IT-Forensik - PaymentTraces: Gone but not forgotten!

Bernhard Birnbaum, Glenn Diebetz, Pascal Heiroth, Tobias Heitmüller, Sönke Otten

Zusammenfassung—Im Zentrum dieses Projekts steht die Fragestellung, inwiefern datenschutzrelevante Daten bei app-basierten Bezahlvorgängen mit Hilfe von forensischen Methoden auf verschiedenen Datenströmen nachgewiesen werden können.

Index Terms—IT-Security, Privacy, Payment, Apps, Open Source

1 Motivation

Die zunehmende Verbreitung von app-basierten Zahlungssystemen und Kreditkartenzahlungen erfordert internetbasierte Zugriffe auf die Systeme der Zahlungsdienstleister, um die IT-Sicherheit zu gewährleisten. Leider werden bei diesen Bezahlvorgängen oft unnötige und vor allem datenschutzrelevante Daten übermittelt oder Drittanbieter kontaktiert, was sowohl die IT-Sicherheit als auch die Privatsphäre beeinträchtigen kann [1].
Neben traditionellen Zahlungsmethoden wie Zahlung auf Rechnung, Bankeinzug und Kreditkarte gewinnt die Abwicklung über einen Vermittler zunehmend an Beliebtheit. Kunden benötigen dafür ein Konto bei dem Vermittler, auf dem persönliche Daten wie Bankdaten hinterlegt werden. Die Vorteile liegen in der Flexibilität, Einfachheit und Schnelligkeit, jedoch müssen Kunden auch ein neues Konto mit personenbezogenen Daten eröffnen und es können Kommunikationsprobleme auftreten [2]. Um diesen Bedrohungen entgegenzuwirken, ist es notwendig, eine IT-forensische Untersuchung von app-basierten Bezahlvorgängen durchzuführen.
Unser Ziel nach A.1 ist es, im Rahmen eines Demonstrators den Bezahlvorgang von app-basierten Zahlungen IT-forensisch zu untersuchen und die gesamte Bearbeitungskette zu dokumentieren. Hierfür verwenden wir ein virtualisiertes Ökosystem für App-Zahlungen. Dabei werden app-basierte Zahlungsvorgänge initiiert und mit live- und post-mortem IT-forensischen Methoden begleitet, um den Massenspeicherdatenstrom (DS_T), den Hauptspeicherdatenstrom (DS_M) und den Netzwerkdatenstrom (DS_N) zu analysieren. Die Untersuchung des DS_N orientiert sich an einem bereits bestehenden Untersuchungsaufbau [3], wir weiten die Analyse aber auf DS_M und DS_T aus. Zudem wurde eine bestehende Ontologie, im *MitreAtt&ck*-Schema, entlang der Ergebnisse der Untersuchungen ergänzt und erweitert.
Im Rahmen dieser Arbeit werden zwei Zahlungsdienstleister (Amazon und Google) näher untersucht, wobei insgesamt vier Bezahlvorgänge näher betrachtet werden. Unser Ziel ist es, mögliche Bedrohungen, Schwachstellen und verletzte Sicherheitsaspekte aufzudecken, um die IT-Sicherheit und den Schutz der Privatsphäre in app-basierten Zahlungssystemen zu verbessern.

2 Stand der Technik

2.1 Sicherheit von Online-Zahlungsdiensten [Pascal]

Im Bereich der Zahlungssysteme gibt es unzählige Stimmen und Ansätze, die in Hinsicht auf die Sicherheit und die Vertrauenswürdigkeit der gebotenen Dienste betrachtet werden können.
Einige etablierte Dienstleister wie *PayPal* oder *AmazonPay* bieten eine Verschlüsselung der Zahlungsdaten an, um die Sicherheit der Transaktionen zu erhöhen. Diese Verschlüsselungstechnologien helfen dabei, die Vertraulichkeit der sensiblen Daten zu schützen und einen sichereren Datentransfer zu gewährleisten [2]. Allerdings gibt es auch bei etablierten Dienstleistern schwerwiegende Sicherheitslücken: Ein Beispiel hierfür ist *PayPal*, bei dem Betrüger nach dem Diebstahl von E-Mail-Adressen und Passwörtern in Online-Shops einkaufen können, insofern die Zwei-Faktor-Authentifizierung deaktiviert ist. Daher ist es wichtig, auf eine starke Passwortrichtlinie zu bestehen und Sicherheitsfunktionen wie die Zwei-Faktor-Authentifizierung stets zu aktivieren, um das Risiko eines unbefugten Zugriffs so gering wie möglich zu gestalten [4].
Das Bundesamt für Sicherheit in der Informationstechnik (BSI) rät dazu, sowohl die Zugangsdaten als auch die Banktransaktionen regelmäßig auf unauthorisierte Aktivitäten zu überprüfen. Somit kann man verdächtige Aktivitäten frühzeitig erkennen und unter Umständen Maßnahmen ergreifen, um hohe finanzielle Schäden zu vermeiden [5]. Eine der sichersten Varianten bleibt nach wie vor der Kauf auf Rechnung, wobei die Bezahlung erst nach dem Erhalt des Produkts erfolgt. Dadurch haben Kunden die Möglichkeit, die Ware zunächst zu prüfen, bevor eine Bezahlung stattfindet, wodurch das Risiko eines Betrugs oder einer fehlerhaften Transaktion minimiert wird.
Diese verschiedenen Stimmen im Bereich der Zahlungssysteme verdeutlichen die Bedeutung von Sicherheitsvorkehrungen und die Notwendigkeit, verantwortungsbewusst mit Zugangsdaten und Transaktionen zu hantieren. Durch eine Kombination aus sicheren Verschlüsselungstechnologien, vertrauenswürdigen Zahlungsmethoden und bewusstem Umgang mit Zugangsdaten können Endverbraucher dazu beitragen, die Sicherheit ihrer Zahlungsvorgänge selbständig zu erhöhen.

2.2 IT-forensisches Daten- und Vorgehensmodell [Pascal]

Im Bereich der forensischen Untersuchungen von IT-Systemen existiert ein Daten- und Vorgehensmodell, zur systematischen Beschreibung der Daten in IT-Systemen [6]. Damit wird vor allem die Vorfallsaufklärung unterstützt und eine strukturierte Herangehensweise bei der Untersuchung von Daten ermöglicht. Für die in dieser Arbeit durchgeführten Analysen sind hauptsächlich die folgenden drei Datenströme relevant:
Der Netzwerkdatenstrom (DS_N) umfasst die Daten, die über das Netzwerk des IT-Systems kommuniziert werden (Pakete bzw. Kommunikationsprotokolldaten). Der Hauptspeicherdatenstrom (DS_M) bezieht sich auf die Daten, die im Hauptspeicher des IT-Systems verarbeitet werden (temporäre Daten bzw. Prozessdaten). Der Massenspeicherdatenstrom (DS_T) betrifft die Daten, die auf den Massenspeichern des IT-Systems abgelegt sind (Festplatten, SSDs oder andere Speichermedien).
Um eine IT-forensische Untersuchung zu dokumentieren, können die über die Datenströme übermittelten Daten in Datentypen [7] klassifiziert werden. Aus entsprechenden Datentypen können mit geeigneten Analyse-Werkzeugen widerum andere Datentypen abgeleitet werden. Dadurch wird eine strukturierte Analyse und Evaluation der gesammelten Daten ermöglicht.
Durch die Verwendung dieses Daten- und Vorgehensmodells sowie der Aufteilung der Daten in die drei Datenströme und die Beschreibung der Untersuchung als eine Funktion auf Datentypen kann strukturiert und systematisch vorgegangen werden, um IT-Systeme forensisch zu untersuchen und besonders relevante Informationen für die Vorfallsaufklärung zu gewinnen.

2.3 Werkzeugauswahl [Pascal]

Bei der forensischen Untersuchung von IT-Systemen ist die Auswahl geeigneter Werkzeuge von ziemlich großer Bedeutung. Im Folgenden werden alle bei dieser Arbeit verwendeten Werkzeuge zur Umgebungsaufsetzung, Datenakquise und -analyse kurz vorgestellt und erläutert.
Für die Datenakquise wurden zwei Analyseumgebungen basierend auf dem *Testerstick* aufgesetzt: Der statische *Testerstick* [8] stellt eine stabile und vordefinierte Umgebung für IT-forensische Untersuchungen zur Aufzeichnung des Haupt- und Massenspeichers (DS_M, DS_T) bereit. Der dynamische *Testerstick* [9] ermöglicht die Aufzeichnung des entschlüsselten Netzwerkdatenstroms (DS_N) und ist dementsprechend vorkonfiguriert. Die *Testerstick*-Images wurden dabei mit *Rufus* [10], ein Werkzeug zum Erstellen von bootfähigen USB-Laufwerken, auf einen USB-Stick geschrieben. Mit dem Tool *gparted* [11] (ein Partitionseditor) können Anpassungen von Partitionen auf den Speichermedien durchgeführt werden. *VirtualBox* [12] ist eine Open-Source-Virtualisierungssoftware, die die Erstellung und Verwaltung von virtuellen Maschinen ermöglicht und somit eine geeignete Umgebung für IT-forensische Untersuchungen darstellt. *Android x86* [13] ist eine Portierung des Android-Betriebssystems für x86-Architekturen. Es kann als virtuelle Maschine (VM) in *VirtualBox* verwendet werden, um Android-Systeme IT-forensisch zu untersuchen.
Für die Analyse des Netzwerkdatenstroms (DS_N) stehen die folgenden Werkzeuge zur Verfügung: *Wireshark* [14] ist ein leistungsstarkes Tool zur Analyse des Netzwerkdatenstroms auf verschiedenen Protokollebenen. Es ermöglicht unter anderem eine detaillierte grafische Auswertung des Netzwerkverkehrs. Das Tool *mitmproxy* [15] ermöglicht es, die den Netzwerkverkehr zwischen Client und Server zu entschlüsseln und auf Anwendungsebene mitzulesen.
Für die Untersuchung des Hauptspeichers (DS_M) sind die folgenden Werkzeuge relevant: Mit dem Werkzeug *strings* [16] kann der Hauptspeicher nach ASCII-Zeichenfolgen durchsucht werden. Der Hexeditor *GHex* [17] ermöglicht es ebenfalls, den Abzug des Speichers zu durchsuchen. *Volatility* [18] ist ein bekanntes Werkzeug für die Analyse des Hauptspeichers, allerdings wird bis heute kein *Android-x86* unterstützt (siehe Abs. 6.2.1).
Für die Auswertung des Massenspeichers (DS_T) werden die folgenden Werkzeuge genutzt: Mit der Open-Source-Software *Autopsy* [19] kann eine umfassende forensische Analyse des Massenspeichers durchgeführt werden. Es bietet eine benutzerfreundliche Oberfläche und verschiedene Analysefunktionen. Das Werkzeug *ExtUndelete* [20] ist auf die Wiederherstellung gelöschter Dateien in EXT-basierten Dateisystemen spezialisiert, weshalb es sich insbesondere für die Analyse von Linux-basierten Systemen (*Android-x86*) geeignet.
Die Auswahl der richtigen Werkzeuge ist dabei entscheidend, um effektive IT-forensische Untersuchungen durchführen zu können. Die genannten Tools bieten eine breite Palette an Funktionen und unterstützen bei der Analyse verschiedener Aspekte von IT-Systemen.

2.4 Ontologie: Mitre Attack [Tobias]

Die MITRE ATTACK Ontologie ist ein umfassendes Wissensmodell, das speziell für die Analyse und das Verständnis von Cyberangriffen entwickelt wurde. Sie liefert eine strukturierte und standardisierte Darstellung von Angriffstechniken, Taktiken und Verfahren (TTPs), die von Angreifern genutzt werden, um in Computer- und Informationssysteme einzudringen. Die MITRE ATTACK Ontologie wurde erstmals im Jahr 2013 von der MITRE Corporation, einem gemeinnützigen Forschungsunternehmen mit Fokus auf Informationstechnologie und Cybersicherheit, entwickelt. Das Ziel der Ontologie besteht darin, ein gemeinsames Vokabular und eine gemeinsame Wissensbasis zu schaffen, um Cyberangriffe besser verstehen, analysieren und darauf reagieren zu können. Die Anwendung der MITRE ATTACK Ontologie ist in verschiedenen Bereichen der Cybersicherheit weit verbreitet. Sie wird von Sicherheitsfachleuten, Incident-Response-Teams, Sicherheitsanalysten und Forschern genutzt, um Angriffe zu klassifizieren, zu dokumentieren und zu bekämpfen. Die Ontologie ist in einer Vielzahl von Sicherheitswerkzeugen und -plattformen integriert, um bei der Erkennung und Abwehr von Angriffen zu helfen. Die Vorteile der MITRE ATTACK Ontologie liegen in ihrer Standardisierung und ihrer umfangreichen Abdeckung von Angriffstechniken. Durch die Verwendung eines einheitlichen Frameworks können Sicherheitsfachleute und Forscher leichter zusammenarbeiten,

Informationen austauschen und bewährte Verfahren entwickeln. Die Ontologie ermöglicht eine detaillierte Analyse von Angriffen, um Schwachstellen zu identifizieren, Präventionsmaßnahmen zu verbessern und effektive Verteidigungsstrategien zu entwickeln. Darüber hinaus bietet die MITRE ATTACK Ontologie eine wertvolle Ressource für die Ausbildung von Sicherheitspersonal und die Entwicklung von Schulungen. Sie ermöglicht es Unternehmen und Organisationen, ihre Verteidigungsfähigkeiten zu stärken, indem sie das Verständnis für die Taktiken und Techniken von Angreifern verbessern und effektive Gegenmaßnahmen entwickeln. Insgesamt ist die MITRE ATTACK Ontologie zu einem wichtigen Instrument im Bereich der Cybersicherheit geworden, das dabei hilft, die Bedrohungslandschaft besser zu verstehen, Verteidigungsstrategien zu entwickeln und die Sicherheit von Computer- und Informationssystemen zu verbessern [21] [22].

3 Konzept

3.1 Vorüberlegungen [Bernhard]

Bei dieser Untersuchung werden drei Datenströme betrachtet, an denen das mobile Endgerät des Nutzers während eines Bezahlvorgangs beteiligt ist. Dafür ist ein Vollzugriff (Root-Rechte) auf das Android-System erforderlich, weshalb eine virtualisierte Lösung verwendet wird. Die VM läuft dabei, wie auch alle Analysetools, in der Untersuchungsumgebung des bereitgestellten Testersticks. Durch die Virtualisierung wird es später sehr einfach, den Haupt- und Massenspeicher (DS_M, DS_T) des Systems auszulesen. In einem realen Szenario würde ein mobiles Endgerät bei einem Bezahlvorgang eine direkte Verbindung mit dem Zahlungsdienstleister aufbauen. Um den Netzwerkdatenstrom (DS_N) auszulesen wird der gesamte Traffic von *Wireshark* mitgeschrieben und zusätzlich über einen Proxy (*mitmproxy*) geleitet, welcher zwischen dem virtuellen Android-System und dem Server des Zahlungsdienstleisters in der Untersuchungsumgebung positioniert wird.
Das Zusammenwirken der verschiedenen beteiligten Systeme ist in Abbildung 1 als IT-Systemlandschaft schematisch dargestellt.

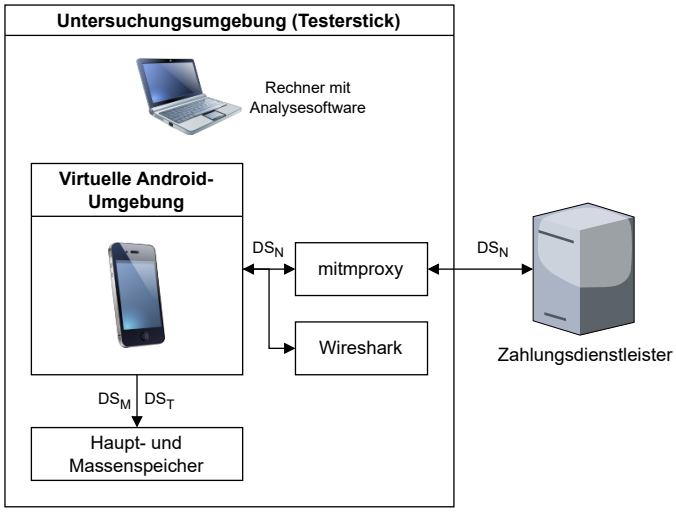


Abbildung 1: Darstellung der IT-Systemlandschaft

3.2 Aufbau der Analyse-Infrastruktur [Bernhard]

Dieser Abschnitt behandelt die aus der vorgestellten IT-Systemlandschaft abgeleitete Analyse-Infrastruktur.
Basierend auf den bereitgestellten *Testerstick*-Images der Arbeitsgruppe AMSL werden insgesamt drei voneinander isolierte Untersuchungsumgebungen benötigt, um alle geforderten Analysen durchführen zu können: statisch, dynamisch und post-mortem.
Die statische und dynamische Umgebung sind für die **Datenakquise** gedacht, in denen jeweils die Bezahlvorgänge ausgeführt und aufgezeichnet werden. Die **dynamische** Umgebung (`d`) ist dabei primär für die Erfassung des Netzwerkdatenstroms (DS_N) der Anwendungsschicht ausgelegt, da über einen Proxy der Datenverkehr m.H. eines entsprechenden Zertifikats entschlüsselt wird. Aus der **statischen** Umgebung (`s`) soll der Haupt- und Massenspeicher (DS_M, DS_T) gesichert und zusätzlich ein zweites Mal der Netzwerkdatenstrom (verschlüsselt, aber auf allen Protokollebenen) aufgezeichnet werden. Die dritte und letzte benötigte Umgebung stellt alle Werkzeuge für eine **post-mortem-Analyse** (`pm`) bereit. Damit sollen die in der statischen und dynamischen Umgebung gesammelten Daten weiter untersucht werden.

3.2.1 Vorbereitung der Downloader- und Analyse-VMs

Bevor die eigentlichen Untersuchungsumgebungen aufgesetzt und konfiguriert werden, sollten die zu untersuchenden Apps für ein effizientes Vorgehen zuerst heruntergeladen und dann isoliert in jeweils eine Analyse-VM mit *Android x86* installiert werden. Anschließend kann die Analyse-VM in die statische bzw. dynamische Umgebung importiert werden, um mit der Datenakquise zu beginnen. Die bis zu diesem Punkt notwendigen vorbereitenden Schritte sowie die Struktur der Analyseumgebungen sind in Abbildung B.1 schematisch dargestellt. Ausgangspunkt dafür stellt die vom AMSL bereitgestellte *VirtualBox*-Appliance `Android x86-32.ova` dar, welche bereits ein fertig installiertes *Android 7.1*-System beinhaltet.
Um die gegebenen VMs so leichtgewichtig wie möglich anzubieten, ist der Speicherplatz der virtuellen Festplatten stark begrenzt. Aus diesem Grund müssen - bevor eine App installiert werden kann - zunächst die virtuelle Festplatte, die darauf befindliche Partition und auch das Dateisystem vergrößert werden. Desweiteren müssen, um die Kompatibilität mit der dynamischen Untersuchungsumgebung zu gewährleisten, in den Netzwerkeinstellungen der Android-VM weitere Anpassungen vorgenommen sowie das MITM-Zertifikat installiert werden. Abschließend wird ein Sicherungspunkt der Android-VM angelegt, da ab diesem Punkt mehrere Instanzen dieser VM benötigt werden. Die zur Umsetzung notwendige Schrittfolge wird später in Abs. 4.1.1 ausführlich erläutert.
Ausgehend vom Sicherungspunkt wird im nächsten Schritt eine **Downloader-VM** aufgesetzt. Dies ist unbedingt notwendig, da der APK-Download über den *Google Play Store* läuft, wozu ein Login bei Google im Android-System zwingend benötigt wird. Problematisch daran ist, dass nach einem Login bei Google auf dem Android-System viele Dienste ausgeführt werden. Sollten während der Untersuchung in der Analyse-VM mehr Apps ausgeführt werden

als unbedingt notwendig, kann es unter Umständen zu Wechselwirkungen zwischen den Apps und Hintergrundrauschen kommen, wodurch potentielle Untersuchungsergebnisse verfälscht werden können.
Nach dem Download der benötigten Apps aus dem *Google Play Store* müssen sie aus der Downloader-VM extrahiert werden. Die APKs können dann in einem nächsten Schritt ausgehend vom zuvor angelegten Sicherungspunkt jeweils in „frische" Android-Systeme, die **Analyse-VMs**, ohne Google-Login installiert werden. Eine vollständige Liste aller notwendigen Schritte und Befehle zum Herunterladen und Installieren der APKs ist in Abschnitt 4.1.2 zu finden.
Im letzten Schritt der Vorbereitung können alle Analyse-VMs für die Analyse als OVA exportiert werden und stehen dann bereit für die Untersuchung. Das OVA-Format (Open Virtualization Format) ist für das Exportieren von VMs prädestiniert, da zusätzlich zum vollständigen Systemzustand (d.h. Betriebssystem, Speicher, Anwendungen) die Konfiguration der VM mit abgespeichert wird. Da außerdem eine Kompression angewandt wird, sind OVA-Dateien relativ platzsparend.

3.2.2 Untersuchungsumgebungen und Werkzeuge

Ausgangspunkt für die in dieser Arbeit durchgeführten Untersuchungen waren die *Testerstick*-Images des AMSL. Daraus wurden insgesamt drei voneinander separat laufende Untersuchungsumgebungen (alle *Debian 11*) abgeleitet:
In der **dynamischen Untersuchungsumgebung** (`d`) wird der Netzwerkverkehr über die Software *mitmproxy* umgeleitet. Dabei wird ein MITM-Zertifikat verwendet, um den Netzwerkdatenstrom (DS_N) auf der Anwendungsschicht zu entschlüsseln. In diesen Daten kann später nach auffälligen Informationen und Schlüsselwörtern zum Bezahlvorgang gesucht werden.
In der **statischen Umgebung** (`s`) hingegen kann der Netzwerkdatenstrom (DS_N) zwar nicht entschlüsselt werden, er wird allerdings trotzdem mit *Wireshark* erfasst, um zusätzliche Ebenen der Netzwerkkommunikation zu erfassen. Der Fokus liegt hier allerdings auf dem Massen- und Hauptspeicher (DS_T, DS_M), wobei beide Datenströme mit Utility-Funktionen von *VirtualBox* gesichert werden können.
Für die spätere Analyse der gesammelten Daten stellt die **post-mortem-Umgebung** (`pm`) eine Auswahl von Analysetools bereit. Alle Werkzeuge werden im folgenden Abs. 3.2.3 zu den betroffenen Datenströmen bzw. Umgebungen zugeordnet.

3.2.3 Einordnung in forensisches Datenmodell

Die in Tabelle 1 dargestellte Einordnung in das Datenmodell nach Kiltz [6] p.65 ist durch folgende Punkte motiviert:
Der Netzwerkdatenstrom DS_N besteht zunächst aus Kommunikationsprotokolldaten (DT_5); bei einem Bezahlvorgang werden allerdings auch Daten zur Authentifizierung des Nutzers (Sitzungsdaten, DT_7) übertragen, welche aber nur von *mitmproxy* entschlüsselt werden können.
Der Hauptspeicher DS_M liegt nach dem Dump zunächst als Rohdaten (DT_1) vor. Im Optimalfall können durch gezielte Schlüsselwortsuchen Daten der ausgeführten App (Prozessdaten, DT_6) oder auch Sitzungsdaten (DT_7) extrahiert werden.
Über den Massenspeicher DS_T können über das Dateisystem Metadaten (DT_3) sowie Konfigurationsdaten (DT_4), allerdings auch Nutzerdaten (DT_8) ausgelesen werden.

Tabelle 1: Übersicht der verwendeten Analysetools mit Zuordnung zu betroffenen Datenströmen, Datenarten und Umgebungen

Werkzeuge	Datenströme	Datenarten		Umgebung
		Eingang	Ausgang	
VirtualBox	DS_T, DS_M	n.a.	DT_1	`s`, `d`
Wireshark	DS_N	DT_5	DT_5	`s`, `pm`
mitmproxy	DS_N	DT_5	DT_5, DT_7	`d`, `pm`
strings	DS_M	DT_1	DT_6, DT_7	`pm`
GHex	DS_M	DT_1	DT_6, DT_7	`pm`
Autopsy	DS_T	DT_1	DT_3, DT_4, DT_8	`pm`
ExtUndelete	DS_T	DT_1	DT_3, DT_4, DT_8	`pm`

3.3 Methodik der Datenakquise [Glenn]

In dieser Sektion des Reports gehen wir näher darauf ein, wie wir die Apps analysiert haben und wie wir dabei vorgegangen sind. Zuerst brauchen wir eine Analyse-VM im Rahmen dieses Reports haben wir eine Android VM benutzt, die die benötigten Apps bereits vorinstalliert hatte. Im weiteren Schritt kann man die Untersuchung in mehrere Phasen einteilen wie auch in B.2 beschrieben, wir starten mit der ersten Phase der Datenakquise:
Phase 1 - Datenakquise: In dieser Phase ist unser Hauptziel das sammeln von so vielen nützlichen Daten wie möglich. Wir können diese Phase in 2 Umgebungen unterteilen, einmal die dynamische Umgebung und einmal die statische Umgebung. In der dynamischen Umgebung starten wir zuerst unsere Analyse-VM und verbinden uns dann mit einem Proxy um die Aufzeichnung in *mitmproxy* zu starten, danach führen wir unseren Bezahlvorgang durch und speichern die Aufzeichnung als `.mitm`-Datei ab. Danach können wir die Datenakquise für die statische Umgebung starten, dazu starten wir wieder unsere Analyse-VM und danach starten wir eine Wireshark Aufzeichnung bevor wir den Bezahlvorgang starten. Dann führen wir den Bezahlvorgang durch und speichern die Wireshark Datei ab denn diese werden wir zur Analyse des Netzwerkdatenstroms benutzen. Dann dumpen wir den Hauptspeicher der laufenden VM und kopieren die virtuelle Disk aus der VM, um den Massenspeicher zu analysieren. Nachdem wir nun die Daten gesammelt haben können wir mit der nächsten und letzten Phase weitermachen.

3.4 Methodik der Analyse [Glenn]

Nachdem wir die Daten gesammelt haben können wir sie analysieren; im folgenden wird unsere Vorgehensweise beschrieben:
Phase 2 - Analyse: In dieser Phase analysieren wir die in Phase 1 gewonnenen Daten aus den 3 Hauptdatenströmen. Für die Analyse des Netzwerks haben wir nun einmal die `.mitm`-Datei von *mitmproxy* die wir analysieren können und wir haben die `.pcapng`-Datei von *Wireshark*. Für die Analyse des Hauptspeichers nutzen wir *strings* um die `.elf`-Datei auszuwerten die wir durch den Dump bekommen haben

und um alle Zeichenketten aus der Datei zu extrahieren. Für die Analyse des Massenspeichers benutzen wir *Autopsy* und *ExtUndelete*, dabei können wir gelöschte Dateien auf dem Massenspeichermedium wiederherstellen und schauen, was alles jemals gespeichert wurde.
Nachdem wir alle Daten gesammelt und ausgewertet haben können wir die Ergebnisse der Auswertung ausführlich und übersichtlich zusammenstellen und unsere Untersuchung beenden.

3.5 Vorüberlegungen zur Erweiterung der MITRE ATTACK Ontologie [Tobias]

Um die Ontologie auf den Bezahlvorgang anzuwenden, muss zunächst eine detaillierte Analyse des Bezahlvorgangs durchgeführt werden. Es geht darum, die verschiedenen Aktivitäten, Prozesse oder Methoden zu identifizieren, die während des Bezahlvorgangs auftreten. Besonderes Augenmerk sollte dabei auf spezifische Schritte, Interaktionen oder Datentransfers gelegt werden. Anschließend erfolgt der Vergleich der identifizierten Aktivitäten oder Prozesse mit den Taktiken und Techniken in der MITRE ATTACK Ontologie. Dabei wird nach ähnlichen oder verwandten Begriffen gesucht, die den Bezahlvorgang beschreiben könnten. Die Suche in der Ontologie erfolgt durch die Verwendung der Suchfunktion, um nach relevanten Begriffen oder Schlagwörtern zu suchen, die mit den identifizierten Aktivitäten oder Prozessen zusammenhängen könnten. Diese Begriffe werden in das Suchfeld eingegeben und die Ergebnisse werden geprüft. Die Zuordnung der identifizierten Aktivitäten oder Prozesse zu den entsprechenden Taktiken oder Techniken in der Ontologie erfolgt durch eine sorgfältige Überprüfung der Beschreibungen. Es wird festgestellt, welche Taktiken oder Techniken am besten zu den identifizierten Aktivitäten passen. Um die Richtigkeit der Zuordnung zu gewährleisten, wird eine Verifizierung durchgeführt. Es wird sichergestellt, dass die ausgewählte Taktik oder Technik in der Ontologie die Aktivitäten und Prozesse des Bezahlvorgangs genau widerspiegelt. Dabei werden die Beschreibungen und Beispiele überprüft. Die identifizierten Taktiken oder Techniken werden schließlich festgehalten und in der Analyse dokumentiert. Dabei werden die entsprechenden Begriffe aus der MITRE ATTACK Ontologie verwendet, um eine klare und einheitliche Terminologie zu gewährleisten [23].

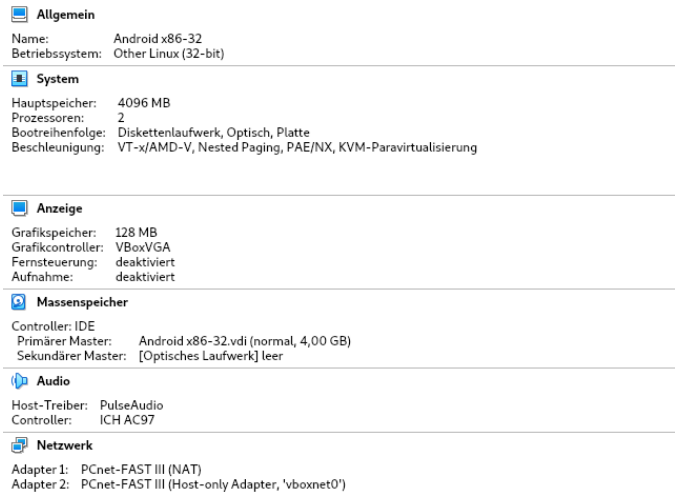
4 Implementierung

4.1 Aufsetzen der Android Analyse-VMs [Bernhard]

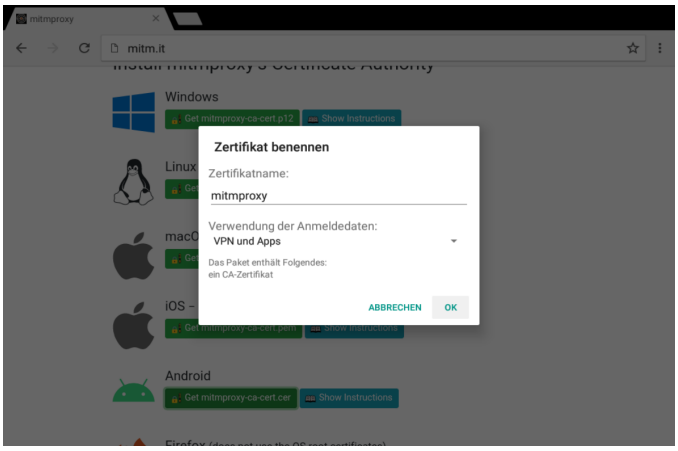
In Abschnitt 3.2.1 wurde motiviert, alle zu untersuchenden Apps in jeweils eine eigenständige Analyse-VM zu installieren. In diesem Abschnitt wird eine konkrete Schrittfolge gegeben, welche Anpassungen vom Ausgangspunkt der „`Android x86-32.ova`“ bis zur fertig konfigurierten Analyse-VM-Appliance (siehe Abbildung B.1, oberer Abschnitt) umgesetzt werden müssen.

4.1.1 Importieren der Appliance und VM-Konfiguration

1) **Appliance importieren**:
   Um eine Appliance in *VirtualBox* (Version 7.0.2) zu importieren, wird folgender Menüpunkt ausgewählt: `Datei` → `Appliance importieren ...` Anschließend kann der Pfad zur `.ova`-Datei angegeben werden; Voreinstellungen übernehmen, mit `Fertigstellen` bestätigen.
2) **Vergrößern des Speicherplatzes der VM**:
   Um die virtuelle Festplatte zu vergrößern, wird `Datei` → `Werkzeuge` → `Virtuelle Medien` aufgerufen und „`Android x86-32.vdi`“ ausgewählt, dann kann in den `Eigenschaften` der Regler `Größe` angepasst und mit `Sichern` bestätigt werden.
   Um die Partition und das Dateisystem zu vergrößern, muss in den Einstellungen der VM unter dem Punkt `Massenspeicher` → `Sekundäres IDE-Gerät 0` ein Live-ISO eines beliebigen debian-basierten Linux-Systems ausgewählt und gestartet werden. Anschließend wird mit dem Befehl `sudo apt-get install gparted -y && sudo gparted` das Partitionierungstool *gparted* auf dem Live-System installiert und gestartet. Im GUI kann nun die Partition und das Dateisystem der virtuellen Festplatte des Android-Systems vergrößert werden.
   Nach dem Verändern der Festplatten kann es notwendig sein, den Bootloader zu aktualisieren, damit das System weiterhin normal starten kann. In diesem Fall muss zusätzlich `sudo update-grub` ausgeführt werden.
   Insofern der Prozess erfolgreich durchlaufen wurde, kann das Live-System heruntergefahren werden und aus den Einstellungen der VM entfernt werden. Sollte während des Vergrößerns eine Fehlermeldung durch bereits vergebene UUIDs auftreten, kann der virtuellen Festplatte mit dem Befehl `VBoxManage internalcommands sethduuid 'Android x86-32.vdi'` eine neue UUID zugewiesen werden [24].
3) **Konfiguration der VM**:
   Nachdem die Festplatte und das Dateisystem vergrößert wurden, können die Einstellungen der VM konfiguriert werden. Für alle in dieser Arbeit durchgeführten Analysen wurden die Parameter entsprechend Abbildung 2a gesetzt. Für den Host-only Adapter muss unter `Datei` → `Werkzeuge` → `Netzwerk-Manager` „`vboxnet0`“ mit `192.168.56.1` hinzugefügt werden. Als letztes muss die Android-VM gestartet werden, um im System das MITM-Zertifikat zu installieren. Im



(a) Android-VM-Einstellungen



(b) Installationsbestätigung des mitm-Zertifikats

Abbildung 2: Konfiguration der Android-VM

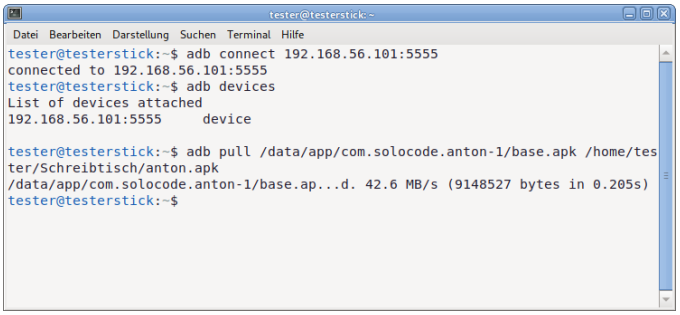
Android-System wird z.B. über den Google Chrome-Browser http://mitm.it aufgerufen. Insofern *mitmproxy* ausgeführt wird, kann von dort das Zertifikat für Android-Systeme heruntergeladen werden. Nach dem Download und dem Bestätigen der Meldung (siehe Abbildung 2b) wird es automatisch installiert.

4) **Sicherungspunkt erstellen**:
An diesem Punkt wird von der konfigurierten VM ein Sicherungspunkt erzeugt (unter `Sicherungspunkte` → `Erzeugen`), von dem aus wie in Abschnitt 3.2.1 konzeptioniert eine **Downloader-VM** sowie für alle zu untersuchenden Apps jeweils eine **Analyse-VM** erstellt werden kann.

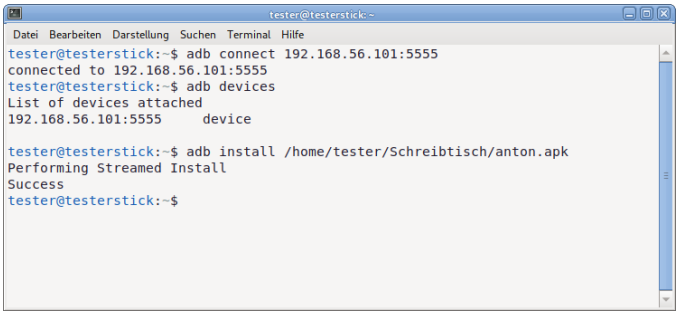
4.1.2 *Installation von Apps*

Ausgehend vom Sicherungspunkt wird zum einen eine **Downloader-VM** erstellt, in der dann die Apps aus dem *Google Play Store* heruntergeladen werden. Zum anderen werden anschließend die heruntergeladenen APKs von diesem Sicherungspunkt ausgehend in einzelne **Analyse-VMs** installiert.

1) **Downloader-VM erstellen**: Nach dem Starten der VM wird der *Google Play Store* geöffnet. An dieser Stelle ist ein Login mit einem Google-Konto verpflichtend. Nach dem Login können alle zu untersuchenden Apps installiert werden. Die APK-Dateien aller installierter Apps werden an folgender Stelle im Android-System der Downloader-VM abgelegt: `/data/app/*/base.apk`. In Abbildung 3a ist zu sehen, wie die APK für eine App aus der Downloader-VM mit *adb* geborgen werden kann (exemplarisch für die *Anton-App*). Ein Versuch, die Apps über den *Aurora Store* zu installieren (ohne Google-Login), welcher über *F-Droid* installiert werden kann, schlug auch nach mehreren Versuchen fehl (siehe Abs. 4.6.1).
2) **Analyse-VM erstellen**: Die Analyse-VM muss für jede App separat gebaut werden. Zunächst wird die VM auf den zuvor gesetzten Sicherungspunkt zurückgesetzt, um den Google Login auf der VM rückgängig zu machen. Anschließend kann, wie in Abbildung 3b gezeigt, die entsprechende APK mit *adb* installiert werden. Danach kann die Android-VM heruntergefahren werden. Über das Menü `Datei` → `Appliance exportieren ...` wird die Analyse-VM abschließend gespeichert, um später für die Bezahlvorgänge in jeweils beide Untersuchungsumgebungen zur Datenakquise importiert zu werden.



(a) Befehle zum Bergen einer APK aus der Downloader-VM



(b) Befehle zum Installieren einer APK in die Analyse-VM

Abbildung 3: APK-Transfer mit *adb* von Downloader-VM in Analyse-VM

4.2 Untersuchungsumgebungen [Bernhard]

Die Einteilung der Untersuchungsumgebungen in statisch, dynamisch und post-mortem wurde bereits konzeptuell in Abs. 3.2.2 beschrieben und ist ebenfalls in Abbildung B.1 dargestellt. Wie aus Tabelle 1 entnommen werden kann, müssen in den drei Umgebungen jeweils verschiedene Tools installiert werden.

Für diese Arbeit wurde das gegebene *Testerstick*-Image „`Testerstick.img`“ mit dem Tool *Rufus* auf einen USB-Stick geschrieben. Auch hier der vorhandene Speicherplatz für die Arbeit mit großen VMs nicht ausreichend. Deshalb muss, ähnlich wie in Abs. 4.1.1 Schritt 2 bereits ausgeführt, die Partition sowie das Dateisystem des Systems vergrößert werden. Am effektivsten ist dies mit einem Debian Live-System und *gparted* möglich.

Für die **statische Untersuchungsumgebung** sind bereits alle benötigten Tools (*VirtualBox* und *Wireshark*) auf dem bereitgestellten Image installiert. An diesem Punkt können die in Abs. 4.1.2 erstellen Analyse-VMs importiert werden (in *VirtualBox* unter `Datei → Appliance importieren ...`). In der statischen Umgebung reicht es, wenn der Netzwerk-Adapter der Analyse-VM auf „NAT“ oder „Netzwerkbrücke“ gestellt ist.

Auf der **dynamischen Umgebung** ist *mitmproxy* ebenfalls bereits installiert. Um tatsächlich den gesamten Netzwerkverkehr des Systems über den Proxy zu leiten, muss in den Systemeinstellungen unter `System → Einstellungen → Internet und Netzwerk → Netzwerkvermittlung` die in Abbildung 4 gezeigte Konfiguration angewandt werden. Zu beachten ist außerdem, dass der Netzwerk-Adapter wie in Abbildung 2a gezeigt konfiguriert ist.

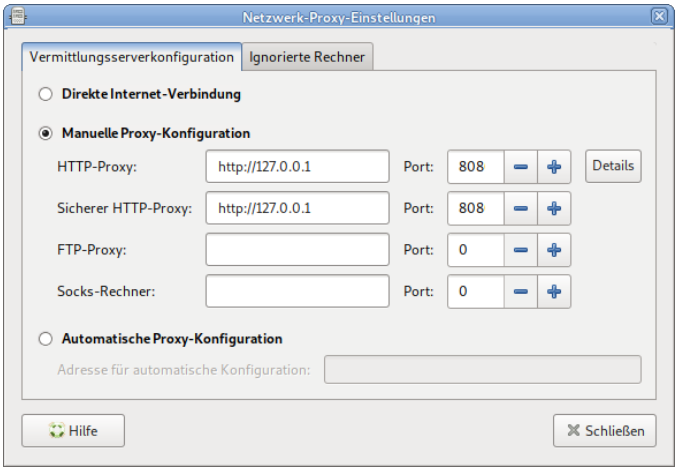


Abbildung 4: Konfiguration der Netzwerkvermittlung für *mitmproxy* in Debian Systemeinstellungen (Ports sind in beiden Fällen 8080)

In der **post-mortem-Umgebung** werden die meisten Werkzeuge benötigt. Um die Aufzeichnungen zum Netzwerkdatenstrom (DS_N) zu untersuchen, werden *Wireshark* bzw. *mitmweb* benötigt, welche beide bereits auf dem *Testerstick*-Image installiert sind. Für die Rohdaten-Untersuchung (DT_1) des Hauptspeichers (DS_M) durchzuführen, wird *GHex* bzw. *strings* verwendet. Beide Tools können mit dem Befehl `sudo apt-get install ghex strings -y` installiert werden kann. Für die Analyse des Massenspeichers (DS_T) wird primär *Autopsy* benötigt; *extundelete* muss mit dem Befehl `sudo apt-get install extundelete -y` installiert werden.

4.3 Umsetzung der Datenakquise [Glenn]

Für das Extrahieren der Daten von den Bezahlvorgängen haben wir zwei verschiedene Ansätze verfolgt (siehe Abb. B.2). Einmal haben wir eine statische Umgebung in der wir das Netzwerk, den Hauptspeicher und den Massenspeicher analysieren können. Ebenso haben wir eine dynamische Umgebung in der wir uns über *mitmproxy* verbinden, um zur Laufzeit zu überprüfen mit welchen Adressen wir uns verbinden während wir den Bezahlvorgang durchführen. Im folgenden werden die beiden Umgebungen weiter erklärt:

1) **statische Umgebung**:
   Für die Sammlung der Daten nutzen wir *Wireshark* und die Funktionen von *VirtualBox* zur Diskextrahierung auf das wir im folgenden näher eingehen werden.
   - **Wireshark**:
     Mit *Wireshark* ist es uns möglich den Netzwerkverkehr aufzuzeichnen und für die Analyse in einer `.pcapng`-Datei zu speichern. Dadurch können wir sehr genau nachvollziehen mit welchen IP-Adressen wir uns verbinden und mit welchen Protokollen wir kommunizieren.
   - **VirtualBox** (DS_M):
     Nachdem wir mit dem Befehl `vboxmanage debugvm 'Win7' dumpvmcore {filename file.elf` den Hauptspeicher gedumped haben bekommen wir ein Abbild des Hauptspeichers in einer `.elf`-Datei. Diese beinhaltet alle Zeichenketten die aus dem Hauptspeicher gedumped werden konnten. Das ermöglicht es uns die Zeichenketten zu extrahieren und somit den Inhalt des Hauptspeichers zu analysieren.
   - **VirtualBox** (DS_T):
     Wenn wir den Bezahlvorgang abgeschlossen haben fahren wir die Virtuelle Maschine herunter und können die virtuelle Disk (`.vdi`) extrahieren, dazu sei gesagt das wir die `.vdi` vorher noch in ein anderes Dateiformat umwandeln müssen damit wir sie mit Autopsy analysieren können. Das bewerkstelligen wir indem wir auf die Datenspeicherung von *VirtualBox* gehen wo alle virtuellen Disks aufgelistet sind, diese befindet sich unter **Werkzeuge**, dort gehen wir dann auf kopieren und wählen den Dateityp VHD (Virtual Hard Disk) oder VMDK (Virtual Machine Disk) aus.
2) **dynamische Umgebung**:
   In der dynamischen Umgebung verwenden wir die Android VM und *mitmproxy* mit dem wir uns verbinden.
   - **mitmproxy**:
     Während der Ausführung des Bezahlvorgangs wird fortlaufend in eine `.mitm`-Datei reingeschrieben mit welchen Adressen wir uns verbinden sowie die nicht verschlüsselten Daten die wir übertragen.

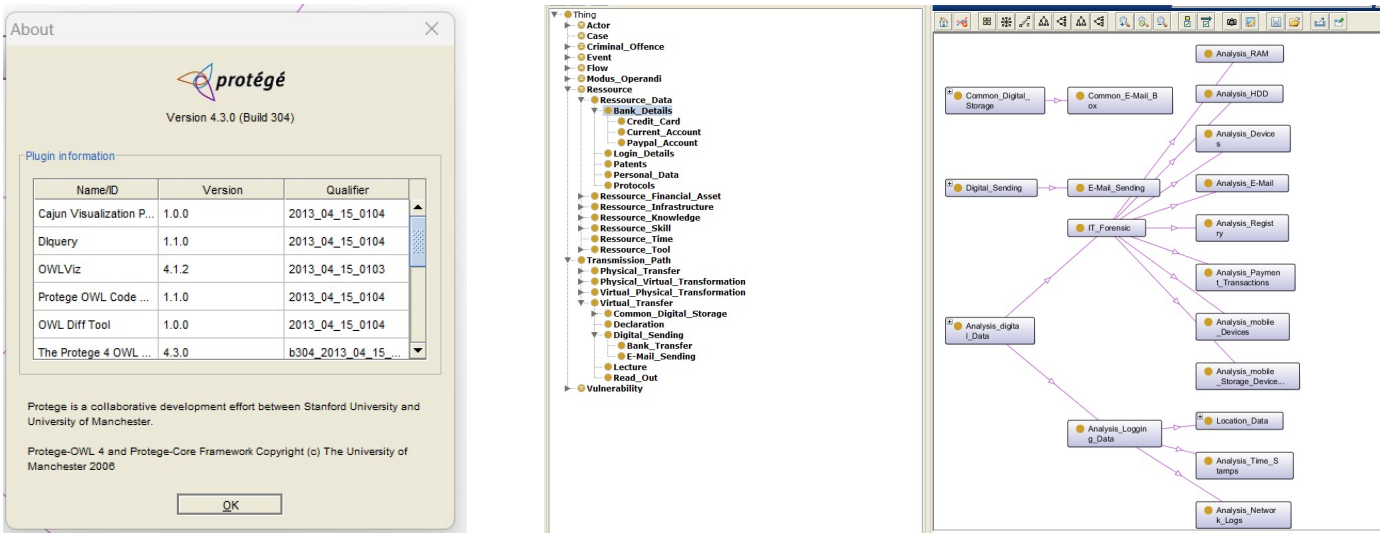
4.4 Umsetzung der Analyse [Glenn]

Nachdem wir die Daten gesammelt haben, können wir sie analysieren, im Rahmen dieser Arbeit benutzen wir:

- **strings**: Nachdem wir den Hauptspeicher mit dem Befehl `vboxmanage debugvm 'Win7' dumpvmcore {filename file.elf` gedumped haben, können wir die Datei mit *strings* analysieren, indem wir den Befehl `strings -n 5 file.elf > file.txt` ausführen. Dabei extrahiert *strings* alle ASCII-Zeichenketten, die es in der Datei `file.elf` finden kann, und schreibt sie in die Datei `file.txt`. Die Extrahierung der Zeichenketten würde aber zu lange dauern, deshalb bietet uns *strings* einige Parameter und Möglichkeiten dies zu beschleunigen. Wir spezifizieren das wir nur Zeichenketten extrahieren wollen die eine Minimumlänge von 5 haben, damit wir nur die relevanten Zeichenketten bekommen.
- **Autopsy**: Wenn wir den Bezahlvorgang abgeschlossen haben fahren wir die Virtuelle Maschine herunter und können die virtuelle Disk (`.vmdk`) in *Autopsy* als Quelle hinzufügen. Somit können wir den Massenspeicher auf zum Beispiel E-Mail-Adressen und sämtliche Dateiformate die gespeichert wurden, untersuchen. Dabei gibt *Autopsy* uns mehrere Optionen, wir können nach Keywords suchen wie zum Beispiel Passwörtern oder E-Mail-Adressen. Es gibt einen komplett eigenen Android Analyzer den wir ebenfalls installiert haben für die Analyse.
- **ExtUndelete**: Wenn wir die virtuelle Disk extrahiert haben aus der VM können wir mit *ExtUndelete* vom Datenträger alle gelöschten Dateien wiederherstellen und diese zur weiteren Analyse benutzen. Im weiteren ist eine detaillierte Schrittfolge, wie man mit *ExtUndelete* die Analyse durchführen kann:
  1) **Umwandeln in img**:
     `vboxmanage clonehd massstorage.vdi massstorage.img --format RAW`
  2) **Offset auslesen**: `fdisk -lu massstorage.img`; davon dann den Wert der Spalte „Beginn“ mit 512 multiplizieren (i.d.R. $63 \cdot 512 = 32256$)
  3) **Einhängepunkt erstellen**:
     `sudo mkdir /mnt/disk.img.partition0`
  4) **Img-Datei als Laufwert einhängen**:
     `sudo mount -o loop,offset=32256 massstorage.img /mnt/disk.img.partition0` [25]
  5) **extundelete starten**:
     `extundelete /dev/loop0 --restore-all`
- **GHex**: Nachdem wir die `.elf`-Datei gedumped haben können wir mit *GHex* einen Blick auf die Hex-Darstellung der Datei werfen, um mögliche sensible Daten herauszufiltern.

4.5 Implementierung der MITRE ATTACK Ontologie [Tobias]

Die Mitre Attack Ontologie wurde dem Projektteam von der Arbeitsgruppe AMSL der Otto-von-Guericke Universität zur Verfügung gestellt und wird mithilfe von *protégé* [26] genutzt bzw. bearbeitet (siehe Abb. 5a). Die Anwendung *protégé* ist eine Softwareplattform zur Arbeit mit ontologischen Modellen, einschließlich der MITRE ATTACK Ontologie. *protégé* bietet eine benutzerfreundliche Umgebung zur Erstellung, Bearbeitung und Visualisierung von Ontologien und ermöglicht es Benutzern, komplexe Wissensmodelle effektiv zu verwalten. In Bezug auf die MITRE ATTACK Ontologie bietet *protégé* eine umfassende Palette von Funktionen, um die Arbeit mit dieser Ontologie zu erleichtern. Es ermöglicht Benutzern, die verschiedenen Taktiken, Techniken und Beziehungen in der Ontologie zu erkunden und zu visualisieren. Durch die intuitive Benutzeroberfläche von *protégé* können Benutzer die Ontologie durchsuchen, Konzepte hinzufügen, Eigenschaften definieren und Beziehungen zwischen den Konzepten herstellen. Darüber hinaus unterstützt *protégé* den Import und Export von Ontologien in verschiedenen Formaten, was die Interoperabilität mit anderen Werkzeugen und Plattformen erleichtert. Es ermöglicht



(a) Version 4.3.0 - *protégé*

(b) Beispielhafte Ansicht in *protégé*

Abbildung 5: Beispielbilder *protégé*

auch die Zusammenarbeit und den gemeinsamen Zugriff auf Ontologien, indem es Versionierungsfunktionen und die Integration mit Versionskontrollsystemen bietet. Dank der Flexibilität und Anpassungsfähigkeit von *protégé* können Benutzer die MITRE ATTACK Ontologie nach ihren individuellen Anforderungen erweitern und anpassen. Sie können benutzerdefinierte Klassifikationen, Eigenschaften und Beziehungen definieren, um die Ontologie an ihre spezifischen Anwendungsfälle anzupassen. *protégé* kann über die Homepage der Firma über den Downloadbutton heruntergeladen und installiert werden. Für diese Arbeit wurde die Version 4.3.0 verwendet [21].
In Abbildung 5b ist eine beispielhafte Ansicht der gegebenen Ontologie im Top-Down-Menü sowie als grafische Darstellung zu sehen.

4.6 Probleme bei der Umsetzung [Bernhard]

4.6.1 Aurora-Store

Der *Aurora Store* von *F-Droid* [27] bietet die Möglichkeit, Android-Apps zu installieren, ohne einen Google-Account zu besitzen. Da dies eine datensparsame Alternative darstellt wurde zunächst versucht, die zu untersuchenden Apps darüber zu installieren.
Dabei sind allerdings mehrere Probleme entstanden, die nicht vollständig gelöst werden konnten: Zum einen laden die App-Vorschläge nur partiell, wodurch keine Apps gezielt gesucht werden können. Desweiteren bricht der Versuch, eine App zu installieren, ohne Fehlermeldung ab bzw. friert auf unbestimmte Zeit ein.
Aufgrund dieser Probleme wurde letztendlich der Ansatz mit einer Downloader-VM und dem Google-Play-Store umgesetzt.

4.6.2 Wiederholter Absturz von Apps

Einige für die Untersuchung interessante Apps wie *PayPal*, *giropay*, oder auch Spiele stürzen in der virtuellen *Android-x86*-Umgebung wiederholt sofort nach dem Start ab.
Vermutlich ist dieses Problem entweder auf eine Inkompatibilität mit *Android-x86* oder den Rooting-Umstand der Umgebung zurückzuführen. Die betroffenen Apps könnten erweiterte Sicherheitsprüfungen oder ein AntiCheat-System implementiert haben, welche den Rooting-Umstand erkennen und die App beenden (Safety-Check).
Wie diesem Problem bei zukünftigen Arbeiten entgegnet werden könnte, wird in Abs. 6.2.2 skizziert.

4.6.3 Detektion des mitm-Zertifikats

Durch den in dieser Arbeit implementierten Untersuchungsaufbau bedingt sind die Online-Features einiger Dienste beeinträchtigt oder funktionieren nicht, da der Netzwerkdatenstrom über *mitmproxy* geleitet wird.
Einige Apps scheinen das Zertifikat vor einem Verbindungsaufbau zu überprüfen und blockieren einen Datenaustausch, wenn ein abweichenden Zertifikat (mitm-Zertifikat) erkannt wird. In einer weiterführenden Arbeit könnte diskutiert werden, welche Möglichkeiten es gibt, diesem Problem zu begegnen (siehe Abs. 6.2.3).

5 Evaluierung

5.1 Ergebnisse der App-Analysen

5.1.1 App-Kauf im Amazon Store [Pascal]

Der Bezahlvorgang in der App *Amazon App Store* ist unser erster Versuch gewesen, die drei Datenströme zu analysieren. Dafür war es zunächst notwendig eine Bankverbindung, mit dem Swift-Code, der IBAN, dem Kontoinhaber, sowie einer Rechnungsadresse im entsprechenden Amazon Profil zu hinterlegen und dem Betreiber somit ein SEPA Lastschrift Mandat zu erteilen. Weiterhin musste für die Zahlung der digitalen Einkäufe die sogenannte „1-Click Bezahlung“ von Amazon eingerichtet werden. Damit ist es - wie der Name schon sagt - möglich, Bestellungen mit nur einem Klick zu bezahlen. Beispielsweise wird diese Methode auch genutzt, wenn Bestellungen über Amazon Alexa, also rein per Spracheingabe, getätigt werden. Um sicher zu gehen, dass der Bezahlvorgang erfolgreich duchgeführt werden kann, wurde die App *A Day at the Beach HD*, zunächst ohne jegliche Aufzeichnungen der Analysetools, erfolgreich in der virtuellen Android Umgebung gekauft. Im nächsten Schritt wurde dann die App *Kung Fu Panda 2 Movie Storybook* auf dieselbe Weise gekauft, dieses Mal allerdings mit Aufzeichnung der entsprechenden Analsysetools.

5.1.1.1 Netzwerkdatenstrom (DS_N)

Der Mitschnitt durch das Programm *Wireshark* ergab insgesamt zwölf einzigartige DNS-Anfragen. Die intersanntesten waren unter anderem `api.amazon.de`, `cloudfront.net`, `device-metrics-us-ud.amazon.com`, `m.media-amazon.com` sowie außerdem `pascololliphone-30`. `api.amazon.de` ist Teil der Amazon API und bietet Entwicklern, unter Anderem Zugang zu Informationen oder Beständen eines bestimmten Produkts [28]. `cloudfront.net` wird betrieben von AWS - Amazon Web Services und dient im Grunde dazu Bilder, Videos und sonstige webrelevante Inhalte möglichst schnell an jeden Endverbraucher auf der ganzen Welt zur Verfügung zu stellen. Zum Einsatz kommt dafür ein großes internationales Netz an Serverkapazitäten [29]. `device-metrics-us-ud.amazon.com` wird von Amazon dazu genutzt, bestimmte Metriken der Endgeräte der Benutzer zu erfassen. Dabei werden zum Beispiel der Gerätetyp oder der verwendete Browser dazu genutzt, um das Nutzungsverhalten zu analysieren und durch bespielsweise personenbezogene Inhalte darauf zu reagieren [30]. `m.media-amazon.com` wird von Amazon dazu genutzt, um zum Beispiel Bilder, Videos und Bewertungen der Produkte zu hosten und eine schnelle Bereitstellung derer, für die Webseiten von Amazon, zu gewährleisten [31]. `pascololliphone-30` kennzeichnet den Namen eines Apple iPhones im Netzwerk der Untersuchungsumgebung. Offensichtlich hat dieses zu dem Zeitpunkt der Untersuchung nach anderen Apple Geräten im Netzwerk gesucht und so auch die Untersuchungsumgebung mit abgefragt. Erkennbar ist, dass alle DNS-Anfragen, bis auf die des iPhones zu Amazon selbst oder von Amazon betriebenen Dienste geleitet werden. Dies ist in sofern gut, da dadurch potenzielle Risiken durch Dritte nahezu ausgelschlossen

werden. Allerdings ist es aus datenschutzrechtlicher Sicht auch bedenkenswert, dass Amazon zum Beispiel durch die Metrik des Endgeräts personalisierte Werbung schalten kann.
Die Analyse mit *mitmproxy* war leider nicht möglich gewesen. Die Anmeldung im *Amazon App Store* während des aktivierten Tools wurde ohne Fehlermeldung immer wieder verworfen. Bei deaktivierem Tool klappte die Anmeldung hingegen ohne Probleme. Bedauerlicher Weise konnte aber kein Bezahlvorgang mit aktiviertem Tool durchgeführt und analysiert werden. Sobald ein Download oder der Kauf einer App gestartet werden sollte wurde die in (Abbildung 6) zu sehende Fehlermeldung ausgegeben.

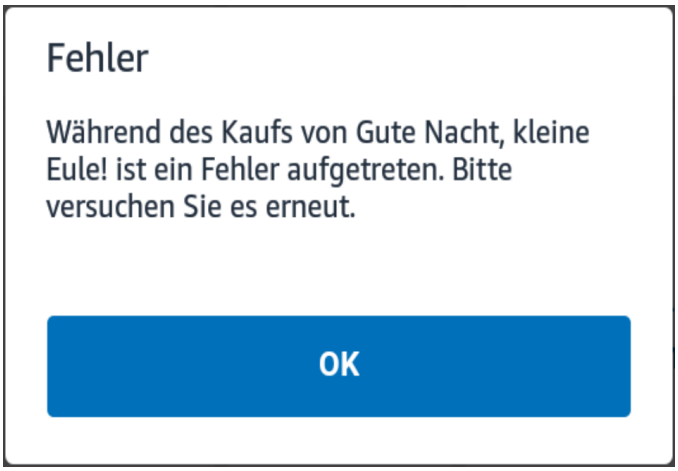


Abbildung 6: Fehlermeldung im *Amazon App Store* bei aktiviertem *mitmproxy*

Unsere Vermutung ist es, dass der *Amazon App Store* aufgrund des veränderten Zertifikats (von *mitmproxy*) jegliche Anmeldungen, Zahlungen und Downloads blockiert. Dies führt wiederum zu der Erkenntnis, dass die App gegen Angriffe dieser Art abgesichert zu sein scheint.

5.1.1.2 Hauptspeicherdatenstrom (DS_M)

Die Suche nach sensiblen Daten, wie etwa dem Passwort des *Amazon* Accounts oder der IBAN der Bankverbindung, mit dem Hex-Editor *GHex* ergab kein Ergebniss. Diese sensiblen Daten werden somit vermutlich nicht zwischengespeichert.

5.1.1.3 Massenspeicherdatenstrom (DS_T)

Die Analyse des Massenspeichers mit dem Analysetool *Autopsy* hat insgesamt 26 Ergebnisse erbracht. Zur besseren Übersicht wurden diese in verschiende Kategorien eingeordnet:
Zu der Kategorie SSL/TLS können die Adressen `appro@openssl.org` und `sre@letsencrypt.org` eingeordnet werden. *OpenSSL* [32] bietet dabei eine Open-Source Softwarebibliothek zur Implementierung von SSL/TLS-Protokollen an und *Let's Encrypt* [33] kümmert sich als Zertifizierungsstelle um sichere Kommunikation im Netz, durch die kostenfreie Bereitstellung entsprechender SSL/TLS-Protokolle.
Die Mail-Adressen `ct-logs@cloudflare.com`, `google-ct-logs@googlegroups.com` und `trustasia-ct-logs@trustasia.com` lassen sich zur Kategorie CT - Certificate Transparency zusammenfassen. Certificate Transperancy, als Sicherheitsmechanismus, soll die Prüfbarkeit und Transpazenz von SSL/TLS erhöhen [34]. Cloudflare verwaltet dabei die Logs von CT [35], ebenso *Google Groups* [36] und *TrustAsia* [37] in diesem Fall.
Zur nächsten Kategorie CDN (Content Delivery Network) gehören verschiedene Adressen von *Cloudflare*. *Cloudflare* bietet ihre Dienste vor allem in den Bereichen der Internetsicherheit und Bereitstellung von Content Delivery Network an. Die Namen vor der Domain könnten dabei entweder Mitarbeiter oder Servernamen von *Cloudflare* sein [38].
`ctops@digicert.com` und `ctops@sectigo.com` fallen in die Rubrik Certificate Operations. *DigiCert* [39] ist hierbei ebenso wie *Sectigo* [40] für die Verwaltung und den Betrieb der SSL/TLS - Protokolle verantwortlich.
Die Kategorie Frameworks umfasst die Adressen `framework@*.art`, wie z.B. `framework@boot-apache-xml.art`. Diese Kategorie beinhaltet die verschiedensten Frameworks, beispielsweise bietet das Open-Source-Framework Apache XML eine Verarbeitung von XML Dateien [41]. Auch im Massenspeicher ergab die Suche nach sensiblen Daten, wie etwa dem Passwort des Amazon-Accounts oder der IBAN der Bankverbindung, mit dem Hex-Editor *GHex* kein Ergebniss. Diese sensiblen Daten werden somit vermutlich nicht im Klartext gespeichert.

5.1.1.4 Fazit und betroffene Sicherheitsaspekte

Positiv zu erwähnen ist, dass keine Drittanbieter in den Bezahlvorgang eingebunden wurden sind. Jegliche Dienste, die in den Vorgang involviert waren, lassen sich auf Amazon zurückführen. Auch, wenn es für die Analyse sicher noch einige interessante Erkenntnisse hervorgebracht hätte, lässt sich durch die Verwehrung des Zugriffs mittels *mitmproxy* sagen, dass der *Amazon App Store* vor solchen Angriffen, von den untersuchten Apps, am besten geschützt zu sein scheint. Allerdings erhebt auch Amazon selbst Daten, die nicht unmittelbar von dem Kaufvorgang betroffen sind. Beispielsweise werden die Gerätemetriken ausgelesen, um noch besser personalisierte Werbung zu schalten. Dieses Vorgehen verletzt dabei die Sicherheitsaspekte der **Vertraulichkeit** sowie der **Privatsphäre**.

5.1.2 Anton App: In-App-Abo über Google Play mit PayPal [Glenn]

Der Bezahlvorgang bestand aus dem Kauf eines In-App Abos über den *Google Play Store* mithilfe eines PayPal Accounts, dies stellt den zweiten Versuch des Analysierens eines Bezahlvorgangs in dem Projekt PaymentTraces dar. Um diesen Bezahlvorgang durchzuführen musste erst einmal ein Google-Account angelegt werden oder sich mit einem bestehenden Account angemeldet werden. Im Rahmen der Analyse haben wir uns dazu entschieden einen bereits bestehenden Account zu benutzen und somit stehen uns keine Informationen zur Erstellung des Accounts zur Verfügung. Es ist erwähnenswert das in diesem Account trotzdem persönliche Daten gespeichert werden, welche beim Bezahlvorgang unverschlüsselt weitergegeben werden

können. Wir haben in der statischen Umgebung die drei Datenströme Hauptspeicher, Massenspeicher und Netzwerk auswerten können, dabei wurde der Netzwerkdatenstrom mit *Wireshark* abgefangen und analysiert, der Massenspeicher wurde mit *Autopsy* analysiert und der Hauptspeicher wurde gedumpt und mit *strings* analysiert. In der dynamischen Umgebung konnten wir mithilfe von *mitmproxy* zur Laufzeit die Verbindungen noch genauer analysieren, da der Datenstrom entschlüsselt wird. Im folgenden gehen wir genauer auf die Ergebnisse ein.

5.1.2.1 Netzwerkdatenstrom (DS_N)

Insgesamt wurden 1046 Pakete während des gesamten Bezahlvorgangs mit *Wireshark* aufgezeichnet, dazu muss allerdings gesagt werden, das die Authentifizierung des Bezahlvorgangs zweimal aufgrund eines falschen Passworts fehlschlug, weshalb diese Zahl inflationär zu betrachten ist. Insgesamt konnten neun einzigartige DNS-Anfragen festgestellt werden:

- `content.anton.app`
- `content-2.anton.app`
- `logger-lb-4.anton.app`
- `apis-db-logger-s-lb-1.anton.app`
- `www.googleapis.com`
- `connectivitycheck.gstatic.com`
- `play-fe.googleapis.com`
- `apis-payment.anton.app`
- `connectivity-check.ubuntu.com`

Dabei können wir `connectivity-check.ubuntu.com` und `connectivitycheck.gstatic.com` vernachlässigen da diese Systemadressen sind die durch eine Routine aufgerufen werden. `content.anton.app` und `content-2.anton.app` sind Anfragen die auftreten wenn man die App öffnet. `logger-lb-4.anton.app` und `apis-db-logger-s-lb-1.anton.app` sind die DNS Anfragen die auftreten, wenn man sich mit seinem Account Token in seinen Account einloggt. Sehr interessant sind die Anfragen an Google und keine Anfragen an PayPal. Man kann davon ausgehen dass die Accountinformationen von PayPal die übertragen wurden auf den Google Servern gespeichert ist, da keine DNS Anfrage an die PayPal Server erfolgte. Mit dem *mitmproxy* konnten wir herausfinden das der Account Token der bei der Erstellung des Accounts zugeteilt wird beim einloggen unverschlüsselt gesendet wird. Desweiteren konnten wir die E-Mail Adresse in der Aufzeichnung von *mitmproxy* finden. Weitere Informationen konnten nicht extrahiert werden.

5.1.2.2 Massenspeicher (DS_T)

Die Hauptspeicher Analyse mit *Autopsy* ergab das sich die E-Mail-Adresse die beim Bezahlvorgang genutzt wurde auf dem System befand und das 26 mal. Zusätzlich konnte man unter dem Reiter „Web Accounts“ Bearer-Authentication Tokens auslesen, es wurden vier Cookies im System gesetzt alle 4 davon stammen von `google.com`. Eine weitere Analyse mit *ExtUndelete* ergab keine weiteren Ergebnisse.

5.1.2.3 Hauptspeicher (DS_M)

Mit *strings* konnten zunächst aus dem RAM-Dump alle Zeichenketten extrahiert werden. Aus der daraus resultierenden Datei konnte man die E-Mail-Adresse des Bezahlvorgangs herauslesen, sonstige Informationen wie das Passwort waren nicht zu finden. Eine weitere Analyse mittels *GHex* lieferte keine weiteren Ergebnisse.

5.1.2.4 Fazit und betroffene Sicherheitsaspekte

Beim Bezahlvorgang zum Kauf von Anton Plus mit PayPal wurden soweit nur Daten erhoben die auch für den Bezahlvorgang benötigt wurden. Was kritisch zu sehen ist, ist die Speicherung der Daten auf den Google Servern, die Übertragung der E-Mail-Adresse im entschlüsselten Datenstrom und die Übertragung der Authentication Tokens und des Account Tokens was eine Verletzung der Sicherheitsaspekte der **Privatsphäre** und **Vertraulichkeit** darstellt.

5.1.3 App-Kauf im Google Play Store mit PaySafeCard [Sönke]

Der Bezahlvorgang des Kaufens einer App im *Google Play Store* mit einer PaySafeCard beziehungsweise mit *MyPaySafe* stellt den dritten Versuch des Analysierens eines Bezahlvorgangs in diesem Projekt dar. Um diesen Bezahlvorgang durchzuführen, musste zuerst ein Account bei Google angelegt werden oder sich mit einem bereits bestehenden Google Account angemeldet werden. Wir haben für die Umsetzung einen bereits bestehenden Account genutzt und daher keine Aufzeichnungen über diesen Anmeldevorgang aufgezeichnet, da dies auch keinen Mehrwert für die Analyse des Bezahlvorgangs hervorbringt. Jedoch ist dies dennoch zu erwähnen, da dieser Account persönliche Daten enthält, welche beim Durchführen des Bezahlvorgangs eventuell unverschlüsselt übertragen werden könnten und somit zu Analyseergebnissen beitragen könnten. Um nun eine PaySafeCard als Bezahlmethode verwenden zu können, war es nicht möglich, diese direkt auf den Google Play Account aufzuladen, sondern man benötigte hierfür einen weiteren Account bei *MyPaySafe*. Hierfür wurde auch wieder ein bereits bestehender Account genutzt. Das Aufladen der PaySafeCard auf den MyPaysafe Account und der Kauf der App *Universe in a Nutshell* sollten Teil der Analyse werden, jedoch stellte das Aufzeichnen des Aufladens der PaySafeCard einige Probleme dar. Der Anmeldevorgang mit dem *MyPaySafe* Account im *Google Play Store* war nicht möglich, was auf ein nicht akzeptiertes Zertifikat von seitens *MyPaySafe* zurückzuführen ist. Der Grund hierfür ist wahrscheinlich *mitmproxy*, da die Zertifikate als nicht vertrauenswürdig eingestuft wurden. Jedoch stellte sich auch beim Versuch ohne Proxy heraus, dass die Anmeldung nicht möglich war, da *MyPaysafe* einen Browser aufruft, der in der virtuellen Maschine nicht geladen werden konnte, wodurch auch eine Aufnahme mit *Wireshark* entfällt. Daher musste die Aufladung der PaySafeCard auf ein externes Gerät ausgelagert werden, wodurch die Analyse des Aufladevorgangs entfällt, da dort ein Mitschnitt mit *Wireshark* ein so großes Rauschen enthält, sodass die Daten unbrauchbar zur Analyse waren. Die Aufzeichnung des App-Kaufs der App *Universe in a Nutshell* im *Google Play Store* konnten jedoch mit *Wireshark* und *mitmproxy* für die Analyse des Netzwerkdatenstroms sowie Abzüge des Massen- und Hauptspeicher für die Analyse dieser durchgeführt werden.

5.1.3.1 Netzwerkdatenstrom (DS_N)

Der folgende IO-Graph (Abbildung 7) der *Wireshark*-Aufzeichnung liefert einen ersten Eindruck über den Verlauf des Bezahlvorgangs und stellt damit den zeitlichen Verlauf der Untersuchung dar.

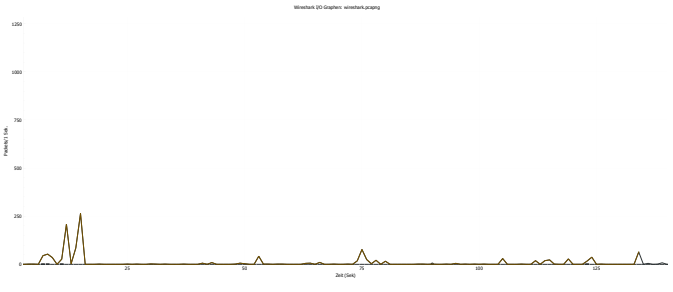
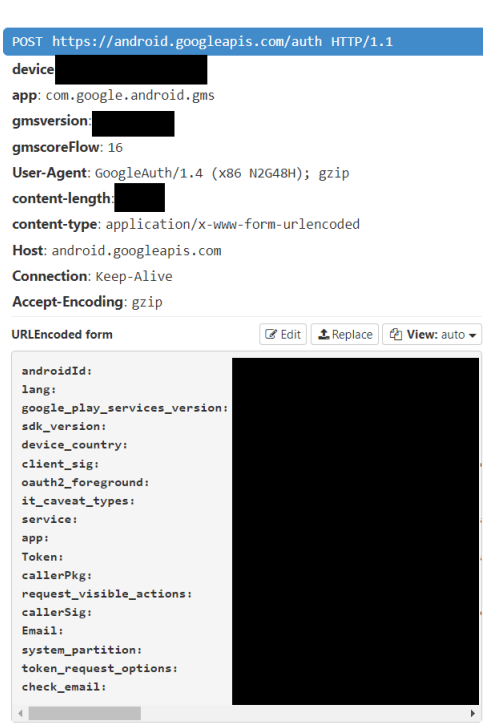


Abbildung 7: IO-Graph des *MyPaySafe*-Bezahlvorgangs

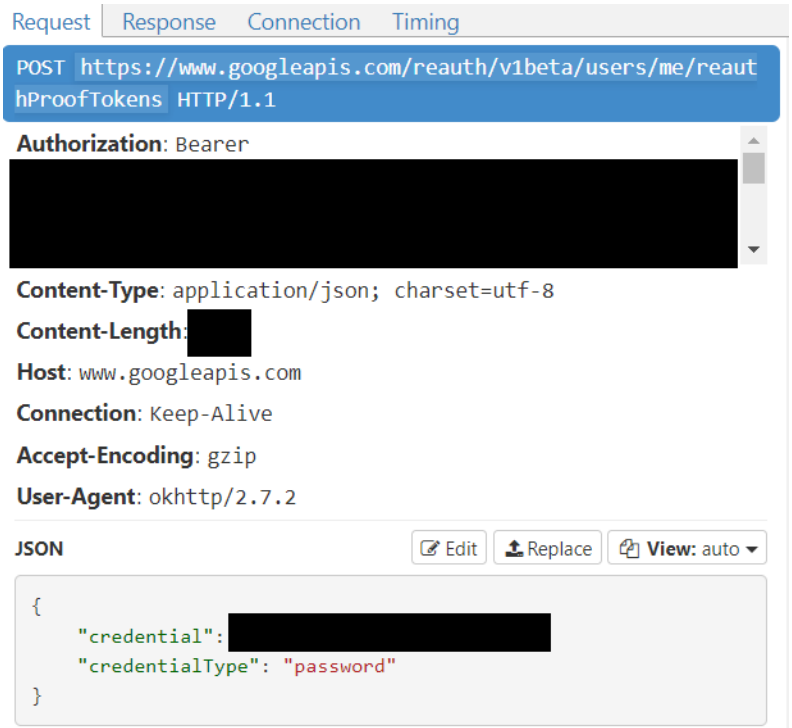
Im Folgen werden die Ausschläge auf den IO-Graphen den Phasen des Bezahlvorgangs zugeordnet. Die Aufzeichnung beginnt mit Browsern im *Google Play Store* und dem Aufrufen der *Universe in a Nutshell* Store Seite, was die Ausschläge zwischen Sekunde 0-35 erklärt, da dort die Store Seite lädt. In Sekunde 40 wird dann auf den Kaufen Button gedrückt, was den dort entstehenden leichten Anstieg erklärt. Zwischen Sekunde 50-60 wird dann der Kauf nochmals bestätigt, was den weiteren Anstieg erklärt, da dort dann das Fenster mit den möglichen Zahlungsmethoden geladen wird Darauf folgt die Auswahl der Zahlungsmethode *PaySafeCard* in Sekunde 75, welche einen Peak hervorruft. Diese wird dann in Sekunde 95 nochmals bestätigt, was sich in einem weiteren leichten Anstieg zeigt. In den Zeitraum zwischen Sekunde 100-125 wird dann mehrfach das Passwort falsch eingeben, bis es dann in Sekunde 120 richtig eingegeben wird, was die dort ansteigenden und abfallenden Datenströme erklärt. In Sekunde 135 ist der Kauf und damit der gesamte Bezahlvorgang abgeschlossen. Dieser Verlauf weist auf ein mögliches Sicherheitsproblem hin, da ein Großteil der Daten in Sekunde 75 übertragen wird. Zu diesem Zeitpunkt wurde die Zahlungsmethode auf *Paysafecard* geändert, aber noch kein Passwort eingegeben. Dies wird nämlich erst in Sekunden 120 eingeben. Ein weiterer möglicher Hinweis auf ein Sicherheitsproblem könnte die der etwas geringere Peak in Sekunde 60 sein, wo der Kauf noch einmal bestätigt wird, da dort eventuell schon Nutzerdaten des Google-Accounts und eventuell Daten der vorher standardmäßig festgelegten zahlungsmethode PayPal übertragen wurden ohne das ein Passwort abgefragt wurde. Insgesamt wurden im gesamten Bezahlvorgang 1262 Pakete mit insgesamt 755.43 KB an Daten übertragen. Dabei wurden insgesamt 624 Paket mit 203.21 KB an Daten vom Client empfangen und von diesen 638 Packte mit insgesamt 552.21 KB Daten an verschiedene Server gesendet. Auffällig dabei war, dass die meisten Daten an die folgenden zwei IP-Adressen gesendet wurden. Von der IP-Adresse `172.217.19.78` wurden insgesamt 380.9 KB gesendet und empfangen ,womit sie die IP-Adresse mit den meisten Datenverkehr darstellt. `172.217.19.78` ist die IP-Adresse eines Google LLC Servers in Kalifornien, USA [42]. Die zweitmeiste Kommunikation mit 204 Byte an gesendeten und empfangenden Daten stellt die IP-Adresse `142.250.181.214` dar, welche auch einem Google LLC Server zuzuordnen ist. Dieser steht jedoch in Hamburg, Deutschland [42]. Während des gesamten Bezahlvorgangs wurden folgende einzigartige DNS-Anfragen aufgerufen.

- `0.debian.pool.ntp.org`
- `1.debian.pool.ntp.org`
- `2.android.pool.ntp.org`
- `android.clients.google.com`
- `android.googleapis.com`
- `app-measurement.com`
- `lh3.googleusercontent.com`
- `mail.google.com`
- `people-pa.googleapis.com`
- `play-fe.googleapis.com`
- `play.googleapis.com`
- `region1.app-measurement.com`
- `rr2---sn-i5h7lner.gvt1.com`
- `rr2---sn-i5heen7r.gvt1.com`
- `www.google.com`
- `www.googleapis.com`
- `www.gstatic.com`

Von diesen konnten drei nicht eindeutig Google zugeordnet werden. Bei diesen nicht Google zuordnungsbaren DNS Anfragen handelt es sich um Anfragen zur Zeitsynchronisation. Diese drei sind `0.debian.pool.ntp.org` , `1.debian.pool.ntp.org` sowie `2.android.pool.ntp.org` , welche alle Anbieter von Network Time Protocol (NTP) sind. Das bedeutet, dass während des gesamten Bezahlvorgangs nur Google-Dienste involviert waren, was die Untersuchung mit *mitmproxy* bestätigt. Bei den restlichen 14 Anfragen handelt es sich größtenteils um unbedenkliche Anfragen wie z.b `android.googleapis.com` oder `play-fe.googleapis.com`, welche die Google APIs vom *Google Play Store* aufrufen. Jedoch lassen sich dort auch zwei Tracker finden. Einer dieser Tracker ist `app-measurement.com`, welcher Teil von Google Analytics (Teil von *Google Firebase* [43]) ist und während des Bezahlvorgangs insgesamt vier Mal aufgerufen wird. Des Weiteren wird der Tracker `region1.app-measurement.com` drei Mal aufgerufen. Dieser übermittelt die Daten, dass ein Kauf abgeschlossen wurde sowie die Stückzahl des Produktes und die Währung in der dieser gekauft wurde. Weitere Analysen mit *mitmproxy* haben gezeigt, dass die Login-Daten des Google-Kontos im entschlüsselten Netzwerkprotokoll gefunden werden konnten. Diese wurde mehrfach an `https://android.googleapis.com/auth` via `POST`-Anfrage geschickt (siehe Abbildung 8a). Desweiteren wurde auch das Passwort mit einer `POST`-Anfrage an `https://www.googleapis.com/reauth/v1beta/users/me/reauthProofTokens` geschickt (Siehe Abbildung 8b). Dieses wurde als JSON-Format im Klartext übertragen. Dadurch dass nur Google Dienste involviert waren und keine Anfrage an *MyPaySafe* zum Abfragen der Kontodaten geschickt wurde, lässt sich daraus schließen, dass diese Information auf einem Google-Server gespeichert sein muss. Daher ist es wahrscheinlich, dass der Vorgang der Transaktion Vom Google-Server aus bei *MyPaySave* autorisiert wurde.



(a) Anfrage der `android.googleapis` mit der übertragenden E-Mail-Adresse



(b) Anfrage zum Übermitteln des Passworts

Abbildung 8: Ergebnisse der Analyse der *mitmproxy*-Aufzeichnung

5.1.3.2 Hauptspeicher (DS_M)

Die Suche nach sensiblen Informationen wie Passwörtern oder Benutzernamen für das *MyPaySafe*-Konto mittels des Hex-Editors *GHex* ergab keine Ergebnisse. Mit *strings* wurden zunächst aus dem RAM-Dump alle Zeichenketten extrahiert. Danach wurden alle Zeichenketten, die kürzer als das genutzte Passwort oder die Emailadresse waren, aus dem Datensatz gelöscht. Darauf wurde der Datensatz stichprobenartig geprüft, dabei fand sich häufig die verwendete E-Mail-Adresse wieder. Das Passwort konnte jedoch durch diese stichprobenartige Suche nicht gefunden werden. Es lässt sich jedoch vermuten, dass bei genauer Untersuchung des Datensatzes dies auch aufzufinden ist.

5.1.3.3 Massenspeicher (DS_T)

Die Massenspeicher-Untersuchungen mit *ExtUndelete* ergaben keine relevanten Erkenntnisse. Die Untersuchungen mit *Autopsy* haben jedoch ergeben, dass die E-Mail-Adresse 742 mal im Dateisystem aufzufinden war. Desweiteren konnten die schon von *mitmproxy* aufgezeichneten Bearer-Auth-Tokens auch mit *Autopsy* gefunden werden. Dies stellt eine Sicherheitslücke dar, welche genutzt werden könnte, um die zwei Faktoren Authentifizierung zu überwinden.

5.1.3.4 Fazit und betroffene Sicherheitsaspekte

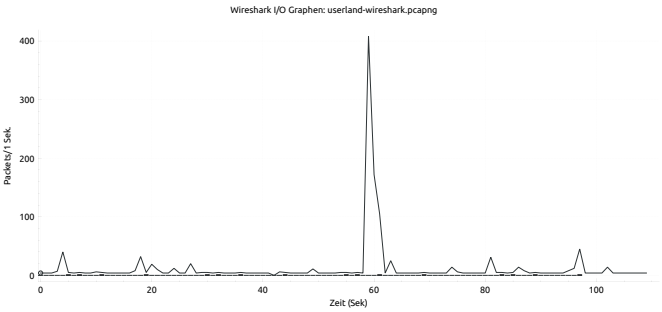
Beim Bezahlvorgang zum Kauf einer App mit *PaySafeCard* im *Google Play Store* sind keine Drittanbieter eingebunden, was positiv ist. Allerdings wurden zwei Tracking-Dienste von Google aktiviert, was zu einer Verletzung der Sicherheitsaspekte der **Privatsphäre** und **Vertraulichkeit** führt. Dabei wurden Daten erhoben und gesammelt, die für die Ausführung des Dienstes nicht notwendig sind. Zusätzlich konnten die Bearer-Auth-Tokens mitgelesen werden, was es unter Umständen ermöglicht, die Zwei-Faktor-Authentifizierung zu umgehen (siehe Abs. 6.2.4). Dadurch wird die Sicherheit des Systems beeinträchtigt. Ein weiteres Problem besteht darin, dass die E-Mail-Adresse und das Passwort im Klartext übertragen wurden. Dies stellt ein Sicherheitsrisiko dar, da diese Informationen bei einem sogenannten Man-in-the-Middle-Angriff ausgenutzt werden könnten. Eine besser verschlüsselte Übertragung (nicht nur HTTPS) könnte dieses Problem leicht beheben. Diese Schwachstelle verletzt den Sicherheitsaspekt der **Authentizität**.

5.1.4 UserLAnd: In-App-Kauf über Google Play mit PayPal [Bernhard]

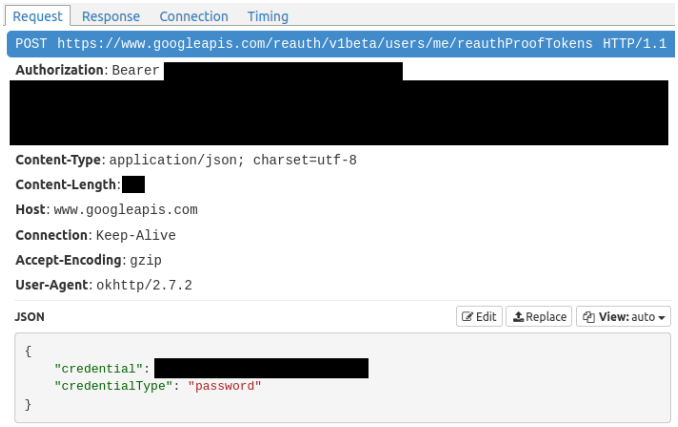
Bei diesem Bezahlvorgang wurde in der Linux-Emulator-App *UserLAnd* die Pro-Version gekauft. Die Transaktion ist über einen In-App-Kauf mit Google Play verbundenem PayPal-Konto ausgeführt worden.

5.1.4.1 Netzwerkdatenstrom (DS_N)

Die *Wireshark*-Aufzeichnung gibt m.H. des IO-Graphs (Abbildung 9a) einen Eindruck über den zeitlichen Verlauf der Untersuchung.
Folgend werden kurz die einzelnen Phasen der Aufzeichnung zu den Ausschlägen des IO-Graphs zugeordnet: Der aufgezeichnete Bezahlvorgang beginnt mit dem Öffnen der *UserLAnd*-App zwischen 15-30s. Bei ca. 60s wurde der Button in der App zum kaufen der Premium-Version geklickt,



(a) IO-Graph des Bezahlvorgangs in der *UserLAnd*-App



(b) Anfrage zum Übermitteln des Google-Passworts zur Authentifizierung der Transaktion

Abbildung 9: *UserLAnd* DS_N-Auswertung: *Wireshark* IO-Graph und mit *mitmproxy* entschlüsselte HTTP-Anfrage

wodurch der Peak ausgelöst wurde. In der Zeit zwischen 65-90s wurde das Google-Passwort zum Bestätigen der Transaktion eingegeben, das PayPal-Passwort wurde nicht abgefragt. Der letzte Ausschlag bei 95s ist die eigentliche Authentifizierung des Kaufs durch Übermitteln des Passworts.
Über den gesamten Bezahlvorgang hinweg wurden insgesamt 1386 Pakete mit 327 KB Daten über das Netzwerk übertragen. 633 Pakete mit 213 KB wurden dabei vom Client empfangen und 753 Pakete mit 113 KB an verschiedene Server gesendet. Die meiste Kommunikation fand dabei mit der IP-Adresse `142.250.185.138` statt, ein Server der Google LLC in Kalifornien, USA [42].
Während des Bezahlvorgangs wurden 9 verschiedene DNS-Hosts angefragt, wovon 6 eindeutig zu Google zuzuordnen sind. Die weiteren Hosts sind:

- `2.android.pool.ntp.org`
- `app-measurement.com`
- `region1.app-measurement.com`

Der Dienst `app-measurement.com` ist ein CNAME-Tracker von Google (Teil von *Google Firebase* [43]) und sendet Analytics. `2.android.pool.ntp.org` hingegen ist ein Anbieter des Network Time Protocol (NTP) und ist für die Uhrzeitsynchronisation verantwortlich.
Die Analyse der *mitmproxy*-Aufzeichnung bestätigt, dass während des gesamten Bezahlvorgangs ausschließlich Google-Dienste involviert waren. Folgende URLs sind dabei eindeutig Tracking-Diensten zuzuordnen:

- `app-measurement.com`
- `region1.app-measurement.com`
- `ssl.google-analytics.com`

Dabei wird an `region1.app-measurement.com` übermittelt, dass ein In-App-Kauf abgeschlossen wurde und auch, welche Artikel wie oft für welchen Preis in welcher Währung gekauft wurden. An `ssl.google-analytics.com` hingegen wurden lediglich „normale“ Analytics gesendet, wie beispielsweise welche Systemversion auf dem Gerät läuft, wie groß die Bildschirmauflösung ist, welcher Ländercode gesetzt ist sowie weitere Daten zur verwendeten Hardware. Ob die Anfrage an `ssl.google-analytics.com` direkt mit dem Kauf in Verbindung steht, kann nicht eindeutig festgestellt werden; es könnte sich auch um eine Routine-Übermittelung handeln.
Der Fakt, dass ausschließlich Google-Dienste beteiligt waren ist zunächst positiv hervorzuheben. Dies impliziert allerdings ebenfalls, dass das PayPal-Passwort auf dem Google-Server gespeichert sein muss, da keine Anfrage direkt an PayPal ging. Die Transaktion wurde deshalb wahrscheinlich vom Google-Server aus bei PayPal autorisiert, nachdem das Google-Passwort zum Bestätigen des Bezahlvorgangs übermittelt wurde.
Desweiteren konnten die vollständigen Login-Daten des Google-Kontos in den entschlüsselten Netzwerkprotokolldaten gefunden werden: Die E-Mail-Adresse wurde an `https://android.clients.google.com/auth` via `POST`-Anfrage übermittelt. Das Passwort wurde einzeln in einer `POST`-Anfrage an `https://www.googleapis.com/reauth/v1beta/users/me/reauthProofTokens` als JSON-Format im Klartext geschickt (siehe Abbildung 9b). Außerdem kann man sehen, dass die API-Authentifizierung m.H. von Bearer-Tokens umgesetzt wird.

5.1.4.2 Hauptspeicher (DS_M)

Mit *strings* konnten zunächst aus dem RAM-Dump alle Zeichenketten extrahiert werden. Anschließend wurde eine Schlüsselwortsuche mit *grep* durchgeführt. Die Mail-Adresse des verwendeten Google-Kontos konnte dabei insgesamt 27263 mal gefunden werden. Eine Suche nach der Zeichenkette „`"password","credential":`“ liefert das Passwort des genutzten Google-Kontos in Form des JSON-Strings, welcher bereits im Netzwerkdatenstrom gefunden werden konnte.
Durch die weitere Analyse mit *GHex* konnten keine weiteren Ergebnisse gefunden werden.

5.1.4.3 Massenspeicher (DS_T)

Die Massenspeicheruntersuchung mit Autopsy hat ergeben, dass die Google-E-Mail-Adresse 637 mal im Dateisystem

existiert. Im Abschnitt „Web Accounts“ sind alle auf dem Gerät gespeicherten Bearer-Auth-Tokens aufgelistet. Dieselben Tokens wurden bereits von *mitmproxy* aufgezeichnet und können u.U. dazu verwendet werden, die 2FA auszuhebeln, was eine gravierende Sicherheitslücke darstellt.
Durch eine Untersuchung mit *ExtUndelete* konnten keine weiteren relevanten Informationen gewonnen werden.

5.1.4.4 Fazit und betroffene Sicherheitsaspekte

Die über den Bezahlvorgang hinweg übertragenen Datenmengen sind insgesamt nicht außergewöhnlich hoch. Es sind keine Drittanbieter in den Vorgang eingebunden, allerdings waren während dem Vorgang zwei Tracking-Dienste von Google aktiv. Dadurch sind die Sicherheitsaspekte der **Vertraulichkeit** sowie der **Privatsphäre** betroffen, da Daten erhoben, übermittelt und mit Diensten geteilt werden, welche nicht zwingend notwendig für den Kaufvorgang sind.
Das größere Problem stellt die Übertragung des Passworts dar. Dies wird dabei im Klartext - lediglich TLS verschlüsselt - an den Google server geschickt. Insofern es ein Angreifer schafft einen MITM-Angriff durchzuführen, hätte er bereits die E-Mail und das Passwort des verwendeten Google-Kontos, was einen Angriff auf den Sicherheitsaspekt der **Authentizität** bedeuten würde.
Es könnte weitergehend getestet werden, ob mit den mitgelesenen Bearer-Auth-Tokens, welche ebenfalls im DS_T zu finden sind, eine 2-FaKtor-Authentifizierung umgangen werden könnte (siehe Abs. 6.2.4).

5.2 Einordnung in KillChain [Glenn]

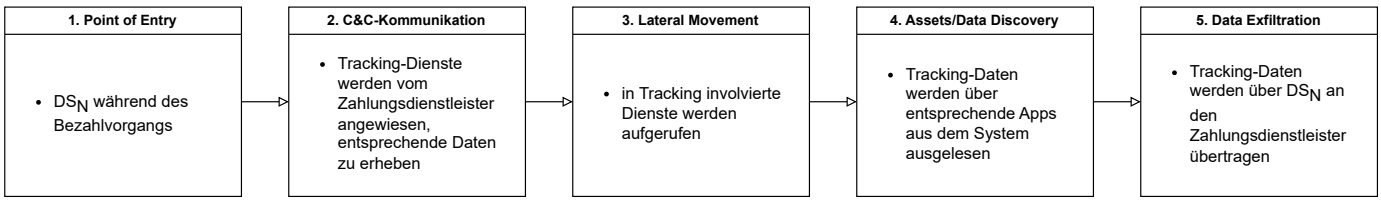
In diesem Abschnitt geht es darum die Bezahlvorgänge im Allgemeinen in das Modell der KillChain einzuordnen. Wie auch in Abbildung 10 gezeigt wird die KillChain in 5 Schritte eingeteilt, um den Dauervorfall von Privatsphärenverletzungen zu beschreiben. Die KillChain wird eigentlich nur für Angriffsvektoren genutzt, dennoch werden durch Tracking-Dienste Privatsphärenverletzungen begangen und durch unnötig erhobene Daten Informationen erhoben. Wenn diese Daten auf den Servern kompromittiert oder von einem Angreifer mitgelesen werden, können diese dafür genutzt werden, um bessere Social Engineering Angriffe durchzuführen, weshalb wir solche Vorfälle in der KillChain mit einordnen können.

1) **Point of Entry**: Der Einstiegspunkt für die Datenerhebung durch die Zahlungsdienstleister stellt der **Netzwerkdatenstrom** (DS_N) dar, sobald ein Bezahlvorgang initiiert wird.
2) **C&C-Kommunikation**: Der Zahlungsdienstleister gibt dabei vor, auf **welche** Tracking-Dienste auf dem System zugegriffen werden sollen.
3) **Lateral Movement**: Während des Bezahlvorgangs werden verschiedene Tracking-Dienste **gestartet**.
4) **Asset/Data Discovery**:
   Die gestarteten Tracking-Dienste greifen auf das System zu, um **Informationen auszulesen**.
5) **Data Exfiltration**: Die gesammelten Daten werden über DS_N an den die Tracking-Dienste des Zahlungsdienstleisters **übermittelt**.

Ein Exploit wird während dieses Vorfalls nicht ausgenutzt.

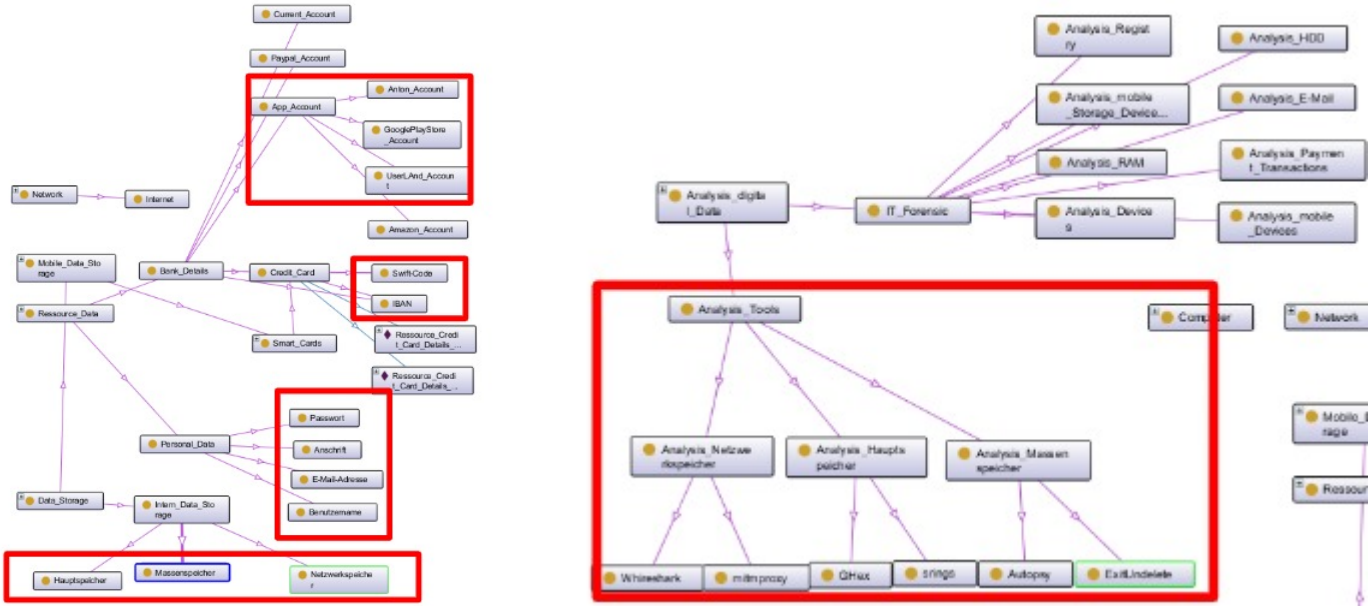
5.3 Erweiterung der Ontolgie [Tobias]

Aufgrund der technischen Probleme, die die Auswertung der Zahlungsvorgäng erschwert und vor allem verzögert haben, konnte das beschriebene Konzept nur in Teilen angewendet werden. Die zur Verfügung gestellte Ontologie wurde analysiert und an den folgenden Stellen erweitert. Zunächst wurden die in diesem Projekt verwendetet Apps mit den jeweiligen Accounts zu den „Bank-Details“, welche über „Ressource Data“ zu finden sind, hinzugefügt. Da für die Erstellung der jeweiligen Accounts verschiedene persönliche Informationen benötigt wurden, ist der Block „Personal Data“ um genau diese mit den Entitys „Benutzername“, „Passwort“, „E-Mail-Adresse“ und „Anschrift“ erweitert wurden. Der Amazon-Bezahlvorgang lief als SEPA-Lastschriftmandat ab, wozu zum einen die „IBAN“ und zum anderen der „Swift-Code“ benötigt wurden, die ab sofort der Entity „Credit Card“ zugeordnet sind. In dieser Arbeit wurden der Hauptspeicher, der Massenspeicher und der Netzwerkspeicher analysiert. In den „Ressource Data“, war der Unterpunkt „Data Storage“ mit dem erneuten Unterpunkt „Intern Data Storage“ zu finden, der im nächsten Schritt mit den neuen Punkten „Hauptspeicher“, „Massenspeicher“ und „Netzwerkspeicher“ erweitert worden ist. Diese Änderung sind in Abbildung 11a zu sehen. Alle rotmarkierten Felder sind neu, der Rest war bereits vorhanden. Der zeite große Schritt der Erweiterung der Ontologie bezog sich auf den Punkt „Analysis digital Data“. Der bereits vorhandene Unterpunkt „IT Foresic“ der die Analyse des RAM und bspw. der E-Mail beinhaltete, ist in Abbildung 11b



1. Reconnaissance: Nutzer hat bereits ein Konto bei dem entsprechenden Zahlungsdienstleister
2. Weaponization: mit dem Kauf eines Produktes stimmt der Nutzer den AGB zu und willigt der Datenerhebung ein
3. Delivery: wird initiiert durch einen Bezahlvorgang
4. Exploitation: -
5. Installation: die App(s), die die Daten erheben und versenden sind bereits installiert
6. Command&Control: der Zahlungsdienstleister bestimmt, welche Daten übermittelt werden
7. Actions on Objectives: datenschutzrelevante Daten werden übertragen

Abbildung 10: KillChain-Einordnung der Privatsphärenverletzungen als Dauervorfall bei App-Bezahlvorgängen



(a) Erweiterung der Ontologie - 1

(b) Erweiterung der Ontologie - 2

Abbildung 11: Übersicht der erweiterten Ontologie

enthalten, da diese Unterpunkte in dieser Arbeit eine Rolle gespielt haben. Für die durchgeführten Analysen wurden für jeden Speicher zwei verschiedene Tools verwenden, die in der Ontologie bislang noch keinen Platz gefunden hatten. Aus diesem Grund wurde Oberpunkt eine Subkategorie hinzugefügt mit dem Namen „Analysis Tools“, die genau die Tools auflistet, die in dieser Arbeit verwendet wurden. Die überarbeitete Ontologie wird der Abgabe mit zugefügt.

6 Zusammenfassung und Ausblick

6.1 Zusammenfassung der Untersuchungsergebnisse

Alle relevanten Erkenntnisse dieser Arbeit wurden aus dem Netzwerkdatenstrom DS_N ausgelesen. Zunächst positiv zu werten ist, dass in **keinem** der untersuchten Bezahlvorgänge **Drittanbieter** in die Kommunikation involviert sind. Bei Google- sowie Amazon-Käufen sind allerdings **eigene Analytic-Dienste** eingebunden, trotzdem sind die während eines Kaufvorgangs kommunizierten Datenmengen relativ gering. Dadurch werden bei allen Bezahlvorgängen die Sicherheitsaspekte der **Vertraulichkeit** bzw. **Privatsphäre** in gewissen Teilen **verletzt**, da für den Kauf selbst unnötige Daten ausgetauscht werden.
Bei Käufen in Verbindung mit dem *Google Play Store* wurden mehrere potentielle Sicherheitsrisiken gefunden:

- E-Mail-Adresse und Passwort sind im entschlüsselten Netzwerkdatenstrom im Klartext enthalten
- PayPal/MyPaySafe Passwörter werden wahrscheinlich auf dem Google-Server gespeichert
- Kauf ist mit *mitmproxy*-Zertifikat möglich (MITM-Angriffsvektor)
- potentielle Möglichkeit, 2FA zu umgehen (Auth-Tokens)

Sollte die E-Mail-Adresse und das Passwort des Google-Kontos bei einen MITM-Angriff gestohlen werden, ist zusätzlich der Sicherheitsaspekt der **Authentizität** verletzt.
Im Vergleich zu den Bezahlvorgängen mit *Google Play Store* sind Transaktionen im *Amazon Store* insofern sicherer, da das Zertifikat des Servers vor Durchführung des Bezahlvorgangs überprüft wird. Dadurch ist es Amazon möglich, Man-in-the-Middle-Angriffe zuverlässig abzuwehren, allerdings kann deshalb auch *mitmproxy* nicht verwendet werden, wodurch die vorgestellte Untersuchungsmethodik nicht vollständig anwendbar ist.

6.2 Ausblick für zukünftige Arbeiten

6.2.1 *Volatility*

Für die Untersuchung des Hauptspeichers DS_M würde in dieser Arbeit lediglich eine Keyword-Suche im Datenstrom durchgeführt. Eine strukturiertere Analyse des Arbeitsspeichers ist mit *Volatility* möglich, allerdings sind benötigte Symboltabellen für *Android-x86* noch nicht verfügbar, weshalb das Tool nicht verwendet werden konnte.

6.2.2 *App-Kompatibilität verbessern*

Die Suche von zu untersuchenden Apps gestaltete sich schwieriger als gedacht, da viele Apps in der virtuellen *Android-x86*-Umgebung nicht funktionsfähig sind. Eine mögliche Alternative könnte die Virtualisierungslösung *GenyMotion* sein. Da diese Software allerdings nicht quelloffen ist und Features hinter einer PayWall sind, wurde die Software für diese Arbeit nicht weiter betrachtet. Eine weitere Möglichkeit besteht darin, ein Framework zum Verbergen des Rooting-Umstands (z.B. *Xposed* [44] oder *Magisk* [45]) des genutzten Android-Systems zu installieren. Da es möglich ist, dass Apps als Sicherheitsmaßnehme einen SafetyNet-Check [46] anfragen, könnte dadurch die Kompatibilität verbessert werden.

6.2.3 Analyseumgebung verbessern

Das für diese Untersuchung genutzte Setup unterteilt sich in die dynamische und statische Umgebung, da die gestellten *Testerstick*-Images genutzt worden sind und dies so vorgeben. Diese Teilung ist nicht zwingend notwendig, weshalb beide Untersuchungsumgebungen in einem nächsten Schritt vereinigt werden könnten. Desweiteren könnten in einer weiterführenden Arbeit nach möglichen Alternativen gesucht werden, um die Detektion des mitm-Zertifikats zu unterbinden und trotzdem den Netzwerkdatenstrom zu entschlüsseln.

6.2.4 Google Play Store: Angriffsvektor Bearer-Auth-Tokens

Von den drei untersuchten Datenströmen wurden in DS_N und DS_T zur E-Mail-Adresse zuzuordnende Bearer-Auth-Tokens gefunden. Dies könnte ein Angreifer nutzen und mit Hilfe der Tokens den Authentifizierungsprozess vollständig umgehen. Sollte es tatsächlich möglich sein, die 2FA zu überwinden wäre zusätzlich der Sicherheitsaspekt der Nicht-Abstreitbarkeit verletzt, weil dadurch das Google-Konto vollständig übernommen werden könnte.

LITERATUR

[1] S. Kiltz, R. Altschaffel, T. Lucke, and J. Dittmann, "Introduction to being a privacy detective: Investigating and comparing potential privacy violations in mobile apps using forensic methods," Nov. 2020. [Online]. Available: https://www.thinkmind.org/articles/securware_2020_2_80_30032.pdf

[2] Haendlerbund, "Vor- und nachteile von online-bezahldiensten." [Online]. Available: https://www.haendlerbund.de/de/ratgeber/online-handel-business/4160-vorteile-nachteile-online-bezahldienste

[3] S. Ezennaya-Gomez, E. Blumenthal, M. Eckardt, J. Krebs, C. Kuo, J. Porbeck, E. Toplu, S. Kiltz, and J. Dittmann, "Revisiting online privacy and security mechanisms applied in the in-app payment realm from the consumers' perspective," Aug. 2022. [Online]. Available: https://dl.acm.org/doi/pdf/10.1145/3538969.3543786

[4] Verbraucherzentrale, "Bezahlen beim online-shopping: Vor- und nachteile von bezahldiensten," Jan. 2023. [Online]. Available: https://www.verbraucherzentrale.de/wissen/digitale-welt/onlinedienste/bezahlen-beim-onlineshopping-vor-und-nachteile-von-bezahldiensten-61294

[5] BSI, "Bezahlmethoden: Pro und contra." [Online]. Available: https://www.bsi.bund.de/DE/Themen/Verbraucherinnen-und-Verbraucher/Informationen-und-Empfehlungen/Online-Banking-Online-Shopping-und-mobil-bezahlen/Online-Shopping/Bezahlen-im-Internet/bezahlen-im-internet_node.html

[6] S. Kiltz, "Data-centric examination approach (dcea) for a qualitative determination of error, loss and uncertainty in digital and digitised forensics," Sep. 2020. [Online]. Available: http://dx.doi.org/10.25673/34647

[7] BSI, "Leitfaden it-forensik, p. 83," Mar. 2011. [Online]. Available: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Cyber-Sicherheit/Themen/Leitfaden_IT-Forensik.pdf?__blob=publicationFile&v=1

[8] "Testerstick mit statischer analyseumgebung," May 2023.

[9] "Testerstick mit dynamischer analyseumgebung," May 2023.

[10] "Rufus." [Online]. Available: https://rufus.ie/de/

[11] "gparted." [Online]. Available: https://wiki.ubuntuusers.de/GParted/

[12] "Oracle virtualbox." [Online]. Available: https://www.virtualbox.org/

[13] "Android x86." [Online]. Available: https://www.android-x86.org/

[14] "Wireshark - the world's most popular network protocol analyzer." [Online]. Available: https://www.wireshark.org/

[15] "mitmproxy." [Online]. Available: https://mitmproxy.org/

[16] "strings." [Online]. Available: https://github.com/GNOME/ghex

[17] "Ghex." [Online]. Available: https://github.com/GNOME/ghex

[18] "Volatility." [Online]. Available: https://github.com/volatilityfoundation/volatility

[19] "Autopsy." [Online]. Available: https://www.sleuthkit.org/autopsy/

[20] "Extundelete." [Online]. Available: https://extundelete.sourceforge.net/

[21] "Getting started with mitre attack." [Online]. Available: https://attack.mitre.org/resources/getting-started/

[22] "Was ist mitre attack?" [Online]. Available: https://www.rapid7.com/de/cybersecurity-grundlagen/mitre-attack/

[23] "Was ist die mitre attack ontologie?" [Online]. Available: https://www.redlings.com/de/ratgeber/mitre-attack

[24] "Changing uuid for virtualbox virtual machines," Aug. 2018. [Online]. Available: https://tothecore.sk/2018/08/20/changing-uuid-for-virtualbox-virtual-machines/

[25] "How to mount a disk image from the command line?" [Online]. Available: https://unix.stackexchange.com/questions/316401/how-to-mount-a-disk-image-from-the-command-line

[26] "Protégé." [Online]. Available: https://fortext.net/tools/tools/protege

[27] "Aurora store." [Online]. Available: https://f-droid.org/de/packages/com.aurora.store/

[28] Amazon, "Amazon api gateway." [Online]. Available: https://aws.amazon.com/de/api-gateway/

[29] ——, "Was ist amazon cloudfront." [Online]. Available: https://docs.aws.amazon.com/de_de/AmazonCloudFront/latest/DeveloperGuide/Introduction.html

[30] ——, "Monitoring os metrics with enhanced monitoring." [Online]. Available: https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_Monitoring.OS.html

[31] ——, "Gallery." [Online]. Available: https://m.media-amazon.com/images/G/01/AmazonStores/Help/en/Gallery.html

[32] P. S. Dipl.Ing. (FH) Stefan Luber, "Was ist openssl." [Online]. Available: https://www.security-insider.de/was-ist-openssl-a-698279/

[33] L. Encrypt, "Über let's encrypt." [Online]. Available: https://letsencrypt.org/de/about/

[34] D. Beattie, "Was ist certificate transparency? ein update zu ct." [Online]. Available: https://www.globalsign.com/de-de/blog/was-ist-certificate-transparency

[35] Cloudflare, "Certificate transparency monitoring." [Online]. Available: https://developers.cloudflare.com/ssl/edge-certificates/additional-options/certificate-transparency-monitoring/

[36] "certificate-transparency-community-site." [Online]. Available: https://github.com/google/certificate-transparency-community-site/blob/master/docs/google/known-logs.md

[37] T. C. L. P. Team, "Trustasia ct log." [Online]. Available: https://ct.trustasia.com/blog/english/

[38] Cloudflare, "Was ist ein cdn? — wie funktionieren cdns?" [Online]. Available: https://www.cloudflare.com/de-de/learning/cdn/what-is-a-cdn/

[39] DigiCert, "Tls/ssl-zertifikate." [Online]. Available: https://www.digicert.com/de/tls-ssl/tls-ssl-certificates

[40] Sectigo, "Certificate authority and pki soultions." [Online]. Available: https://www.sectigo.com

[41] A. Foundation, "Xml at the apache foundation." [Online]. Available: https://xml.apache.org

[42] "Ip address lookup." [Online]. Available: https://www.iplocation.net/ip-lookup

[43] Google, "Google firebase." [Online]. Available: https://firebase.google.com/docs/reference/android/com/google/android/gms/measurement/AppMeasurement

[44] "Xposed framework." [Online]. Available: https://forum.xda-developers.com/f/xposed-general.3094/

[45] "Magisk." [Online]. Available: https://github.com/topjohnwu/Magisk

[46] "Safetynet." [Online]. Available: https://developer.android.com/training/safetynet/attestation

ANHANG A

A.1 Aufgabenstellung

Einleitung/Motivation:

App-basierte Zahlungssysteme (digital wallets, buy now – pay later, Keditkartenzahlungen) benötigen zu ihrer IT-sicheren Bearbeitung netzbasierte Zugriffe auf Systeme von Zahlungsdienstleistern. Oftmals jedoch werden auch nicht notwendige, datenschutzrelevante Daten bei Bezahlvorgängen zusätzlich übermittelt oder Drittanbieter kontaktiert und dorthin Informationen transferiert [EBE+20]. Dies hat sowohl Auswirkungen auf die IT-Sicherheit als auch auf die Privatsphäre. Eine forensische Untersuchung von App-basierten Bezahlvorgängen soll diese Bedrohungen aufdecken.

Aufgaben: Idee, Ansatz und Ausgangspunkt:

Ziel ist es, im Rahmen eines Demonstrators den Bezahlvorgang von App basierten Bezahlvorgängen forensisch zu untersuchen und dabei die komplette Bearbeitungskette (Eingangsdaten, Konfigurationsdaten, Ausgangsdaten) nach einem existierenden Modell zu dokumentieren [Kil20]. (1) Hierzu soll für den gesamten Demonstrator ein virtualisiertes Ökosystem für App Payments auf Basis existierender Versuche [EBE+22] eingerichtet und angepaßt werden, welches diesen Anforderungen genügt. (3) Es sollen App-basierte Zahlungsvorgänge initiiert werden und mit live+post mortem IT-forensischen Methoden auf den Datenströmen Massenspeicher und Netzwerk begleitet werden (3) Der gesamte Untersuchungsverlauf ist als Kette von Untersuchungsmethoden und deren Ein- und Ausgabedaten zu beschreiben [Kil20]. (4) Eine existierende Ontologie, welche das Mitre Att@ck Schema [Mit23] integriert, soll entlang der Untersuchungsergebnisse ergänzt/erweitert werden.

Mögliche teaminterne Aufgabenaufteilung:

Alle: Einrichtung/Anpassung der VM und Untersuchungslandschaft, Planung und Durchführung von Bezahlvorgang und forensischer Untersuchung, Dokumentation des forensischen Prozesses und Abgleich/Einordnung in Ontologie sowie Kill-Chain, pro Team-Mitglied 1-2 Bezahlvorgänge

Erwartetes Ergebnis:

Tabelle mit Datenarten + Methoden über gesamten Untersuchungsverlauf, überarbeitete Ontologie, Report

Grundkenntnisse:

Einrichtung und Betrieb von Virtualisierungslösungen (Virtualbox, Genymotion), Erstellung und Anpassung von Ontologien (Protoege), Skriptprogrammierung, IT-Forensik

Abbildung A.1: Aufgabenstellung

ANHANG B

B.1 Diagramm: Aufbau der Untersuchungsinfrastruktur

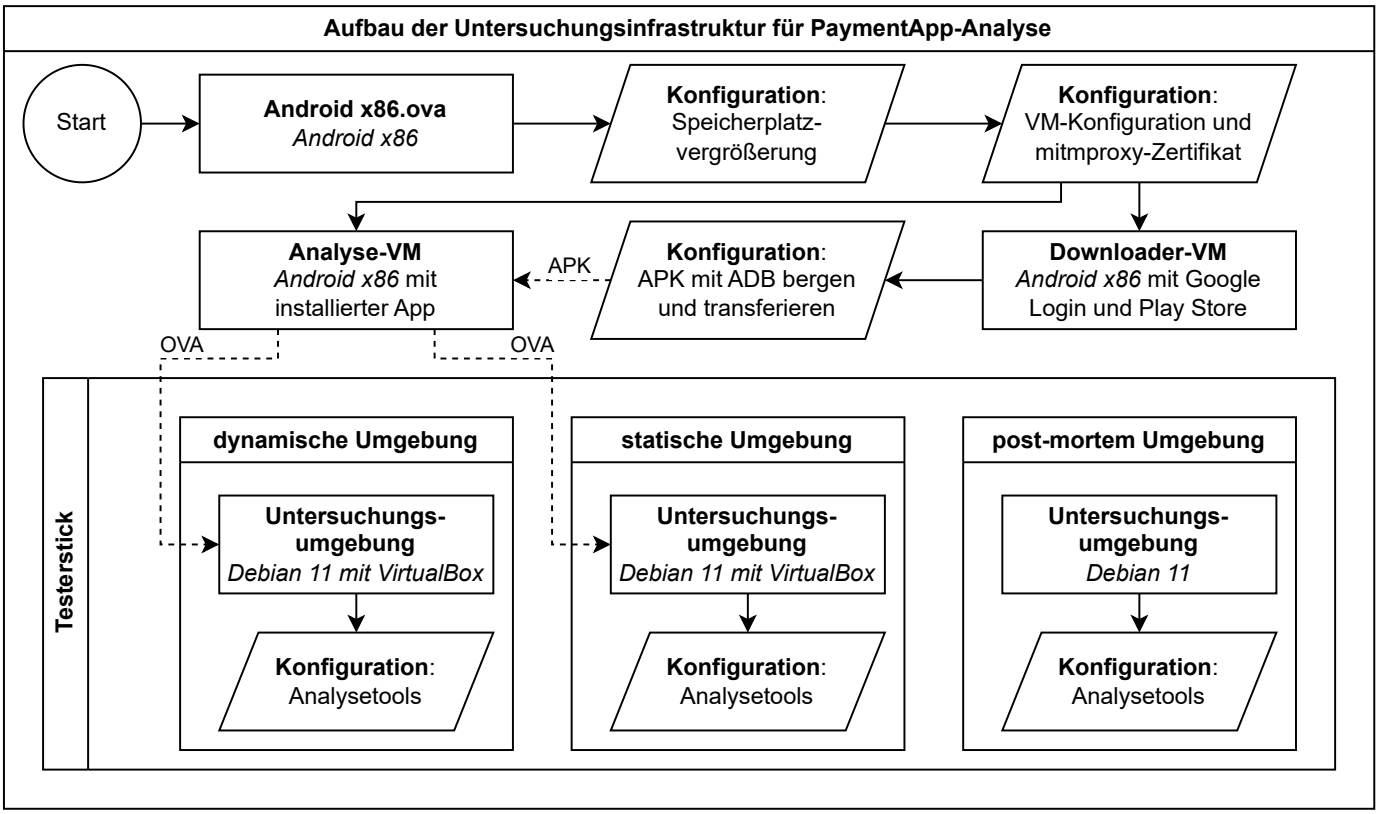


Abbildung B.1: Darstellung der Untersuchungsinfrastruktur

B.2 Diagramm: Untersuchungsmethodik

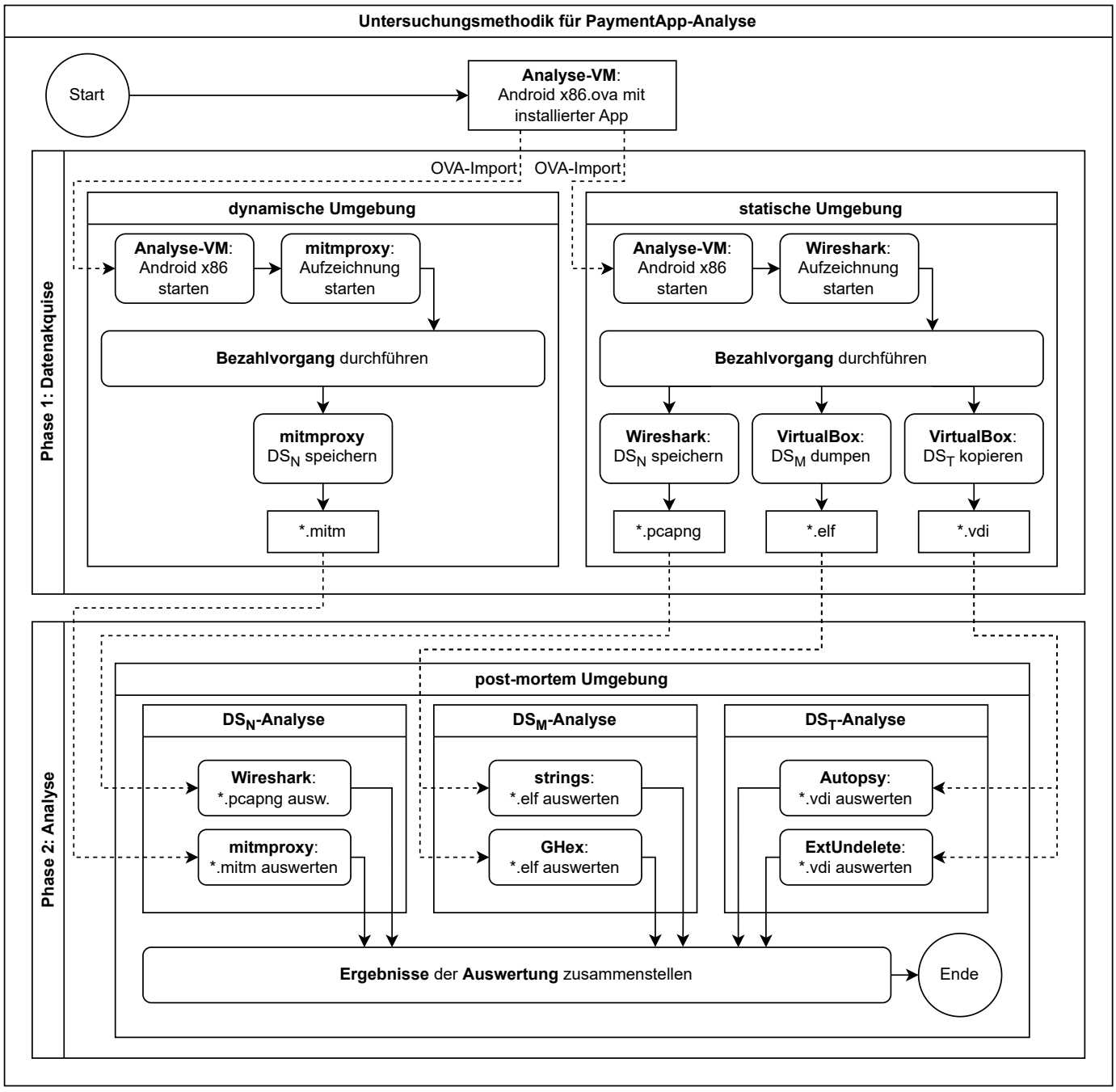


Abbildung B.2: Darstellung der Untersuchungsmethodik als Ablaufdiagramm

ANHANG C

C.1 Ergebnistabelle - App-Kauf im Amazon Store

Tabelle 2: Ergebnistabelle - App-Kauf im Amazon Store

Datenstrom	Werkzeug	Ergebnisse
DS_N	*Wireshark*	• 12 einzigartige DNS-Anfragen, darunter `api.amazon.de`, `cloudfront.net`, `device-metrics-us-ud.amazon.com` sowie `m.media-amazon.com`
DS_N	*mitmproxy*	• Analyse, aufgrund fehlender/alter Zertifikate nicht möglich
DS_M	*strings*	• Suche nach Passwort/IBAN - keine Ergebnisse
DS_T	*Autopsy*	• 26x verschiedene E-Mail Adressen zu Kategorien: SSL/TLS, Certificate Transparency, Content Delivery Network, Frameworks und Certificate Operations

C.2 Ergebnistabelle - Anton App: In-App-Abo über Google Play mit PayPal

Tabelle 3: Ergebnistabelle - Anton App: In-App-Abo über Google Play mit PayPal

Datenstrom	Werkzeug	Ergebnisse
DS_N	*Wireshark*	• Server in Deutschland lokalisiert, IP Adressen von Hetzner Online Gmbh werden genutzt
DS_N	*mitmproxy*	• nur Google Dienste involviert → PayPal Passwort muss auf Google Server gespeichert sein • Account Token ist im Klartext • E-Mail Adresse ist im Klartext
DS_M	*strings*	• E-Mail Adresse ist im Klartext zu finden
DS_T	*Autopsy*	• E-Mail Adresse auslesbar • Bearer-Auth-Tokens

C.3 Ergebnistabelle - App-Kauf im Google Play Store mit PaySafeCard

Tabelle 4: Ergebnistabelle - App-Kauf im Google Play Store mit PaySafeCard

Datenstrom	Werkzeug	Ergebnisse
DS_N	*Wireshark*	• Server in Kalifornien, USA lokalisiert • Server in Hamburg, Deutschland lokalisiert • tracker `app-measurement.com` identifiziert • tracker `egion1.app-measurement.com` identifiziert
DS_N	*mitmproxy*	• Einzige involvierte Partei ist Google → MYPaySafe muss auf Google gespeichert sein • E-Mailadresse und Passwort in Klartext (man in the middle attack) • Bearer-Auth-Tokens können ausgelesen werden (man in the middle attack)
DS_M	*strings*	• Stichprobensuche: E-Mailadresse im Klartext
DS_T	*Autopsy*	• Bearer-Auth-Tokens

C.4 Ergebnistabelle - UserLAnd: In-App-Kauf über Google Play mit PayPal

Tabelle 5: Ergebnistabelle - UserLAnd: In-App-Kauf über Google Play mit PayPal

Datenstrom	Werkzeug	Ergebnisse
DS_N	*Wireshark*	• Server in Kalifornien, USA lokalisiert • CNAME-Tracker `app-measurement.com` identifiziert
DS_N	*mitmproxy*	• weiteren Tracker `ssl.google-analytics.com` identifiziert • nur Google-Dienste involviert → PayPal-Passwort muss auf Google-Server gespeichert sein • vollständige Login-Daten des verwendeten Google-Accounts: E-Mail und Passwort im Klartext • Bearer-Auth-Tokens werden verwendet und können u.U. zur Umgehung der 2FA genutzt werden
DS_M	*strings*	• 27263x Google-E-Mail-Adresse • Suche nach „`"password","credential":`“ liefert Google-Passwort im Klartext
DS_T	*Autopsy*	• 637x Google-E-Mail-Adresse • Bearer-Auth-Tokens