David Gries – Mittelwaldstraße 13 – 63776 Mömbris
Matrikelnummer: 2217395, E-Mail: s170417@th-ab.de

Entwicklung und Programmierung eines automatisierten Backup-Ladesystems für iOS-Geräte unter Verwendung eines Einplatinencomputers

Studienarbeit von David Gries

21. Juli 2021

Technische Hochschule Aschaffenburg - Fakultät Ingenieurwissenschaften
Betreuer: Prof. Dr.-Ing. Alexander Czinki
Würzburger Straße 45, 63743 Aschaffenburg

Erklärung

Ich versichere hiermit, dass ich meine Studienarbeit mit dem Thema: „Entwicklung und Programmierung eines automatisierten Backup-Ladesystems für iOS-Geräte unter Verwendung eines Einplatinencomputers“ selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

Ich versichere hiermit zudem, dass die eingereichte elektronische Fassung mit der gedruckten Fassung übereinstimmt.

_____ _____

Ort, Datum Unterschrift

Management Summary

English

The following thesis describes the implementation of a mobile charging device that automatically creates backups of iOS-devices during each charging process. In contrast to comparable products, it does not focus on file synchronization or media backups, but full system backups as a secure alternative for cloud-backups. This prevents data loss if the device is damaged or lost without the need of additional effort. To create these backups, a single-board computer with a Linux system and open-source software is used to store all data on a USB stick.

Deutsch

In dieser Studienarbeit wird die Umsetzung eines Ladegeräts beschrieben, welches automatisiert vollständige Sicherungen von iOS-Geräten bei jedem Ladevorgang erstellt. Der Fokus liegt im Gegensatz zu vergleichbaren Produkten nicht auf der Dateisynchronisation oder Sicherung ausgewählter Medien, sondern vollständige Backups zur Vermeidung von Datenverlust ohne zusätzlichen Aufwand als sichere Alternative zu Cloud-Backups. Zum Erstellen dieser Sicherungen wird ein Einplatinencomputer mit einem Linux-System und Open-Source-Software verwendet, der die Daten der Sicherung auf einem USB-Stick speichert.

Inhalt

Abbildungen

1. Einleitung

Smartphones sind heute aus dem Alltag nicht mehr wegzudenken. Die hohe Zugänglichkeit und einfache, intuitive Bedienung sorgen für eine steigende Verbreitung und Nutzung in immer mehr Bereichen. Noch vor der Nutzung von Social Media liegt in Deutschland die Verwendung des Smartphones zum Verfassen von E-Mails und die Nutzung des Internet-Browsers (vgl. Statista Global Consumer Survey 2020a). Aber auch sehr sensible Daten, die beispielsweise beim Speichern von Terminen, persönlichen Notizen oder dem Online-Banking anfallen, werden von den Nutzern oft ohne Sicherheitsbedenken auf dem Smartphone und später bei Cloud-Providern gespeichert. Während Geräte mit dem Android-Betriebssystem in Deutschland den größten Marktanteil einnehmen, ist aktuell ein verstärktes Wachstum iOS-basierter Smartphones von Apple zu beobachten: 78,2 % der Käufer setzten im ersten Quartal 2020 auf Android-Smartphones und 21,3 % auf % iOS-Geräte. Der iOS-Anteil stieg im Jahr 2021 auf 29,8 % (vgl. Kantar Worldpanel 2021).

Die Popularität von Cloud-basierten Speicherlösungen steigt wegen der einfachen Nutzung und der scheinbar kostengünstigen Datensicherung immer weiter. Dokumente werden dabei oft ausschließlich auf den Servern der Anbieter gespeichert oder mit diesen synchronisiert, während die lokale Speicherung auf dem PC oder Smartphone, auch aufgrund der steigenden Integration von online-Lösungen in die Betriebssysteme und der daraus folgenden einfachen Bedienung, immer seltener wird und infolgedessen oft keine offline-Backups angelegt werden. Bei einer Umfrage aus dem Jahr 2020 in Deutschland gaben 35 % der Befragten an, dass sie ihre Dokumente und Bilder online speichern, während 25 % Cloud-Backups verwendeten (vgl. Statista Global Consumer Survey 2020b). Auch wenn diese Methode eine sinnvolle Möglichkeit zum Speichern nicht vertraulicher Daten darstellt, sollte sie aufgrund von Sicherheitsmängeln, wie beispielsweise der Speicherung des privaten Schlüssels auf dem Server oder sogar das Fehlen einer Verschlüsselung (vgl. Menn 2020), nicht für Backups aller Dateien oder vollständige Betriebssystembackups eines Smartphones genutzt werden. Neben der möglichen höheren Sicherheit von lokalen Backups können auch die mit der Datenmenge steigenden Kosten im Vergleich zu Cloud-Lösungen eingespart werden.

Da Android-Geräte im Gegensatz zu iOS-Smartphones schon die Möglichkeit bieten, auf einen Großteil des Dateisystems und somit die wichtigsten Daten über einen Computer zuzugreifen, ist eine lokale Alternative zu Cloud-Speichern vor allem für iOS-basierte Geräte relevant.

Diese Studienarbeit thematisiert die Entwicklung und Programmierung eines Ladegerätes, das bei jedem Ladevorgang von iOS-Geräten wie dem *iPad* oder *iPhone* ein komplettes, iTunes-kompatibles System-Backup des Geräts erstellt. Dabei handelt es sich um vollständige und wahlweise verschlüsselte Backups des Systems und der Dateien, die mit der Software *iTunes* von Apple unter Microsoft Windows oder dem Apple *Finder* bei Verwendung eines aktuellen macOS-Systems kompatibel sind und somit einfach wiederhergestellt werden können.

Dazu wird im Ladegerät oder dem daran angeschlossenen Kabel ein Speichermedium integriert, auf dem die Sicherungen abgelegt werden können und ein Einplatinencomputer, der das Erstellen und die Verwaltung der Backups steuert.

Die vorliegende Arbeit fasst zuerst aktuelle und frühere Entwicklungen ähnlicher Geräte zusammen. Anschließend wird auf bestehende Möglichkeiten zum Erstellen von Backups eingegangen. Den Kern bildet die technische Umsetzung des Geräts und dessen Software. Zuletzt folgt ein Ausblick über den Optimierungsbedarf und eine mögliche Veröffentlichung.

Die Entwicklung und Konstruktion eines Systemgehäuses für das Ladegerät werden in einer separaten Arbeit behandelt (vgl. Steinbeck 2021).

2. Entwicklungsstand

In der Vergangenheit wurde oft der Versuch unternommen, Geräte zur unkomplizierten Erstellung lokaler Backups für Mobilgeräte zu entwickeln. Folgend werden einige Produkte, die den Anforderungen dieser Arbeit ähneln, beschrieben und deren jeweilige Nachteile oder Inkompatibilitäten mit der angestrebten Umsetzung herausgearbeitet.

2.1. Patente für Backup-Ladegeräte

Ladegeräte, die ein Backup der Smartphone-Daten erstellen, nehmen aufgrund der fortschreitenden Verbreitung von Cloud-Backups eine untergeordnete Rolle auf dem Markt ein. Um einen besseren Überblick über Produkte zu erhalten, die diese Funktion erfüllen, werden folgend Patente aufgeführt, die das Ziel eines lokalen Backups mittels einem in ein Ladegerät integrierten Speicher verfolgen. Diese sind in der Datenbank des Online-Portals *Espacenet* enthalten, die Daten von weltweiten Patentanmeldungen enthält und sich somit zur Analyse des aktuellen Entwicklungsstands eignet.

2.1.1. Patent JP2000324237A

Charger with data backup function for mobile phone and data backup unit connected to the charger (2000)

Die Entwicklung von Backup-Ladegeräten für Handys begann schon viele Jahre vor der Verbreitung von Cloud-Backups und sogar dem Erscheinen der ersten Smartphones. Im Gegensatz zu aktuellen Lösungen lag der Fokus hauptsächlich auf der unkomplizierten Sicherung, die wegen der damals kaum vorhandenen Cloud-Lösungen für Privatanwender, insbesondere wegen des Fehlens von online-Backup-Funktionen, lokal realisiert werden musste.

Das Patent *JP2000324237A*, das Ende des Jahres 2000 veröffentlicht wurde, stellt eine der ersten Lösungen zur unkomplizierten Sicherung eines Mobiltelefons dar. Das Ziel ist, eine Alternative zum nur für erfahrene Nutzer möglichen Backup eines Handys am Computer zu schaffen. Dazu wird ein Ladegerät verwendet, das den Anschluss eines externen Speichermediums erlaubt. Das Backup erfolgt somit bei jedem Ladevorgang des mobilen Endgeräts automatisch und erfordert geringe Nutzerintervention. Eine Synchronisation von Daten ist hierbei vom Handy auf das Ladegerät und in umgekehrter Richtung möglich. Außerdem ist die Sicherung mehrerer Geräte, die durch eine Identifikationsnummer unterschieden werden können, möglich. Neben der Anschlussmöglichkeit an haushaltsübliche Stromnetze kann das Gerät mit einem KFZ-Ladeadapter oder einem externen Akkumulator verbunden werden. Der Status des Ladevorgangs wird durch eine Kontrollleuchte signalisiert. Zur Erstellung eines Backups muss eine PIN am Mobilgerät eingegeben werden (vgl. Goddo KK 2000).

Diese Lösung ist wegen der nicht vorhandenen Kompatibilität mit aktuellen Mobiltelefonen aufgrund von aktuellen Standards der Dateisysteme und Hardware-Anschlüsse nicht mehr in der

vorgesehenen Form nutzbar. Außerdem fehlen Informationen darüber, ob eine Verschlüsselung der Nutzerdaten stattfindet.

2.1.2. Patent WO03005690A1

Data transfer device for connection of charging adapter for mobile telephone (2003)

Das Ziel einer unkomplizierten Backup-Möglichkeit wird auch im Patent *WO03005690A1* aus dem Jahr 2003 verfolgt. Als Alleinstellungsmerkmale dienen bei diesem Gerät die Möglichkeit eines Backup-Exports und der einfache Transfer von Daten zwischen Handy und Ladegerät. Da das Gerät einen internen Speicher besitzt, ist ein Backup mit oder ohne die Verwendung eines externen Speichermediums möglich. Das Gerät besitzt ein eingebautes Statusdisplay zur Visualisierung der wichtigsten Daten. Außerdem soll der aktuelle Status auf dem Bildschirm des Mobiltelefons angezeigt werden. Ein Ladeadapter ist nicht integriert, jedoch stellt ein Spannungs-Umschalter die Möglichkeit zur Verwendung einer Vielzahl von Netzteilen mit unterschiedlichen Spezifikationen sicher. Es können auch hier Adapter für KFZ und Batterien verwendet werden. Als Speichermedium wird optional eine Speicherkarte verwendet (vgl. Kobayashi 2003a).

Neben den gleichen Nachteilen wie das vorherige Patent ist hier die manuelle Bedienung des Spannungs-Umschalters zu erwähnen, die für unnötige Komplexität bei der Bedienung sorgt und zu einer Fehlbedienung führen kann.

2.1.3. Patent US2003098670A1

Cellular phone charger with data backup function and cellular phone data backup device (2003)

Im Vergleich zu den erwähnten Patenten ist hier die Integration von zwei Knöpfen im Ladegerät zum Erstellen und Wiederherstellen von Sicherungen vorhanden. Diese sorgen für eine sehr einfache Bedienmöglichkeit für den Endnutzer. Es werden zwei mögliche Umsetzungen beschrieben: die Integration aller Komponenten in eine Ladestation, in die das Gerät eingelegt werden kann oder ein vom Backup-Gerät getrennter Netzstecker. Die Sicherung erfolgt entweder inkrementell, sodass nur geänderte Daten gesichert werden oder bei jedem Ladevorgang vollständig (vgl. Kobayashi 2003b).

Aufgrund der einfachen Bedienung ist diese Lösung gut für unerfahrene Nutzer geeignet. Auch hier ist die Inkompatibilität mit aktuell verbreiteter Hardware zu erwähnen, durch die sich das Gerät heute nicht nutzen lässt.

2.1.4. Patent JP2004032480A

Battery charger of portable telephone (2004)

Während die automatische Backup-Funktion ähnlich zu den oben aufgeführten Geräten funktioniert, wird für die Wiederherstellung der Daten ein anderer Ansatz verfolgt: Die Lösung zielt

primär auf eine Wiederherstellung bei Verlust des Handys oder einem nicht behebbaren Schaden. Auch hier wird bei jedem Laden des Geräts eine Sicherung über einen im Ladeadapter integrierten Mikrocontroller erstellt, jedoch erfolgt die Wiederherstellung auf andere Art: Im Ladegerät ist ein - eventuell portables - Speichermedium integriert, das bei Verlust oder Schaden des alten Handys zu einem Händler gebracht wird, der die Wiederherstellung des Backups auf einem neuen Handy mit Hilfe von spezieller Hardware durchführt (vgl. Nakajima Tsushinki Kogyo KK 2004).

Während diese Methode die Wahrscheinlichkeit eines Fehlers bei der Bedienung durch den Nutzer minimiert, ist hier eine Bindung an den Händler vorhanden. Außerdem kann nicht sichergestellt werden, dass die Daten nicht auf dem Computer, der zur Wiederherstellung genutzt wird, gespeichert werden. Dies kann zu eingeschränkter Datensicherheit und Privatsphäre führen.

2.1.5. Patent JP2004274528A

Charging apparatus and method (2004)

Auch bei diesem Patent wird ein Ladegerät mit Speichermedium beschreiben, das bei jeder Dateiänderung auf dem Mobilgerät ein automatisches Backup erstellt. Hier liegt der Fokus besonders auf der Unterstützung mehrerer Handys. Dazu wird ein Identifikationsmerkmal benötigt, das sich bei jedem Mobiltelefon unterscheidet. Dieses lässt sich beispielsweise durch die Verwendung der Handynummer, die auf dem Gerät gespeichert ist und im Normalfall nur einmal vergeben wird, realisieren. Wenn die auf dem Ladegerät gespeicherten Identifikationsinformationen mit denen des Mobilgeräts übereinstimmen, wird eine Datensicherung erstellt. Zum Zurücksetzen kann ein Schalter auf dem Ladegerät verwendet werden (vgl. TDK Corp. 2004).

Der Ansatz eignet sich grundsätzlich für die Umsetzung eines Smartphone-Backup-Ladegeräts. Es wird jedoch kein vollständiges Systembackup erstellt.

2.1.6. Patent US2006158154A1

Method and apparatus for backing up data from cell phones and other hand-held devices (2006)

Hier soll eine Möglichkeit zur automatisierten Datensicherung eines Mobilgeräts, beispielsweise eines Handys oder einer Kamera, beim Laden in einer Ladestation geschaffen werden. Beim Anschluss soll entweder ein Backup oder eine 2-Wege-Synchronisation der gespeicherten Daten erfolgen. Der Beginn der Übertragung erfolgt hierbei entweder automatisch beim Verbinden oder manuell. Sollte eine Synchronisierung statt eines Backups implementiert werden, werden die Dateien beider Datenspeicher verglichen und jeweils die neuste Datei auf das andere Gerät kopiert. Fehlende Dateien werden ebenfalls auf das entsprechende Gerät kopiert. Bei automatisierter Backup-Funktion wird die Synchronisation von der Ladestation aus gestartet, sobald ein Gerät durch einen Sensor erkannt wird. Bei Anschluss der Ladestation an einen PC kann eine

Datensicherung entweder auf den integrierten Speicher der Station übertragen werden oder auf den des PCs. Die Erkennung mehrerer Geräte ist möglich, wobei dann die Daten entweder auf separaten Speichermodulen oder in separaten Ordnern gesichert werden. Bei jedem Ladevorgang muss hierfür eine Identifikation des verwendeten Geräts erfolgen. Bei Verlust des Mobilgeräts oder beschädigten Dateien erfolgt beim Anschluss eine automatische Übertragung von der Station auf das Mobilgerät (vgl. Maurilus 2006).

Dieses Gerät ist im Vergleich zur angestrebten Lösung verstärkt auf eine Dateisynchronisation und nicht auf eine Sicherung fokussiert. Die Problematik besteht hierbei in der Übertragung geänderter Dateien vom Ladegerät auf das Mobilgerät, was ein ungewolltes Überschreiben zur Folge haben kann. Auch die Kompatibilität mit vielen Geräten und Unterstützung mehrerer Modi führt zu einer hohen Bedienkomplexität.

2.1.7. Patent AU2008320924A1

Cable with memory (2009)

Dieses Patent wurde nach der Vorstellung des ersten iPhones im Januar 2007 veröffentlicht und zielt somit als erstes hier erwähntes Patent auf die Nutzung mit modernen Smartphones statt einfacher Handys.

Das Patent beschreibt ein Smartphone-Ladekabel, das zwischen den zwei Anschlussenden ein Speichermodul integriert hat. Außerdem wird der Ansatz eines modularen Datenspeichers verfolgt, der die Nutzung eines wechselbaren Speichermediums mit mehreren Geräten erlauben soll (vgl. Goel 2009).

Das Ladegerät soll mit aktuellen Smartphones kompatibel sein und sowohl iOS-, als auch Android-Geräte unterstützen. Die Datensicherungen enthalten dabei ausgewählte Dateien und bilden kein vollständiges Systembackup. Es handelt sich um eine Lösung ohne Stromzufuhr oder integrierten Ladecontroller. Der Datentransfer kann bidirektional erfolgen, was eine Sicherung oder das Wiederherstellen einzelner Dateien ermöglicht. Im Vergleich zu vorherigen Lösungen kann aufgrund des erwähnten Ausbleibens eines Netzsteckers eine sehr kompakte Bauform des Geräts umgesetzt werden. Im Kabel ist ein nicht weiter beschriebenes Speichermodul enthalten. Dieses kann entweder nicht entfernbar oder beispielsweise in Form eines wechselbaren Flash-Speichermediums umgesetzt werden. Ein integrierter Mikrocontroller ist laut Beschreibung für viele Funktionen des Geräts nicht zwingend notwendig, wenn die Steuerung des Geräts vom Smartphone aus erfolgt. Wird ein Mikrocontroller eingesetzt, kann die Kontrolle der Datenspeicherung folglich von diesem statt vom Smartphone übernommen werden. Das Gerät wird auf einer Seite mit einer Stromquelle wie beispielsweise einem separaten Netzteil oder einem Computer über die USB-Schnittstelle und mit einem zweiten Stecker mit einem kompatiblen Smartphone verbunden. Die Steuerung der Datensicherung und -wiederherstellung wird hier durch Eingabe am Smartphone oder automatisch beim Anschluss eines Geräts übernommen. Die Verbindung mit einer externen Stromquelle ist hierbei optional. Nach dem Backup

oder bei Inkompatibilität mit dem Smartphone könnte das Kabel wie ein normales USB-Verbindungskabel genutzt werden. Bei Umsetzung mit einem wechselbaren Speichermedium sollte dieses bevorzugt eine mit dem Smartphone kompatible Form haben, beispielsweise eine Mikro-SD-Karte, die in das Smartphone eingesteckt wird. Dabei sollten alle wichtigen Nutzerdaten auf dem wechselbaren Speichermedium gespeichert werden. Somit wäre die Nutzung dieser SD-Karte mit anderen Mobilgeräten möglich, beispielsweise bei leerem Akku oder Schaden am Gerät (vgl. Goel 2009).

Dieser Ansatz verfolgt hauptsächlich das Ziel eines einfachen Backups unter Vernachlässigung von Sicherheit und Privatsphäre des Nutzers. Es wird nicht auf eine mögliche Verschlüsselung zum Schutz der Nutzerdaten eingegangen. Auch der Vorschlag, die SD-Karte bei leerem Akku mit einem Gerät einer anderen Person zu nutzen, zeigt den fehlenden Fokus auf Datensicherheit, da diese Daten vom anderen Nutzer beabsichtigt oder unbeabsichtigt durch ein Schadprogramm kopiert werden können. Die Ziele werden heute durch die Existenz von Cloud-Backups, die zur Datenübertragung auf neue Geräte oder Kopie auf mehrere Smartphones genutzt werden können, fast vollständig ersetzt. Auch die weite Verbreitung mobiler, externer Batterien zum Laden von Smartphones senkt die Wahrscheinlichkeit eines leeren Akkus.

Aufgrund dieser Nachteile, der heute starken Zugriffsbeschränkungen des iOS-Speichers über USB und der fehlenden Möglichkeit zur Verwendung eines wechselbaren Speichermediums mit iOS-Geräten mit Zugriff auf relevante Daten sind die Funktionen heute stark eingeschränkt.

2.1.8. Patent US2012330888A1

Data backup device (2012)

Eine Lösung, die das Ziel einer Offline-Dateisynchronisation als Alternative zu Online-Sicherungen verfolgt, wird hier beschreiben. Das Produkt ist auf portable Geräte wie Smartphones oder Tablets spezialisiert, könnte aber auch mit Kameras oder anderen portablen Geräten verwendet werden. Es soll neben der Synchronisationsmöglichkeit eine unkomplizierte Alternative zu manuellen, lokalen Backups geschaffen werden, wobei die Sicherheits- und Datenschutzbedenken bei der Nutzung von Cloud-Backups ausbleiben (vgl. Cruz 2012).

Das Backup wird wie bei den anderen Geräten durch Datenübertragung beim Laden erstellt. Neben der Stromzufuhr zum Laden und einem eventuell benötigten zweiten Anschluss für die Datenübertragung zwischen den Geräten ist ein internes Speichermedium und eine kabellose Schnittstelle zur Kommunikation mit dem Gerät im lokalen Netzwerk vorhanden. Dabei wird dem Ladegerät eine eigene IP-Adresse zugewiesen, was einen Datenaustausch über das Netzwerk erlaubt. Somit ist die Interaktion mit einem zweiten Computer oder beispielsweise der Download eines bestehenden Cloud-Backups möglich. Mit dieser Methode können auch Daten auf das Ladegerät aufgespielt werden, die beim nächsten Ladevorgang synchronisiert werden. Zum Erstellen einer Sicherung werden die Daten des aktuellen Backups beim Verbinden mit denen auf dem Gerät verglichen. Die Wahl der Parameter wie Häufigkeit der Sicherung oder

einzuschließende Verzeichnisse erfolgt über ein Programm auf dem Endgerät. Das Ladegerät selbst kann auch einige Parameter des Backups, wie die Verwaltung der Rechte verschiedener Geräte, steuern. Statt einer reinen Backup- und Wiederherstellungsfunktion können drei Modi zur Datenübertragung gewählt werden: Neben der Erstellung eines Backups können Daten vom Ladegerät auf ein neues Gerät übertragen werden oder Daten zwischen beiden Geräten synchronisiert werden, sodass am Ende alle Daten auf beiden Geräten vorhanden sind, wobei jeweils die neuste Datei bestehen bleibt.

Die Synchronisation soll herstellerübergreifend möglich sein, wobei als Beispiel die Synchronisierung von Kontakten zwischen Android- und iOS-Smartphones aufgeführt wird. Ein Datenabgleich mit weiteren Geräten wie beispielsweise einem Fernseher zur Übertragung von Aufnahmen wird auch angestrebt. Zudem ist ein Austausch über eine USB-Schnittstelle mit dem Computer möglich, womit auch automatisierte Prozesse, wie die Kontaktsicherung eines per Mail gesendeten Kontakts, ermöglicht werden können. Ein Backup mehrerer Geräte mit dem gleichen Ladegerät ist auch denkbar (vgl. Cruz 2012).

Wie im Patent *US2006158154A1* (Kapitel 2.1.7) steht auch hier die Dateisynchronisation im Vordergrund. Das Gerät bietet viele Möglichkeiten zur Sicherung und Synchronisation, wodurch die Bedienung und Einrichtung komplex ist. Weitere genannte Nachteile einer Dateisynchronisierung treten hier ebenfalls auf.

2.1.9. Patent EP3236358A1

Data backup and charging device for communication devices (2017)

Hier soll auch das Problem der Notwendigkeit eines zusätzlichen Geräts oder einer Abhängigkeit vom Mobilfunknetz zum Erstellen eines Daten-Backups gelöst werden. Die Technik zum Laden und Erstellen der Sicherung wird in einem einzigen Gehäuse untergebracht. Die Stromversorgung des Speichers erfolgt entweder durch das Mobilgerät oder das Netzteil. Eine kabellose Verbindung zum Laden und Sichern ist denkbar. Eine Verschlüsselung eines Teils der Dateien und eine Kompression ist möglich. Der Typ dieser kann dabei vom Nutzer selbst gewählt werden und die Entschlüsselung erfolgt mittels eines selbst gewählten Passworts. Die Auswahl der zu sichernden Daten erfolgt möglicherweise über ein Programm auf dem Smartphone. Jedes angeschlossene Gerät kann über einen „Unique Identifier“, also eine eindeutige und gerätespezifische Zeichenfolge identifiziert werden. Optional ist eine Datenübertragung auf ein zweites Mobilgerät möglich. Die Wiederherstellung einer Sicherung kann auf jedem Gerät erfolgen, solange dessen freier Speicher größer als der benötigte Speicher des Backups ist. Bei nicht ausreichendem Speicherplatz ist zudem ein partieller Restore möglich. Außerdem ist die Integration eines Akkus im Netzteil denkbar. Ein Statusindikator, der beispielsweise den Ladestatus anzeigt, kann auch im Gerät verbaut werden. Das Gerät soll mit einer Vielzahl verschiedener Mobilgeräte wie Smartphones und Tablets kompatibel sein. Als internes Speichermedium wird ein nonvolatiler und nicht wechselbarer Speicher verwendet. Optional wird auch ein Modul zum

Senden von Standortdaten des Ladegeräts an das Smartphone verbaut, das bei Verlust oder Diebstahl des Geräts verwendet werden kann. Die Backups erfolgen nach Erstellen eines vollen Initialbackups entweder inkrementell oder differenziell. Es können außerdem mehrere Backup-Versionen des gleichen Geräts gesichert werden. Umgekehrt ist auch die Verwendung mehrerer Ladegeräte mit dem gleichen Mobilgerät möglich (vgl. Lior/Israeli 2017).

Das Ladegerät wird vom Mobilgerät als externes Speichermedium erkannt, wodurch nicht alle Speicherorte des Dateisystems zugänglich sind. Außerdem ist ein Programm auf dem mobilen Endgerät zwingend notwendig, um das Backup durchzuführen. Der im Patent aufgeführte Ansatz eines manuellen und zeitplanabhängigen Backups unterscheidet sich deutlich vom Erstellen eines Backups bei jedem Ladevorgang und erfordert einen im Vergleich hohen Konfigurations- und Wartungsaufwand. Auch die Wahl vieler Parameter wie beispielsweise der Typ der Verschlüsselung macht das Gerät eher für erfahrene Nutzer geeignet.

2.1.10. Patent DE102019000928A1

Verfahren für ein dezentrales Backupsystem im Mehrfach-USB-Ladegerät (2020)

Dieses Patent befasst sich mit der Entwicklung eines dezentralen Backup-Ladegeräts, das eine einfache und sichere Alternative zu Cloud-Diensten schaffen soll. Hier liegt der Fokus vor allem auf der einfachen Bedienbarkeit und Datensicherheit im Vergleich zu Cloud-Diensten. Das Gerät soll nicht mobil sein, um die Wahrscheinlichkeit eines Diebstahls oder Verlust zu minimieren und so geringe Kosten aufweisen, dass der Nutzer Geräte an mehreren Standorten platzieren kann. Es erfolgt eine symmetrische Verschlüsselung aller Ordner. Auch eine Synchronisation von unter mehreren Anwendern geteilten Ordnern ist möglich, die beispielsweise über das interne Netzwerk eines Unternehmens abgeglichen werden. Zum Wiederherstellen eines Backups und vor Erstellen der ersten Sicherung muss das Gerät durch einen Sicherheitsmechanismus, beispielsweise in Form eines Fingerabdrucksensors, freigeschaltet werden. Die Synchronisation eines Ordners zwischen Ladegerät und Mobilgerät erfolgt, wenn dieser auf beiden Geräten das gleiche Identifikationsmerkmal aufweist. Auf jedem zu synchronisierenden Endgerät wird ein Programm benötigt, das für den Backup-Prozess und das Erstellen der Ordner verwendet wird. Jedem Gerät wird außerdem eine eindeutige Identifikationsnummer zugewiesen, um den Backup-Prozess zu automatisieren (vgl. Berberich 2020).

Hier liegt der Fokus vor allem auf der Sicherung einzelner, ausgewählter Dateien. Auch eine Synchronisation zwischen mehreren Geräten zur Kollaboration soll umgesetzt werden (vgl. Berberich 2020). Es handelt sich also nicht um eine reine Backup-Lösung für Privatanwender, sondern hauptsächlich eine Synchronisationslösung mit Verschlüsselung, die beispielsweise in einem Haushalt oder wahrscheinlicher in Unternehmen genutzt werden kann. Eine Problematik ist hierbei ebenfalls der komplexe Einrichtungs- und Wiederherstellungsprozess. Auch ein vollständiges Systembackup wird hier nicht umgesetzt, da nur selektiv synchronisiert wird.

2.2. Open-Source-Projekte

Die Automatisierung von Backup-Prozessen wurde auch von einigen Open-Source-Projekten thematisiert. Die veröffentlichten Lösungen wenden sich allerdings hauptsächlich an erfahrene Nutzer, die den Prozess automatisieren wollen und stellen keine einfache Lösung zur Sicherung des Geräts dar. Auch der Wiederherstellungsprozess ist meist nicht beschrieben.

2.2.1. iPiBackup

Ein Ansatz zum automatisierten Erstellen eines iPhone-Backups während des Ladevorgangs verfolgt auch das auf der Webseite *github.com*, auf der Software-Quellcode veröffentlicht und verwaltet werden kann, verfügbare Projekt *iPiBackup*. Die letzte Aktualisierung erfolgte im September 2019 und das Projekt ist in der bereitgestellten Form mit aktueller iOS-Version nicht nutzbar. Eine Anleitung zur Verwendung des Codes mit einem Einplatinencomputer wie dem *Raspberry Pi Zero* ist enthalten. Das Programm erstellt mithilfe des Programms *idevicebackup2*, das in der Software-Bibliothek *libimobiledevice* (vgl. Kapitel 2.3) enthalten ist, ein Backup des iPhones auf der SD-Karte eines Computers, beispielsweise des *Raspberry Pi Zero*. Das Backup wird gestartet, nachdem das Mobilgerät mit dem Einplatinencomputer verbunden wurde und wird somit bei jedem Ladevorgang erstellt. Zum Anschluss des iPhones wird der Mikro-USB-Port des *Raspberry Pi* verwendet, wofür ein Adapter auf USB benötigt wird. Die benötigte Software erfolgt über den Paketmanager des Betriebssystems (vgl. Perez 2018a). Damit das Backup automatisiert erstellt wird, müssen weitere Schritte wie das Erstellen des Ziel-Dateisystems und -Ordners und das automatische Einhängen dieses durchgeführt werden, die für viele unerfahrenen Nutzer zu komplex oder aufwändig sind oder dem Ziel eines einfachen Backups widersprechen.

Zum Erstellen einer Systemsicherung muss nach der Einrichtung das Smartphone an den USB-Port des *Raspberry Pi* angeschlossen werden und bei erster Verbindung der Code zur Entschlüsselung des iPhones eingegeben werden. Daraufhin wird der Sicherungsprozess automatisch gestartet und die Dateien werden auf der SD-Karte des Einplatinencomputers gespeichert (vgl. Perez 2018b). Eine Verschlüsselung ist standardmäßig nicht aktiviert. Auch der Zugriff auf die Backup-Daten in einer nach Medientyp oder App geordneten Struktur ist nicht möglich.

Zur Wiederherstellung kann laut Beschreibung des Projekts entweder das Programm *idevicebackup2* direkt auf dem *Raspberry Pi* oder die von Apple zur Verfügung gestellte Software verwendet werden. Aufgrund der Verwendung des FAT-Dateisystems zur Speicherung der Daten kann die SD-Karte ohne zusätzliche Programme mit macOS, Linux und Windows verwendet werden (vgl. Perez 2018c).

Das Fehlen der Verschlüsselung, die Komplexität des Einrichtungsprozesses und zusätzliche Schritte zur Verwendung der aktuellen iTunes-Version machen diese Lösung nur für Nutzer zugänglich, die sich mit Linux-basierten Systemen wie dem verwendeten Raspbian (heute

Raspberry Pi OS) auskennen. Auch die Installation von Betriebssystem- und Softwareupdates, um die Sicherheit des Systems und Kompatibilität mit neuen iOS-Versionen zu gewährleisten, wird hier nicht beschrieben.

2.2.2. Raspberry-Pi-for-iPhone-Backup

Das Projekt *Raspberry-Pi-for-iPhone-Backup*, das von Justin Pearson auf *github.com* veröffentlicht wurde, verwendet einen *Raspberry Pi Zero* zum Erstellen eines iOS-Backups. Im Vergleich zur oben genannten Lösung wird hier nicht *idevicebackup2* verwendet, sondern *ifuse*, was ebenfalls als Teil von *libimobiledevice* installiert wird. Zusätzlich zur Backup-Funktion ist eine Statusanzeige in Form mehrerer LEDs vorhanden, die den aktuellen Status der Verbindung, des Backups und weitere Parameter visualisiert (vgl. Pearson 2018a). Da das Projekt aktuelle iOS-Versionen unterstützen soll, wird der aktuelle verfügbare Quellcode von *libimobiledevice*, der nicht über den Paketmanager von der auf den *Raspberry Pi* angepassten Linux-Distribution *Raspbian* installiert werden kann, auf das Gerät geladen und dort kompiliert (vgl. Pearson 2018b). Der Start des Backups erfolgt mithilfe eines Skripts, das gestartet wird, sobald ein iOS-Gerät mit dem Raspberry Pi verbunden wird. Zum Verwenden des Projekts muss das iOS-Gerät nach der Installation der benötigten Pakete mit dem *Raspberry Pi* verbunden werden. Anschließend wird das iOS-Dateisystem mit dem Programm *ifuse* in das des Einplatinencomputers eingehängt. Somit sind Daten verschiedener Apps, Downloads, Fotos und weitere wichtige Dateien zugänglich. Nach dem Einhängen muss eine Kopie der Daten auf die SD-Karte erfolgen, woraufhin das Dateisystem wieder ausgehängt werden kann (vgl. Pearson 2018c).

Damit dieser Prozess später automatisiert erfolgt, wird ein Programm zur Verfügung gestellt, das die Schritte beim Erkennen eines kompatiblen iOS-Geräts durchführt. Die Nutzung der Status-LEDs erfordert weitere Schritte, auf die hier aufgrund des komplexen Einrichtungsprozesses nicht eingegangen wird (vgl. Pearson 2018d).

Hier bestehen neben den gleichen Problemen wie die des Projekts *iPiBackup* folgende Schwierigkeiten: Das Programm *ifuse* kann nur teilweise auf den internen Speicher des iOS-Geräts zugreifen, da nur bestimmte Apps den Zugriff erlauben. Außerdem sind, auch wenn eine App grundsätzlich Dateizugriff gewährt, nicht alle Nutzerdaten im zugänglichen Teil des Systems gespeichert. Zudem fehlen Backups der Einstellungs-Daten, Online-Accounts und weiterer Teile des iOS-Betriebssystems. Das Programm ist somit nicht als alleinige Backup-Lösung geeignet und muss durch weitere, vollständige Systemsicherungen komplementiert werden. Ein Vorteil im Vergleich zum oben genannten Projekt besteht jedoch in der einfachen Zugänglichkeit der Dateien und einer nach Apps oder Dateityp geordneten Ordnerstruktur.

2.3. Problematik bestehender Lösungen

Aufgrund des stark limitierten Zugriffs auf das iOS-Dateisystem bei Verbindung über USB ist ein Backup der darüber einfach zugänglichen Daten oft nicht ausreichend. Selbst wenn alle Daten zugänglich wären, wäre die Wiederherstellung durch viele manuelle Eingriffe wie der Installation von Apps und die Anmeldung verschiedener Online-Benutzerkonten für die meisten Endanwender zu aufwändig oder komplex, weshalb diese Lösung unpraktisch ist. Somit verbleibt nur die Nutzung von Apples nativem Backup-Prozess, der neben der Speicherung von App-Daten vieler Apps auch eine einfache Möglichkeit zur kompletten Wiederherstellung der Nutzerdaten bietet. Da der Quellcode dieser Lösung aber nicht öffentlich zugänglich ist und offiziell nur auf Windows-Computern oder Macs benutzt werden kann, ist hier die Integration problematisch.

Es existieren nur wenige veröffentlichte Projekte, die das Apple-Protokoll beispielsweise durch die Verwendung des Programms *idevicebackup2* nutzen. Bei den genannten kommerziellen Lösungen sorgen Probleme wie die Unvollständigkeit der Backups, fehlende Verschlüsselung, keine Unterstützung für aktuelle Betriebssysteme oder die Notwendigkeit manueller Eingriffe, beispielsweise über Apps, für eine zu starke Limitierung des Funktionsumfangs bei alltäglicher Nutzung.

Auch die Open-Source-Lösungen weisen Mängel auf, die das Nutzererlebnis und die Sicherheit stark einschränken können. Bei aktuellen Projekten fehlt beispielsweise eine standardmäßig aktivierte Verschlüsselung, was gerade bei unerfahrenen Nutzern zu einer geringeren Sicherheit als beim Verwenden von Cloud-Lösungen sorgen kann.

Diese Einschränkungen sind wahrscheinlich vor allem der fehlenden öffentlichen Dokumentation von Apples Backup-Protokoll und der nicht vorhandenen Unterstützung für Linux-Systeme geschuldet.

In den folgenden Kapiteln werden bestehende Möglichkeiten zur Sicherung eines iOS-Geräts aufgeführt und ein möglicher Lösungsansatz für ein Gerät, das ohne Aufweisen der oben genannten Mängel funktionieren soll, beschrieben.

3. Backup-Software

Die einzige offiziell unterstützte Möglichkeit zum Erstellen von iOS-Backups ist die Nutzung von Apple-Software. Jedoch haben sich aufgrund fehlender Funktionen oder keiner Unterstützung von Betriebssystemen außer Microsoft Windows oder Apple macOS Alternativen wie die Open-Source-Bibliothek *libimobiledevice* etabliert.

3.1. Offiziell unterstützte Möglichkeiten zur Sicherung von iOS-Geräten

Mit dem aktuell schnellen Fortschreiten der Digitalisierung von wichtigen Dokumenten und Dateien nehmen Backups eine immer bedeutendere Rolle ein. Im folgenden Kapitel wird auf den Stand der Technik bei der Erstellung von Backups für Apple-Geräte mit dem iOS-Betriebssystem und auf die Verfügbarkeit von momentan am Markt erhältlichen Backup-Ladegeräten eingegangen.

Aktuell werden von Apple offiziell zwei Optionen zum Erstellen von kompletten System-Backups angeboten, die neben den persönlichen Dateien auch eine Sicherung betriebssystemrelevanter und versteckter Dateien beinhalten (vgl. apple.com 2021a):

3.1.1. Online-Backups über iCloud

Eine Möglichkeit ist die Sicherung des Smartphones in Apples eigenem Cloud-Service iCloud. Diese Backups werden laut Apple verschlüsselt und sind nur für den Besitzer des iCloud-Accounts zugänglich. Eine Ende-zu-Ende-Verschlüsselung, die essenziell für die Datensicherheit beim Übertragen ist, wird nicht vollständig unterstützt. Während ein Teil der Daten Ende-zu-Ende-verschlüsselt übertragen werden, sind beispielsweise E-Mails und Dateien beim Zugriff über *icloud.com* ausgeschlossen, was ein erhöhtes Sicherheitsrisiko mit sich bringt. Ein weiterer wichtiger Punkt ist hier die 2-Faktor-Authentifizierung, ohne die die Ende-zu-Ende-Verschlüsselung komplett deaktiviert ist (vgl. apple.com 2020a).

Die Option, einzelne Dateien wiederherzustellen oder zu betrachten besteht bei dieser Möglichkeit des Backups nicht. Außerdem wird zwingend ein iPhone oder iPad benötigt, um bei Datenverlust wieder Zugang zu den Dateien zu erlangen. Bei Verlust der Zugangsdaten werden Möglichkeiten zum Zurücksetzen dieser geboten, was die Sicherheit der Daten durch mehr Angriffsmöglichkeiten beeinträchtigt (vgl. apple.com 2020b). Während alle relevanten Dateien wie Fotos, Nachrichten etc. gespeichert werden, handelt es sich beim iCloud-Backup nicht um ein vollständiges Backup inklusive aller Systemdateien. Es werden unter anderem Einstellungen, die Anordnung der Apps auf dem Home-Screen, Nachrichten, Fotos und Videos, Klingeltöne und getätigte Käufe im Backup eingeschlossen (vgl. apple.com 2021b). Dabei entscheiden jedoch die App-Entwickler, ob ihre App-Daten im Backup enthalten sein dürfen oder ausgeschlossen werden. So schließen manche Entwickler meist aufgrund von Sicherheitsbedenken oder Software-Limitierungen des iOS-Systems ihre Applikationen von Backups aus. Als Beispiel kann hier die

App *Authenticator* genannt werden (vgl. Rubin 2014). Ausgeschlossen vom Backup sind außerdem Daten, die schon separat in iCloud gespeichert sind, beispielsweise Kontakte oder Bilder.

Die im Backup eingeschlossenen Daten umfassen nur Dateien, die zum Sicherungszeitpunkt auf dem Mobilgerät gespeichert sind. Lokal gelöschte Dateien werden bei der nächsten Sicherung auch am Sicherungsziel gelöscht. Die Größe des Backups hängt vom aktuell belegten Speicher auf dem iOS-Gerät ab. Es wird immer nur die neuste Version des Backups zur Verfügung gestellt, weshalb sich die Eignung auf den Wechsel, Verlust oder die Funktionsfehler des Smartphones beschränkt. Eine automatische, tägliche Erstellung von Backups ist bei dieser Variante möglich. Manche Apps implementieren ihre eigenen Backup-Lösungen, weshalb die Einheitlichkeit fehlt, was beim Endnutzer zu unnötiger Komplexität bei der Wiederherstellung führt. Die Wiederherstellung kann hier nur über eine WLAN-Verbindung und den Download des kompletten Backups erfolgen. Eine Bindung an das Gerät besteht nicht, weshalb der Restore auf dem gleichen Gerät erfolgen, aber auch beispielsweise zur Einrichtung eines neuen Geräts verwendet werden kann, welches nicht vom gleichen Typ sein muss. Bei Fehlschlagen oder Verlust aller Zugangsdaten gibt es keine Möglichkeit zur Wiederherstellung der Daten. Selbstständige Fehlerbehebung ist somit nicht möglich.

3.1.2. Lokale Backups über iTunes

Neben der Möglichkeit eines Cloud-Backups bietet Apple auch die Möglichkeit, eine lokale Sicherung des Geräts anzulegen. Diese ist im Desktop-Betriebssystem macOS standardmäßig integriert. Zum Erstellen des Backups wird in der neusten Version des Systems das Dateiverwaltungsprogramm *Finder* verwendet (vgl. apple.com, 2021d). Für Windows-Systeme kann die Software *iTunes* verwendet werden. Eine offizielle Methode, Backups auf anderen Betriebssystemen zu erstellen, existiert nicht. Die Möglichkeit zur Verschlüsselung lokaler Backups ist optional vor deren Erstellung gegeben. Wie beim iCloud-Backup können einzelne Dateien der lokalen Sicherungen nicht mit einfachen Mitteln durchsucht werden. Deren Ordnerstruktur ist für Endanwender unübersichtlich und nur bei unverschlüsselten Backups zugänglich. Im Gegensatz zum iCloud-Speicher ist hier die Erstellung mehrerer Sicherungsversionen möglich. So können auch ältere Sicherungen wiederhergestellt werden. Zur Erstellung von Backups muss das Endgerät mindestens einmal mit dem Computer per USB-Kabel verbunden werden und die Rechte müssen durch Eingeben des Passcodes am Mobilgerät erteilt werden. Danach sind Speicherungen über USB oder WLAN möglich, wobei sich die Geräte zur Nutzung der kabellosen Variante im gleichen Netzwerk befinden müssen und dieses die Kommunikation von Geräten im lokalen Netzwerk erlauben muss. Diese Option für die Erstellung von Backups ist nicht auf das Verschieben oder Verändern der Ordnerstruktur ausgelegt. Auch die Verwaltung ist ohne die von Apple zur Verfügung gestellte Software schlecht möglich. Die Wiederherstellung erfolgt über eine USB-Verbindung zwischen dem Mobilgerät und dem Computer, auf dem die Dateien gespeichert sind. Ist die Verschlüsselung aktiviert, sind diese bei Verlust der Passphrase nicht mehr zugänglich (vgl. apple.com 2021c).

3.2. Backups über Linux-basierte Betriebssysteme

Weil der Funktionsumfang der von Apple zur Verfügung gestellten Software stark limitiert und keine Unterstützung von Unix-ähnlichen oder Unix-basierten Betriebssystemen neben Apples eigenem macOS-System vorhanden ist, hat sich das seit 2007 entwickelte Open-Source-Projekt *libimobiledevice* vor allem unter Linux-Nutzern durchgesetzt. Dabei handelt es sich nach eigener Beschreibung um eine plattformübergreifende Software-Bibliothek zur Kommunikation mit iOS-Geräten (vgl. libimobiledevice.org 2021a). Der aktuelle Quellcode des Projekts ist öffentlich auf der Webseite *github.com* verfügbar. Wegen der eigenen Implementierung von Kommunikationsprotokollen ist das Projekt unabhängig von Apple. Außerdem werden keine Programme auf dem Smartphone benötigt, um den vollen Funktionsumfang zu nutzen. Die weite Verbreitung und das lange Bestehen des Projekts hat eine hohe Wahrscheinlichkeit einer schnellen und fortlaufenden Weiterentwicklung zur Folge, die die Unterstützung neuer Betriebssystemversionen und somit eine lange Lebensdauer der Software des zu entwickelnden Ladegeräts gewährleistet. Laut der offiziellen Webseite des Projekts ist die Bibliothek seit langem mit Ausnahme seltener Fälle ohne Unterbrechung mit iOS-Geräten kompatibel und weist folglich eine hohe Wahrscheinlichkeit auf, auch zukünftige Versionen des Betriebssystems zu unterstützen (vgl. libimobiledevice.org 2021b). Außerdem wir im Gegensatz zu anderen Projekten kein sogenannter „Jailbreak“, mit dem durch Ausnutzen offener Sicherheitslücken mehr Verzeichnisse des iOS-Betriebssystem zugänglich gemacht werden, benötigt, um auf das Dateisystem zuzugreifen (vgl. libimobiledevice.org 2020). Dadurch werden keine zusätzlichen Sicherheitslücken geschaffen und es kann immer die aktuelle iOS-Version verwendet werden.

Der Funktionsumfang umfasst neben der Erstellung von Backups Methoden zur Fehleranalyse und zum Zugriff auf Teile des Dateisystems eines iOS-Geräts. Dies kann beim Ladegerät beispielsweise auch zur Extraktion von Bildern und Medien aus den erstellten Backups und Übertragung auf ein externes Speichermedium genutzt werden. Auch das Auslesen einer Identifikationsnummer kann zum Auseinanderhalten mehrerer Geräte verwendet werden.

3.2.1. Kompatibilität mit Apple-Software

Bei Verwendung von *libimobiledevice* sind die erstellten Backups vollständig mit den Apple-Lösungen kompatibel. Somit können diese auch von Nutzern - wenn dies gewünscht ist - auf einfache Weise wiederhergestellt werden. Dazu muss die Sicherung lediglich auf den PC übertragen werden, wozu zum Beispiel ein USB-Stick verwendet werden kann. Die Wiederherstellung kann aber auch direkt von Linux-Computern aus erfolgen.

Das in *iTunes* oder dem Apple *Finder* bei neuen macOS-Versionen festgelegte Passwort für die Verschlüsselung wird auch zur Wiederherstellung des Backups und zur Extraktion der Dateien mittels *libimobiledevice* verwendet. Somit kann ein vollständig kompatibler Austausch zwischen der Apple Software und dem Open-Source-Projekt erfolgen.

3.2.2. Eignung für das Projekt

Das Projekt *libimobiledevice*, insbesondere das enthaltene Programm *idevicebackup2* ist grundsätzlich zur Erstellung eines iOS-Backups geeignet. Bei Verwendung der Standardeinstellung wird eine vollständige Sicherung inklusive App-Daten und Systemdateien erstellt. Hierbei spielt die Möglichkeit einer Verschlüsselung durch Setzen eines Backup-Passworts neben der erhöhten Sicherheit eine wichtige Rolle: manche Apps werden wegen mangelnder Sicherheit oder anderer technischer Limitierungen von unverschlüsselten Backups ohne Warnung ausgeschlossen, was eine Wiederherstellung unmöglich macht.

Durch die mögliche Nutzung des vollen Funktionsumfangs von *libimobiledevice* über die Konsole des Betriebssystems - im vorliegenden Fall eine Linux-Distribution - ist eine Automatisierung des Backups umsetzbar. Das Programm *idevicebackup2* enthält außerdem eine Funktion zur Identifikation des angeschlossenen Geräts über einen „Unique Device Identifier“ (*UDID*), wobei es sich um eine Zeichenfolge zur eindeutigen Identifikation eines Geräts handelt. Somit ist eine Funktion zur Unterscheidung angeschlossener Geräte umsetzbar.

Sobald ein Initialbackup erstellt wurde, erfolgen die weiteren Backups inkrementell. Das bedeutet, dass nur Daten, die sich nach dem letzten Backup geändert haben, neu gespeichert werden. Bereits gesicherte Dateien werden übersprungen, was zu einer deutlichen Reduzierung der Backup-Dauer und weniger Schreibvorgängen führt, die wiederum die Lebensdauer des verwendeten Speichers im Vergleich zu vollständigen Backups erhöhen.

4. Technische Umsetzung

Das grundlegende Ziel dieser Arbeit ist der Entwurf eines Ladegeräts, das ohne komplizierten Einrichtungsprozess automatisch ein Backup eines iOS-Geräts erstellt. Dieses ist im Optimalfall verschlüsselt und einfach für den Nutzer zugänglich, beispielsweise durch Verwendung eines USB-Sticks oder einer SD-Karte.

Die Daten sollen in der Form vorliegen, die für die Wiederherstellung über offizielle Lösungen von Apple geeignet ist. Die Verwendung mehrerer Geräte an einem Ladegerät sollte bestehen, wobei diese durch den sogenannten „Unique Device Identifier“ unterschieden werden können. Zur Vermeidung unnötiger Komplexität für den Bediener beim Erstellen, aber auch bei der Wiederherstellung der Backups, beschränkt sich die grundlegende Funktionalität des Ladegeräts auf das automatische Erstellen von Backups und die Wiederherstellung der letzten erstellten Sicherung. So kann ein Ausgleich zwischen Nutzen und Einfachheit der Bedienung geschaffen werden. Komplexere Funktionen wie die Wiederherstellung über einen kabellosen Zugriff auf die Kommandozeile des Ladegeräts oder eine partielle Datenwiederherstellung sollen möglich sein, aber aufgrund der Fehleranfälligkeit und Bedienung nur von erfahrenen Anwendern durchgeführt werden. Eine einfache Möglichkeit zur Wiederherstellung wird hier durch die automatische Sicherung des Backups auf ein externes Speichermedium umgesetzt, mit dem die beschriebenen Schritte anschließend über einen Desktop-PC durchgeführt werden können.

Im Vergleich zu bestehenden Lösungen liegt der Mehrwert in der einfachen Handhabung beim Erstellen, Wiederherstellen und Verwalten von Backups, wobei dies mit mehreren Geräten möglich sein soll. Die Wartung der Hardware soll einfach möglich sein und auf ein Minimum reduziert werden. Während alle wichtigen Funktionen einfach gehalten werden, sodass sie auch von unerfahrenen Anwendern genutzt werden können, sollen erweiterte Funktionen trotzdem für erfahrene Anwender zugänglich sein.

Um ein automatisiertes Backup erstellen zu können, wird ein Computer benötigt, der die Programme der Bibliothek *libimobiledevice* vollständig unterstützt. Außerdem müssen Schnittstellen für ein wechselbares Speichermedium und den Anschluss des Smartphones zur Verfügung stehen.

4.1. Verwendete Hardware

Zur Einsparung von Herstellungskosten und Verringerung der Baugröße soll das Gerät in ein Ladekabel integriert werden, das mit einem bestehenden Netzstecker kompatibel ist, sodass es beispielsweise mit dem beiliegenden Netzteil des Mobilgeräts verwendet werden kann. Ein sehr geringer Stromverbrauch des Computers ist daher von großer Bedeutung, um genug elektrische Leistung für das Aufladen des Geräts zur Verfügung zu stellen, ohne dass ein separates Netzteil für den integrierten Einplatinencomputer benötigt wird. Einplatinencomputer mit Prozessoren,

die auf der ARM-Prozessorarchitektur basieren, eignen sich dafür besonders, da diese sehr effizient arbeiten und mit sehr geringer elektrischer Leistung auskommen.

Neben der geringen Baugröße und dem niedrigen Stromverbrauch sind aber auch die Hardware-Schnittstellen relevant. So werden Schnittstellen für den Anschluss des iOS-Geräts und eines wechselbaren Speichermediums benötigt. Außerdem sollte ein interner Speicher mit Platz für das Betriebssystem und die Backup-Software vorhanden sein. Dies kann entweder durch ein Gerät mit mehreren Partitionen oder die Verwendung von zwei getrennten Massenspeichermedien realisiert werden.

4.1.1. Wahl des Einplatinencomputers

Aufgrund der genannten Anforderungen scheint ein *Raspberry Pi* für das Projekt geeignet. Die technischen Daten und Spezifikationen sind öffentlich verfügbar und das Datum, bis zu dem die Produktion und der Verkauf mindestens stattfindet, ist festgelegt (vgl. The Raspberry Pi Foundation o. D. b). Wegen der geringen Baugröße (vgl. Abbildung 1), niedrigen Kosten und einem integrierten WLAN-Modul eignet sich dabei das Modell *Raspberry Pi Zero W* besonders. Dieser weist neben der hohen Energieeffizienz eine Mikro-USB-Schnittstelle auf, an die das Ladekabel für das zu sichernde Gerät angeschlossen werden kann. Außerdem ist ein Slot für eine Mikro-SD-Karte vorhanden, die als interner Speicher für das Betriebssystem und eine eventuelle Sicherung des Gerätebackups genutzt werden kann. Die geringen Kosten und die Erlaubnis zur Integration des Computers in Geräte, die anschließend verkauft werden, sprechen für die Verwendung dieses Einplatinencomputers.

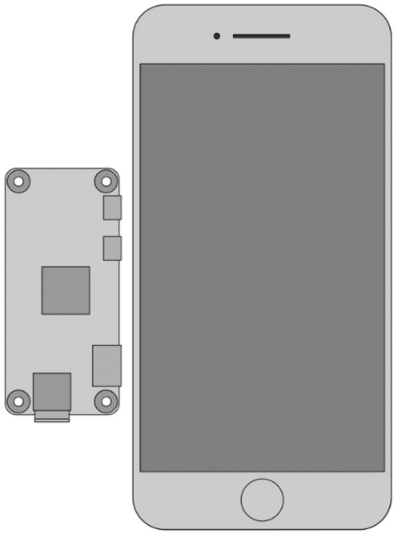

Abbildung 1: Raspberry Pi Zero W, iPhone 7 – Größe
Quelle: eigene Abbildung

Für das Backup wird keine hohe Rechenleistung benötigt, weshalb die verbaute Single-Core CPU mit einer maximalen Taktfrequenz von 1 Gigahertz und der integrierte, 512 Megabyte große Arbeitsspeicher ausreichen. Die WLAN-Schnittstelle kann für eine direkte Verbindung mit einem Desktop-PC genutzt werden, um beispielsweise den Einrichtungsprozess eines neuen iOS-Geräts zu starten.

Der *Raspberry Pi Zero W* ist aufgrund der beschriebenen Eigenschaften gut für das Ladegerät geeignet und weist nach ersten Tests keine signifikanten Nachteile auf.

4.1.2. Wahl des Speichermediums

Für das Backup-Ladegerät ist sowohl ein Speichermedium für das Betriebssystem als auch für die Sicherung des Backups selbst nötig, wobei diese wie erwähnt entweder separat oder in

verschiedenen Partitionen auf demselben Medium gesichert werden können. Da die Backups durch die regelmäßige Nutzung viele Schreibvorgänge zur Folge haben, sollte das Speichermedium für das Backup im Idealfall physisch von dem des Betriebssystems getrennt sein. Somit können Ausfälle und Funktionsstörungen der Software aufgrund eines fehlerhaften Speichers reduziert werden.

Bei der Nutzung von zwei separaten Speichermedien wird die Mikro-SD-Karte sowohl für das Betriebssystem als auch die Programme und deren Konfigurationsdateien verwendet. Die USB-Schnittstelle ist somit für den Anschluss eines USB-Sticks oder einer USB-Festplatte frei, die für die Sicherung des Backups selbst und weiteren Medien verwendet werden kann. Somit muss zur Datenübertragung auf einen Desktop-Computer lediglich der USB-Stick entfernt werden. Dadurch entfällt die Notwendigkeit eines Mikro-SD-Karten-Slots am Desktop.

4.1.3. Hardware-Schnittstellen

Für das Backup wird neben einer Schnittstelle für die Stromversorgung des Einplatinencomputers ein Anschluss für mindestens ein Smartphone und ein externes Massenspeichermedium benötigt, falls dieses physisch vom internen Speicher getrennt sein soll. Als interner Speicher ist die Verwendung einer Mikro-SD-Karte vorgesehen (vgl. Abbildung 2: **4**). Zur Stromversorgung des Einplatinencomputers und des zu sichernden Geräts wird ein Mikro-USB-Input (vgl. Abbildung 2: **2**) des *Raspberry Pi* verwendet, der aufgrund der fehlenden Anschlüsse zur Datenübertragung nur für diesen Zweck verwendet werden kann. Der zweite Mikro-USB-Anschluss (vgl. Abbildung 2: **3**) wird zur Übertragung der Daten und Laden des iOS-Geräts verwendet. Die maximal mögliche Strom- und Spannungsversorgung ist hier nicht limitiert und hängt somit ausschließlich vom verwendeten Netzteil ab.

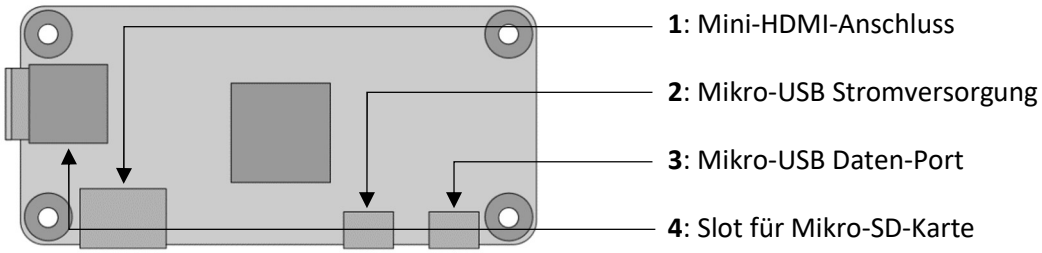

Abbildung 2: Raspberry Pi Zero W – Schnittstellen
Quelle: eigene Abbildung

Um eine zweite USB-Schnittstelle für den Anschluss eines externen Speichermediums zur Verfügung zu stellen, wird ein USB-Hub benötigt. Dieser wird indirekt mit der gleichen Schnittstelle wie der Daten-Port über zwei Pins auf der Unterseite des Ports verbunden. Mögliche Umsetzungen für ein Gehäuse und Anschlüsse werden in der Studienarbeit „Entwicklung und Konstruktion eines Systemgehäuses für ein automatisiertes Backup-Ladesystem für iOS-Geräte“ von Dennis Steinbeck (vgl. Steinbeck 2021) beschrieben.

4.2. Zugriff auf das System des Raspberry Pi

Um den *Raspberry Pi* bedienen zu können, ist ein Zugriff auf dessen Kommandozeile notwendig. Dazu muss eine Verbindung zwischen dem Gerät, von dem die Steuerung ausgeht und dem Einplatinencomputer hergestellt werden. Alternativ erfolgt die Steuerung über eine USB-Tastatur und ein externes Display.

4.2.1. Anschluss eines Displays

Eine Methode, mit der auf das Gerät zugegriffen werden kann, ist der Anschluss eines externen Bildschirms an den *Raspberry Pi Zero*. Dazu kann der vorhandene Mini-HDMI-Anschluss (vgl. Abbildung 2: **1**) verwendet werden. Das Anschließen einer Tastatur ist über einen freien Mikro-USB-Port am Hub des *Raspberry Pi* möglich. Die Methode kann nur angewandt werden, wenn der Einplatinencomputer aus dem Gehäuse entnommen wird.

Aufgrund der Umständlichkeit ist diese Art der Bedienung hauptsächlich für eine Reparatur oder Fehleranalysen des Geräts geeignet.

4.2.2. WLAN

Ein schnellerer und komfortablerer Zugriff ist über eine direkte WLAN-Verbindung zwischen dem *Raspberry Pi* und einem Desktop-Computer möglich. Der Einplatinencomputer öffnet hier über die integrierte WLAN-Schnittstelle ein eigenes Netzwerk, in das man sich mit einem WLAN-fähigen Computer oder Smartphone einwählen kann. Ein Passwort zur Authentifizierung sollte hier beim Endprodukt individuell vergeben werden und mittels eines aktuellen Sicherheitsprotokolls wie WPA3 geschützt werden.

Der Login und die Bedienung erfolgen anschließend über das *Secure-Shell* Protokoll, für dessen Verwendung auf Windows, macOS und vielen verbreiteten Linux-basierten Betriebssystemen bereits ein Programm vorinstalliert ist. Nach Verbindung mit dem WLAN-Netzwerk kann über die Konsole des jeweiligen Betriebssystems, beispielsweise die *Powershell* unter Windows oder das macOS-*Terminal*, eine Verbindung mit dem *Raspberry Pi* hergestellt werden. Dazu wird das Programm *OpenSSH* verwendet (vgl. openssh.com o. D.). Der Befehl zum Verbinden lautet beispielsweise „ssh backup@10.0.0.5“, wobei „backup“ der Benutzername für den Login und „10.0.0.5“ die IP-Adresse des *Raspberry Pi* im lokalen Netzwerk ist.

4.3. Erstellen von Backups

Die in *libimobiledevice* enthaltenen Programme sind auf die Verwendung mit der Kommandozeile des jeweiligen Betriebssystems ausgelegt. Während dies die Automatisierung erleichtert, erschwert es vor allem für unerfahrene Benutzer die Bedienung. Deshalb müssen die Nutzereingriffe beim Erstellen des Backups mit dem Ladegerät und der Übertragung der Daten minimiert werden.

4.3.1. Installation der Software-Bibliothek libimobiledevice

Auf dem Raspberry Pi Zero wird aufgrund der Kompatibilität der enthaltenen Software mit der Hardware und der hohen Wahrscheinlichkeit einer bestehenden Weiterentwicklung die Linux-Distribution *Raspberry Pi OS* installiert. Die in den offiziellen Paketquellen dieses Betriebssystems enthaltene Version von *libimobiledevice* ist, je nach Entwicklungsstand der aktuellen iOS-Version, nicht mit der neusten iOS-Version kompatibel. Um dieses Problem zu umgehen, wird der auf *github.com* veröffentlichte Quellcode der aktuellen Version des Programms heruntergeladen und auf dem Raspberry Pi kompiliert.

4.3.2. Automatisierung des Backup-Prozesses

Das Ziel des Backup-Ladegeräts ist eine automatisierte, vollständige Sicherung des kompletten Systems. Deshalb muss das Backup automatisch starten, sobald ein iOS-Gerät an den *Raspberry Pi* angeschlossen wird. Der Prozess soll bei Anschluss eines Geräts bei laufendem System, also wenn der Einplatinencomputer mit dem Netzteil, aber anfangs nicht mit einem Smartphone verbunden ist, starten. Weiterhin ist ein Start des Backups auch bei Verbindung des iOS-Geräts vor Systemstart und anschließender Verbindung mit der externen Spannungsversorgung erwünscht. Die Erkennung und Identifikation bei Verwendung mehrerer Geräte mit dem gleichen Ladegerät soll außerdem vollständig autonom erfolgen und die Daten sollen getrennt in einem gerätespezifischen Verzeichnis abgelegt werden.

Die Erkennung eines angeschlossenen Geräts unter Linux kann mit sogenannten „udev“-Regeln erfolgen. Innerhalb dieser kann in einer Datei ein Kommandozeilenbefehl definiert werden, der bei Anschluss eines Geräts mit bestimmten Eigenschaften vom System ausgeführt wird (debian.org 2005). Die Software-Bibliothek *libimobiledevice* nutzt eine solche Regel zum Starten eines USB-Multiplex Hintergrundprozesses „usbmuxd“ (vgl. Anhang 5) über den Service-Manager der auf dem Betriebssystem vorinstallieren Software-Suite *systemd*, wenn ein iOS-Gerät angeschlossen wird (vgl. debian.org 2010). Die Funktionalität dieses Prozesses wird für bestimmte – hier nicht weiter relevante – Funktionen von *libimobiledevice*-Programmen benötigt. Die Erkennung, ob es sich beim verbundenen Gerät um ein iOS-Gerät handelt, erfolgt im vorliegenden Fall über den „Unique Device Identifier“, dessen Zeichenfolge einem bestimmten Schema folgt und somit in der „udev“-Regel festgelegt werden kann. Somit kann sichergestellt werden, dass der Service bei Verbinden eines iOS-Geräts, unabhängig vom aktuellen Zustand des Systems, gestartet und beim Trennen beendet wird.

Der erwähnte Hintergrundservice kann, da er wie beschrieben bei jedem Verbinden eines iOS-Geräts gestartet und beim Trennen gestoppt wird, zur Erkennung des verbundenen Geräts genutzt werden. Diese Erkennung könnte alternativ auch über eine eigene „udev“-Regel realisiert werden, die sich aber nach Tests als unzuverlässig herausgestellt hat. Daher erfolgt die Identifikation hier über einen weiteren Service. Dieser wird automatisch ausgeführt, sobald der Service „usbmuxd“ erfolgreich gestartet wurde. Der Hintergrundservice führt beim Start mehrere in

einem Shell-Skript definierte Kommandozeilenbefehle aus, die für die Sicherung des iOS-Geräts benötigt werden (vgl. Anhang 4).

4.3.3. Detaillierter Ablauf des Backup-Prozesses

Beim Verbinden eines iOS-Geräts wird durch den oben beschriebenen Prozess das Backup-Skript gestartet. Dieses prüft zuerst, ob ein externes Speichermedium vorhanden ist. Ist dies der Fall, wird mittels des in *libimobiledevice* enthaltenen Programms *ideviceinfo* der „Unique Device Identifier“ ausgelesen. Der Name des Ordners, der von *idevicebackup2* während des Sicherungsprozesses erstellt wird, entspricht dem Identifier des zu sichernden Geräts. Nach Auslesen des Identifiers wird ein Backup erstellt, sofern ein USB-Stick eingesteckt ist und dessen gesamter Speicherplatz größer als der benötigte Speicher aller auf dem iOS-Gerät befindlichen Daten ist.

Abbildung 3: iTunes – Ändern des Sicherungspassworts
Quelle: eigene Abbildung

Eine Prüfung, ob die Sicherungsverschlüsselung aktiv ist, ist nicht automatisiert möglich. Daher sollte diese unbedingt vor der ersten Sicherung über das Programm *iTunes* auf Windows, *Finder* auf macOS oder die Software *idevicebackup2* bei Nutzung eines Linux-Systems aktiviert werden (vgl. Abbildung 3). Zur Aktivierung unter Linux kann ein dafür erstelltes Skript genutzt werden (vgl. Anhang 4). Dieses fragt den Nutzer nach einem Passwort, das anschließend zur Ver- und Entschlüsselung des Geräts genutzt wird. Nach Passworteingabe wird der Sicherungsprozess gestartet. Beim ersten Backup wird explizit ein vollständiges Backup erstellt. Dieser Vorgang muss nur einmal durchgeführt werden und ist einfach umsetzbar.

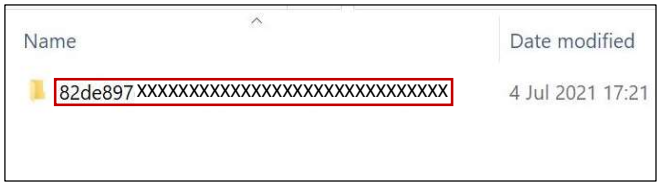

Abbildung 4: Unique Device Identifier
Quelle: eigene Abbildung

Bei Verwendung mehrerer iOS-Geräte werden mehrere Ordner mit den jeweiligen Identifiern der Mobilgeräte erstellt (vgl. Abbildung 4). Durch die Verschlüsselung ist der Dateizugriff nur für Nutzer, die das Passwort des jeweiligen Backups kennen, möglich.

Aus Sicht des Nutzers wird – nach Durchlaufen des beschriebenen Einrichtungsprozesses – bei jedem Anschließen des Mobilgeräts an das Ladegerät automatisch und ohne Notwendigkeit jeglicher Eingaben ein vollständiges Backup des Geräts erstellt. Der Zahlencode des Geräts muss bei erster Verbindung nach Bestätigen der Meldung auf dem iOS-Gerät, ob es sich beim Ladegerät um ein vertrauenswürdiges Gerät handelt, einmalig vom Nutzer eingegeben werden. Weitere Sicherungen erfordern keine Nutzereingriffe und erfolgen inkrementell, wodurch die Sicherungszeit stark reduziert werden kann. Die Dauer hängt von der Menge der seit dem letzten Backup geänderten Dateien ab und liegt bei wenigen Minuten. Während des Prozesses wird auf

dem iOS-Gerät ein Symbol am oberen Rand oder bei aktuellen iPhones ab dem Modell *iPhone X* im Kontrollzentrum des Systems angezeigt.

4.4. Wiederherstellung der Sicherung

Um eine Möglichkeit zur vollwertigen Nutzung der Sicherungen zu schaffen, kann das Backup auf einen Desktop-Computer übertragen werden. Während beim Sicherungsprozess ein Programm- oder Bedienfehler keinen Dateiverlust auf dem Mobilgerät zur Folge hat, ist der Wiederherstellungsprozess komplexer und erfordert eine Möglichkeit zur Überwachung des Prozesses. Daher sollte dieser vorzugsweise über Desktop-Computer durchgeführt werden. Die Umsetzung der Wiederherstellung über einen Knopf am Gerät ist zwar technisch möglich, jedoch muss dafür wegen der fehlenden einfachen Überwachungsmöglichkeit die Software zuerst viele Nutzertests durchlaufen, damit ein reibungsloser Ablauf gewährleistet werden kann.

4.4.1. Wiederherstellung über Desktop-Systeme

Wird ein USB-Stick mit dem Ladegerät verbunden, sollte auf diesem vorher eine mit Windows, macOS und Linux-Systemen kompatible „exFAT“-Partition erstellt, auf der die Sicherung erfolgt. Dieser Stick kann, wenn eine Wiederherstellung durchgeführt werden soll, mit einem Computer verbunden werden. Auf diesem wird anschließend der Backup-Ordner an die Stelle, die von der mit dem jeweiligen Betriebssystem kompatiblen Apple-Software durchsucht wird, kopiert. Auf Windows-Systemen ist diese unter dem Pfad „%USERPROFILE%\MobileSync\Backup“ zu finden, auf Mac OS muss der Ordner nach „~/Library/Application Support/MobileSync/Backup/“ kopiert werden (vgl. apple.com 2021d). Auf Linux-Systemen ist der Pfad irrelevant, da er bei der Wiederherstellung explizit angegeben wird. Anschließend kann das Backup über die jeweilige Software wiederhergestellt werden. Unter Windows funktioniert die Wiederherstellung beispielsweise folgendermaßen:

Zuerst muss der Pfad, unter dem sich die *iTunes*-Backups befinden, im *Windows Explorer* geöffnet werden (vgl. Abbildung 5). Vor Anschluss eines iOS-Geräts sollte die automatische Backupfunktion deaktiviert werden, um ein Überschreiben des kopierten Sicherungsordners zu verhindern (vgl. Abbildung 6 und 7).

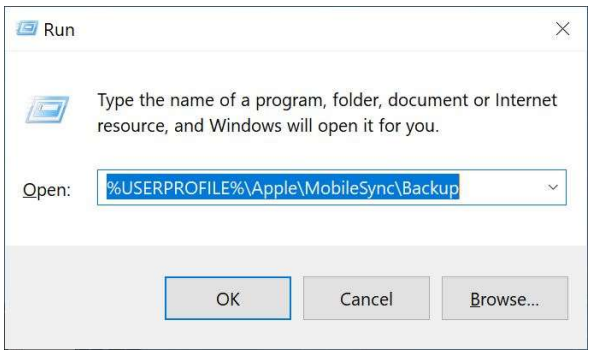

Abbildung 5: iTunes – Backup Pfad
Quelle: eigene Abbildung

Der Ordner mit dem Namen des gewünschten „Unique Device Identifier“, der die Sicherung enthält, wird dann an in das Verzeichnis unter dem geöffneten Pfad kopiert.

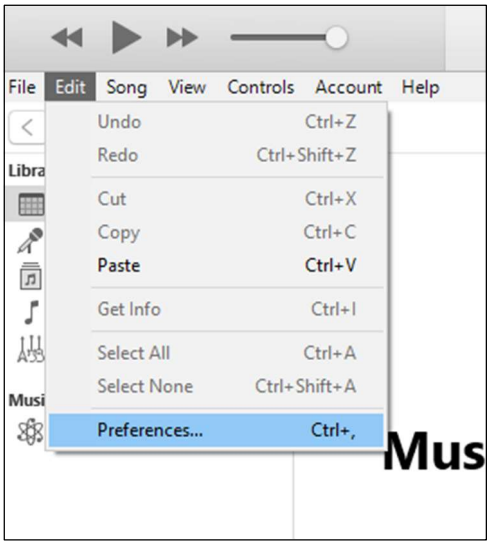

Abbildung 6: iTunes – Auto Backup 1
Quelle: eigene Abbildung

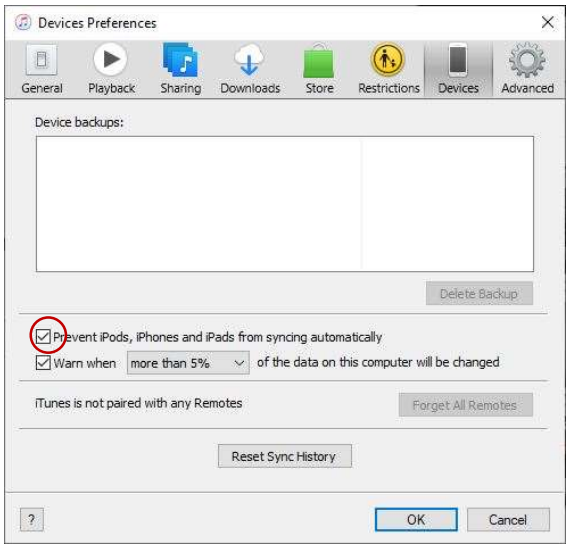

Abbildung 7: iTunes – Auto Backup 2
Quelle: eigene Abbildung

Der *UDID* kann ebenfalls über die Software *iTunes* ausgelesen werden, indem das iOS-Gerät an den Computer angeschlossen wird und in *iTunes* mit der Maus auf „Serial Number“ geklickt wird, bis an dieser Stelle der *UDID* angezeigt wird (vgl. Abbildung 8 und 9).

Abbildung 8: iTunes – Seriennummer
Quelle: eigene Abbildung

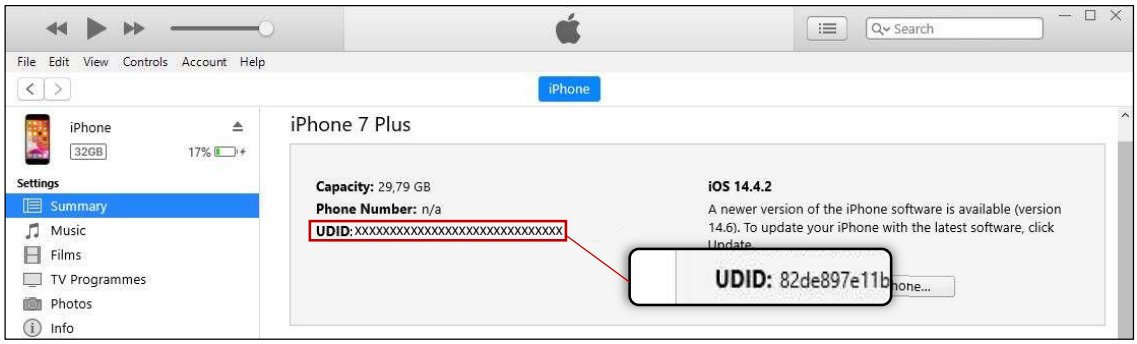

Abbildung 9: iTunes – UDID
Quelle: eigene Abbildung

Nach Import des Backup-Ordners und Neustart von *iTunes* kann das Backup über die Schaltfläche „Restore iPhone“ wiederhergestellt werden.

4.4.2. Wiederherstellung über das Ladegerät

Alternativ erfolgt die Wiederherstellung über ein Skript auf dem Ladegerät selbst. Diese Möglichkeit ist für erfahrene Anwender geeignet und bietet Optionen zur vollständigen oder partiellen Wiederherstellung des Betriebssystems. Dazu muss der Nutzer Zugriff auf das System des Einplatinencomputers haben, was in *Kapitel 4.2* erklärt wird. Das Gerät muss für die Wiederherstellung angeschlossen sein und alle Backup-Prozesse müssen vollständig abgeschlossen sein. Dies kann an dem fehlenden Sicherungssymbol auf dem iOS-Gerät erkannt werden oder über eine Statusabfrage des Backup-Services (vgl. Anhang 4). Zur Wiederherstellung kann das Programm *idevicebackup2* verwendet werden. Zum Auflisten der wichtigsten Funktionen wird der Befehl „idevicebackup2 --help“ auf dem System ausgeführt. Für eine vollständige Wiederherstellung des Systems auf dem gleichen Gerät wird zuerst der Identifier des angeschlossenen Geräts mit dem Programm *ideviceinfo* ausgelesen. Sollte die Wiederherstellung auf einem anderen Gerät gewünscht sein, muss der Identifier des Wiederherzustellenden Backups bekannt sein.

Anschließend erfolgt der Restore beispielsweise mit dem Befehl „idevicebackup2 --source *UDID* restore --system --settings *DESTINATION* -i“ (*UDID* wird durch den Identifier des Geräts ersetzt, *DESTINATION* durch den Pfad der Sicherung), was eine vollständige Wiederherstellung inklusive aller Apps und Systemdateien zur Folge hat. Dieser Vorgang kann auch durch ein Wiederherstellungsskript automatisiert werden (vgl. Anhang 2). Nach dem Prozess muss der Nutzer den auf dem iOS-Gerät beschriebenen Schritten folgen, um die Wiederherstellung abzuschließen. Eventuell ist ein erneutes Einloggen in Online-Accounts nötig.

4.5. Redundanz

Da die Sicherung nur auf selten eintretende Fälle wie Verlust des iOS-Geräts oder einen Schaden, der den Zugriff auf die Dateien verhindert, ausgelegt ist, ist eine einzige Version der Sicherung im Normalfall ausreichend. Wegen der geringen Baugröße und aufgrund der Kosten ist die Integration eines zweiten Speichers für eine Sicherungskopie nicht sinnvoll umsetzbar. Bei Verwendung eines externen Speichermediums und ausreichend großer SD-Karte ist eine zweite Sicherung auf dieser Karte theoretisch möglich, was aber deren Lebensdauer durch viele Schreibvorgänge stark verringert. Auch die Sicherung über WLAN auf einem weiteren Gerät im gleichen Netzwerk ist denkbar, aber für die meisten Anwender nicht einfach umsetzbar.

Die einfachste Lösung stellt somit bei Bedarf eine periodische Sicherung der auf dem USB-Stick vorhandenen Daten mit einem Desktop-PC dar. Dazu müssen lediglich die auf dem Stick vorhandenen Ordner auf den Computer kopiert werden. Die Wiederherstellung erfolgt dann wie oben beschrieben, wobei die gewünschte Sicherung je nach verwendetem Betriebssystem an die geeignete Stelle kopiert werden muss oder unter Linux-Systemen der entsprechende Pfad angegeben wird.

5. Ausblick

Während die grundlegenden Anforderungen an die Funktionalität des Backup-Ladegeräts erfüllt werden, besteht Optimierungsbedarf bei der Bedienung und Wartung des Geräts. Auch ein Verkauf erfordert die Beachtung der rechtlichen Bedingungen von Software und der verwendeten Hardware.

5.1. Möglichkeit zum Verkauf

Da das Gerät sowohl mit der Linux-Distribution *Raspberry Pi OS*, als auch der Software-Bibliothek *libimobiledevice* ausgeliefert werden muss, sind deren Lizenzen für die Veröffentlichung relevant. Die Software-Bibliothek ist unter der „Lesser General Public License“ in Version 2.1 veröffentlicht. Diese erlaubt die freie Verbreitung und Integration der Programme unter Beachtung einiger Bedingungen. Die Software-Bibliothek darf, solange der Quellcode nicht modifiziert wird, von Programmen unabhängig von deren Lizenz, genutzt werden. Da der Quellcode von *libimobiledevice* hier nicht verändert wird, sondern die Funktionen der enthaltenen Programme nur automatisiert ausgeführt werden, fallen die in dieser Arbeit genutzten Skripte nicht unter die Lizenz und unterliegen infolgedessen nicht deren Bedingungen. Somit muss lediglich der Quellcode von *libimobiledevice* öffentlich verfügbar sein und auf die Nutzung der Bibliothek hingewiesen sowie eine Kopie der Lizenz angehängt werden (vgl. Free Software Foundation Inc. 1999). Das Betriebssystem *Raspberry Pi OS* basiert auf der Linux-Distribution *Debian*. Das System und enthaltene Programme sind hauptsächlich unter der „General Public License“ veröffentlicht. Dieses darf in kommerziellen Produkten unverändert genutzt werden, solange auch die Lizenzen aller enthaltenen Programme die Nutzung erlauben. Programme, die keiner Open-Source Lizenz unterliegen, sind für dieses Projekt irrelevant und müssen somit nicht im Endprodukt enthalten sein (vgl. Free Software Foundation Inc. 2007).

Die technische Dokumentation und alle Hardware-Spezifikationen des Raspberry *Pi Zero W* selbst sind öffentlich verfügbar (vgl. The Raspberry Pi Foundation o. D. a). Die Integration in das Produkt erfordert keinen Erwerb einer Lizenz. Optional kann ein Logo mit der Aufschrift „Powered by Raspberry Pi“ auf Anfrage auf der Website der Raspberry Pi Foundation genutzt werden (vgl. The Raspberry Pi Foundation o. D. b). Bei Nutzung innerhalb eines Gehäuses muss die elektromagnetische Verträglichkeit des Endprodukts untersucht werden. Diese Thematik wird in der Studienarbeit „Entwicklung und Konstruktion eines Systemgehäuses für ein automatisiertes Backup-Ladesystem für iOS-Geräte“ von Dennis Steinbeck (vgl. Steinbeck 2021) ausführlich behandelt.

5.2. Vereinfachen der Bedienung

Das Backup-Ladegerät stellt in der beschriebenen Form eine einfache Möglichkeit zum Erstellen vollständiger Sicherungen von iOS-Geräten zur Verfügung. Jedoch müssen bei der Einrichtung des Geräts einige Schritte über die Kommandozeile oder Desktop-Computer ausgeführt werden. Diese können zu Bedienfehlern und somit zu Einschränkungen der Funktionalität führen. Auch der Zeitaufwand ist dadurch erhöht. Nach der Einrichtung laufen die Prozesse automatisiert ab, weshalb diese Schritte nur einmal durchgeführt werden müssen und bei korrekter Eingabe keinen Nachteil im Betrieb zur Folge haben.

Um die Einrichtung und Verwaltung für unerfahrene Nutzer zu vereinfachen, ist die Entwicklung einer grafischen Benutzeroberfläche denkbar. Diese kann beispielsweise eine Übersicht aller gesicherten Backups enthalten. Auch das Setzen des Backup-Passworts und Verbinden des Ladegeräts mit dem WLAN des Heimnetzwerks über eine grafische Oberfläche ist denkbar. Diese Schritte müssen nach aktuellem Stand über die Kommandozeile ausgeführt werden.

Ein automatisiertes Software-Update ist außerdem für den Erhalt der Funktionalität aufgrund der fortschreitenden Entwicklung des iOS-Betriebssystems vorteilhaft.

5.3. WLAN-Übertragung

Das WLAN-Modul des *Raspberry Pi Zero* kann neben der Steuerung des Ladegeräts zusätzlich als Schnittstelle zur Datenübertragung fungieren. Der Up- und Download von Dateien ist aktuell schon beispielsweise über *ssh* und das Programm *rsync* möglich, jedoch eignet sich diese Methode nur schlecht für unerfahrene Anwender. Bei Integration einer Möglichkeit zur Übertragung mithilfe einer grafischen Bedienoberfläche und eventuellem automatischen Import der Backup-Dateien in *iTunes* oder *Finder* kann der Prozess der Wiederherstellung und Verwaltung stark vereinfacht werden. Auch weitere Einstellungen wie die Sicherungshäufigkeit, Art des Backups und Ändern des Verschlüsselungs-Passworts könnten so realisiert werden.

6. Zusammenfassung

Die Herausforderungen, die heute durch den Fokus auf Cloud-Backups und die geringe öffentlich verfügbare Dokumentation des Backup-Protokolls von Apple entstehen, erschweren die Realisierung eines Ladegeräts mit Backup-Funktion. Besonders vollständige Systembackups sind nur schwer umsetzbar. Auch wenn in der Vergangenheit viele Patente mit dem Ziel eines solchen Geräts angemeldet wurden, fehlt es den Produkten an essenziellen Funktionen wie einer Verschlüsselung oder einfachen Bedienung, die die Geräte zu einer sinnvollen Alternative für Online-Sicherungen machen. Patentanmeldungen und Geräte auf dem Markt verfügen entweder über unzureichende Sicherungsfunktionen, zu komplexe Bedienkonzepte, starken Fokus auf beispielsweise den Einsatz in Unternehmen oder die Synchronisierung mehrerer Geräte. Open-Source-Lösungen können hingegen auf die Nutzerbedürfnisse angepasst werden, stellen aber aufgrund der dafür nötigen Computerkenntnisse für die meisten Endanwender keine Alternative dar und erschweren sogar teilweise den Backup-Prozess im Vergleich mit der Erstellung über einen Desktop-Computer.

Die offiziellen Lösungen können für einfache Anwendungen wie gelegentliche manuelle Sicherungen und den Import von Dateien genutzt werden. Aufgrund der Lizenzen, der fehlenden Unterstützung von Linux-Betriebssystemen und keiner Möglichkeit zur Automatisierung eignen sich die Programme *Finder* unter macOS oder *iTunes* auf Windows nicht für ein regelmäßiges Systembackup. Eine bessere Alternative bietet hier die in das iOS-Betriebssystem integrierte Cloud-Backup-Funktion, die eine Sicherung vieler auf dem Gerät befindlicher Daten automatisiert auf den Servern von Apples Cloud-Service *iCloud* durchführen kann. Diese ist jedoch aufgrund der Abhängigkeit von *iCloud*, fehlender Sicherungen mancher Teile des Systems und dem Fehlen einer vollständigen Ende-zu-Ende-Verschlüsselung nur bedingt als Alternative zu den Offline-Lösungen geeignet.

Durch die Nutzung von Programmen der Open-Source Software-Bibliothek *libimobiledevice* kann aufgrund der weiten Verbreitung des Projekts und infolgedessen ständigen Weiterentwicklung die Wahrscheinlichkeit einer bestehenden Funktionalität des Backup-Ladegeräts erhöht werden. Die Kompatibilität der Sicherungen und deren Verschlüsselungen mit der von Apple entwickelten Software zur Verwaltung von iOS-Geräten stellt eine hohe Flexibilität bei der Wiederherstellung und Verwaltung der Backups sicher. Somit ist gewährleistet, dass die Backups über Windows, macOS und Linux wiederhergestellt werden können. Auch wenn die Bibliothek *libimobiledevice* nicht von Apple unterstützt wird, stellt sie bei der Backup-Verwaltung durch die erfahrungsgemäß hohe Stabilität eine sichere Alternative zu iTunes oder den macOS *Finder* dar. Durch die Linux-Unterstützung ist die im Ladegerät benötigte Automatisierung der Prozesse gut umsetzbar. Da kein „Jailbreak“ am iOS-Gerät zur Verwendung der Software durchgeführt werden muss und somit immer die aktuelle Betriebssystemversion genutzt werden kann, werden keine zusätzlichen Sicherheitslücken bei der Verwendung von *libimobiledevice* geschaffen.

Der Einplatinencomputer *Raspberry Pi Zero W* stellt die geeigneten Hardware-Schnittstellen und den Prozessor für das Backup bereit. Aufgrund der geringen Baugröße und des sehr niedrigen Stromverbrauchs ist die Integration in das Ladekabel problemlos umsetzbar, womit eine portable Backup-Lösung geschaffen wird, die mit dem beiliegenden Netzteil des iOS-Geräts genutzt werden kann. Der Zugriff auf das System ist aufgrund der integrierten WLAN-Schnittstelle problemlos möglich und kann zukünftig auch zur einfachen Dateiübertragung auf einen Desktop-Computer genutzt werden. Durch die Verwendung eines im Ladegerät integrierten USB-Hubs sind Schnittstellen für das Laden und ein externes Speichermedium in Form eines USB-Sticks vorhanden, der aufgrund der Formatierung im *exFAT*-Dateisystem mit allen gängigen Betriebssystemen kompatibel ist. So kann die Wiederherstellung eines Backups bei einem Geräteschaden oder -wechsel von einem Desktop-Computer aus erfolgen. Auch eine Wiederherstellung über das Ladegerät selbst ist mithilfe des Zugriffs über WLAN möglich.

Bei einem möglichen Verkauf des Geräts muss auf die Verwendung des *Raspberry Pi Zero W* hingewiesen werden. Zudem sind die Software-Lizenzen zu beachten, was aber aufgrund der Open-Source Lizenzen keine weiteren Kosten zur Folge hat. Durch das festgelegte Datum, bis zu dem der *Raspberry Pi* mindestens produziert wird, kann die Produktion des Ladegeräts gut geplant werden (vgl. raspberrypi.org o. D.).

Zukünftig ist die Umsetzung einer grafischen Benutzeroberfläche zur Sicherungsverwaltung und -wiederherstellung, eine schnellere Möglichkeit zum Aktivieren der Verschlüsselung, Updates der Programme und des Betriebssystems sowie eine einfache Möglichkeit zur Datenübertragung über die WLAN-Schnittstelle des Einplatinencomputers denkbar. Diese Eigenschaften machen das Gerät bei einer möglichen Veröffentlichung für Endnutzer einfacher bedienbar und somit attraktiver.

Das Ladegerät stellt durch die einfache Bedienung und das Erstellen vollständiger Systemsicherungen eine Alternative zu Cloud-Backups dar. Aufgrund der vollständigen Automatisierung nach der Ersteinrichtung und der einfachen Möglichkeit einer Backup-Verschlüsselung steht das Gerät bei der Benutzerfreundlichkeit und Datensicherheit einer Online-Lösung in nichts nach.

Quellen

[1] apple.com (2020a): iCloud security overview, 09. April, support.apple.com, URL: https://support.apple.com/en-us/HT202303, Abruf am 23. Juni 2021

[2] apple.com (2020b): If you forgot your Apple ID password, 30. November, support.apple.com, URL: https://support.apple.com/en-us/HT201487, Abruf am 27. Juni 2021

[3] apple.com (2021a): Backup methods for iPhone, iPad, and iPod touch, 30. Juni, support.apple.com, URL: https://support.apple.com/en-us/HT204136, Abruf am 27. Juni 2021

[4] apple.com (2021b): What does iCloud back up, 04. Februar, support.apple.com, URL: https://support.apple.com/en-us/HT207428, Abruf am 27. Juni 2021

[5] apple.com (2021c): If you can't remember the password for your encrypted backup, 26. Januar, support.apple.com, URL: https://support.apple.com/en-us/HT205220, Abruf am 27. Juni 2021

[6] apple.com (2021d): Locate backups of your iPhone, iPad, and iPod touch, 23. März, support.apple.com, URL: https://support.apple.com/en-us/HT204215, Abruf am 05. Juli 2021

[7] Berberich, Olaf (2020): Verfahren für ein dezentrales Backupsystem im Mehrfach-USB-Ladegerät (DE102019000928A1), 13. August, Deutsches Patent- und Markenamt, worldwide.espacenet.com, URL: https://worldwide.espacenet.com/patent/search/family/071738713/publication/DE102019000928A1?q=pn%3DDE102019000928A1, Abruf am 23. Juni 2021

[8] Cruz, Arnaldo Zael (2012): Data backup device (US2012330888A1), 27. Dezember, U.S. Patent and Trademark Office, worldwide.espacenet.com, URL: https://worldwide.espacenet.com/patent/search/family/047362782/publication/US2012330888A1?q=pn%3DUS2012330888A1, Abruf am 23. Juni 2021

[9] debian.org (2005): udev – Linux dynamic device management, 17. Februar, wiki.debian.org, URL: https://wiki.debian.org/udev, Abruf am 23. Juni 2021

[10] debian.org (2010): systemd – system and service manager, 17. November, wiki.debian.org, URL: https://wiki.debian.org/systemd, Abruf am 23. Juli 2021

[11] Free Software Foundation, Inc. (1999): GNU Lesser General Public License, Februar, www.gnu.org, URL: https://www.gnu.org/licenses/old-licenses/lgpl-2.1.en.html, Abruf am 29. Juni 2021

[12] Free Software Foundation, Inc. (2007): GNU General Public License, 29. Juni, www.gnu.org, URL: https://www.gnu.org/licenses/gpl-3.0.html, Abruf am 29. Juni 2021

[13] Goddo KK (2000): Charger with data backup function for mobile phone and data backup unit connected to the charger (JP2000324237A), 24. November, Japan Patent Office, worldwide.espacenet.com, URL: https://worldwide.espacenet.com/patent/search/family/026403662/publication/JP2000324237A?q=pn%3DJP2000324237A, Abruf am 23. Juni 2021

[14] Goel, Anil (2009): Cable with memory (AU2008320924A1), 07. Mai, Australian Patent Office, worldwide.espacenet.com, URL: https://worldwide.espacenet.com/patent/search/family/040329121/publication/AU2008320924A1?q=pn%3DAU2008320924A1, Abruf am 23. Juni 2021

[15] Kantar Worldpanel (2021): Sales market share of leading smartphone operating systems in Germany first quarter 2020 and 2021, Mai, www.statista.com, URL: https://www.statista.com/statistics/461997/smartphone-os-market-shares-of-sales-in-germany/, Abruf am 24. Juni 2021

[16] Kobayashi, Toshihiro (2003a): Data transfer device for connection of charging adapter for mobile telephone (WO03005690A1), 16. Januar, World Intellectual Property Organization, worldwide.espacenet.com, URL: https://worldwide.espacenet.com/patent/search/family/019040110/publication/WO03005690A1?q=pn%3DWO03005690A1, Abruf am 23. Juni 2021

[17] Kobayashi, Toshihiro (2003b): Cellular phone charger with data backup function and cellular phone data backup device (US2003098670A1), 29. Mai, U.S. Patent and Trademark Office, worldwide.espacenet.com, URL: https://worldwide.espacenet.com/patent/search/family/025536062/publication/US2003098670A1?q=pn%3DUS2003098670A1, Abruf am 24. Juni 2021

[18] libimobiledevice.org (2020): API Documentation, 12. Juni, docs.libimobiledevice.org, URL: https://docs.libimobiledevice.org/libimobiledevice/latest/index.html, Abruf am 22. Juni 2021

[19] libimobiledevice.org (2021a): libimobiledevice, libimobiledevice.org, URL: https://libimobiledevice.org/#get-started, Abruf am 27. Juni 2021

[20] libimobiledevice.org (2021b): FAQ – Does the library to device communication break every time a new iOS is out?, libimobiledevice.org, URL: https://libimobiledevice.org/#faq, Abruf am 22. Juni 2021

[21] Lior, Ben David & Israeli, Chai (2017): Data backup and charging device for communication devices (EP3236358A1), 25. Oktober, Europäisches Patentamt, worldwide.espacenet.com, URL: https://worldwide.espacenet.com/patent/search/family/058108451/publication/EP3236358A1?q=pn%3DEP3236358A1, Abruf am 23. Juni 2021

[22] Maurilus, Jean R. (2006): Method and apparatus for backing up data from cell phones and other hand-held devices (US200615815A1), 20. Juli, U.S. Patent and Trademark Office, worldwide.espacenet.com, URL: https://worldwide.espacenet.com/patent/search/family/036683199/publication/US2006158154A1?q=pn%3DUS2006158154A1, Abruf am 23. Juni 2021

[23] Menn, Joseph (2020): Apple dropped plan for encrypting backups after FBI complained – sources, 21. Januar, www.reuters.com, URL: https://www.reuters.com/article/us-apple-fbi-icloud-exclusive/exclusive-apple-dropped-plan-for-encrypting-backups-after-fbi-complained-sources-idUSKBN1ZK1CT, Abruf am 23. Juni 2021

[24] Nakajima Tsushinki Kogyo KK (2004): Battery charger of portable telephone (JP2004032480A), Japan Patent Office, 29. Januar, worldwide.espacenet.com, URL: https://worldwide.espacenet.com/patent/search/family/031182418/publication/JP2004032480A?q=pn%3DJP2004032480A, Abruf am 23. Juni 2021

[25] openssh.com (o. D.): OpenSSH Users, www.openssh.com, URL: https://www.openssh.com/users.html, Abruf am 23. Juni 2021

[26] Pearson, Justin (2018a): Raspberry Pi for iPhone Backup – 6. Indicate Process with Blinkt LEDs, 08. Juli, github.com, URL: https://github.com/justinpearson/Raspberry-Pi-for-iPhone-Backup#6-indicate-progress-with-blinkt-leds, Abruf am 23. Juni 2021

[27] Pearson, Justin (2018b): Raspberry Pi for iPhone Backup – 1. Install libimobiledevice from source, 08. Juli, github.com, URL https://github.com/justinpearson/Raspberry-Pi-for-iPhone-Backup#1-install-libimobiledevice-from-source, Abruf am 23. Juni 2021

[28] Pearson, Justin (2018c): Raspberry Pi for iPhone Backup – 2c. Mount Phone's files on RPi, 08. Juli, github.com, URL: https://github.com/justinpearson/Raspberry-Pi-for-iPhone-Backup#2c-mount-phones-files-on-rpi, Abruf am 23. Juni 2021

[29] Pearson, Justin (2018d): Raspberry Pi for iPhone Backup – Configure udev to run the backup script upon the phone plug-in event, 08. Juli, github.com, URL: https://github.com/justinpearson/Raspberry-Pi-for-iPhone-Backup#4-configure-udev-to-run-the-backup-script-upon-the-phone-plug-in-event, Abruf am 23. Juni 2021

[30] Perez, Yves-Alexis (2018a): iPibackup: backup your iPhone while charging it - Hardware, 29. April, github.com, URL: https://github.com/corsac-s/ipibackup#hardware, Abruf am 23. Juni 2021

[31] Perez, Yves-Alexis (2018b): iPibackup: backup your iPhone while charging it - Backup, 29. April, github.com, URL: https://github.com/corsac-s/ipibackup#backup, Abruf am 23. Juni 2021

[32] Perez, Yves-Alexis (2018c): iPibackup: backup your iPhone while charging it - Restore, 29. April, github.com, URL: https://github.com/corsac-s/ipibackup#restore, Abruf am 23. Juni 2021

[33] raspberrypi.org (o. D.): Obsolescence Statement, www.raspberrypi.org, URL: https://www.raspberrypi.org/products/raspberry-pi-zero-w/, Abruf am 01. Juli 2021

[34] Rubin, Matt (2014): Kommentar (2) unter: Backup/Restore (onto new devices), 02. Oktober 2014, github.com, URL: https://github.com/mattrubin/authenticator/issues/6, Abruf am 22. Juni 2021

[35] Statista Global Consumer Survey (2020a): Smartphone app usage by type in Germany 2020, November, www.statista.com, URL: https://www.statista.com/forecasts/998679/smartphone-app-usage-by-type-in-germany, Abruf am 23. Juni 2021

[36] Statista Global Consumer Survey (2020b): Which of these online services have you used in the past 12 months?, November, www.statista.com, URL: https://www.statista.com/forecasts/998832/cloud-service-usage-in-germany, Abruf am 23. Juni 2021

[37] Steinbeck, Dennis (2021): *Entwicklung und Konstruktion eines Systemgehäuses für ein automatisiertes Backup-Ladesystem für iOS-Geräte*

[38] TDK Corp (2004): Charging apparatus and method (JP2004274528A), 30. September, Japan Patent Office, worldwide.espacenet.com, URL: https://worldwide.espacenet.com/patent/search/family/033125682/publication/JP2004274528A?q=pn%3DJP2004274528A, Abruf am 23. Juni 2021

[39] The Raspberry Pi Foundation (o. D. a): Raspberry Pi hardware, www.raspberrypi.org, URL: https://www.raspberrypi.org/documentation/hardware/raspberrypi/README.md, Abruf am 29. Juni 2021

[40] The Raspberry Pi Foundation (o. D. b): Trademark rules and brand guidelines, www.raspberrypi.org, URL: https://www.raspberrypi.org/trademark-rules/, Abruf am 23. Juni 2021

Anhang 1: backup.sh

```
#!/bin/bash

## Hauptprogramm zum Erstellen eines iOS-Backups auf einem externen
USB-Stick mittels "libimobiledevice"

# Speicherort der Sicherung
DIR="/backup.tmp"
DEV="/dev/sda1"

# Auslesen der Medienspeicherkapazität des iOS-Geräts und der Spei-
cherkapazität des USB-Sticks
DISKIOS=$(/home/iosback/usr/bin/ideviceinfo -q com.apple.disk_usage
| grep TotalDataCapacity | awk '{print $2}')
DISKUSB=$(blockdev --getsize64 /dev/sda1)

mkdir $DIR

# Auslesen des UDID des iOS-Geräts
ID=$(/home/iosback/usr/bin/idevice_id | awk '{print $1}')

# USB-stick unter Backup-Speicherort einhängen
mount $DEV $DIR && mkdir $DIR/$ID

# Überprüfen, ob die Kapazität des USB-Sticks ausreicht
if [ $DISKUSB -ge $DISKIOS ] && [ -a $DEV ]; then

      # Starten des Sicherungsprozesses
      until /home/iosback/usr/bin/idevicebackup2 backup $DIR; do
            echo failed, retry in 10s
            sleep 10
      done

# Warnung, wenn der Speicherplatz des USB-Sticks nicht ausreicht
elif [ -a $DEV ]; then
      printf "Speicherplatz nicht ausreichend! Benötigt: $DISKIOS
[Byte]; verfügbar: $DISKUSB [Byte]\n"
      exit 1

# Warnung, wenn kein USB-Stick vorhanden ist
else
      printf "Kein USB-Laufwerk vorhanden!\n"
      exit 1
fi

# Aushängen des USB-Sticks
sleep 20 && umount $DIR
```

Anhang 2: restore.sh

```
#!/bin/bash

## Programm zum Wiederherstellen eines gespeicherten Backups über
das Ladegerät

# Speicherort der Sicherung
DIR="/backup.tmp"
DEV="/dev/sda1"

# Auslesen des UDID des iOS-Geräts
ID=$(/home/iosback/usr/bin/idevice_id | awk '{print $1}')

# USB-stick unter Backup-Speicherort einhängen
sudo mount $DEV $DIR

# Überprüfen, ob ein USB-Stick vorhanden ist, welcher eine Siche-
rung des angeschlossenen Geräts enthält
if [ -d $DIR/$ID ] && [ -a $DEV ]; then

      # Wiederherstellung der Sicherung starten
      idevicebackup2 --source $ID restore --system --settings $DIR
-i

# Fehlermeldung, wenn keine Sicherung des Geräts vorhanden ist
elif [ -a $DEV ]; then
      printf "Keine Sicherung für das angeschlossene Gerät vorhan-
den!\n"

# Fehlermeldung, wenn kein USB-Stick angeschlossen ist
else
      printf "Bitte schließen sie einen USB-Stick an\n"
fi

# Aushängen des USB-Sticks
sleep 20 && sudo umount $DIR
```

Anhang 3: firstbackup.sh

```
#!/bin/bash

## Programm zum manuellen Erstellen der ersten Sicherung und Akti-
vieren der Backup-Verschlüsselung

# Speicherort des Backups
DIR="/backup.tmp"
DEV="/dev/sda1"

# Auslesen des UDID des iOS-Geräts
ID=$(/home/iosback/usr/bin/idevice_id | awk '{print $1}')

# Überprüfen, ob bereits eine Sicherung des angeschlossenen Geräts
auf dem Stick vorhanden ist
if [[ -d $DIR/$ID ]]; then
      printf "Ein Backup dieses Geräts existiert bereits!\n"
else

# Starten des Backups und Aktivieren der Verschlüsselung, wenn
keine Sicherung vorhanden ist
printf "Folgen Sie zum Erstellen des ersten Backups den folgenden
Schritten. Somit wird eine Verschlüsselung sichergestellt.\nBitte
nur ein iOS-Gerät anschließen\n"

idevicebackup2 -i encryption on && printf "\n"

# USB-stick unter Backup-Speicherort einhängen
sudo mount $DEV $DIR

# Erstellen des Zielverzeichnisses
sudo mkdir $DIR/$ID

# Aushängen des USB-Sticks
sudo umount $DEV

printf "Trennen Sie das Gerät vom Ladekabel.\n"

fi
```

Anhang 4: iosbackup.service

```
[Unit]
Description=Automatisches iOS-Backup

[Service]
Type=simple

# Wartezeit, um das Gerät sicher zu erkennen
ExecStartPre=/bin/sleep 10

# Starten des Backup-Hauptprogramms
ExecStart=/home/iosback/scripts/backup.sh

[Install]

# Service nach dem Start des "usbmuxd"-Services starten
WantedBy=usbmuxd.service
```

Anhang 5: usbmuxd udev

```
# usbmuxd (Apple Mobile Device Muxer listening on /var/run/usbmuxd)

# systemd should receive all events relating to device
SUBSYSTEM=="usb", ENV{DEVTYPE}=="usb_device", ENV{PROD-
UCT}=="5ac/12[9a][0-9a-f]/*|5ac/8600/*", TAG+="systemd"

# Initialize iOS devices into "deactivated" USB configuration state
and activate usbmuxd
SUBSYSTEM=="usb", ENV{DEVTYPE}=="usb_device", ENV{PROD-
UCT}=="5ac/12[9a][0-9a-f]/*|5ac/8600/*", ACTION=="add",
ENV{USBMUX_SUPPORTED}="1", ATTR{bConfigurationValue}="0",
OWNER="usbmuxd", ENV{SYSTEMD_WANTS}="usbmuxd.service"

# Make sure properties don't get lost when bind action is called
SUBSYSTEM=="usb", ENV{DEVTYPE}=="usb_device", ENV{PROD-
UCT}=="5ac/12[9a][0-9a-f]/*|5ac/8600/*", ACTION=="bind",
ENV{USBMUX_SUPPORTED}="1", OWNER="usbmuxd", ENV{SYS-
TEMD_WANTS}="usbmuxd.service"

# Exit usbmuxd when the last device is removed
SUBSYSTEM=="usb", ENV{DEVTYPE}=="usb_device", ENV{PROD-
UCT}=="5ac/12[9a][0-9a-f]/*|5ac/8600/*", ACTION=="remove",
RUN+="/usr/sbin/usbmuxd -x"
```

Anhang 6: Verzeichnisstruktur CD