A
Media = clu8_17033B
PDK = Nvidia_17.45.06-8589238
MediaEngine = MM2_Media_1.0.61.4
exFAT-Library = Tuxera_2016/02/04-14:42:21-UTC
devp-iso-nmx-mib2 = 2014/08/15_05-51-30_UTC
putl = 1.0.0
uaputl = 2016/08/03-11:44:10-EDT
BlueSDK = 4.3.1 + AV 2.2.2u1
RealVNC = 2.8.0.3986-QNX
AH6A-FW = 03.001 (App: 01.000.01)
ALS6A-FW = 02.011 (App: 01.001.48)
ALS6A-BOLO = 84084
Bluetooth-Firmware = Id: w8787-Ax, RF878X, FP44, 14.44.35.p233
TelephoneDriver = 10.754.162
Radiodata = 2.8.0
GracenoteSDK = 3.05.4
OS = QNX 6.5.0
Toolchain = QNX_OS_650SP1_2017.05.05A
```

Bild 57: Inhalt der Datei *version\_info.txt*

Im Namen des Branches wird der Automobilhersteller *Seat* genannt, das Projekt selbst

hat den Namen des Automobilherstellers *Skoda* in der Dateibenennung. Selbstverständlich kann dies ein Zufall sein, jedoch gehören beide Automobilhersteller zu dem Volkswagen Konzern. Ebenfalls wird der OEM (Hersteller) mit der Abkürzung „SK“ angegeben, die wahrscheinlich für *Skoda* oder die *Slowakei* steht. Beide Annahmen stützen die Vermutung, dass es sich um den Hersteller *Skoda* handelt, da dieser seine Fahrzeuge in der Slowakei fertigt.

Diese Erkenntnis zusammengefasst mit sämtlichen anderen Hinweisen, die im Laufe der Analyse erarbeitet wurden, deuten mit einer hohen Wahrscheinlichkeit darauf hin, dass dieses Infotainmentsystem aus einem PKW des Automobilherstellers *Skoda* stammt.

Darüber hinaus werden für sämtliche Treiber, Frameworks und Programme die jeweiligen eingesetzten Versionen mitgeliefert. Ebenso auch das eingesetzte Betriebssystem wie auch die Angabe der OS Version. Auch in diesem Fall stimmen die bisher gesammelten Informationen mit dem in diesem Dokument angegebenen Betriebssystem wie auch der Version überein. Damit kann bestätigt werden, dass das eingesetzte Betriebssystem ein QNX Neutrino RTOS in der Version 6.5.0 ist.

### Partitionsübersicht

In dem Kapitel 4.2 konnten insgesamt 14 Partitionen ermittelt werden. Demgegenüber sind in der Untersuchung mittels der QNX Neutrino RTOS VM im Kapitel 4.4 nur insgesamt 12 Partitionen erkannt und untersucht worden.

Die vollständige Untersuchung aller Partitionen hat gezeigt, dass 13 verschiedene Partitionen existierten. Denn die Partitionen 1 und 12 der anfänglichen Untersuchung befinden sich unter den Partitionen der weiterführenden Analyse, alleinig die Partition 10 konnte durch den Einsatz der QNX Neutrino RTOS VM nicht untersucht werden.

Außerdem wird angenommen, dass eine der anfänglich 14 erkannten Partitionen den nicht-allozierten Bereich des externen Datenträgers darstellt. Diese Annahme stützt sich darauf, dass der externe Datenträger auf den das zu untersuchende Image im Rohformat geschrieben wurde und deutlich größer als das Image selbst ist.

Eine Gesamtübersicht der Partitionen, sowie dessen Einsatzzweck und Beschreibung ist in der nachfolgenden Tabelle 7 dargestellt.

Benennung	Einsatz	Beschreibung
hd2t177	Systempartition	Betriebssystem, Images, Systemprogramme, Dienste, Bibliotheken und Konfigurationsdateien
hd2t178	Infotainment	Radiosender

---

Fortsetzung auf der nächsten Seite

Fortsetzung der vorherigen Seite

<b>Benennung</b>	<b>Einsatz</b>	<b>Beschreibung</b>
hd2t178.1	Navigationssystem	Sprachausgabe, Mediendateien und personenbezogene Daten (Adressbuch)
hd2t178.2	Infotainment	Konfigurations-Datenbanken und Konfigurationsdaten für Gracenote
hd2t178.3	Infotainment	Datenbanken für Musik, Podcasts und Hörbücher
hd2t178.4	Infotainment	Bilddateien der Albumcover
hd2t178.5	Infotainment	Adressbuch, Bilder und Truffles-Datenbank
hd2t178.6	Navigationssystem	Systemdateien
hd2t178.7	Navigationssystem	Datenbank für Navigationssystem (Standorte, gefahrene Strecken etc.)
hd2t178.8	Navigationssystem	Konfigurationsdateien, Kartenmaterial und Datenbanken für europäische Länder- und Städtenamen
hd2t178.9	Unklar	Nur ein verstecktes und leerer „boot“ Verzeichnis
hd2t179	Systempartition	Verzeichnisse für Kernel, Applikationen und temporäre Dateien
Part. 10	Infotainment	Adressbuch, Bilder und Truffles-Datenbank
Part. 14	-	Nicht-allozierter Bereich der Festplatte

Tabelle 7: Gesamtübersicht der Partitionen und dessen Einsatzzweck

Bei den insgesamt 13 Partitionen handelt es sich um Dateisysteme der Klasse *Block*. Zu dieser Klasse gehören beispielweise Festplatten und CD-Roms, das sind traditionell Block-basierte Geräte.

Darüber hinaus handelt es sich um die Dateisysteme *QNX Power-Safe POSIX partition* und *QNX Power-Safe POSIX partition (secondary)*, diese sind auch unter dem Namen *QNX6* Dateisystem bekannt. Diese Information ist der Dokumentation zur Standardbenennung von Dateisystemen innerhalb des QNX Neutrino RTOS zu entnehmen. Die *QNX Power-Safe POSIX partition* Dateisysteme sind proprietäre QNX Dateisysteme, die Stromausfälle überstehen, ohne dass Daten verloren gehen oder beschädigt werden [63]. Ein Auszug der verschiedenen QNX-Dateisystem-Typen ist der folgenden **Tabelle 8** zu entnehmen [64].

<b>Typ</b>	<b>Dateisystem</b>
7	Previous QNX version 2 (pre-1988) oder OS/2 HPFS oder Windows NT
8	QNX 1.x and 2.x (“qny”)

Fortsetzung auf der nächsten Seite

Tabelle 7 – Fortsetzung der vorherigen Seite

<b>Typ</b>	<b>Dateisystem</b>
9	QNX 1.x and 2.x (“qnz”)
79	QNX POSIX partition
177	QNX Power-Safe POSIX partition (secondary)
178	QNX Power-Safe POSIX partition (secondary)
179	QNX Power-Safe POSIX partition

Tabelle 8: QNX Partitionstypen und Dateisysteme

### Betriebssystem

Das eigentliche Betriebssystem des Infotainmentsystem befindet sich innerhalb des sicherstellten Dateiarchivs *main\_stage2.ifs.lzo*. Die technische Funktionsweise des Infotainmentsystems wird im Folgenden dargestellt.

Im Gegensatz zu einem herkömmlichen Desktop-Computer ist das gesamte Betriebssystem nicht direkt auf dem Datenträger installiert, sondern in einem .ifs-Image (Image Filesystem) auf einer Partition abgelegt. Das .ifs-Image kann, abgesehen von dem eigentlichen Betriebssystem, weitere Programme und Bibliotheken beinhalten.

Für die Erstellung dieses Images wird ein BSP benötigt. Das BSP bietet eine Abstraktionsschicht aus hardwarespezifischer Software. Diese Software erleichtert die Implementierung des QNX Neutrino RTOS auf dem jeweiligen Board. Allgemein sind BSPs architekturnspezifisch, boardspezifisch und teilweise boardrevisionsspezifisch. Grundsätzlich dienen sie der Initialisierung und anderer architektur- und hardwarespezifische Aufgaben, die eine lauffähige Umgebung für das System vorbereiten. Typischerweise beinhalten diese einen hardwarespezifischen **Initial Program Loader (IPL)** und den Startup-Code für den Boot-Vorgang [65, 66].

Der IPL wiederum richtet für sich eine minimale Umgebung für die Ausführung seines kompilierten C-Codes ein. Anschließend sucht er das Betriebssystem-Image, lädt dieses in den Speicher, entpackt es und springt zum Startcode des Betriebssystems. Sobald das Betriebssystem startet, werden die verschiedenen untersuchten Partitionen nach Bedarf dynamisch ein- und ausgehängt [67].

### **main\_stage2.ifs.lzo und persist.img**

Für die Untersuchung des Betriebssystem-Images muss das Dateiarchiv extrahiert werden. Dieses sollte mit dem Kommandozeilen-Programm *lzip* möglich sein. Jedoch erkennt das Programm die Datei nicht als .lzo-Dateiarchiv. Erstmalig wurde das Programm

*lzop* in der Version 1.04 verwendet, aber auch ein Versuch mit einer älteren Version des Programmes ist gescheitert. Der abgesetzte Befehl sowie die Antwort des Programmes können dem **Bild 58** entnommen werden.

```
user@forensics:~/Desktop/img_restore$ sudo lzop -d main_stage2.ifs.lzo
[sudo] password for user:
lzop: main_stage2.ifs.lzo: not a lzop file
```

Bild 58: Fehlerhafter Extraktion des Dateiarchivs

Aus diesem Grund wird das Dateiarchiv, dass das Betriebssystem enthält, mit *Binwalk* weiter untersucht. Die Untersuchung mit *Binwalk* zeigt lediglich, dass das Dateiarchiv einige 32 Bit Binärdateien, XML-Dateien und weitere verschlüsselte Daten enthält. Eine weitere Untersuchung mit dem *fdisk* Tool zeigt, wie im **Bild 59** dargestellt, dass die Datei eine Größe von 28,81 MiB umfasst und eine Sektorgröße von 512 Byte besitzt.

```
user@forensics:~/Desktop/img_restore$ fdisk -l main_stage2.ifs.lzo
Disk main_stage2.ifs.lzo: 28,81 MiB, 30188032 bytes, 58961 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
```

Bild 59: Untersuchung des Betriebssystem-Images mittels *fdisk*

Es sind keine weiteren Informationen über das Betriebssystem innerhalb des Betriebssystem-Images zu eruieren.

Die Untersuchung des *persist.img* mit *Binwalk* zeigt eine mit zlib komprimierte Datei. Mittels dem *fdisk* Tool wird ermittelt, dass dieses Image 8 MiB groß ist und ebenfalls eine Sektorgröße von 512 Byte besitzt. Jedoch kann auch dieses mit bekannten Linux-Kommandozeilen Programmen nicht dekomprimiert werden und somit keine weitere Untersuchung erfolgen.

Für eine tiefergehende Analyse des Betriebssystems selbst bedingt es dem Reverse Engineering. Dies ist nicht teil des Themas dieser Arbeit und würde den zeitlichen Rahmen dieser sprengen. Jedoch wäre es ein sehr interessanter Ansatz, der mittels einer weiteren Forschungsarbeit vertieft werden könnte.

### Personenbezogene Daten

In diesem Kapitel werden die extrahierten personenbezogenen Daten aufbereitet und dargestellt.

Insgesamt wurden zwei Datenbanken extrahiert, die personenbezogene Daten enthalten. Begonnen wird mit der Detailanalyse des Adressbuchs und im zweiten Teil werden die

## 4.5. PRESENT PHASE

---

Daten der Navigations-Datenbank extrahiert, aufbereitet und der Verlauf der navigierten Strecke visualisiert.

### Adressbuch

Zuerst wird das lokale Profil des Adressbuchs gesichtet, dies zeigt als Gerätenamen den Namen *iPhone von Anton* mit der Gerät Identifikationsnummer *70:70:0D:96:25:DC* an. Ein Verlauf der Gesprächshistorie ist nicht vorhanden.

Im nächsten Schritt werden die Kontakte mit den dazugehörigen Adressen und Telefonnummern mit der folgenden SQL-Abfrage exportiert und innerhalb der Software *LibreOffice Calc* formatiert und leserlich aufbereitet.

Vorname	Nachname	Telefonnummer	Straße	Postleitzahl	Stadt
[BE]LUG		+49 30 45490376			
ADAC		+49 800 222222			
Alex	Lahr	+49 161 1183445			
Ann-Kathrin	Munsberg	+49 159 5557291			
Annette	Dummamn	030 33052126			
Annika	Poensgen	01518 4478923			
Antonio	Loehr	030 52205835			
Christina	Brauer	0172 1129843			
Claus	Baumgartner	01518 610236			
Daniel	Ortner	0176 2325887			
Dierk	Schreiter	+49 151 5589641			
Emily	Jaster	0158 10040530			
Felix	Schlaich	+49 134 3459721			
Finn	Zienkiewicz	+49 157 5910077			
Henry	Kalinowski	+49 176 883667			
Ich		+49 172 4546640			
Irene	Tänzer	+49 172 15129993			
Isabelle	Rolfs	+49 1642 539556			
Jann	Faber	030 225572			
Johannes	Dummamn	0151 5866698			
Katharina	Koskova	0176 83941002			
Konstantin	Gustenberg	0176 6972354			
Lennard	K.	0151 4667823			
Lisa	M.	+49 176 43445572			
Luca	Kühling	0151 5588116			
Luisa	Kotulla	+49 151 88833445			
Mama		+49 176 5529654			
Marcus	Schrard	+49 172 7711517			
Marion	Rössler	+49 176 4788546			
Michael	Köst	+49 151 115144			
Papa		+49 176 5521087			
Peter (Lackierer)		+49 151 114785			
Sarah Schatz		+49 151 4344564			
Siegfried	Bär	+49 151 5395424			
Silke	Bellmann	+49 30 46368691			
Simon	Keller	+49 30 8822579			
Steven	Brecht	+49 151 2265223			
Svenja	Meiling	+49 1756 505424			
Thorben	Bruder	+49 30 4558960			
Vera	Schwester	030 74562			
Willi	Dür	0176 77789631			
Wolfgang	Lambert	0176 2255367			

Bild 60: Aufbereitung der extrahierten Tupel aus der *Adressbuch* Datenbank

In dem **Bild 60** ist zu erkennen, dass sämtliche Telefonnummern und Adressen den jeweiligen Kontakten zuordnungsbar sind. Dadurch das sämtliche Haustelefonnummern aus dem Raum Berlin stammen, kann die Annahme getroffen werden, dass der Besitzer des iPhones mit einer hohen Wahrscheinlichkeit aus dem Raum Berlin stammt. Das gesamte Adressbuch des iPhones ist rekonstruierbar und auslesbar.

### SQL-Abfrage

```
Select firstName AS Vorname, lastName AS Nachname,
       displayedNumber AS Telefonnummer, street AS Straße, city AS Stadt,
       postalCode AS Postleitzahl, city AS Stadt
  FROM Person AS person
 LEFT JOIN Address AS address ON address.fk_person_id = person.id
INNER JOIN Phone AS phone ON phone.fk_person_id = person.id
```

### Navigation

Als Nächstes wird die Datenbank des Navigationssystems mittels SQL abgefragt und anschließend werden die Daten in der Software *LibreOffice Calc* formatiert und aufbereitet. Das **Bild 61** zeigt die extrahierten und aufbereiteten Daten.

Nummer	Latitude	Longitude	UNIX-Time	Deutsche Zeit	Beschreibung
	52,539380	13,613955	1557403834	09.05.2019 13:10:34	Start
0	52,538869	13,611147	1557403848	09.05.2019 13:10:48	Zwischenpunkt
1	52,538404	13,609855	1557403888	09.05.2019 13:11:28	Zwischenpunkt
2	52,537632	13,609742	1557403901	09.05.2019 13:11:41	Zwischenpunkt
3	52,537770	13,608694	1557403940	09.05.2019 13:12:20	Zwischenpunkt
4	52,539553	13,608509	1557403962	09.05.2019 13:12:42	Zwischenpunkt
5	52,540892	13,619361	1557403999	09.05.2019 13:13:19	Zwischenpunkt
6	52,536596	13,620718	1557404048	09.05.2019 13:14:08	Zwischenpunkt
7	52,533339	13,619675	1557404071	09.05.2019 13:14:31	Zwischenpunkt
8	52,531538	13,617958	1557404135	09.05.2019 13:15:35	Zwischenpunkt
9	52,531424	13,615148	1557404168	09.05.2019 13:16:08	Zwischenpunkt
10	52,527083	13,609418	1557404198	09.05.2019 13:16:38	Zwischenpunkt
11	52,529341	13,590187	1557404221	09.05.2019 13:17:01	Zwischenpunkt
12	52,517363	13,588626	1557404263	09.05.2019 13:17:43	Zwischenpunkt
13	52,514190	13,562748	1557404287	09.05.2019 13:18:07	Zwischenpunkt
14	52,503024	13,561691	1557404298	09.05.2019 13:18:18	Zwischenpunkt
15	52,477690	13,559545	1557404436	09.05.2019 13:20:36	Zwischenpunkt
16	52,469847	13,552593	1557404452	09.05.2019 13:20:52	Zwischenpunkt
17	52,474043	13,547132	1557404468	09.05.2019 13:21:08	Zwischenpunkt
18	52,478075	13,535781	1557404481	09.05.2019 13:21:21	Zwischenpunkt
19	52,481880	13,527142	1557404502	09.05.2019 13:21:42	Zwischenpunkt
20	52,481380	13,526136	1557404513	09.05.2019 13:21:53	Zwischenpunkt
21	52,481599	13,525264	1557404531	09.05.2019 13:22:11	Zwischenpunkt
22	52,478428	13,524406	1557404548	09.05.2019 13:22:28	Zwischenpunkt
23	52,469414	13,513754	1557404571	09.05.2019 13:22:51	Zwischenpunkt
24	52,470606	13,511389	1557404582	09.05.2019 13:23:02	Zwischenpunkt
25	52,469985	13,508932	1557404597	09.05.2019 13:23:17	Zwischenpunkt
26	52,469255	13,506707	1557404612	09.05.2019 13:23:32	Zwischenpunkt
27	52,469696	13,505994	1557404617	09.05.2019 13:23:37	Zwischenpunkt
28	52,469273	13,504964	1557404629	09.05.2019 13:23:49	Zwischenpunkt
	52,468503	13,504529	1557404650	09.05.2019 13:24:10	Ziel

Bild 61: Aufbereitung der extrahierten Tupel aus der *Navigation* Datenbank

#### 4.5. PRESENT PHASE

---

Es werden die Koordinatenpaare sowie die dazugehörigen UNIX-Zeitstempel der einzigen gespeicherten Fahrt (Trip) exportiert. Zusätzlich wird der Linux-Timestamp in deutsche Zeit (UTC + 1) konvertiert, dafür wird die folgende Formel verwendet:

$$(UNIX-Zeitstempel/86400)+25569+(1/24)$$

Für eine einfache Veranschaulichung und einer besseren Nachvollziehbarkeit der gefahrenen Route werden die Koordinaten-Paare auf der Website <https://www.mapcustomizer.com/> auf einer Karte dargestellt. Das Resultat dieser Visualisierung ist dem **Bild 62** zu entnehmen.

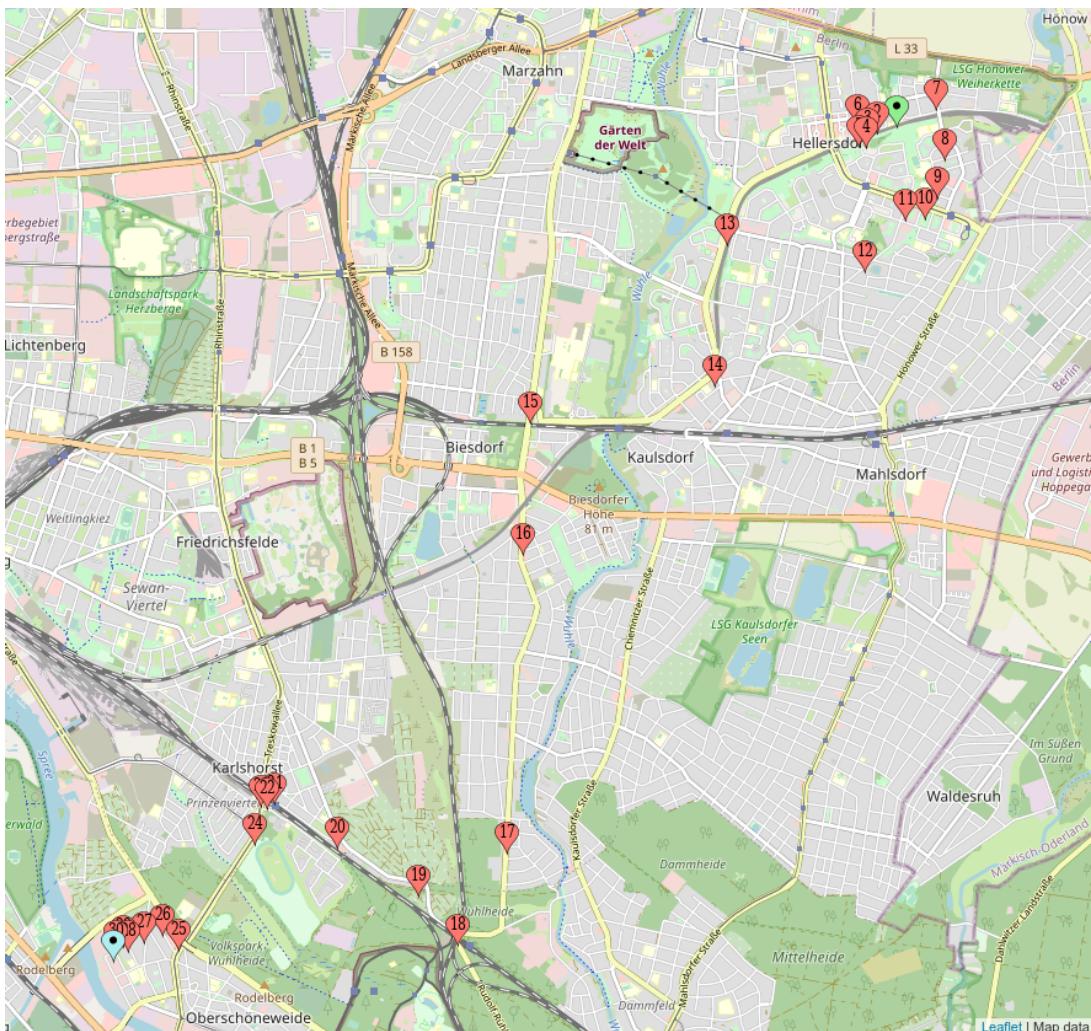


Bild 62: Visualisierung der rekonstruierten Strecke im Navigationssystem

Hierfür werden die jeweiligen Koordinaten-Paare als Punkte auf der Karte visualisiert. Dabei zeigt der grüne Punkt den Ort an, an dem die Fahrt begonnen wurde. In rot sind sämtliche Zwischenpunkte dargestellt, die von dem Navigationssystem automatisch gespeichert wurden. Das Ziel der Route ist mit einer blauen Markierung dargestellt.

## 4.6. VIRTUALISIERUNG

---

Somit kann die gesamte durch das Navigationssystem geführte Strecke rekonstruiert werden. Die rekonstruierte Fahrt liegt in der Stadt Berlin. Dies stützt die These, dass der Besitzer des IPHones aus dem Raum Berlin stammt.

### Personenbeziehbare Daten

Die letzte Kategorie extrahierter Daten bilden die personenbeziehbaren Daten. In diese Kategorie fällt die extrahierte Medien-Datenbank. Sie beinhaltet einige Songs sowie weitere Informationen zu diesen. Beispielsweise sind das, das dazugehörige Album, das Albumcover, das Genre und die jeweiligen Interpreten.

Ein Ausschnitt der extrahierten Daten ist in dem folgenden **Bild 63** dargestellt.

Dateiname	Titel	Interpret	Album	Genre	Jahr
007-the_disco_boys_-_i_love_you_so-ministry.mp3	I Love You So	The Disco Boys	German ODC Top40	Pop	2007
01 - Step By Step.mp3	Step By Step	New Kids On The Block	Step By Step Hit Collection, Vol. 2: The Best	Pop Alternative Musik	1990
01 Hungry For Love.wma	Hungry For Love		Unbekanntes Album (16.03.2011 20:01:04)		2005
01 Titelnummer 1.wma	Titelnummer 1				
01. Modern Talking - Win The Race.mp3	Win The Race	Modern Talking	America	Pop	2001
02 - In The Heat Of The Night - Sandra.mp3	In The Heat Of The Night	Sandra	18 Greatest Hits	Pop	1992
02 Don't Take Away My Heart.wma	Don't Take Away My Heart	Modern Talking	2000: Year of the Dragon	Dance	2000
03 A Train To Nowhere.wma	A Train To Nowhere		Hit Collection, Vol. 2: The Best	Alternative Musik	2005
04 - Recognizer - (Musicfire.in).mp3	Recognizer - Musicfire.in	Daft Punk/Musicfire.in	Tron: Legacy [Original Motion Picture Soundtrack]	Electronic, Pop/ Rock/Stage	2010
05 05 05 Titel 5.wma			- Musicfir		
05 Spur 5.wma	Spur 5		Unbekanntes Album (07.06.2008 10:36:43)		
06 - Loreen - Sandra.mp3	Loreen	Sandra	18 Greatest Hits	Pop	1992
07 - Midnight Man - Sandra.mp3	Midnight Man	Sandra	18 Greatest Hits	Pop	1992
07-Too Young-Blue System.mp3	Too Young	Blue System	The 2nd Album - Body Heat Disco		1988

Bild 63: Aufbereitung der extrahierten Tupel aus der *Medien* Datenbank

Das sind Daten, die für sich alleinstehend keine große Aussagekraft haben, jedoch in ihrer Anzahl durchaus Rückschlüsse auf eine Person ziehen lassen. Hinzu kommt, dass auch in der Medien-Datenbank durchaus belastende Daten in Form von illegal beschaffter Musik oder Filmen ermittelt werden könnten.

## 4.6 Virtualisierung

Für die Virtualisierung des Infotainmentsystem wird, wie im Kapitel 3.4.6 beschrieben, die Software *QEMU* mit dem Kernelmodul *KVM* eingesetzt. Im Folgenden werden die Maßnahmen zur Vorbereitung der Virtualisierung ausgeführt.

## 4.6. VIRTUALISIERUNG

---

Im ersten Schritt wird kontrolliert, dass im BIOS die Funktion *Intel Virtual Manager* aktiviert wird. Anschließend wird geprüft, ob das vorliegende Ubuntu Betriebssystem Virtualisierung unterstützt. Dafür wird der folgende Befehl ausgeführt: `egrep -c '(vmx|svm)' /proc/cpuinfo`. Ist die Systemantwort größer als 0 wird Virtualisierung unterstützt.

Im nächsten Schritt wird geprüft, ob auch KVM Virtualisierung unterstützt wird, dafür wird der Befehl `sudo kvm-ok` abgesetzt. Wenn auch dies unterstützt wird, sieht das Ergebnis wie in dem folgenden Screenshot 64 aus.

```
user@forensics:~$ sudo kvm-ok
[sudo] password for user:
INFO: /dev/kvm exists
KVM acceleration can be used
```

Bild 64: Prüfung auf die Unterstützung von KVM Virtualisierung

Weiterhin wird mit dem Befehl `sudo systemctl status libvird` geprüft, ob auch der Virutalisierungs-Daemon läuft. In der Ausgabe sollte nun unter dem Punkt *Active: active (running)* stehen.

Zuletzt wird abgefragt, ob die KVM Module geladen wurden, dafür wird der Befehl `lsmod | grep -i kvm` abgesetzt. Das Ergebnis in der Ausgabe sollte wie folgt im Bild 65 aussehen.

```
user@forensics:~$ lsmod | grep -i kvm
kvm_intel           282624  0
kvm                 663552  1 kvm_intel
```

Bild 65: Prüfung der geladenen KVM Module

Das Infotainmentsystem soll direkt von dem externen Datenträger aus virtualisiert werden, auf dem das Infotainment-Image im Rohformat geschrieben wurde. Dafür wird dieser an die Forensik-Maschine angeschlossen. Eine Überprüfung der angeschlossenen Geräte zeigt, dass das Betriebssystem für den externen Datenträger die Kennung `/dev/sdb` vergeben hat.

Die KVM-Virtualisierung wird über die Kommandozeile mit dem Befehl `kvm [options] [disk_image]` gestartet. Der korrekte Befehl zum Starten der Virtualisierung lautet `kvm -hda /dev/sdb1 -hdb /dev/sdb2 -hdc /dev/sdb3`. Diese Schreibweise des Befehls bezieht sich auf das Virtualisieren mittels Festplatten-Images und ist die Kurzform eines längeren Befehls. Durch die Angabe `-hda` (Hard Disk 0) weiß KVM mit dem Befehl umzugehen.

## 4.6. VIRTUALISIERUNG

---

Jedoch wird unter der Verwendung dieses Befehls eine Warnung ausgegeben. Die Warnung weist auf die Funktionalität bezüglich der CPU hin. Die Lösung ist die Erweiterung des Befehls um die Optionen *-cpu host*. Darüber hinaus wird die Bootreihenfolge mit der Option *-boot c* spezifiziert, das *c* steht dabei für die Festplatte.

Die letzte Erweiterung des Befehls bezieht sich auf die RAM-Größe, die für die Virtualisierung bereitgestellt wird. Der Standardwert beträgt **512 Mega Byte (MB)**. Zur Sicherstellung der Funktionsweise und eine reibungslose Virtualisierung wird die Kapazität auf **2048 MB** erhöht.

Damit lautet der vollständige Befehl:

```
kvm -hda /dev/sdb1 -hdb /dev/sdb2 -hdc /dev/sdb3 -cpu host -m 2048  
-boot c.
```

Nach dem Absetzen des Befehls startet *QEMU* und arbeitet bis zum Auffinden des *QNX v1.2b Boot Loaders*. Allerdings wird auch mit diesem Befehl noch eine Warnung bezüglich der fehlenden Angabe des Format-Typs des Images ausgegeben.

Aus diesem Grund wird der Kurzform-Befehl ausformuliert und jeder einzelne Parameter manuell gesetzt. Damit lautet der Befehl wie folgt:

```
sudo kvm -drive format=raw,file=/dev/sdb1,index=0,media=disk -drive  
format=raw,file=/dev/sdb2,index=1,media=disk -drive format=raw,file=  
/dev/sdb3,index=2,media=disk -cpu host -boot c -m 2048.
```

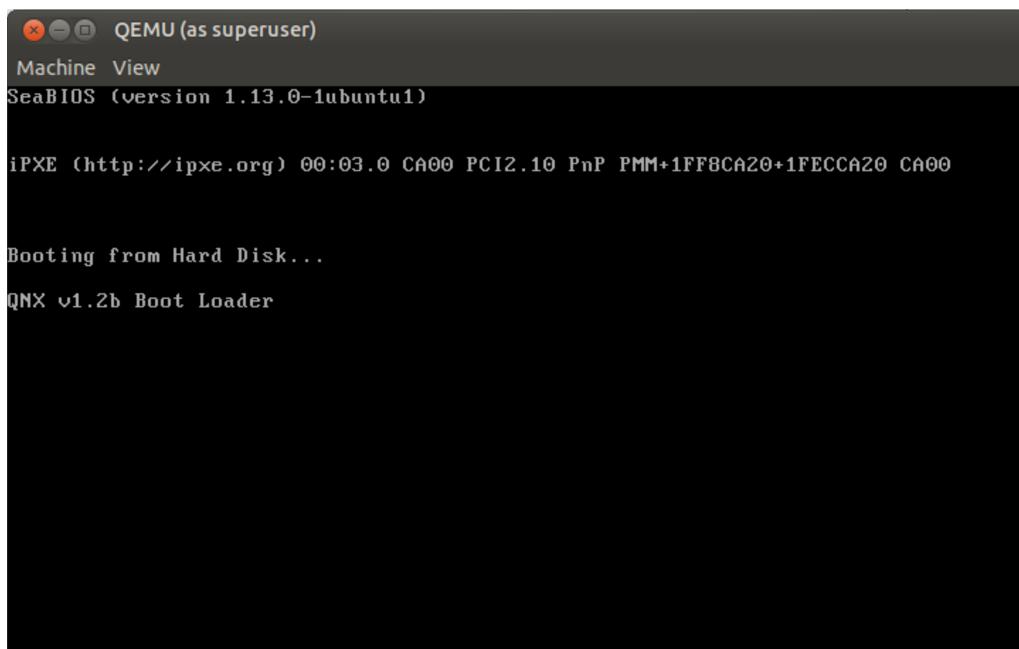


Bild 66: Virtualisierung des Infotainmentsystems

## 4.6. VIRTUALISIERUNG

---

Jetzt startet *QEMU* ohne Warnungen und Hinweise, aber wieder nur bis zum Auffinden des *QNX v1.2b Boot Loaders*. Der Bootvorgang selbst wird nicht beendet, die Virtualisierung bleibt an diesem Punkt hängen. Dieses Verhalten ist in dem Screenshot 66 dargestellt.

Für den zweiten Ansatz der Virtualisierung wird das Infotainment-Image aus dem *Expert Witness Disk Image Format* mit dem Kommandozeilen-Befehl `xmount -in ewf Infotainment.E0? -cache /tmp/info.ovl -out raw /ewf` in das DD-Format konvertiert. Anschließend wird versucht, das Infotainment-Image *Infotainment.dd* für die Virtualisierung zu verwenden. Leider schlägt auch dieser Versuch fehl. Mit diesem Ansatz werden weder der *QNX v1.2b Boot Loaders* noch ein Boot Device erkannt.

Für den dritten Ansatz wird das *Infotainment.dd* Image mit dem Befehl `dd if=/ewf/Infotainment.dd of=/tmp/infotainment.img` in das Image-Format *.img* geschrieben. Auch der Versuch das *infotainment.img* Image für die Virtualisierung zu verwenden schlägt fehl. Somit ist das selbe Ergebnis wie mit dem zweiten Ansatz erzielt worden.

Deshalb kann zum aktuellen Zeitpunkt keine Virtualisierung des Infotainmentsystems durchgeführt werden. Es kann festgehalten werden, dass *QEMU* das QNX Betriebssystem bzw. den Bootloader erkennt und damit auch ein startbares Gerät (Boot Device) gefunden wird. Lediglich der Startvorgang kann nicht sauber ausgeführt werden. Die Vermutung liegt nahe, dass das Problem das fehlende BSP ist.

Denn wie im Kapitel 4.5 beschrieben, sorgen diese dafür eine lauffähige Umgebung für das System vorzubereiten. Weiterhin beinhaltet das BSP den hardwarespezifischen IPL und den Startup-Code für den Boot-Vorgang. Ohne den Startup-Code und die lauffähige Umgebung wird eine Virtualisierung nicht möglich sein. Darüber hinaus muss das Betriebssystem-Image, das im lzo-Format vorliegt, für den Bootvorgang zuerst in den Arbeitsspeicher geladen und dort extrahiert werden, bevor es starten kann.

Dementsprechend müsste ein Image erstellt werden, indem das spezifische Hardware-Board emuliert und das BSP sowie auch das Betriebssystem QNX Neutrino RTOS installiert sind. Weiterhin sollte das Image nicht archiviert werden. Aus diesem Grund und aufgrund der begrenzten Zeit für die vorliegende Forschungsarbeit wird das Virtualisieren des Infotainmentsystems an dieser Stelle nicht weiter fortgeführt.

Dieses Kapitel zeigt wie komplex und aufwändig eine Virtualisierung sein kann. Jedoch bietet dieses Thema auch einen Ansatz für eine weitere Forschungsarbeit, die im Bereich der digitalen Forensik, der Virtualisierung von Embedded Systems und dem Betriebssystem QNX Neutrino RTOS liegt.

## 4.7 Physikalischer Analyseansatz

Dieses Unterkapitel beinhaltet die Umsetzung des im Kapitel 3.6 beschriebenen Konzeptes zur physikalischen Auswertung des QNX Neutrino RTOS. Dafür wird mit dem ersten Ansatz begonnen und die von QNX im Downloadbereich bereitgestellte Installations-CD in Form einer ISO-Datei heruntergeladen. Anschließend wird diese mit dem Befehl `dd if=/home/user/Downloads/qnx6.5.iso of=/dev/sdb bs=1 status=progress` auf einen USB-Stick geschrieben.

Nach dem Fertigstellen werden die BIOS Optionen und die Boot-Reihenfolge ansprechend angepasst, sodass der Computer in erster Instanz von dem USB-Stick startet. Die verwendete Forensik-Maschine erkennt jedoch kein startbares Medium. Das bedeutet, dass die verbaute Hardware nicht unterstützt wird. Da noch zwei weitere Computer zum Testen zur Verfügung stehen, werden auch deren BIOS-Optionen angepasst. Jedoch wird auch bei diesen Computern kein startbares Medium erkannt. Auch in diesem Falle ist die Hardware zu aktuell.

Damit ausgeschlossen werden kann, dass es nicht an dem USB-Stick liegt, wird die ISO-Datei auf eine DVD gebrannt. Auch diese wird an keinem der drei verfügbaren Computer erkannt. Dadurch kann ausgeschlossen werden, dass das Problem ursächlich in der Verwendung des USB-Sticks zu suchen ist.

Um einen Defekt der bereitgestellten ISO-Datei zu überprüfen, wird im *VMWare Player* eine VM mit einem Linux-Kernel der Version 4.X eingerichtet. Als startbares Medium wird die heruntergeladene ISO-Datei *qnx6.5.iso* angegeben. Das folgende Bild 67 zeigt, dass die ISO-Datei nicht beschädigt ist.



Bild 67: Installation von QNX 6.5 mit Installations-CD

Der Installations-Assistent startet und verlangt im ersten Schritt der Installation eine QNX-Lizenz für die Version 6.5. Damit ist die These bestätigt, dass die Hardware in den verfügbaren Computern zu aktuell ist und keine Unterstützung durch QNX 6.5 existiert.

Mehrere Versuche durch E-Mails den Support von QNX in Canada zu erreichen sind unbeantwortet geblieben, weshalb keine erforderliche Lizenz bereitgestellt wurde. Deshalb wird keine ältere Hardware beschafft und der beschriebene Ansatz verworfen.

Obwohl die Umsetzung des physikalischen Ansatzes an dieser Stelle aufgrund der fehlenden Lizenz abgebrochen werden muss, wird versucht das QNX Neutrino RTOS über die Installations-CD (Iso-Datei) zu starten. Das **Bild 68** zeigt die Desktopumgebung des QNX Neutrino Echtzeit-Betriebssystems in der Version 6.5.

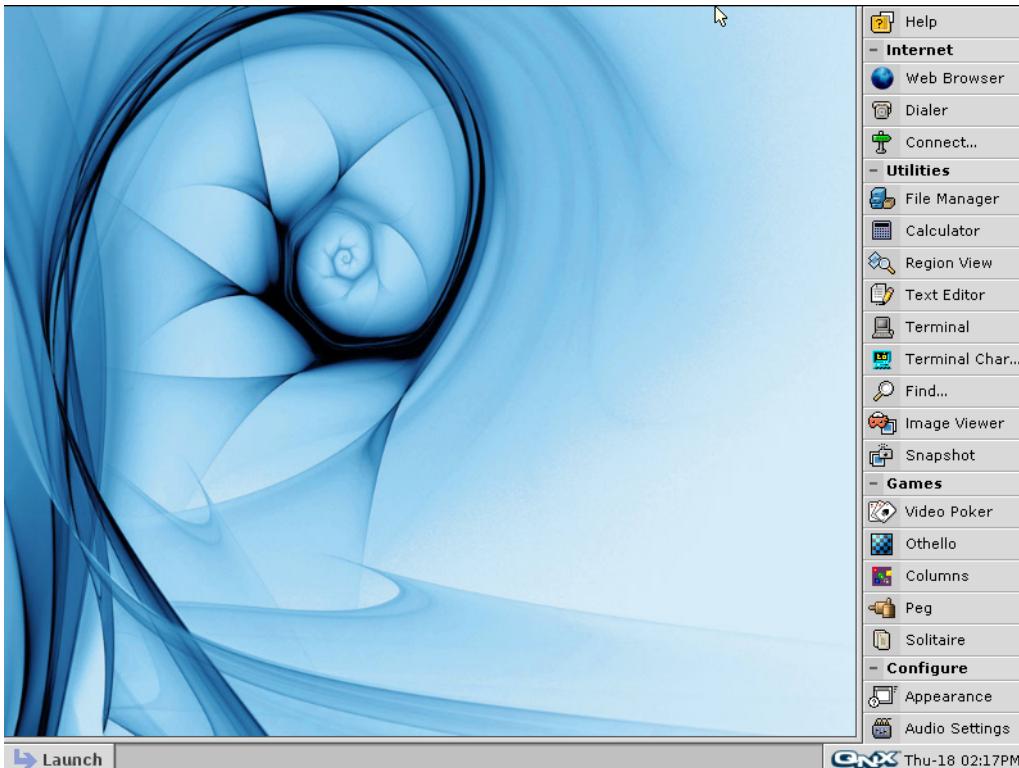


Bild 68: Desktopumgebung des QNX Neutrino RTOS 6.5

Für den zweiten Ansatz wird als Hardware der *Raspberry Pi 4B* mit der 4 GB Arbeitsspeicher Ausführung gewählt. Weil für den Raspberry Pi keine ISO-Datei bereitgestellt wird, muss das passende BSP heruntergeladen werden. Dafür muss während der Installation des QNX Software Centers ein Haken bei „Install experimental Packages“ gesetzt werden, andernfalls werden sämtliche Pakete, die sich noch im „experimental“ Status befinden, nicht zum Download bereitgestellt.

Nach dem Download des „QNX SDP 7.0 BSP for Raspberry Pi4 (raspberrypi-bcm2711-rpi4)“ BSPs wird die „QNX Momentics IDE 7.0“ aus dem QNX Software Center heruntergeladen und installiert. Anschließend muss entschieden werden, ob der Build-Prozess des BSPs über die Kommandozeile oder über die *QNX Momentics IDE 7.0* realisiert wer-

den soll. Zur Veranschaulichung der einzelnen Schritte und zur Verkürzung des gesamten Prozesses wird im Folgenden die *QNX Momentics IDE 7.0* verwendet.

### **Importvorgang und Projekterstellung**

Nach dem Start der *QNX Momentics IDE 7.0* kann der Importvorgang des BSPs über die Schaltfläche „File“ und dann „Import“ gestartet werden. Anschließend muss das „QNX“ Verzeichnis mittels Doppelklick erweitert werden, sodass die Option „QNX Source Package and BSP (archive)“ ausgewählt werden kann. Weil der auszuwählende Menü-Eintrag etwas versteckt ist, wird der gesuchte Menü-Eintrag in **Bild 69** gezeigt.

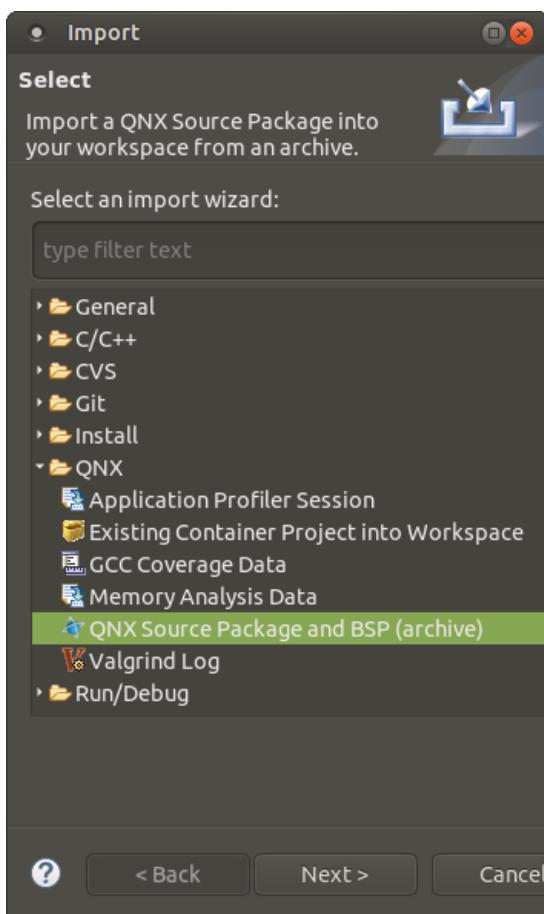


Bild 69: Importierung des BSPs in die *QNX Momentics IDE 7.0*

Weiter geht es über die „Next“-Schaltfläche. In einem weiteren Fenster öffnet sich der Datei-Explorer, damit wird das heruntergeladene BSP im ZIP-Format ausgewählt. Anschließend kann ein individueller Projektname und der gewünschte Quellpfad zu dem anzulegenden Projekt vergeben werden. Bestätigt werden diese Eingaben mit der „Finish“-Schaltfläche. Abschließend wird ein Projekt für dieses BSP von der *QNX Momentics IDE 7.0* angelegt und der Importvorgang abgeschlossen.

### **Build-Prozess**

Der folgende Abschnitt beschreibt den Build-Prozess des BSPs. Dieser beinhaltet den Build aus dem Quellcode wie auch das IFS-Image. Die Voraussetzung für den Build-Prozess ist ein erfolgreich importiertes BSP und das dazugehörige angelegte Projekt.

Gestartet wird der Build-Prozess mit einem rechten Mausklick auf das BSP-Quellprojekt innerhalb des Projekt-Explorers der *QNX Momentics IDE 7.0* und einem anschließenden linken Mausklick auf die Schaltfläche „Build Project“. Dadurch wird das lauffähige BSP erstellt, dass das startfähige IFS-Image und die für die Images erforderlichen Binärdateien enthält. Der Fortschritt während des Build-Prozesses und das Ergebnis des Builds kann in der Konsolenansicht der *QNX Momentics IDE 7.0* überwacht werden.

Verschiedenste Gründe können dafür sprechen an der Build-Datei Anpassungen vorzunehmen. So können der Build-Datei je nach Einsatzzweck des Betriebssystems verschiedene Bibliotheken, Programme und Treiber hinzugefügt oder entfernt werden. Darüber hinaus kann auch Einfluss auf die Konfiguration des QNX Neutrino RTOS genommen werden.

Generell sollte das Betriebssystem-Image an die vorgesehenen Bedingungen individualisiert und so schlank wie möglich gehalten werden. Das führt zu kürzeren Build-Zeiten, weniger Problemen während des Builds und größerer Robustheit im späteren Betrieb. Zusatzsoftware und Dateien sollten möglichst auf dem Dateisystem abgelegt und nicht in dem Betriebssystem-Image implementiert werden.

### **Vorbereitung der microSD-Karte**

Für die Installation und den Betrieb von QNX Neutrino RTOS auf dem Raspberry Pi 4B ist die Nutzung einer microSD-Karte der Klasse 10 (oder UHS-1) empfohlen. Weiterhin muss für die Vorbereitung der microSD darauf geachtet werden, dass sich diese nicht im schreibgeschützen Modus befindet.

Die folgende Aufzählung stellt eine Kurzanleitung zur Vorbereitung der startfähigen microSD-Karte dar. Dieses Vorhaben kann auf der Kommandozeile oder auch mit zusätzlicher Software wie beispielsweise „gparted“ durchgeführt werden.

1. Löschen der existierenden Partitionen auf der microSD.
2. Erstellen einer leeren DOS-Partitionstabelle.
3. Eine neue Partition erstellen:
  - Partitionstyp: Primary
  - Partitionsnummer: 1

Partitionsgröße: Wird nach Bedarf und Abhängigkeit der maximalen Speicherkapazität der microSD gewählt.

Dateisystem: FAT32

### 4. Die neue Partition auf die microSD schreiben.

Nachdem die neue Partition auf die microSD geschrieben wurde, muss die microSD wieder eingehängt werden.

## **Übertragung des QNX Neutrino RTOS Images und der Firmware-Dateien**

Für den ordnungsgemäßen Bootvorgang benötigt der Raspberry Pi 4 noch die folgenden Firmware-Dateien: *-rpi-4-b.dtb, start4.elf, start4db.elf, start4cd.elf, fixup4.dat, fixup4cd.dat, fixup4db.dat*. Die Firmware-Dateien stehen im „stable“ Branch des Raspberry Pi Projekts auf Github zum Download bereit.<sup>3</sup>

Nach dem Download werden diese in das Verzeichnis Projekt\_Quellverzeichnis/images/tools/sdboot\_images verschoben. Anschließend werden die Firmware-Dateien in das Wurzelverzeichnis der microSD kopiert. Zusätzlich muss das IFS-Image „ifs-rpi4.bin“ aus dem Verzeichnis Projekt\_Quellverzeichnis/images in das Wurzelverzeichnis der microSD kopiert werden. Der Name „ifs-rpi4.bin“ des IFS-Images darf nicht verändert werden, weil der Dateiname in der Konfigurationsdatei „config.txt“ fest definiert wird.

## **Startvorgang**

Für den Startvorgang des Raspberry Pi 4Bs muss die microSD-Karte in den Raspberry Pi eingesteckt werden. Die Übertragung der Konsole des QNX Neutrino RTOS erfolgt über die serielle RS-232 Schnittstelle. Diese Schnittstelle wird oftmals auch als COM-Schnittstelle bezeichnet.

Das bedeutet, dass für die Übertragung der Konsole ein RS-232 Adapter für den Raspberry Pi 4B beschafft werden muss. Der Adapter wird für die Versuche innerhalb dieser Arbeit mit sogenannten „Jumper“-Kabeln verbunden. Für einen dauerhaften Einsatz empfiehlt es sich den Adapter mit dem Raspberry Pi fest zu verbinden.

Das folgende **Bild** 70 zeigt eine Übersicht der GPIO<sup>4</sup> Pins des Raspberry Pi 4Bs.

---

<sup>3</sup><https://github.com/raspberrypi/firmware/tree/stable/boot>

<sup>4</sup>General-Purpose Input/Output Pins (GPIO)

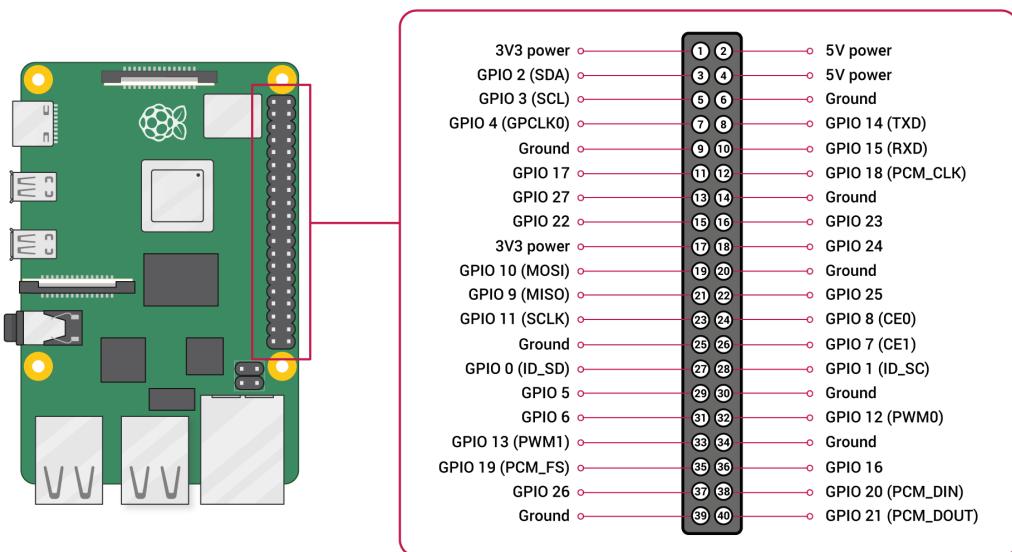


Bild 70: GPIO Pin Layout [68]

Für die serielle RS-232 Schnittstelle wird sowohl der „GPIO 14 (TXD)“ als auch der „GPIO 15 (RXD)“ Pin benötigt. Zusätzlich werden für den Adapter noch der 5V und der Masse (GND) Pin verwendet. Auf dem Adapter sind die jeweiligen Pins beschriftet und werden durch die Jumper-Kabel passend mit dem Raspberry Pi verbunden. Mit einem USB-zu-RS-232 Kabel wird der Adapter des Raspberry Pis an den USB-Port des Computers angeschlossen. Der Aufbau wird in dem folgenden **Foto 71** dargestellt.

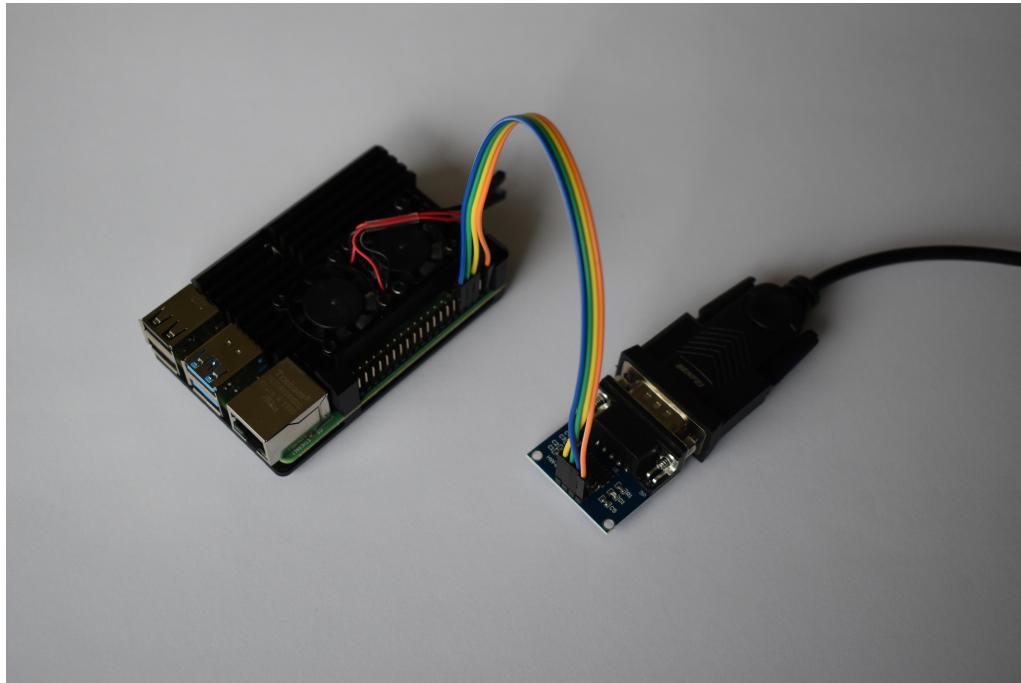


Bild 71: Verdrahtung des Raspberry Pis mit dem Adapter der seriellen RS-232 Schnittstelle

Im nächsten Schritt wird das Ethernet-Kabel in den Raspberry Pi eingesteckt. Jetzt kann der Raspberry Pi 4 über das USB-C Kabel mit Strom versorgt werden.

Der serielle Port des Computers wird mit dem Befehl `ls /dev/ttyUSB*` identifiziert. Für den Zugriff auf den COM-Port muss eine zusätzliche Terminal-Software „Minicom“ installiert werden. Mit dem Befehl `sudo minicom -s /dev/ttyUSB0` startet das Programm. Die notwendigen Einstellungsparameter lauten:

- Baud rate: 115200
- Data: 8 bit
- Stop: 1 bit
- Parity: none
- Flow control: none

Wenn der gesamte Build- und Übertragungs-Prozess erfolgreich verläuft und die Verdrahtung des Adapters ordnungsgemäß durchgeführt wurde, zeigt das Minicom-Terminal die Ausgabe des Raspberry Pis, wie in dem folgenden **Bild 72** dargestellt.

## 4.7. PHYSIKALISCHER ANALYSEANSATZ

---

```
user@forensics: ~
File Edit View Search Terminal Help
MMU: 16-bit ASID 44-bit PA TCR_EL1=b5183519
GICv2: 256 interrupts
GICv2: routing SPIs to gic cpu 0
cpu0: MPIDR=80000000
cpu0: MIDR=410fd083 Cortex-A72 r0p3
cpu0: CWG=4 ERG=4 Dminline=4 Iminline=4 PIPT
cpu0: CLIDR=a200023 LoUU=1 LoC=2 LoUIS=1
cpu0: L1 Icache 48K linesz=64 set/way=256/3
cpu0: L1 Dcache 32K linesz=64 set/way=256/2
cpu0: L2 Unified 1024K linesz=64 set/way=1024/16
cpu0: GICv2 cpu interface 0
Loading IFS...done
cpu1: MPIDR=80000001
cpu1: MIDR=410fd083 Cortex-A72 r0p3
cpu1: CWG=4 ERG=4 Dminline=4 Iminline=4 PIPT
cpu1: CLIDR=a200023 LoUU=1 LoC=2 LoUIS=1
cpu1: L1 Icache 48K linesz=64 set/way=256/3
cpu1: L1 Dcache 32K linesz=64 set/way=256/2
cpu1: L2 Unified 1024K linesz=64 set/way=1024/16
cpu1: GICv2 cpu interface 1
cpu2: MPIDR=80000002
cpu2: MIDR=410fd083 Cortex-A72 r0p3
cpu2: CWG=4 ERG=4 Dminline=4 Iminline=4 PIPT
cpu2: CLIDR=a200023 LoUU=1 LoC=2 LoUIS=1
cpu2: L1 Icache 48K linesz=64 set/way=256/3
cpu2: L1 Dcache 32K linesz=64 set/way=256/2
cpu2: L2 Unified 1024K linesz=64 set/way=1024/16
cpu2: GICv2 cpu interface 2
cpu3: MPIDR=80000003
cpu3: MIDR=410fd083 Cortex-A72 r0p3
cpu3: CWG=4 ERG=4 Dminline=4 Iminline=4 PIPT
cpu3: CLIDR=a200023 LoUU=1 LoC=2 LoUIS=1
cpu3: L1 Icache 48K linesz=64 set/way=256/3
cpu3: L1 Dcache 32K linesz=64 set/way=256/2
cpu3: L2 Unified 1024K linesz=64 set/way=1024/16
cpu3: GICv2 cpu interface 3

System page at phys:000000000012000 user:fffffff8040206000 kern:fffffff8040204000
Starting next program at vfffffff8060059ae0
MMFLAGS=1
All ClockCycles offsets within tolerance
Welcome to QNX Neutrino on the Raspberry Pi 4 (Armv8-A Cortex-A72 core)
Starting watchdog...
Starting random service...
Starting serial driver (/dev/ser1)
Starting serial driver (/dev/ser3)
Starting I2C driver (/dev/i2c1)...
Starting SPI 0 driver (/dev/spi0)...
Starting SDMMC driver for SD card (SDIO0)...
Starting PCI Server...
Starting USB xHCI controller in the host mode (/dev/usb/*)...
Path=0 - bcn2711
    target=0 lun=0 Direct-Access(0) - SDMMC: SC32G Rev: e.6
Starting Network driver...
Starting Audio driver...
#
```

CTRL-A Z for help | 115200 8N1 | NOR | Minicom 2.7.1 | VT102 | Offline | ttyUSB0

Bild 72: Ausgabe des Minicom-Terminals

Zwischen den Versionen QNX Neutrino 6.5 und QNX Neutrino 7.0 hat eine Neuausrichtung des Betriebssystems stattgefunden. Ein Vergleich der Bilder 68 und 72 veranschaulicht die Neuausrichtung des QNX Neutrino RTOS sehr gut. Blackberry hat sich mit dem QNX Neutrino RTOS 7.0 vollständig auf embedded systems und 1-Kern Prozessoren fokussiert. In Zuge dessen wurde die Desktopumgebung entfernt und somit der eigens entwickelte Browser, Datei-Explorer und viele weitere Software, Dienste und Treiber, die

für die Implementierung der Desktopumgebung notwendig waren. Diese Entwicklung zeigt, dass Blackberry das QNX Neutrino RTOS nicht mehr als Desktop-Betriebssystem vorsieht.

## **4.8 Vergleich Standard-Distribution vs QNX Neutrino RTOS**

Damit ein zielgerichteter und substanzialer Vergleich zwischen den Linux Desktop Distributionen und dem QNX Neutrino RTOS aufgestellt werden kann, müssen Rahmenbedingungen und Bewertungskriterien festgelegt werden. Darüber hinaus müssen die gewählten Aspekte nachvollziehbar und quantifizierbar sein.

Systeme können auf unterschiedlichste Weise miteinander verglichen werden, hierbei kommt dem verfolgten Ziel des Vergleichs besondere Bedeutung zu. Das Ziel dieses Vergleichs ist es festzustellen, inwiefern ein Forensiker von dem Einsatz des QNX Neutrino RTOS in Bezug auf die forensische Untersuchung von QNX Betriebssystemen profitieren könnte. Trotzdem sollen auch Parameter wie beispielweise die Kosten, Einarbeitung und weitere Einsatzmöglichkeiten für dieses Betriebssystem mit in die Bewertung eingehen.

Für eine Gegenüberstellung der verschiedenen Betriebssysteme wird die Methode der Nutzwertanalyse angewendet. Dafür werden die Bewertungskriterien erarbeitet und gewichtet. Für das Bewertungsschema der Nutzwertanalyse wird auf das in Kapitel 3.2.1 vorgestellte Bewertungsschema zurückgegriffen. Darüber hinaus gilt die Invertierung des Bewertungsschemas für die Kriterien Kosten und Wartungsaufwand.

### **Bewertungskriterien und dessen Gewichtung**

Für die möglichst objektive Bewertung der Betriebssysteme werden die folgenden Bewertungskriterien zugrunde gelegt.

Speziell für den gewerblichen Einsatz von proprietärer Software ist auf die Lizenzkosten zu achten. Diese sind mit 5% in der Nutzwertanalyse gewichtet.

Ebenfalls nicht zu vernachlässigen ist die Wartbarkeit des eingesetzten Betriebssystems. Software bedarf regelmäßiger Updates. Wird diese aus vielen verschiedenen Quellen installiert, steigt der Wartungsaufwand für das eingesetzte Betriebssystem erheblich. Deshalb wird der Wartungsaufwand mit 15% gewichtet.

Darüber hinaus ist auch die Anpassungsfähigkeit und das Umfeld der Einsatzmöglichkeiten ein starkes Kriterium. Ist dies durch geringen Aufwand mit weiterer Software er-

#### 4.8. VERGLEICH STANDARD-DISTRIBUTION VS QNX NEUTRINO RTOS

weiterbar, kann ein Betriebssystem für viele verschiedene Anwendungen Einsatz finden. Damit liegt die Gewichtung dieses Kriteriums bei 15% der Gesamtbewertung.

Hilfestellungen werden nicht immer von dem geschulten Personal des jeweiligen Softwareherstellers geleistet, sondern viel mehr behelfen sich die meisten Personen mit Hilfestellungen aus Foren, Dokumentationen, Blogs, Büchern und weiterer verschiedener Quellen. Auch der Aspekt der Selbsthilfe ist nicht zu vernachlässigen. Neben den externen Hilfestellungen spielt auch die Ausführlichkeit der Dokumentation eine große Rolle. Weil diesem Kriterium eine besondere Rolle zukommt, fließt die Gewichtung der Dokumentation mit 25% in die Bewertung ein.

Das letzte und wichtigste Kriterium in Bezug auf forensische Untersuchung von QNX Betriebssystemen ist die Auswertbarkeit von QNX Systemen. Aus diesem Grund wird dieses Kriterium mit 40% in der Gesamtbewertung gewichtet.

#### Bewertung der Nutzwertanalyse

Kriterium	Gewichtung	Standard Linux Distro		QNX Neutrino RTOS	
		Punkte	Bewertung	Punkte	Bewertung
Auswertbarkeit QNX	40 %	3	1,2	10	4
Selbsthilfe	25 %	10	2,5	4	1
Anpassungsfähigkeit	15 %	8	1,2	3	0,45
Wartungsaufwand	15 %	3	0,45	10	1,5
Kosten	5 %	8	0,8	3	1
<b>Ergebnis</b>	100 %		6,15		7,95

Tabelle 9: Bewertungsmatrix Nutzwertanalyse des Vergleichs von Standard Linux Distributionen und QNX Neutrino RTOS

Das beste Ergebnis erzielt das QNX Neutrino RTOS mit 7,95 von möglichen 10 Punkten. Mit einer QNX-VM oder einer nativen Installation auf einem externen Datenträger können sämtliche QNX Betriebssysteme forensisch untersucht werden. Dazu zählen nicht nur Infotainmentsysteme von PKWs, sondern auch medizinische Geräte, Systeme der Luftfahrt, des Militärs und viele andere Systeme, in denen ein QNX Betriebssystem läuft.

Negativ zu betrachten ist bei dem Einsatz von einem QNX Neutrino RTOS, dass dieses System nur mit erheblichem Aufwand für weitere Aufgaben erweitert werden kann. Das liegt an der fehlenden Paketverwaltung für die komfortable Installation weiterer Software,

sowie an der fehlenden **Graphical User Interface (GUI)**<sup>5</sup> für das Ausführen von Forensik Software wie beispielsweise *Autopsy*. Weiterhin fehlen sämtliche andere Bibliotheken und Programme die das komfortable Arbeiten mit Linux Betriebssystemen ermöglichen.

Darüber hinaus entstehen spätestens für den gewerblichen Einsatz Lizenzkosten. Die Dokumentation des QNX Neutrino RTOS ist längst nicht so ausführlich und detailliert wie die der meisten nicht proprietären Linux Distributionen. Zusätzlich existiert lediglich ein einziges QNX-Forum für die Informationsbeschaffung und Problemlösung durch andere Benutzer und Enthusiasten. Einzig positiv zum Thema der Hilfestellung und Informationsgewinnung im Vergleich mit den meisten freien Linux Distributionen ist der deutschsprachige Vertriebssupport zu nennen. Stets hinterließ dieser einen sehr professionellen und geschulten Eindruck. Darüber hinaus wurden Anliegen schnell und zielführend beantwortet.

Es ist nicht unmöglich andere Distributionen so anzupassen, dass sich wenigstens QNX-Dateisystem auslesen lassen. Jedoch steigt damit der Wartungsaufwand des eigenen Betriebssystems ungemein, da sämtliche Bibliotheken und Programme manuell aktuell gehalten werden müssen. Diese Lösungsvariante birgt dementsprechend Gefahren mit veralteten Technologien und Bibliotheken zu arbeiten, sodass eventuell nicht alle Spuren aufgefunden oder verfolgt werden können. Das QNX Neutrino RTOS kann bei einer neuen Version von der Hersteller-Webseite heruntergeladen und direkt eingesetzt werden, damit ist der Wartungsaufwand im Vergleich sehr gering.

Damit ist es für Personen, die sich regelmäßig mit derartigen Systemen befassen, nahezu unerlässlich sich ein derartiges Betriebssystem für forensische Arbeit bereitzuhalten. Der wichtigste Punkt jedoch ist die Möglichkeit vollumfängliche forensische Untersuchungen von Systemen mit dem Betriebssystem QNX Neutrino RTOS durchführen zu können.

Wie diese Forschungsarbeit zeigt, ist eine vollständige und vollumfängliche forensische Untersuchung eines QNX Neutrino RTOS Systems erst möglich, wenn auch für die Untersuchung selbst ein QNX Neutrino RTOS System eingesetzt wird. Die Ausnahme dabei bildet die Möglichkeit das eigene System mit einem sehr großen einmaligen Aufwand für die Ersteinrichtung zu erweitern, verbunden mit dem regelmäßigen Aufwand der Aktualisierung der Software. Trotz des großen Aufwandes kann der Fall eintreten, dass einige Bibliotheken oder Zugriffsmöglichkeiten für andere Linux Distributionen und damit auch einige Funktionen nicht zur Verfügung stehen.

---

<sup>5</sup>Grafische Benutzerschnittstelle

## 4.9 Gefahrenanalyse und Sicherheitsbetrachtung

Abschließend folgt eine Betrachtung ausgewählter Angriffsvektoren und Möglichkeiten, die durch die Ergebnisse der vorliegenden Masterthesis festgestellt wurden. Unter den verschiedenen Akteuren in Bezug auf das vernetzte Fahrzeug nimmt der Fahrzeugherrsteller eine zentrale Rolle ein. Die Hersteller-Cloud besitzt 24 Stunden 7 Tage die Woche eine dauerhafte Verbindung zu den einzelnen Fahrzeugen. Die durch die Nutzung des Infotainmentsystems entstehenden Metadaten und die personenbezogenen Daten kann der Hersteller analysieren, um die eigenen Applikationen und Dienste auf dem Infotainmentssystem zu optimieren. Zusätzlich könnten diese Daten für die Schaltung personalisierter Werbung und zur Ausrichtung der eigenen Werbung bezogen auf die verschiedenen Zielgruppen genutzt werden. Darüber hinaus können die fahrzeugbezogenen Daten, die über die Sensorik des Fahrzeugs generiert werden, dazu beitragen Langzeittests und -auswertungen für die Optimierung von verbauten Komponenten zu erstellen. Denkbar wäre auch, dass Hersteller die Daten und Analysen für die Gewinnmaximierung an Dritte verkaufen.

Die sozialen Netzwerke, Streaming-Dienste und weitere Drittanbieter haben durch die vernetzten Fahrzeuge und die damit einhergehende Internetverbindung und Konnektivität zu weiteren mobilen Geräten eine neue Quelle zur Sammlung und Analyse von Metadaten. Denkbare Metadaten wären zum Beispiel die IP-Adresse des Fahrzeugs, das verwendete Betriebssystem, die Bildschirmauflösung, der Aufenthaltsort durch GPS-Tags an Beiträgen im sozialen Netzwerk und viele weitere Metadaten, die Protokollbedingt oder zu Analysezwecken übertragen werden. Darüber hinaus können sämtliche, aus dem Umgang mit dem Internet, bekannte Techniken und Vorgehensweisen zur Überwachung, Verfolgung und Analyse, analog auf das Infotainmentsystem übertragen werden.

Vollzugsbehörden können ebenfalls von diesen Daten profitieren. In einem PKW-Diebstahl Delikt könnte über die GPS-Ortung der aktuelle Aufenthaltsort von Straftätern bzw. des Fahrzeugs ermittelt werden. Die GPS-Ortung könnte ebenfalls dazu genutzt werden, Gefährder über längere Zeiträume zu überwachen, um Aufenthalts- und Wohnort, Kontakte, Verhaltensweisen und Tagesabläufe zu analysieren. Zusätzlich können die auf dem Infotainmentsystem befindlichen personenbezogenen Daten genutzt werden, um Verdachte zu erhärten oder auszuschließen. Wenn illegal beschaffte Daten, zum Beispiel in Form von Filmen, auf dem Infotainmentsystem ermittelt werden, können diese genutzt werden, um strafrechtlich gegen diese Person vorzugehen.

Auch für Versicherungsgesellschaften sind die durch das Fahrzeug und durch die Interaktion mit dem Fahrzeug generierten Daten sehr interessant. Die Daten können den Versicherungsgutachter bei dem Erstellen von Gutachten zu Verkehrsunfällen unterstützen und

das Ergebnis maßgeblich beeinflussen. Anhand der fahrzeugbezogenen Daten kann das Fahrverhalten vor dem Unfall und im Zeitraum des Unfalls untersucht werden und daraus Schlüsse zur mentalen und psychischen Verfassung des Fahrers gezogen werden. Darüber hinaus kann mit Hilfe der Zeitstempel sämtlicher erfasster Daten eine Rekonstruktion der letzten Aktionen und Reaktionen des Fahrers, sowie Interaktionen des Fahrers oder der Insassen erstellt werden. Gegebenenfalls kann dadurch festgestellt werden, wer der Schuldige an dem vorliegenden Verkehrsunfall ist, indem beispielsweise nachgewiesen wird, dass der Fahrer zum Zeitpunkt des Unfalls damit beschäftigt war eine E-Mail zu schreiben.

Zuletzt soll bewertet werden, welche Risiken und Gefahrenpotentiale durch den Eingriff eines Angreifers mit böswilligen Absichten entstehen können. Bei einem Angreifer handelt es sich nicht zwingend um einen böswilligen Hacker mit krimineller Energie. Als Angreifer können viele verschiedene Personen agieren: Beispielsweise der enttäuschte Expartner, Familienangehörige, der Werkstatt-Mitarbeiter, der Fahrzeughändler, der Hersteller oder auch der Mitwagenvermittler.

Bei den verschiedenen Angriffsvektoren muss zwischen physischen Zugriffen auf das Infotainmentsystem und Zugriffen via digitaler Verbindungen (WLAN, Bluetooth, Internet) unterschieden werden. Hat ein Angreifer physischen Zugriff auf das Fahrzeug, ist das Infotainmentsystem für jemanden geübtes mit wenigen Handgriffen ausgebaut und damit der Zugriff auf das Speichermedium hergestellt. So könnten beispielsweise geleaste Fahrzeuge, die zurückgegeben werden, auf Firmengeheimnisse untersucht und im Kontext der Wirtschaftsspionage an den Meistbietenden verkauft werden. Durch die Analyse der personenbezogenen Daten des Infotainmentsystems wären auch Erpressungsversuche dem ehemaligen Fahrer gegenüber oder ein Verkauf der personenbezogenen Daten möglich. Mit physischem Zugriff auf den Datenträger und Umgehen des Schreibschutzes ist es ebenfalls möglich Schadcode und Malware in das System zu indizieren. Auch hier können sämtliche, aus dem Umgang mit dem Internet, bekannte Techniken und Vorgehensweisen zur Überwachung, Verfolgung und Analyse, analog auf das Infotainmentsystem übertragen werden. Das kann dazu führen, dass der Fahrer unwissentlich überwacht wird, es könnte ein Keylogger installiert werden, um Passwörter abzugreifen oder ein Krypto-Trojaner installiert werden. Dabei sind die Möglichkeiten beinahe grenzlos.

Wie in allen anderen Computersystemen kann auch von außerhalb Zugriff auf das System erlangt werden. Allgemein kann festgehalten werden, dass durch die vielen neuen Verbindungsarten, Systeme, Softwarebestandteile und die Vernetzung der Fahrzeuge mit der Umgebung, anderen Fahrzeugen und der Interaktion von Menschen, eine sehr große Anzahl an Parametern und potenzielle Sicherheitsrisiken zusammentreffen. Weil der Ver-

kehrs bereich generell ein sehr sensibler Bereich ist, ist gerade im Bereich der Vernetzung von Fahrzeugen besonders auf die IT-Sicherheit zu achten. Denn im schlimmsten Fall erlangt ein Angreifer über das Infotainmentsystem Zugriff auf die Steuergeräte des Fahrzeugs, sodass Signale der Sensoren manipuliert oder unterbunden werden können. Dadurch wäre es denkbar, dass ein Angreifer über das Internet von zu Hause aus die Kontrolle über ein Fahrzeug übernimmt. Sind diese beiden Systeme nicht ordentlich von einander entkoppelt, könnten daraus dramatische Unfälle resultieren.

Schlussendlich verdeutlicht dieses Worst-Case-Szenario sehr stark, warum ein sehr hohes Maß an Sensibilität für die IT-Sicherheit in der Entwicklung von IT-Systemen für Fahrzeuge und die Herstellung der Fahrzeuge erforderlich ist. Besonders in Bezug auf Schwachstellen in der Software sollten strenge Testverfahren und Kontrollen, sowie regelmäßige Update-Zyklen für die Software der Fahrzeuge implementiert werden. Darüber hinaus sollte die Integrität der einzelnen Computersysteme des Fahrzeugs über die gesamte Lieferkette sichergestellt und verfolgt werden.

Sehr positiv zu bewerten ist, dass Blackberry innerhalb ihrer QNX Betriebssysteme sehr viele Sicherheitsfunktionen und Schutzmechanismen vorsieht und implementiert. Jedoch sind diese automatisch aktiviert und vorkonfiguriert, sondern müssen von den Entwicklern bewusst im Entwicklungsprozess an die vorherrschenden Gegebenheiten angepasst, konfiguriert und aktiviert werden.

Deshalb kann abschließend festgehalten werden, dass die endgültige IT-Sicherheit der Fahrzeuge von den Entwicklern und Experten für eben diese Systeme und Applikationen, wie auch bei vielen anderen Computersystemen, abhängt. Die besten Sicherheitsvorkehrungen sind nutzlos, wenn diese falsch konfiguriert oder erst gar nicht implementiert und aktiviert werden.

# 5 Fazit & Ausblick

## 5.1 Fazit

Die im Rahmen der vorliegenden Arbeit durchgeführte forensische Untersuchung des Infotainmentsystems hat zahlreiche Charakteristika des Blackberry QNX Neutrino Realtime Operation System aufgedeckt und viele neue Erkenntnisse erbracht.

Zu Beginn der Konzeptphase konnten nur wenige Anforderungen an die Vorgehensweise und das Vorgehensmodell der forensischen Untersuchung des Infotainment-Images gestellt werden. Weil zur Zeit keine Forschungsarbeiten zur forensischen Untersuchung von Infotainmentsystemen mit dem QNX Neutrino RTOS existieren, konnte das geplante Vorgehen noch nicht detailliert ausformuliert werden. Aus diesem Grund wurde ein besonderes Augenmerk auf die Flexibilität des Vorgehensmodells und eine experimentelle Vorgehensweise für die manuelle Analyse festgelegt. Rückblickend war die Wahl des *SAP-Modells* eine perfekte Entscheidung, da es sich als flexibel und anpassbar erweist. Das agile Vorgehen hat den Dokumentationsaufwand und aufwändige Iterationszyklen deutlich verkürzt und vereinfacht. Diese Erfahrung bestätigt, weswegen das Vorgehensmodell in der Praxis häufig Anwendung findet und in der Literatur für Forschungsansätze empfohlen wird.

Die Softwareauswahl hat sich ebenfalls als sehr positiv, wenn auch nicht als ausreichend, bewiesen. Mit der im Konzept geplanten Softwareauswahl konnte keine vollumfängliche forensische Untersuchung durchgeführt werden. Erst mit Beginn der Untersuchung des Infotainment-Images ist aufgefallen, dass lediglich drei von insgesamt 13 Partitionen auszuwerten waren. Die für die Funktion eines Infotainmentsystems notwendigen Programme, Bibliotheken und Daten waren nicht auffindbar. Zusätzlich haben die Partitionsgrößen nicht die Gesamtgröße des Images widergespiegelt.

Mit dieser Erkenntnis und der weiteren Konzeptentwicklung für die Erzwingung des Schreibzugriffs wurde die Bedeutung und Notwendigkeit des Einsatzes einer QNX-VM vollkommen unerlässlich. Die dafür erforderliche Beschaffung einer Lizenz hat sich ebenfalls als eine unerwartete Hürde herausgestellt. Die Kontaktaufnahme mit Blackberry QNX hat sich als äußerst schwierig erwiesen, da diese auf E-Mails nicht antworteten, sodass schlussendlich keine Kommunikation mit den Mitarbeitern in Kanada stattgefunden hat. Dahingegen ist der deutsche Vertrieb für QNX-Systeme sehr hilfsbereit gewesen und hat maßgebend dazu beigetragen, dass eine Lizenz für QNX und damit auch die QNX VM zur Verfügung gestellt wurde.

## 5.1. FAZIT

---

Die Literaturrecherche der Spezifikationen von *Autopsy* und weiterer Forensik Softwares ergab, dass eine Unterstützung von QNX Systemen nicht vorhanden ist. Wie erwartet war eine automatisierte forensische Untersuchung mittels *Autopsy* wenig aufschlussreich. Die Erwartungshaltung lag darin, einen groben Überblick über das vorhandene System und die darin gespeicherten Daten zu erlangen. Jedoch wurde der Analysevorgang und damit auch das File Carving schon bei dem Einlesen des Images abgebrochen, sodass keine Resultate erzielt werden konnten.

Dahingegen hat die anschließende manuelle logische forensische Untersuchung mit dem *The Sleuth Kit* schon mehr Ergebnisse erzielt. In dem allozierten Bereich des Infotainment-Images wurden auf insgesamt drei verschiedenen Partitionen personenbezogene Daten in Form von SQLite Datenbanken und weitere Systemdateien und das Betriebssystem selbst ermittelt. Die Untersuchung des nicht-allozierten Bereichs hat weitere spannende Resultate geliefert. So konnten mit der Analyse der E-Mail Merkmale weiterer Verbindungen zwischen dem Image und dem Hersteller Volkswagen und sogar E-Mail-Adressen einiger Mitarbeiter ermittelt werden. Darüber hinaus konnten personenbeziehbare Daten innerhalb einer Medien-Datenbank wiederhergestellt werden.

Zum einen ist die Rekonstruktion der beschädigten SQLite Datenbanken möglich und zum anderen wird im Kapitel 5 gezeigt, dass auch diese Datenbanken interessante Daten beinhalten können. Darüber hinaus wird gezeigt, dass mit einem ausgewählten Rekonstruktionsverfahren nicht sämtliche beschädigte SQLite Datenbanken rekonstruiert werden können, sodass im Zweifelsfall verschiedene Techniken und Programme angewendet werden müssen, um diese Daten wiederherzustellen, damit sie bestmöglich untersucht werden können.

Entgegen der Erwartung stellt die vollumfängliche logische forensische Auswertung der einzelnen Partitionen des Infotainment-Images eine große Hürde dar, die mit Linux-Boardmitteln nicht umsetzbar war. Dies erforderte weitere Literaturrecherche der Dokumentation, sowie Kreativität und viele Versuche im Bereich des Mountens von QNX 6 beziehungsweise QNX Power-Safe POSIX Partitionen. Erst durch die in dem Kapitel 4.3 erarbeitete Vorgehensweise ist eine vollumfängliche forensische Untersuchung aller dreizehn Partitionen des Infotainmentsystems ermöglicht worden. Die weiterführende Analyse erbrachte die notwendigen Erkenntnisse zum Verständnis der Zusammenhänge zwischen der installierten Software, der auf dem Infotainmentsystem gespeicherten Daten und der Funktionsweise des Betriebssystems.

Bezogen auf das ausgewählte Vorgehensmodell wurde die Zusammenfassung in der Present Phase in Bezug auf ihren Mehrwert unterschätzt. Erst nach der vollständigen forensischen Untersuchung beim Zusammenfassen sämtlicher ermittelter Ergebnisse ist das

## 5.2. AUSBLICK

---

Bewusstsein dafür entstanden, dass durch die Dokumentation der zusammengefassten Ergebnisse das Verständnis für das Thema der vorliegenden Arbeit vertieft wurde. Dementsprechend wird der Present Phase in zukünftigen forensischen Untersuchungen von Beginn an eine sehr große Bedeutung und Sorgfalt bei der Dokumentation zugesprochen.

Auch die Virtualisierung stellt eine bisher nicht gelöste Hürde dar. Aufgrund der Funktionsweise des Betriebssystems gibt es einige Parameter die verantwortlich dafür sein können, dass die Virtualisierung des vorhandenen Infotainmentsystems mit aktuellen Techniken nicht möglich ist.

Der Vergleich zwischen Standard-Linux Distributionen und dem QNX Neutrino RTOS, in Bezug auf die forensische Auswertung von QNX Systemen, ist von den in der Analyse Phase gewonnenen Erfahrungen stark geprägt. Wie im Verlauf der forensischen Untersuchung erarbeitet wurde, ist ein QNX Neutrino RTOS zur vollständigen forensischen Untersuchung von QNX Systemen unerlässlich. Jedoch liegen die Ergebnisse der Nutzwertanalyse näher beieinander als erwartet, sodass zwar die klare Empfehlung ausgesprochen wird ein QNX Neutrino RTOS System auf dem Raspberry Pi 4B oder eine QNX VM zur forensischen Untersuchung von QNX Systemen zu nutzen, aber auch die Erweiterung der eigenen Forensik-Distribution durchaus eine Möglichkeit darstellt.

Wird die Aufgabenstellung mit der Zielsetzung verglichen, kann mit Bestimmtheit festgehalten werden, dass die geforderten Ergebnisse erreicht wurden. Auch die auf Basis dieser Arbeit gewonnenen Erkenntnisse in Bezug auf die forensische Auswertung gängiger Infotainmentsysteme spiegeln einen erfolgreichen Projektverlauf wider. So konnten einige besonders schützenswerte personenbezogene Daten unverschlüsselt auf dem Datenträger des Infotainmentsystems ermittelt werden.

Für die Forschung im Bereich der automotiven Forensik stellt diese Arbeit eine sehr umfängliche Basis dar, in der die allgemeinen Funktionsweisen des Betriebssystems, der Dateisysteme und des Infotainmentsystems wissenschaftlich beschrieben sind. Darüber hinaus werden die Speicherorte und -formate für die wichtigsten Dateien aufgezeigt. Durch die erarbeiteten Erkenntnisse und Ergebnisse dieser Masterthesis besteht eine Basis auf die zukünftige Forschungsarbeiten im Bereich der automotiven Forensik aufbauen können.

## 5.2 Ausblick

Das QNX Neutrino RTOS wird weltweit bei mehr als 40 Automobilherstellern für Infotainmentsysteme eingesetzt. Das bedeutet, dass weltweit mehrere Millionen PKWs mit diesem ausgerüstet sind. Zwar gibt es einige Bestrebungen der Automobilhersteller ihre

## 5.2. AUSBLICK

---

eigenen Betriebssysteme für das Fahrzeug und das Infotainment zu entwickeln, jedoch werden zur Umsetzung noch einige Jahre vergehen. Zusätzlich werden derzeit aktuelle Fahrzeuge bis zur Verschrottung weiterhin mit den QNX System ausgestattet bleiben, wodurch auch in zehn Jahren noch viele Millionen PKWs mit diesen Systemen in den Städten und auf den Autobahnen unterwegs sein werden. Dadurch bleibt das Thema QNX in Infotainmentsystemen in den nächsten zehn Jahren mit Sicherheit in der automotiven Forensik stark vertreten. Deshalb lohnt es sich für die Forschung durchaus, sich weiter mit diesem Thema zu beschäftigen.

An einigen Punkten ist diese Arbeit an die Grenzen der aktuellen Möglichkeiten und Techniken gestoßen. Beispielsweise ist die Virtualisierung des in dieser Arbeit behandelten Infotainment-Images aktuell noch nicht möglich. Eine Virtualisierung des untersuchten Systems ist nicht zwangsläufig notwendig, jedoch ist dies speziell im Strafvollzug ein sehr beliebtes Mittel, um aufgefundene Daten und daraus gezogene Erkenntnisse zu visualisieren. Dadurch werden komplexe Zusammenhänge überschaubar und für den Laien verständlich dargestellt. Damit bildet das Thema der Virtualisierung von Infotainmentsystemen einen wichtigen Bestandteil, an denen zukünftigen Forschungsarbeiten ansetzen können. Eine Basis dafür ist in dieser Arbeit erstellt worden.

Eine zukünftige Arbeit könnte die Zusammenhänge der Funktionen des Infotainmentsystems und die jeweils abgelegten Daten weiterführend erörtern. Einerseits könnten die Funktionen ermittelt werden, die für die Ablage bestimmter Daten verantwortlich sind und andererseits könnten die abgespeicherten Daten, die vom Mobiltelefon oder von den Eingaben über das HMI herrühren, ermittelt werden.

Während der Erarbeitung dieser Masterarbeit sind Forschungsarbeiten zu dem Thema der Analyse von fahrzeugbezogenen Daten studiert worden. Auch das ist ein sehr komplexer und spannender Teilbereich der automotiven Forensik. Eine weitere Möglichkeit wäre es, die in dieser Arbeit erarbeiteten Erkenntnisse zu nutzen, um die personenbezogenen Daten des Infotainmentsystems mit den über die BUS-Systeme auszulesenden fahrzeugbezogenen Daten zu vergleichen. Dadurch kann abschließend gezeigt werden, wie viele Daten aus einem Fahrzeug gezogen werden müssen, um Fahrzeugbesitzer, Fahrzeugführer und Insassen zu identifizieren.

# Anhang

## Open Source Software

---

### Nr. Original Wortlaut

---

1 “Free Redistribution

The license shall not restrict any party from selling or giving away the software as a component of an aggregate software distribution containing programs from several different sources. The license shall not require a royalty or other fee for such sale.”[37]

2 “Source Code

The program must include source code, and must allow distribution in source code as well as compiled form. Where some form of a product is not distributed with source code, there must be a well-publicized means of obtaining the source code for no more than a reasonable reproduction cost, preferably downloading via the Internet without charge. The source code must be the preferred form in which a programmer would modify the program. Deliberately obfuscated source code is not allowed. Intermediate forms such as the output of a preprocessor or translator are not allowed.”[37]

3 “Derived Works

The license must allow modifications and derived works, and must allow them to be distributed under the same terms as the license of the original software.”[37]

4 “Integrity of The Author’s Source Code

The license may restrict source-code from being distributed in modified form only if the license allows the distribution of *patch files* with the source code for the purpose of modifying the program at build time. The license must explicitly permit distribution of software built from modified source code. The license may require derived works to carry a different name or version number from the original software.”[37]

5 “No Discrimination Against Persons or Groups

The license must not discriminate against any person or group of persons.”[37]

---

Fortsetzung auf der nächsten Seite

Fortsetzung der vorherigen Seite

---

Nr.	Original Wortlaut
6	“No Discrimination Against Fields of Endeavor The license must not restrict anyone from making use of the program in a specific field of endeavor. For example, it may not restrict the program from being used in a business, or from being used for genetic research.”[37]
7	“Distribution of License The rights attached to the program must apply to all to whom the program is redistributed without the need for execution of an additional license by those parties.”[37]
8	“License Must Not Be Specific to a Product The rights attached to the program must not depend on the program’s being part of a particular software distribution. If the program is extracted from that distribution and used or distributed within the terms of the program’s license, all parties to whom the program is redistributed should have the same rights as those that are granted in conjunction with the original software distribution.”[37]
9	“License Must Not Restrict Other Software The license must not place restrictions on other software that is distributed along with the licensed software. For example, the license must not insist that all other programs distributed on the same medium must be open-source software.”[37]
10	“License Must Be Technology-Neutral No provision of the license may be predicated on any individual technology or style of interface.”[37]

---

Tabelle 10: 10 Kriterien von Open Source Software - Original Wortlaut

Kriterium	Gewichtung	BSI			SAP			Kent			Casey		
		Punkte	Bewertung	Punkte	Bewertung	Punkte	Bewertung	Punkte	Bewertung	Punkte	Bewertung		
Iterationsfähigkeit	40 %	10	4,0	10	4	6	2,4	1	0,4				
Anpassungsfähigkeit	20 %	5	1,0	9	1,8	7	1,4	1	0,2				
Dokumentationsaufwand	20 %	4	0,8	7	1,4	8	1,6	4	0,8				
Komplexität	10 %	4	0,4	9	0,9	8	0,8	2	0,2				
Methodenauswahl	10 %	8	0,8	10	1	10	1	6	0,6				
<b>Ergebnis</b>	<b>100 %</b>		<b>7,0</b>		<b>9,1</b>		<b>7,2</b>		<b>2,2</b>				

Tabelle 11: Bewertungsmatrix Nutzwertanalyse zu den Vorgehensmodellen

## Bewertungsmatrix Vorgehensmodelle

### Analyse Adressbuch DB

Table: Person

	<input type="checkbox"/> id	<input type="checkbox"/> fk_profile_id	<input type="checkbox"/> e_preferredName_gendescn	<input type="checkbox"/> onumbercolss_gendescad	<input type="checkbox"/> invisible	<input type="checkbox"/> firstName	<input type="checkbox"/> rstdName_sound	<input type="checkbox"/> lastName	<input type="checkbox"/> astName_sound
1	1	1	0	1	1	0	0	[BE]LUG	NULL
2	2	1	0	2	1	0	0	ADAC	NULL
3	3	1	0	3	1	0	0	Alex	NULL
4	4	1	0	0	0	0	0	Ann-Kathrin	NULL
5	5	1	0	4	1	0	0	Annette	NULL
6	6	1	0	5	1	0	0	Annika	NULL
7	7	1	0	6	1	0	0	Antonio	NULL
8	8	1	0	7	1	0	0	Christina	NULL
9	9	1	0	8	1	0	0	Claus	NULL
10	10	1	0	9	1	0	0	Daniel	NULL
11	11	1	0	10	1	0	0	Dierk	NULL
12	12	1	0	11	1	0	0	Emily	NULL
								Jaster	NULL

Bild 73: Ansicht der *Person* Relation im *DB Browser for SQLite*

Table: Phone

	<input type="checkbox"/> id	<input type="checkbox"/> order_number	<input type="checkbox"/> fk_person_id	<input type="checkbox"/> phoneType_flag	<input type="checkbox"/> number	<input type="checkbox"/> speeddial_key	<input type="checkbox"/> import_id
1	1	0	1	64	+49 30 ... 45490376	-1	1
2	2	0	2	64	+49 800 ... 222222	-1	1
3	3	0	3	8	+49 161 ... 118344	-1	1
4	4	0	4	64	+49 159 ... 555729	-1	1
5	5	0	5	64	050 ... 35052126	-1	1
6	6	0	6	64	01518 ... 4478923	-1	1
7	7	0	7	2	05200 ... 555729	-1	1
8	8	0	8	64	0123 ... 1129943	-1	1
9	9	0	9	64	01519 ... 655239	-1	1
10	10	0	10	64	0176 ... 2325887	-1	1
11	11	0	11	64	+49 151 ... 5589641	-1	1
12	12	0	12	64	0158 ... 10040530	-1	1
13	13	0	13	8	+49 134 ... 3459721	-1	1
14	14	0	14	64	+49 157 ... 5910077	-1	1
15	15	0	15	64	+49 176 ... 883667	-1	1
16	16	0	16	4	+49 172 ... 4546640	-1	1
17	17	1	17	64	+49 172 ... 15129993	-1	1
18	18	0	18	64	+49 164 ... 539556	-1	1
19	19	0	19	64	030 225572 ... 225572	-1	1
20	20	0	20	64	0151 ... 586669	-1	1
21	21	0	21	64	0176 ... 83941002	-1	1
22	22	0	22	64	0176 ... 6972354	-1	1
23	23	0	23	64	0151 ... 4667823	-1	1
24	24	0	24	64	+49 176 ... 4344572	-1	1
25	25	0	25	64	0151 ... 558116	-1	1
26	26	0	26	4	+49 151 ... 8883345	-1	1
27	27	1	27	64	+49 176 ... 5529054	-1	1

SQL Log

```

Show SQL submitted by Application - Clear
1 PRAGMA foreign_keys = 1;
2 PRAGMA database_list;
3 PRAGMA table_list;
4 PRAGMA main.sync_rate=1;
5 SELECT count(*) FROM "rowid" ORDER BY "rowid";
6 SELECT count(*) FROM "main" ORDER BY "rowid";
7 SELECT count(*) FROM (SELECT * FROM "main" ORDER BY "rowid");
8 SELECT count(*) FROM (SELECT * FROM "main" ORDER BY "rowid");
9 SELECT count(*) FROM (SELECT * FROM "main" ORDER BY "rowid");
10 SELECT count(*) FROM "main" ORDER BY "rowid";
11

```

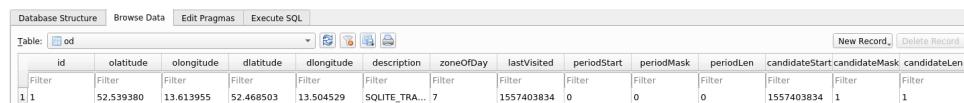
Bild 74: Ansicht der *Phone* Relation im *DB Browser for SQLite*

Table: Address

	<input type="checkbox"/> id	<input type="checkbox"/> order_number	<input type="checkbox"/> fk_person_id	<input type="checkbox"/> addressType_flag	<input type="checkbox"/> street	<input type="checkbox"/> city	<input type="checkbox"/> city_sound	<input type="checkbox"/> region	<input type="checkbox"/> country	<input type="checkbox"/> postalcode	<input type="checkbox"/> geoposition
1	1	0	1	2	Lehrter Straße 53	Berlin	NULL	Deutschland	10557	NULL	

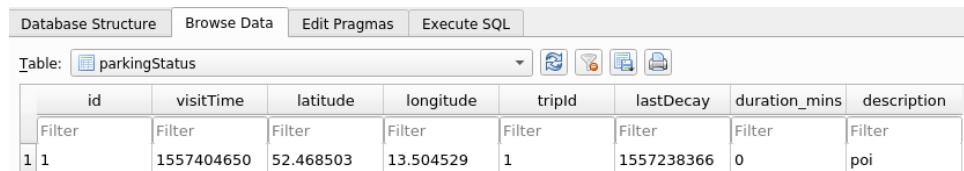
Bild 75: Ansicht der *Address* Relation im *DB Browser for SQLite*

## Analyse Navi DB



The screenshot shows the 'od' relation table in the DB Browser for SQLite. The table has 14 columns: id, olatitude, olongitude, dlatitude, dlongitude, description, zoneOfDay, lastVisited, periodStart, periodMask, periodLen, candidateStart, candidateMask, and candidateLen. A single row is displayed with values: id=1, olatitude=52.539380, olongitude=13.613955, dlatitude=52.468503, dlongitude=13.504529, description='SQLITE\_TRA...', zoneOfDay=7, lastVisited=1557403834, periodStart=0, periodMask=0, periodLen=0, candidateStart=1557403834, candidateMask=1, and candidateLen=1.

Bild 76: Ansicht der *od* Relation im *DB Browser for SQLite*



The screenshot shows the 'parkingStatus' relation table in the DB Browser for SQLite. The table has 8 columns: id, visitTime, latitude, longitude, tripId, lastDecay, duration\_mins, and description. A single row is displayed with values: id=1, visitTime=1557404650, latitude=52.468503, longitude=13.504529, tripId=1, lastDecay=1557238366, duration\_mins=0, and description='poi'.

Bild 77: Ansicht der *parkingStatus* Relation im *DB Browser for SQLite*

## Weiterführende Analyse

### Partition hd2t177

```
drwxrwxr-x 13 root      root          1024 Aug  02  2018 .
drwxr-xr-x  6 root      root          4096 Jan  24 18:01 ..
drwx-----  2 root      root          1024 Aug  02  2018 .boot
-rw xrwxrwx  1 root      root          8943 Jun  25  2018 EsoTraces.esd
-rw xrwxrwx  1 root      root          11283 Jun  25  2018 EsoTracesSec.esd
drwxrwxrwx  7 root      root          1024 Jun  25  2018 armle
-rw xrwxrwx  1 root      root          644 Jun  25  2018 build_parameters.txt
-rw xrwxrwx  1 root      root          397 Jun  25  2018 detailed_changelists_info.txt
drwxrwxrwx  15 root     root          1024 Jun  25  2018 eso
lrwxrwxrwx  1 root      root          22 Aug  02  2018 gemib -> /mnt/app/gemib.factory
drwxrwxrwx  6 root      root          1024 Jun  25  2018 gemib.factory
drwxrwxrwx  3 root      root          1024 Jun  25  2018 hb
drwxrwxrwx  2 root      root          1024 Jan  06 23:18 img_restore
-rw xrwxrwx  1 root      root          34 Jun  25  2018 img_ver.txt
drwxrwxrwx  5 root      root          5120 Jun  25  2018 navigation
-rw xrwxrwx  1 root      root          9 Jun  25  2018 p4changelist.txt
-rw xrwxrwx  1 root      root          8825 Jun  25  2018 p4client.txt
drwxrwxrwx  4 root      root          1024 Jun  25  2018 root
drwxrwxrwx  3 root      root          1024 Jun  25  2018 speech
lrwxrwxrwx  1 root      root          27 Aug  02  2018 streetview -> /mnt/app/streetview.factory
drwxrwxrwx  3 root      root          1024 Jun  25  2018 streetview.factory
-rw xrwxrwx  1 root      root          750 Jun  25  2018 target.properties
drwxrwxrwx  3 root      root          1024 Jun  25  2018 var
-rw xrwxrwx  1 root      root          1587 Jun  25  2018 version_info.txt
```

Bild 78: Vorschau der Dateien und Verzeichnisse auf der Partition hd2t177

### Partition hd2t178

```
drwxr-xr-x  3 sss      sshd          4096 May 14 2020 .
drwxr-xr-x  8 root     root          4096 May 14 2020 ..
drwx-----  2 sss      sshd          4096 May 14 2020 .ssh
```

Bild 79: Vorschau der Dateien und Verzeichnisse auf der Partition hd2t178

### Partition hd2t178.1

```
drwxrwxr-x 7 sss      sshd      32768 Jan 01 1970 .
drwxr-xr-x 6 root     root      4096 Jan 24 18:01 ..
drwx----- 2 sss      sshd      32768 Aug 02 2018 .boot
drwxrwxrwx 3 sss      sshd      32768 Jul 30 2018 common
-rwx-w--w- 1 sss      sshd      98 Jul 30 2018 common.checksum
-rwx-w--w- 1 sss      sshd      34 Jul 30 2018 common.fileinfo
lrwxrwxrwx 1 root    root      19 Jan 01 1970 hmi -> /mnt/app/speech/hmi
drwx----- 2 sss      sshd      32768 Jan 24 18:09 persistence
drwxrwxrwx 3 sss      sshd      32768 Jul 30 2018 sr
drwxrwxrwx 6 sss      sshd      32768 Jul 30 2018 tts
```

Bild 80: Vorschau der Dateien und Verzeichnisse auf der Partition hd2t178.1

### Partition hd2t178.2

```
drwxrwxr-x 5 root    root      32768 Jan 05 19:14 .
drwxr-xr-x 6 root    root      4096 Jan 24 18:01 ..
drwx----- 2 root    root      32768 Aug 02 2018 .boot
-rw-rw-rw- 1 root    root      616 Aug 23 2017 Update.txt
-rwx-w--w- 1 root    root      98 Jul 30 2018 Update.txt.checksum
-rwx-w--w- 1 root    root      32 Jul 30 2018 Update.txt.fileinfo
drwxrwxrwx 2 root    root      32768 Jul 30 2018 config
drwxrwxrwx 2 root    root      32768 Jul 30 2018 database
-rwx-w--w- 1 root    root      99 Jul 30 2018 database.checksum
-rwx-w--w- 1 root    root      32 Jul 30 2018 database.fileinfo
```

Bild 81: Vorschau der Dateien und Verzeichnisse auf der Partition hd2t178.2

### Partition hd2t178.3

```
drwxrwxr-x 3 root    root      32768 Jan 01 1970 .
drwxr-xr-x 6 root    root      4096 Jan 27 00:01 ..
drwx----- 2 root    root      32768 Aug 02 2018 .boot
-rw-rw-r-- 1 root    root      80 Jan 01 1970 .mediadb_2.
-rw-rw-r-- 1 root    root      80 Jan 01 1970 .mediadb_4.
-rw-rw-r-- 1 root    root      80 Jan 01 1970 .mediadb_5.
-rw-rw-r-- 1 root    root      54272 Jan 03 1970 mediadb_2
-rw-rw-r-- 1 root    root      32768 Jan 01 1970 mediadb_4
-rw-rw-r-- 1 root    root      102400 Jan 03 1970 mediadb_5
```

Bild 82: Vorschau der Dateien und Verzeichnisse auf der Partition hd2t178.3

### Partition hd2t178.4

```
drwxrwxr-x 5 root    root      32768 Jan 01 1970 .
drwxr-xr-x 6 root    root      4096 Jan 27 00:01 ..
drwx----- 2 root    root      32768 Aug 02 2018 .boot
drwxrwxrwx 3 root    root      32768 Jan 01 1970 persistent
drwxrwxrwx 2 root    root      32768 Jan 01 1970 tmp
```

Bild 83: Vorschau der Dateien und Verzeichnisse auf der Partition hd2t178.4

### Partition hd2t178.5

```
drwxr-xr-x  4 sss      sshd      4096 May 16 2020 .
drwxr-xr-x  6 root     root      4096 Jan 27 00:01 ..
drwx----- 2 sss      sshd      4096 Aug 02 2018 .boot
-rw-r--r--  1 sss      sshd      91136 Jan 24 18:12 addressbook.db
-rw-r--r--  1 sss      sshd      32768 Jan 01 1970 addressbook.db-shm
-rw-r--r--  1 sss      sshd      1048032 Jan 01 1970 addressbook.db-wal
drwxrwxrwx  4 sss      sshd      4096 Jan 01 1970 pics
-rw-r--r--  1 sss      sshd      24576 Jan 01 1970 picturestore.db
-rw-r--r--  1 sss      sshd      32768 Jan 01 1970 picturestore.db-shm
-rw-r--r--  1 sss      sshd      1048032 Jan 01 1970 picturestore.db-wal
-rw-r--r--  1 sss      sshd      1024 Jan 01 1970 truffles.0.db
-rw-r--r--  1 sss      sshd      32768 Jan 01 1970 truffles.0.db-shm
-rw-r--r--  1 sss      sshd      29376 Jan 01 1970 truffles.0.db-wal
```

Bild 84: Vorschau der Dateien und Verzeichnisse auf der Partition hd2t178.5

### Partition hd2t178.6

```
drwxrwxr-x  4 sss      root      16384 Jan 01 1970 .
drwxr-xr-x  6 root     root      4096 Jan 27 00:01 ..
drwx----- 2 sss      root      16384 Aug 02 2018 .boot
drwx----- 2 sss      root      16384 Jan 01 1970 navigation
```

Bild 85: Vorschau der Dateien und Verzeichnisse auf der Partition hd2t178.6

### Partition hd2t178.7

```
drwxrwxr-x  9 sss      sshd      1024 Jun 16 2020 .
drwxr-xr-x  6 root     root      4096 Jan 27 00:01 ..
drwx----- 2 sss      sshd      1024 Aug 02 2018 .boot
drwxrwxrwx  2 sss      sshd      1024 Jul 30 2018 Parrot
drwx----- 2 sss      sshd      1024 Jan 01 1970 cache
drwx----- 2 sss      sshd      1024 Jun 16 2020 predictiveNav
drw-rw-rw-  3 sss      sshd      1024 Jan 01 1970 rcc
drwxrwxrwx  2 sss      sshd      1024 Jul 30 2018 rhmi
drwx----- 2 sss      sshd      1024 Jan 01 1970 userdata
```

Bild 86: Vorschau der Dateien und Verzeichnisse auf der Partition hd2t178.7

### Partition hd2t178.8

```
drwxrwxr-x  8 sss      root      32768 Aug 02 2018 .
drwxr-xr-x  6 root     root      4096 Jan 27 00:01 ..
drwx----- 2 sss      root      32768 Aug 02 2018 .boot
drwxrwxrwx  3 sss      root      32768 Jan 01 1970 database
drwxrwxrwx  2 sss      root      32768 Jan 01 1970 eggnog
drwxrwxrwx  3 sss      root      32768 Jan 01 1970 speech
drwxrwxrwx  2 sss      root      32768 Jan 01 1970 sr
drwxrwxrwx  3 sss      root      32768 Jan 01 1970 truffles
```

Bild 87: Vorschau der Dateien und Verzeichnisse auf der Partition hd2t178.8

### **Partition hd2t178.9**

```
drwxrwxr-x  3 root      root          32768 Aug  02  2018 .
drwxr-xr-x  6 root      root          4096 Jan 27 00:01 ..
drwx----- 2 root      root          32768 Aug  02  2018 .boot
```

Bild 88: Vorschau der Dateien und Verzeichnisse auf der Partition hd2t178.9

### **Partition hd2t179**

```
drwxrwxr-x  6 root      root          32768 Jan  01  1970 .
drwxr-xr-x  6 root      root          4096 Jan 27 00:01 ..
drwx----- 2 root      root          32768 Aug  02  2018 .boot
drwxrwxrwx  3 root      root          32768 Jan  01  1970 app
drwxrwxrwx  4 root      root          32768 Jan  01  1970 system
drwx----- 3 root      root          32768 Jan  01  1970 tmp
```

Bild 89: Vorschau der Dateien und Verzeichnisse auf der Partition hd2t179

## **Quellcode pythonByteCodeToText**

```
### Script to convert python ByteCode into a
#readable string.
#Therefore you have to paste your python byteCode
#into the variable input_bytes.#
#You have to do manual changes at the byteCode
#if there are problems with the conversion.
input_bytes = b"\xea\x8a\xd1\x00\x06\x9a\x02\x00
+\x07\x00\x00\x00a\x00\x00\x00s\x00\x00\x00r\x00a
+\x00s\x00a\x00a\x00k\x00u\x00r\x00k\x00u\x00\x01\x00"
#variable for converted ascii values
output_numbers = list(input_bytes)

# Initialising list
ini_list = output_numbers

# Printing initial list
```

```
print ("\\nInitial_list:", ini_list)

# Using list comprehension and join
res = ''.join(chr(val) for val in ini_list)

# Print the resultant string
print ("\\nResultant_string:", str(res))
```

# Literatur

- [1] Prof. (FH) Mag. Dr. Helmut Siller. “Gabler Wirtschaftslexikon”. In: *Forensik - Ausführliche Definition im Online-Lexikon* (2018), S. 137–140. URL: <https://wirtschaftslexikon.gabler.de/definition/forensik-53390/version-276483> (besucht am 18.09.2020).
- [2] Prof. Dr. Gabi Dreo et al. *Grundlagen der IT- Forensik*. 2013. URL: [@download/file%20/Seminararbeiten-Forensik-2013.pdf](https://www.unibw.de/technische-informatik/mitarbeiter/professoren/dreo/publikationen/seminararbeiten-forensik-2013.pdf) (besucht am 02.11.2020).
- [3] Bundesamt für Sicherheit in der Informationstechnik (BSI). *Leitfaden IT-Forensik*. BSI, 2011. Kap. Einführung, S. 8.
- [4] Wilhelm Dolle. “Computer-Forensik in der Praxis”. In: Bd. 33. Datenschutz und Datensicherheit - DuD, 2009, S. 183–188.
- [5] IDRIX SARL. *VeraCrypt Volume*. URL: <https://www.veracrypt.fr/en/VeraCrypt%20Volume.html> (besucht am 21.09.2020).
- [6] Dirk Labudde und Michael Spranger. “Datenrekonstruktion mittels Carving”. In: *Forensik in der digitalen Welt*. Springer Verlag, 2017, S. 137–140. ISBN: 978-3-662-53801-2.
- [7] Digitpol. *Vehicle Forensics*. URL: <https://digitpol.com/automotive-forensics/> (besucht am 24.09.2020).
- [8] Volker Johanning und Roman Mildner. *Car IT kompakt. Das Auto der Zukunft – Vernetzt und autonom fahren*. Springer Vieweg, 2015, S. 1–25.
- [9] Peter Rademacher. *Das vernetzte Auto. Der Begriff Car-IT gewinnt in der Automobilindustrie massiv an Bedeutung – die künftige Rolle der Business-IT*. 2012. URL: <https://www.car-it.com/mobility/der-begriff-car-it-gewinnt-in-der-automobilindustrie-massiv-an-bedeutung-die-kunftige-rolle-der-business-it-280.html> (besucht am 24.09.2020).
- [10] DAT. *Anteil der Autos die mit Internetzugang ausgestattet sind in Deutschland im Jahr 2018*. 2019, S. 13. URL: <https://de.statista.com/statistik/daten/studie/707677/umfrage/anteil-der-pkw-mit-internetzugang/> (besucht am 24.09.2020).

- [11] Digitpol. *Prognose des Marktvolumens von fahrzeugbasierten Connected Services nach weltweiten Regionen in den Jahren 2018 bis 2030*. 2019, S. 19. URL: <https://de.statista.com/statistik/daten/studie/984961/umfrage/prognose-des-marktvolumens-von-connected-car-services-nach-weltweiten-regionen/> (besucht am 24.09.2020).
- [12] Konrad Reif. *Batterien, Bordnetze und Vernetzung*. Vieweg + Teubner Verlag, 2010. Kap. Vernetzung im Kfz, S. 120–134. ISBN: 978-3-8348-1310-7.
- [13] Wolfgang Pester. *Das unterschätzte Übergewicht der Premiumautos*. URL: <https://www.welt.de/motor/article138223665/Das-unterschaetzte-Uebergewicht-der-Premiumautos.html> (besucht am 24.09.2020).
- [14] Fabio Martinelli et al. “Human Behavior Characterization for Driving Style Recognition in Vehicle System”. In: *Computers and Electrical Engineering Journal* (2018).
- [15] Boris Tolg und Ansagar Meroth. *Infotainmentsysteme im Kraftfahrzeug*. Vieweg Praxiswissen, 2008. Kap. Vernetzung im Kfz, S. 1–6. ISBN: 978-3-8348-0285-9.
- [16] BMW AG. *Die wichtigsten Fahrerassistenzsysteme im Überblick*. URL: <https://www.bmw.de/innovation/die-wichtigsten-fahrerassistenzsysteme.html> (besucht am 25.09.2020).
- [17] Volker Johanning und Roman Mildner. *Car IT kompakt. Das Auto der Zukunft – Vernetzt und autonom fahren*. Springer Vieweg, 2015, S. 45–61.
- [18] BMW AG. *BMW ConnectedDrive*. URL: <https://www.bmw-connecteddrive.de/app/index.html#/portal/store> (besucht am 25.09.2020).
- [19] Blackberry QNX. *About BlackBerry QNX*. URL: <https://blackberry.qnx.com/en/company/about-qnx> (besucht am 22.09.2020).
- [20] Paul Hansen. *The Company Profile: BlackBerry QNX*. The Hansen Report on Automotive Electronics, 2018, S. 11–18.
- [21] Dirk Fox. *Hypervisor*. Datenschutz und Datensicherheit - DuD, 2012, S. 54.
- [22] Blackberry QNX. *Ultimate Guide to Real-time Operating Systems (RTOS)*. URL: <https://blackberry.qnx.com/en/rtos/what-is-real-time-operating-system/> (besucht am 22.09.2020).
- [23] Blackberry QNX. *QNX Neutrino Real-time Operating System (RTOS)*. URL: <https://blackberry.qnx.com/en/software-solutions/embedded-software/qnx-neutrino-rtos> (besucht am 22.09.2020).

- [24] The SQLite Consortium. *What Is SQLite?* URL: <https://www.sqlite.org/index.html> (besucht am 29.09.2020).
- [25] Jay A. Kreibich. *Using SQLite*. O'Reilly Media Inc., 2010. Kap. What is SQLite?, S. 1–7. ISBN: 978-0-596-52118-9.
- [26] Michael Owens. *The Definitive Guide to SQLite*. Apress, 2006. Kap. Introducing SQLite, S. 1–16. ISBN: 978-1-59059-673-9.
- [27] Studyflix. *B-Baum*. URL: <https://studyflix.de/informatik/b-baum-1435> (besucht am 16.10.2020).
- [28] The SQLite Consortium. *Most Widely Deployed and Used Database Engine*. URL: <https://www.sqlite.org/mostdeployed.html> (besucht am 30.09.2020).
- [29] Christian Meng und Harald Baier. “bring2lite: A Structural Concept and Tool for Forensic Data Analysisand Recovery of Deleted SQLite Records”. In: *Digital Investigation* 29 (Feb. 2019), S. 31–41. ISSN: 1742-2876.
- [30] Li Zhang et al. “Recovering SQLite data from fragmented flash pages”. In: *Annals of Telecommunications* (2019), S. 451–460.
- [31] Yao Liu et al. *Security and Privacy in Communication Networks*. Springer International Publishing, 2016. Kap. SQLite Forensic Analysis Based on WAL, S. 557–574. ISBN: 978-3-319-59608-2.
- [32] usd AG. *Tool for Forensic Data Analysis at DFRWS USA 2019*. URL: <https://herolab.usd.de/en/news-tool-for-forensic-data-analysis-at-dfrws-usa-2019/> (besucht am 09.09.2020).
- [33] Paul Sanderson. *Dealing with records found in SQLite Rollback Journals*. 2015. URL: <https://sqliteforensictoolkit.com/dealing-with-records-found-in-sqlite-rollback-journals/> (besucht am 08.10.2020).
- [34] The SQLite Consortium. *Write-Ahead Logging*. URL: <https://sqlite.org/wal.html> (besucht am 08.10.2020).
- [35] Dominic R. Markowski und Alexandra Kees. *Open Source Enterprise Software*. 2. Aufl. Springer Vieweg, 2019. Kap. Open Source Software, S. 30–32. ISBN: 978-3-658-25218-2.
- [36] Open Source Initiative. *The License Review Process*. URL: <https://opensource.org/approval> (besucht am 09.09.2020).
- [37] Open Source Initiative. *The Open Source Definition*. URL: <https://opensource.org/osd> (besucht am 08.09.2020).

- [38] Bernd Bruegge et al. *Open-Source-Software*. Springer, 2004. Kap. Erscheinungsformen und systematische Kategorisierung, S. 19–21. ISBN: 978-3-642-62077-5.
- [39] Open Source Initiative. *Basics of Open Source*. URL: <https://opensource.org/faq#osd> (besucht am 08. 09. 2020).
- [40] Dirk Labudde und Michael Spranger. “Tatort in der modernen Forensik”. In: *Forensik in der digitalen Welt*. Springer Verlag, 2017, S. 11. ISBN: 978-3-662-53801-2.
- [41] Christian Schawel und Fabian Billing. “SWOT-Analyse”. In: *Top 100 Management Tools: Das wichtigste Buch eines Managers Von ABC-Analyse bis Zielvereinbarung*. Wiesbaden: Springer Fachmedien Wiesbaden, 2018, S. 331–333. ISBN: 978-3-658-18917-4.
- [42] B. Huch et al. *Rechnungswesenorientiertes Controlling. Ein Leitfaden für Studium und Praxis*. 2. Auflage. Physica-Verlag, 1995, S. 145–149. ISBN: 978-3-7908-0094-4.
- [43] Brian Carrier. *The Sleuth Kit - Description*. URL: <https://sleuthkit.org/sleuthkit/desc.php> (besucht am 01. 09. 2020).
- [44] X-Ways Software Technology AG. *X-Ways Forensics/ WinHex - Manual*. URL: <https://www.x-ways.net/winhex/manual.pdf> (besucht am 29. 11. 2020).
- [45] Magnet Forensics Inc. *E-Brochure: MAGNET AXIOM - THE EVOLUTION OF IEF*. URL: [https://irp-cdn.multiscreensite.com/ad68eab3/files/uploaded/MagnetAXIOM\\_e-brochure.pdf](https://irp-cdn.multiscreensite.com/ad68eab3/files/uploaded/MagnetAXIOM_e-brochure.pdf) (besucht am 29. 11. 2020).
- [46] Brian Carrier. *Autopsy*. URL: <http://www.sleuthkit.org/autopsy/> (besucht am 07. 09. 2020).
- [47] Brian Carrier. *Autopsy - Intuitive*. URL: <http://www.sleuthkit.org/autopsy/intuitive.php> (besucht am 07. 09. 2020).
- [48] Roman Trobec et al. *Introduction to Parallel Computing*. Springer Nature Switzerland AG, 2018. Kap. Programming Multi-core and Shared Memory Multiprocessors Using OpenMP, S. 47–49. ISBN: 978-3-319-98833-7.
- [49] Roman Trobec et al. *Introduction to Parallel Computing*. Springer Nature Switzerland AG, 2018. Kap. Why - Every Computer Is a Parallel Computer, S. 3–4. ISBN: 978-3-319-98833-7.
- [50] Simson L. Garfinkel. “Digital Media Triage with Bulk Data Analysis and Bulk\_extractor”. In: *Comput. Secur.* 32.C (Feb. 2013), S. 56–72. ISSN: 0167-4048.
- [51] Simson L. Garfinkel. *bulk-extractor Package Description*. URL: <https://tools.kali.org/forensics/bulk-extractor/> (besucht am 11. 09. 2020).

- [52] ReFirm Labs. *Binwalk The Number 1 Open Source Firmware Extraction Tool*. URL: <https://www.refirmlabs.com/binwalk/> (besucht am 14.09.2020).
- [53] ReFirm Labs. *Binwalk*. URL: <https://github.com/ReFirmLabs/binwalk> (besucht am 14.09.2020).
- [54] *Linux Manual Page - BSD General Commands Manual*. Kap. FILE (1).
- [55] Craig Heffner. *Binwalk Package Description*. URL: <https://tools.kali.org/forensics/binwalk> (besucht am 15.09.2020).
- [56] Hans-Peter Merkel. *Neues aus der OSS Forensik*. 20. Jan. 2021.
- [57] Paolo Bonzini et al. *QEMU Wiki - Main Page*. URL: [https://wiki.qemu.org/Main\\_Page](https://wiki.qemu.org/Main_Page) (besucht am 11.12.2020).
- [58] Truffle Blockchain Group Inc. *Binwalk Package Description*. URL: <https://www.trufflesuite.com/docs/truffle/overview> (besucht am 06.12.2020).
- [59] Kurt L. Hudson und Peter Geelen. *LDAP over SSL (LDAPS) Certificate*. URL: <https://social.technet.microsoft.com/wiki/contents/articles/2980.ldap-over-ssl-ldaps-certificate.aspx> (besucht am 06.12.2020).
- [60] *Linux Manual Page - BSD General Commands Manual*. Kap. DD (1).
- [61] BlackBerry Limited. *On-The-Go (OTG) manager for USB*. URL: <http://www.qnx.com/developers/docs/7.0.0/com.qnx.doc.neutrino.utilities/topic/i/io-usb-otg.html> (besucht am 29.11.2020).
- [62] BlackBerry Limited. *Driver for USB Mass Storage interface*. URL: <http://www.qnx.com/developers/docs/7.0.0/com.qnx.doc.neutrino.utilities/topic/d/devb-umass.html> (besucht am 04.12.2020).
- [63] BlackBerry Limited. *Power-Safe filesystem*. URL: [http://get.qnx.com/developers/docs/6.5.0/topic/com.qnx.doc.neutrino\\_sys\\_arch/fsys.html#QNX6\\_filesystem](http://get.qnx.com/developers/docs/6.5.0/topic/com.qnx.doc.neutrino_sys_arch/fsys.html#QNX6_filesystem) (besucht am 08.12.2020).
- [64] BlackBerry Limited. *System Architecture - Filesystems*. URL: [http://get.qnx.com/developers/docs/6.5.0/topic/com.qnx.doc.neutrino\\_sys\\_arch/fsys.html#FSYSCLASSES](http://get.qnx.com/developers/docs/6.5.0/topic/com.qnx.doc.neutrino_sys_arch/fsys.html#FSYSCLASSES) (besucht am 08.12.2020).
- [65] BlackBerry Limited. *BSP structure and contents*. URL: [https://www.qnx.com/developers/docs/7.0.0/index.html#com.qnx.doc.neutrino.building/topic/bsp/bsp\\_structure.html](https://www.qnx.com/developers/docs/7.0.0/index.html#com.qnx.doc.neutrino.building/topic/bsp/bsp_structure.html) (besucht am 08.12.2020).
- [66] BlackBerry Limited. *QNX Board Support Packages*. URL: <https://blackberry.qnx.com/en/support/qnx-board-support-packages> (besucht am 08.12.2020).

- [67] BlackBerry Limited. *Initial Program Loaders (IPLs)*. URL: [https://www.qnx.com/developers/docs/7.0.0/index.html#com.qnx.doc.neutrino.building/topic/ipl/ipl\\_about.html](https://www.qnx.com/developers/docs/7.0.0/index.html#com.qnx.doc.neutrino.building/topic/ipl/ipl_about.html) (besucht am 08.12.2020).
- [68] Raspberry Pi Foundation. *GPIO*. URL: <https://www.raspberrypi.org/documentation/usage/gpio/> (besucht am 17.02.2020).

# Abbildungsverzeichnis

Bild 1	Visualisierung der CERT-Taxonomie [2]	11
Bild 2	IT Architektur Fahrzeug [8]	16
Bild 3	Fahrzeugsensorik [12]	17
Bild 4	Aufgaben eines Infotainmentsystems [15]	18
Bild 5	Globale Marktaufteilung von Betriebssystemen der Infotainmentsysteme [20]	20
Bild 6	Schematische Darstellung der SQLite Architektur [26]	23
Bild 7	Schematische Darstellung der Arbeitsweise bring2lite [32]	25
Bild 8	Schematischer Aufbau SQLite Datenbank [30]	26
Bild 9	Schematische Darstellung SQLite Rollback-Journal [33]	27
Bild 10	Schematische Darstellung SQLite WAL [31, 34]	28
Bild 11	Abstrahierte Darstellung prozessualer Abläufe der Vorgehensmodelle	34
Bild 12	Schematische Darstellung der SWOT-Analyse	35
Bild 13	SWOT-Analyse des SAP-Modells	36
Bild 14	SWOT-Analyse des Kent, Chevalier, Grance, Dang Modells	36
Bild 15	SWOT-Analyse des BSI-Modells	37
Bild 16	SWOT-Analyse des Casey-Modells	37
Bild 17	Vorgehensweise der Secure Phase	42
Bild 18	Vorgehensweise der manuellen forensischen Untersuchung	44
Bild 19	Lösungsansätze Schreibzugriff	50
Bild 20	Lösungsansätze physikalische Auswertung	52
Bild 21	Version Kommandozeilen-Programm sha512sum	56
Bild 22	SHA512-Hashsumme des Infotainment-Images	56
Bild 23	SHA512-Hashsumme eines Slaves	57
Bild 24	Bildausschnitt der forensische Untersuchung mittels Autopsy	58
Bild 25	Konvertierung des Infotainment-Images in Rohformat	59
Bild 26	Einbinden der Partitionen in ein logisches Device	60
Bild 27	Untersuchung des Rohimages mittels Binwalk	60
Bild 28	Fehlversuch beim Einhängen als QNX4 Partitionen	61
Bild 29	Erfolgreiches Einhängen als QNX6 Partitionen	61
Bild 30	Gefundene SQLite Datenbanken	62
Bild 31	Vorschau der Dateien und Verzeichnisse auf der Partition 1	63
Bild 32	Vorschau der Dateien und Verzeichnisse auf der Partition 10	64
Bild 33	Datenbank und Datenstruktur der <i>Adressbuch</i> Datenbank	65

Bild 34	Profile Relation der <i>Adressbuch</i> Datenbank	65
Bild 35	Datenbank und Datenstruktur der <i>Bilderspeicher</i> Datenbank	66
Bild 36	Datenbank und Datenstruktur der <i>Truffles</i> Datenbank	66
Bild 37	Vorschau der Dateien und Verzeichnisse auf der Partition 12	67
Bild 38	Datenbank und Datenstruktur der <i>Navigations</i> Datenbank	67
Bild 39	Ansicht der <i>Trips</i> Relation im <i>DB Browser for SQLite</i>	68
Bild 40	Screenshot des erfolgreichen Durchlaufs von <i>bulk_extractor</i>	69
Bild 41	Auszug aus den extrahierten E-Mail Headern nach RFC822 Standard	70
Bild 42	Auszug aus Städtenamen Datenbank für das Navigationssystem	71
Bild 43	Auszug aus der Multimedia Sqlite Datenbank	71
Bild 44	Mögliche Wiederherstellungsresultate	73
Bild 45	Ergebnis des Übersetzer-Skripts von Python Bytecode in Textform	74
Bild 46	Version <i>xmount</i>	75
Bild 47	Version <i>fdisk</i>	75
Bild 48	Überprüfung der Konvertierung in Rohformat	76
Bild 49	Version des Befehlszeilen-Programms <i>dd</i>	76
Bild 50	Fehlerhaftes Einhängen des externen Datenträgers in die Forensik-Maschine	77
Bild 51	Fehlermeldung Datenträger in Read-Only Modus	77
Bild 52	Geladene QNX Neutrino RTOS VM	78
Bild 53	Vorschau der Dateien und Verzeichnisse auf der Partition hd2t178.2	79
Bild 54	Erstellter Ordner auf der nicht schreibgeschützten Partition	80
Bild 55	Datei erstellen, beschreiben und löschen	80
Bild 56	Übersicht der eingehängten Partitionen	81
Bild 57	Inhalt der Datei <i>version_info.txt</i>	85
Bild 58	Fehlerhafter Extraktion des Dateiarchivs	89
Bild 59	Untersuchung des Betriebssystem-Images mittels <i>fdisk</i>	89
Bild 60	Aufbereitung der extrahierten Tupel aus der <i>Adressbuch</i> Datenbank	90
Bild 61	Aufbereitung der extrahierten Tupel aus der <i>Navigation</i> Datenbank	91
Bild 62	Visualisierung der rekonstruierten Strecke im Navigationssystem	92
Bild 63	Aufbereitung der extrahierten Tupel aus der <i>Medien</i> Datenbank	93
Bild 64	Prüfung auf die Unterstützung von KVM Virtualisierung	94
Bild 65	Prüfung der geladenen KVM Module	94
Bild 66	Virtualisierung des Infotainmentsystems	95
Bild 67	Installation von QNX 6.5 mit Installations-CD	97
Bild 68	Desktopumgebung des QNX Neutrino RTOS 6.5	98
Bild 69	Importierung des BSPs in die <i>QNX Momentics IDE</i> 7.0	99

## Abbildungsverzeichnis

---

Bild 70	GPIO Pin Layout [68]	102
Bild 71	Verdrahtung des Raspberry Pis mit dem Adapter der seriellen RS-232 Schnittstelle	103
Bild 72	Ausgabe des Minicom-Terminals	104
Bild 73	Ansicht der <i>Person</i> Relation im <i>DB Browser for SQLite</i>	118
Bild 74	Ansicht der <i>Phone</i> Relation im <i>DB Browser for SQLite</i>	118
Bild 75	Ansicht der <i>Address</i> Relation im <i>DB Browser for SQLite</i>	118
Bild 76	Ansicht der <i>od</i> Relation im <i>DB Browser for SQLite</i>	119
Bild 77	Ansicht der <i>parkingStatus</i> Relation im <i>DB Browser for SQLite</i>	119
Bild 78	Vorschau der Dateien und Verzeichnisse auf der Partition hd2t177	119
Bild 79	Vorschau der Dateien und Verzeichnisse auf der Partition hd2t178	119
Bild 80	Vorschau der Dateien und Verzeichnisse auf der Partition hd2t178.1	120
Bild 81	Vorschau der Dateien und Verzeichnisse auf der Partition hd2t178.2	120
Bild 82	Vorschau der Dateien und Verzeichnisse auf der Partition hd2t178.3	120
Bild 83	Vorschau der Dateien und Verzeichnisse auf der Partition hd2t178.4	120
Bild 84	Vorschau der Dateien und Verzeichnisse auf der Partition hd2t178.5	121
Bild 85	Vorschau der Dateien und Verzeichnisse auf der Partition hd2t178.6	121
Bild 86	Vorschau der Dateien und Verzeichnisse auf der Partition hd2t178.7	121
Bild 87	Vorschau der Dateien und Verzeichnisse auf der Partition hd2t178.8	121
Bild 88	Vorschau der Dateien und Verzeichnisse auf der Partition hd2t178.9	122
Bild 89	Vorschau der Dateien und Verzeichnisse auf der Partition hd2t179	122

# Tabellenverzeichnis

Tabelle 1	Anforderungen der forensischen Untersuchung	13
Tabelle 2	10 Kriterien von Open Source Software - Original Wortlaut	30
Tabelle 3	Bewertungsschema der Nutzwertanalyse	38
Tabelle 4	Ausschnitt: Bewertungsmatrix Nutzwertanalyse zu den Vorgehensmodellen	40
Tabelle 5	Analyseergebnis der Partitionen und dessen Einsatzzweck mit Linux-Boardmittel	68
Tabelle 6	Analyseergebnis der Partitionen und dessen Einsatzzweck mit QNX-VM	84
Tabelle 7	Gesamtübersicht der Partitionen und dessen Einsatzzweck	87
Tabelle 8	QNX Partitionstypen und Dateisysteme	88
Tabelle 9	Bewertungsmatrix Nutzwertanalyse des Vergleichs von Standard Linux Distributionen und QNX Neutrino RTOS	106
Tabelle 10	10 Kriterien von Open Source Software - Original Wortlaut	116
Tabelle 11	Bewertungsmatrix Nutzwertanalyse zu den Vorgehensmodellen	117

# Abkürzungsverzeichnis

**BSI** Bundesministerium für Sicherheit in der Informationstechnologie

**BSP** QNX Board Support Package

**bzw.** beziehungsweise

**DFSG** Debian Free Software Guidelines

**engl.** englisch

**EVM** Ethereum Virtual Machine

**GPIO** General-Purpose Input/Output Pins

**GUI** Graphical User Interface

**HMI** Human Machine Interface

**IPL** Initial Program Loader

**LDAPS** Secure Lightweight Directory Access Protocol

**MB** Mega Byte

**MIT** Massachusetts Institute of Technology

**OSI** Open Source Initiative

**OSS** Open Source Software

**PKW** Personenkraftwagen

**POIs** Point of Interest

**QNX Neutrino RTOS** Blackberry QNX Neutrino Realtime Operation System

**SAP** Secure-Analyse-Present

**SIM** Subscriber Identity Module

**TSK** The Sleuth Kit

**UML** Unified Modeling Language

## Abkürzungsverzeichnis

**VM** Virtuelle Maschinen

**VMM** Virtual-Maschine-Monitor

**WAL** Write-Ahead Log

# Thesen

- Durch die Vernetzung der Fahrzeuge stehen Hersteller und Entwickler vor einer ganz neuen Herausforderung.
- Für die forschende Herangehensweise sollte ein flexibles Vorgehensmodell gewählt werden.
- Ohne eine offizielle Unterstützung von QNX kann *Autopsy* keinen sinnvollen Datengehalt zur forensischen Untersuchung beitragen.
- Eine vollumfängliche forensische Untersuchung des QNX Systems ist mit Linux-Bordmitteln nicht möglich.
- Beschädigte SQLite Datenbanken können teilweise rekonstruiert und analysiert werden.
- Ein bereitgestelltes QNX System ist für die forensische Untersuchung nahezu unerlässlich.
- Mit dem Raspberry Pi 4B ist eine mobile Hardware verfügbar, auf der physikalische Auswertungen von QNX Neutrino Echtzeit-Betriebssystemen durchgeführt werden können.