Durchführung einer Relay Attacke durch Interception der Android NFC-Schnittstelle

BACHELORARBEIT

zur Erlangung des akademischen Grades

Bachelor of Science

im Rahmen des Studiums

Software & Information Engineering

eingereicht von

Michael Peyerl
Matrikelnummer 1326027

ausgeführt am
Institut für Information Systems Engineering
Forschungsgruppe Industrial Software
der Fakultät für Informatik der Technischen Universität Wien

Betreuung: Thomas Grechenig
Mitwirkung: Clemens Hlauschek

Wien, 18. Juli 2018

Kurzfassung

Über diese Vorlage: Dieses Template dient als Vorlage für die Erstellung einer wissenschaftlichen Arbeit am INSO. Individuelle Erweiterungen, Strukturanpassungen und Layout-Veränderungen können und sollen selbstverständlich nach persönlichem Ermessen und in Rücksprache mit Ihrem Betreuer vorgenommen werden.

Aufbau: In der Kurzfassung werden auf einer 3/4 bis maximal einer Seite die Kernaussagen der Diplomarbeit zusammengefasst. Dabei sollte zunächst die Motivation/der Kontext der vorliegenden Arbeit dargestellt werden, und dann kurz die Frage-/Problemstellung erläutert werden, max. 1 Absatz! Im nächsten Absatz auf die Methode/Verfahrensweise/das konkrete Fallbeispiel eingehen, mit deren Hilfe die Ergebnisse erzielt wurden. Im Zentrum der Kurzfassung stehen die zentralen eigenen Ergebnisse der Arbeit, die den Wert der vorliegenden wissenschaftlichen Arbeit ausmachen. Hier auch, wenn vorhanden, eigene Publikationen erwähnen.

Wichtig: Verständlichkeit! Die Kurzfassung soll für Leser verständlich sein, denen das Gebiet der Arbeit fremd ist. Deshalb Abkürzungen immer zuerst ausschreiben, in Klammer dazu die Erklärung: z.B: “Im Rahmen der vorliegenden Arbeit werden Non Governmental-Organisationen (NGOs) behandelt, . . . ”. In LaTeXwird diese bereits automatisch durch verwenden des Befehls `\ac` erreicht. Für Details siehe Paket `glossaries`.

Schlüsselwörter

Abstract

About this template: This template helps writing a scientific document at INSO. Users of this template are welcome to make individual modifications, extensions, and changes to layout and typography in accordance with their advisor.

Writing an abstract: The abstract summarizes the most important information within less than one page. Within the first paragraph, present the motivation and context for your work, followed by the specific aims. In the next paragraph, describe your methodology / approach, and / or the specific case you are working on. The third paragraph describes the results and the contribution of your work.

Comprehensibility: People with different backgrounds who are novel to your area of work should be able to understand the abstract. Therefore, acronyms should only be used after their full definition has given. E.g., "This work relates to non-governmental organizations (NGOs), . . . ".

Keywords

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Liste der Listings

Liste der Algorithmen

1 Einleitung

In der Einleitung soll die Zielsetzung der Arbeit beschrieben, ihre Einordnung in einen übergeordneten Kontext hergestellt und die Bedeutung des Themas erörtert werden. Zu diesem Zweck ist die Einleitung in folgende Unterkapitel unterteilt:

- Problemstellung
- Motivation
- Zielsetzung
- Aufbau der Arbeit

Organisatorisches Der Umfang einer Diplomarbeit beträgt üblicherweise 90 bis ca. 120 Seiten und bei Bachelorarbeiten 40 bis ca. 60 Seiten. Beurteilungskriterien für eine Diplomarbeit ist nicht nur die Qualität der praktischen Arbeit, sondern auch Aufbau, Inhalt und Formulierung der schriftlichen Arbeit. Insbesondere sind die Grundregeln wissenschaftlichen Arbeitens (z.B. richtiges Zitieren) zu beachten.

Allgemeines Achtung beim Setzen von Absätze.

Fußzeilen: Bei technischen Arbeiten eher unüblich.
In den Sozialwissenschaften (z.B. BWL) ist es üblich, die Referenz in eine Fußnote zu setzen.
In den technischen Wissenschaften ist es üblich, im Text ein Kürzel (Dorn et al. 1999) oder eine Zahl [1] zu verwenden, um dann im Anhang der Arbeit alle Referenzen detailliert aufzuführen, wobei jede Referenz mit dem Kürzel beginnt.

Bei einer wissenschaftlichen Arbeit wird Wert auf die Einhaltung formaler Aspekte des guten Schreibens gelegt. Es ist hilfreich, wenn man seinen eigenen Schreibstil kritisch in Bezug auf folgende Punkte überprüft:

- Einfachheit (das Keep It Sober and Significant (KISS)-Prinzip gilt nicht nur für die Softwareentwicklung, sondern auch für wissenschaftliche Arbeiten: Mach es schlicht und wesentlich/Keep It Sober and Significant)
- Gliederung/ logische Ordnung (vom Allgemeinen zum Konkreten, nachvollziehbare Kette von einer Fragestellung/ einem Problem über die Bearbeitung und Zerlegung in Detailprobleme zur Lösung und der Ableitung von Erkenntnissen)
- Kürze/Prägnanz (keine Schachtelsätz, Wiederholungen vermeiden – “Don’t repeat yourself”)
- Anregende Zusätze (erläuternde/ interessante/ spannende Praxisbeispiele etc.)
- Sprache/Stil (kein Erzählstil, möglichst keine “ich”-Form, objektiv)
- Korrekte Formeln und Abbildungen
- Korrekte Zitierweise (siehe Kapitel Literaturverzeichnis)

- Einhaltung definierter Rahmenbedingungen (z.B. diese Vorlage)
- Vermeiden von Anglizismen. Grundsätzlich sollte in einer in Deutsch verfassten Diplomarbeit immer das deutsche Wort verwendet werden, wenn es unmissverständlich und akzeptiert ist. Ein Eindeutschen um jeden Preis ist allerdings zu vermeiden, da dies zu schwer lesbaren Texten führt. Beispiele: Der Begriff "Unterbrechung des Programmflusses" ist angebrachter als "Interrupt des Programmflusses", und "verdeckte Kanäle" ist als akzeptierter deutscher Begriff für "Covert Channels" zu verwenden. Umgekehrt ist aber "Cursor" das bessere Wort als "Lichtmarke". Anglizismen können als Stilmittel genutzt werden, etwa wenn man von einer "Quick and Dirty" Implementierung spricht.

Typographie Grundsätzlich sollten nicht mehr als drei unterschiedliche Schriftarten verwendet werden. Grundsätzlich sollte die Typographie sowie die Anordnung von Bildern, Tabellen, etc. LaTeXüberlassen werden. Allerdings können Silbentrennungen durch \- gesteuert werden.

Abweichungen nach persönlichem Geschmack und in Rücksprache mit Ihrem Betreuer sind zulässig. Als Stilmittel werden üblicherweise nur die Fettschrift und Kursivschrift verwendet - Unterstreichungen, Schattenschriften etc. sind zu vermeiden. Blocksatz bitte mit automatischer Silbentrennung verwenden, Rechtschreibprüfung aktivieren.

Generell kann bei der Einleitung eine modifizierte Version des Exposés als Basis verwendet werden.

1.1 Problemstellung

Formulierung der Problemstellung, Einbettung in das Forschungsumfeld und Theorie, auf die sich die Arbeit beziehen. Tendenziell kurz, allgemeiner und sehr gut verständlich – detaillierter im Kapitel "Grundlagen".

Die formulierte Fragestellung soll das Interesse an der Lösung wecken (eine langweilige oder triviale Problemstellung lässt meistens auch eine eher weniger interessante wissenschaftliche Arbeit erwarten).

1.2 Motivation

In diesem Kapitel wird der Forschungsbedarf aufgezeigt. Nach dem Lesen dieses Kapitels sollten folgende Punkte klar dargestellt sein:

- Aktueller Stand der Wissenschaft in Bezug auf die zuvor formulierte Problemstellung und klare Darstellung, was hier unzureichend/offen ist.
- GGf. Darstellung des größeren Forschungsbereichs, in den die Diplomarbeit eingebettet ist.
- Darlegung der Bedeutung des Themas für den Stand oder die Weiterentwicklung eines Bereichs der Informatik (z.B. Datenbanksysteme, Mobile Anwendungen, Java-Programmierung, Rechenzentrumsbetrieb, ...) oder eines Fachbereichs (z.B. Bankwesen, Wertpapierhandel, Gesundheitswesen, Transportwesen, Flugsicherheit ...). Erklärung, was durch die Lösung des Problems z.B. kostengünstiger/schneller/hochwertiger/sicherer/anwendbarer/"schöner" etc. wird.

1.3 Zielsetzung

Nachdem die Problemstellung und die Motivation zu deren Lösung formuliert wurden, wird in diesem Kapitel das zu erarbeitende Resultat beschrieben.

2 Grundlagen und Hintergründe

Dieses Kapitel beschreibt Grundlagen und Konzepte, die eine wichtige Rolle als Fundament für das Verständnis der Arbeit einnehmen. In erster Linie wird die Kommunikationstechnologie Near Field Communication (NFC) und ihre Funktionsweise vorgestellt. Es soll behandelt werden, wie NFC technisch umgesetzt ist und mithilfe welcher Komponenten und Methoden die drahtlose Kommunikation erfolgt. Außerdem soll auf die Ausführung der Technologie auf mobilen Android Geräten eingegangen und Sicherheitsrisiken und -konzepte beschrieben werden. Ein äußerst relevanter Punkt im Zusammenhang mit dieser Arbeit ist die Relay Attacke. Das Prinzip hinter diesem Angriff und dessen praktische Ausführung werden ebenfalls behandelt.

Im Zuge der praktischen Implementierung der Relay Attacke wird mit dem Android Betriebssystem gearbeitet. Dabei werden neben der Entwicklung von Android Applikationen auch Eingriffe in den Android Quellcode selbst notwendig sein. Um die praktische Implementierung der Relay Attacke zu verstehen, ist daher ein grundlegendes Verständnis der Android Architektur sowie der Komponenten zur Entwicklung einer Android Applikation essentiell. Im nachfolgenden Kapitel wird daher vor der Diskussion der mit der NFC Technologie verwandten Themen eine Erläuterung der Grundlagen des Android Betriebssystems gegeben.

2.1 Android Architektur

Das Android Betriebssystem ist mit einem Marktanteil von rund 85% [18] das beliebteste Betriebssysteme für mobile Plattformen weltweit. Im Jahr 2018 sind unterschiedlichste Mobilgeräte wie Smartphones, Tablets, Smartwatches, Fernseher und Autos mit dem Betriebssystem ausgestattet. Android wird von der Firma Google entwickelt, wobei der Quellcode des Betriebssystems vollständig Open Source vorliegt und von jeder beliebigen Person heruntergeladen und modifiziert werden kann. Darüber hinaus werden von Google umfassende Tools zur Verfügung gestellt, die es erlauben, Anwendungen für die Android Plattform zu entwickeln und diese der Öffentlichkeit zur Verfügung zu stellen. In den nachfolgenden Kapiteln werden die grundlegenden Komponenten derartiger Anwendungen vorgestellt. Darüber hinaus wird ebenfalls der Aufbau des Android Betriebssystems erläutert.

2.1.1 Architektur des Android Betriebssystems

Das Android Betriebssystem ist ein Software Stack, der auf einem Linux Kernel aufgebaut ist [12]. Dieser Linux Kernel wurde modifiziert und optimiert, um den Anforderungen einer mobilen Plattform mit beschränkten Ressourcen bestmöglich gerecht zu werden. Abbildung **XXX** gibt einen Überblick über den Android Software Stack.

Abbildung 2.1: Architektur des Android Software Stacks

Von oben nach unten betrachtet, erfüllen die unterschiedlichen Schichten folgende Aufgaben:

Das Application Framework

Das Application Framework stellt Komponenten und Services zur Verfügung, die Entwickler in eigenen Applikationen verwenden können. Das Framework besteht aus Klassen, die in der Pro-

grammiersprache Java entwickelt wurden und die essentielle Bausteine für alle auf dem Android System vorhandenen Applikationen darstellen [12]. Neben unzähligen weiteren Funktionen stellen diese Bausteine Wege zur Verfügung um das User Interface zu bauen, Datenbanken zu erstellen, mit der Kamera zu interagieren, Daten über das Internet, Bluetooth, WLAN oder NFC zu übertragen, Spiele zu entwickeln oder beispielsweise Musik abzuspielen. Android System Applikationen wie die Email, Kalender oder Kontakte Anwendung werden ebenfalls mit diesen Komponenten erstellt. Auch die NFC Applikation, die für das Senden und Empfangen von Daten über die NFC Schnittstelle verantwortlich ist, ist eine gewöhnliche Android Applikation, die mithilfe der Framework Komponenten erstellt wurde.

Binder IPC

Der Binder Inter-Process Communication Mechanismus stellt eine Möglichkeit zur Verfügung, zwischen verschiedenen Prozessen zu kommunizieren. Auf diese Art und Weise können Android Framework Komponenten, Methoden aus der weiter unten liegenden Systemservice-Schicht aufrufen, um mit System Services zu kommunizieren [8]. Dieser Mechanismus kann auch benutzt werden, um zwischen verschiedenen Android Applikationen zu kommunizieren.

Der Android Binder Mechanismus wird mit sogenannten Bound Services (siehe Kapitel **XXX**: Architektur von Android Anwendungen) realisiert und beschreibt eine Client-Server-Architektur. Der Service stellt hierbei, den Service dar, während die mit ihm kommunizierende Komponente den Client realisiert. Services besitzen ein Messenger-Objekt, das wiederum mit einem Handler initialisiert wird. Dieser dient als Callback für ankommende Nachrichten. Beim Binden des Services wird ein IBinder-Objekt von diesem an den Client weitergereicht. Dies ist ein Objekt das die Handler-Instanz des Services darstellt. Der Client erstellt nun seinerseits ein Objekt der ServiceConnection-Klasse, die mithilfe des IBinders eine Verbindung zum Service aufbaut. Bei der erfolgreichen Herstellung der Verbindung wird im Client wiederum ein Messenger Objekt erstellt, das ab diesem Moment genutzt werden kann um Message-Objekte an den Service zu senden, der diese in der Handler-Callback Methode empfängt. Um Nachrichten vom Service an den Client senden zu können, enthalten Message-Objekte ein replyTo-Feld das wiederum mit einem Messenger, der mit einem Handler als Callback initialisiert wurde, gesetzt werden muss, um Nachrichten erhalten zu können [2].

Folgende Abbildungen sollen den Binder IPC Mechanismus verdeutlichen:

Abbildung 2.2: Binder IPC Datenfluss

System Services

System Services sind Komponenten, die meist in naher Verbindung mit der darunterliegenden Hardware liegen und die Framework-Komponenten mit wichtigen Systeminformationen versorgen oder Funktionalität zur Verwendung bestimmter Hardware zur Verfügung stellen [8, 42]. Aufgrund der Nähe zu bestimmter Hardware sind viele System Services in C oder C++ programmiert und verwenden C sowie C++ Bibliotheken [42]. Die Services in dieser Schicht kommunizieren über den Binder IPC Mechanismus mit der Applikationsebene.

Hardware Abstraction Layer

Diese Ebene wird hauptächlich von Herstellern von Hardware und Mobilgeräten verwendet, um Interfaces für die Hardware des Geräts zur Verfügung zu stellen und diese mit Gerätetreibern zu verknüpfen. Ein HAL definiert ein Standard Interface für diese Operationen, mit dessen Hilfe es

möglich ist, Funktionalität für höhere Schichten zur Verfügung zu stellen, ohne das Betriebssystem zu beeinflussen [10, 42].

Linux Kernel

2.1.2 Architektur von Android Anwendungen

2.2 Near Field Communication (NFC)

Near Field Communication ist eine drahtlose Datenübertragungstechnologie ähnlich wie Bluetooth oder WLAN, mit der auf einfachem und schnellem Weg kleinere Datenmengen über kurze Distanzen übertragen werden können. Im Vergleich zu anderen kontaktlosen Übertragungsmöglichkeiten muss bei NFC allerdings keine Kopplung der Geräte, die an der Datenübertragung teilnehmen wollen, stattfinden. Befindet sich ein NFC-kompatibles Gerät in der Nähe eines anderen, wird das Senden und Empfangen von Informationen automatisch gestartet, was die Benützung der Technologie sehr angenehm und einfach gestaltet.

Erstmals wurde die Entwicklung von NFC im Jahr 2002 von Philips und Sony bekanntgegeben [6], die zwei Jahre darauf gemeinsam mit Nokia das NFC Forum, einen Non-Profit Industrieverband zur Entwicklung der Technologie und deren Spezifikation, gründeten [20, 29]. Die Organization hat bis zum Jahr 2018 insgesamt 16 Spezifikationen der NFC Technologie veröffentlicht und zählt zahlreiche einflussreiche Unternehmen wie Apple, Google, Intel, Sony, Visa, Mastercard und viele weitere zu seinen Mitgliedern [29].

Zusätzlich zu den Spezifikationen des NFC Forums ist NFC durch den ISO 18092 Standard sowie ETSI TS 102 190 Standard standardisiert, der die Kommunikationsmodi für das NFC Interface und das NFC Protokoll beschreibt [22, 41].

2.2.1 Die Funktionsweise von NFC

Die Basis für die Funktionsweise von NFC bildet das Prinzip der magnetischen Induktion [45]. Dieses allgemein bekannte Prinzip besagt, dass ein sich ändernder elektrischer Strom ein Magnetfeld hervorruft und umgekehrt ein sich änderndes Magnetfeld, elektrischen Strom in einem Leiter erzeugt. Ein NFC Chip besteht daher grundlegend aus einer Spule, die mit elektrischem Strom versorgt werden kann und dadurch ein elektromagnetisches Feld hervorruft. Dieses besitzt eine festgelegte Frequenz im 13,56 MHz Bereich [45] und kann in der nahen Umgebung des Erzeugers genutzt werden, um einen Kommunikationskanal aufzubauen. NFC-fähige Geräte können grundsätzlich in zwei Modi operieren: aktiv und passiv.

Passiver Modus

Im passiven Modus nehmen an der Kommunikation sowohl ein aktiver als auch ein passiver Partner teil. Der aktive Teilnehmer versorgt den NFC Chip des Gerätes mit elektrischem Strom wodurch ein elektromagnetisches Feld erzeugt wird. Wird ein passiver NFC Chip ohne eigene Stromversorgung, auch Tag genannt, in die Nähe des aussendenden Gerätes gebracht, verursacht dies in der Spule des passiven Chips wiederum einen elektrischen Strom, der für den Betrieb genutzt werden kann. Mithilfe dieser Methode können Daten übertragen werden indem der aktive Erzeuger sein Feld direkt verändert. Diese Änderungen können als Daten wahrgenommen und gesendet werden. Der passive Kommunikationspartner modifiziert das Feld durch das Entziehen von Energie, wodurch Feldschwankungen, die vom Feldverursacher als Informationen identifiziert werden können, entstehen [26].

Aktiver Modus

Im aktiven Modus generiert und verändert jeder der Kommunikationspartner sein eigenes elektromagnetisches Feld, was vom jeweils anderen Teilnehmer als Information interpretiert werden kann [26].

Geräte, die NFC unterstützen, sind typischerweise in der Lage zwischen aktivem und passivem Modus zu wechseln und befinden sich standardmäßig im passiven Zustand [7]. Vor der Initiierung einer Verbindung über NFC wird jedem Kommunikationsteilnehmer eine aktive bzw. passive Rolle zugeteilt [26].

Neben den Modi, in welchen ein Gerät, das zur Datenübertragung mittels NFC fähig ist, sich befinden kann, unterscheidet man bei der Kommunikation über die drahtlose Technologie zwischen den drei Modi Peer-to-Peer, Read/Write und Card Emulation, die nachfolgend genauer beleuchtet werden. Um eine Datenübertragung über diese Modi zu ermöglichen, besitzt ein NFC-fähiges Gerät einen Host Controller, ein Secure Element (SE) sowie den NFC Chip. Der Chip sorgt für die Umwandlung der digitalen Signale in die analogen Feldveränderungen und umgekehrt. Das Secure Element ist eine sichere und modifikationsgeschützte Umgebung zur Ausführung von Code und beinhaltet typischerweise mehrere Applikationen, die gestartet werden können. Die Implementierung des SE kann auch vollständig softwarebasiert erfolgen und im Regelfall ist es dazu in der Lage, echte Smartcards zu emulieren. Der Host Controller ist verantwortlich für die Steuerung des NFC Chips sowie für die Kommunikation mit dem Secure Element [26]. Diese Komponente sorgt demnach dafür, dass die Signale des elektromagnetischen Feldes, die vom NFC Chip empfangen werden, nach deren Umwandlung an das SE weitergeleitet werden, um dort bestimmte Anweisungen auszuführen.

Die Kommunikation über NFC kann in einem der folgenden drei Modi durchgeführt werden:

Peer-to-Peer Modus

Befinden sich zwei Geräte im Peer-to-Peer Modus, können Daten über eine bidirektionale Verbindung in beiden Richtungen untereinander ausgetauscht werden. Diese Daten können Nachrichten, Kontakte, Bilder oder jegliche andere Art von Informationen beinhalten. Beim Senden der Nachrichten befindet sich der Sender im aktiven Modus und erzeugt somit sein eigenes elektromagnetisches Feld. Der Empfänger der Nachrichten befindet sich beim Empfangen der Informationen im passiven Modus [26, 43].

Read/Write Modus

Ein NFC Gerät, dass sich im Read/Write Modus befindet, ist in der Lage, Informationen von passiven NFC Tags, die keine eigene Energieversorgung besitzen, zu lesen, sowie Daten auf ihnen zu speichern bzw. zu modifizieren. Auf den Tags können sich unterschiedlichste Informationen wie Weblinks oder WLAN Verbindungsinformationen befinden. Je nach Art der gespeicherten Information führt das NFC Gerät diverse Aktionen, wie beispielsweise ein Video im Webbrowser zu öffnen oder Treuepunkte auf einer Kundenkarte zu verändern, aus [26, 43].

Card Emulation Modus

Der Card Emulation Modus dient dazu, eine kontaktlose Smartcard zu emulieren. Smartcards verfügen über keine eigene Energieversorgung weshalb sich das sie emulierende NFC Gerät im passiven Modus befindet. Card Emuation ist das Gegenteil des Read/Write Modus, da hier das NFC Gerät die exakt umgekehrte Rolle bei der Datenübertragung einnimmt. Mithilfe dieser Datenübertragungsmethode kann eine kontaktlose Zahlung mit dem NFC Gerät anstelle einer echten Karte

durchgeführt werden. Durch die Virtualisierung der Smartcard wird darüber hinaus die Möglichkeit geboten, vielfache unterschiedliche Karten auf demselben Gerät zu simulieren und Zahlungen damit durchzuführen [26].

2.2.2 Zahlungen mittels NFC

Um kontaktlose Zahlungen über NFC durchzuführen, werden in erster Linie eine Smartcard sowie ein Smartcard-Reader benötigt, die das NFC Protokoll unterstützen. Nachfolgend sollen diese Komponenten sowie der Ablauf einer kontaklosen Zahlung über die NFC Technologie genauer erläutert werden.

Smartcards

Unter einer Smartcard, die auch als Integrated Circuit Card (ICC) bzw. Chipkarte bezeichnet wird, versteht man in erster Linie eine Plastikkarte, die einen integrierten Schaltkreis besitzt. Dieser kann sowohl zur Speicherung von Dateien sowie zur Verarbeitung von Instruktionen und zur Ausführung von Programmen verwendet werden [25]. Beispiele für Smartcards sind handelsübliche Debit- und Kreditkarten sowie Subscriber Identity Module Karten (SIM Karten), die in Mobiltelefonen Verwendung finden. Üblicherweise besteht eine Chipkarte aus einem Read Only Memory (ROM) Speicher oder einem Flash Speicher, einem Electrically Erasable Programmable ROM (EEPROM) Speicher und einer Central Processing Unit (CPU). Auf diesen Hardware Komponenten befindet sich darüber hinaus ein Dateisystem sowie ein Betriebssystem [33]. Da dies gleichermaßen die Kernbestandteile eines Computers sind, könnte man eine Smartcard ebenfalls als solchen bezeichnen. Wie auf Computern befinden sich auf einer Smartcard unterschiedliche ausführbare Applikationen, die unabhängig voneinander ausgewählt und gestartet werden können [33].

Das Dateisystem einer Chipkarte besteht aus einem übergeordneten Master File (MF), das vergleichbar mit dem Root Ordner eines Linux Betriebssystems bzw. eines MS-DOS Systems ist und alle anderen Dateien und Verzeichnisse sowie Dedicated Files (DF) und Elementary Files (EF) enthält. Als Dedicated Files bezeichnet man hierbei wiederum Verzeichnisse während Elementary Files einzelne Dateien beinhalten. Eine einzelne Applikation auf der Chipkarte befindet sich hierbei in einem einzelnen DF. Variationen eines Programms können sich in untergeordneten DFs befinden während sich die Programmdaten selbst in den EFs finden [33]. Die nachfolgende Abbildung dient zur Veranschaulichung des Dateisystems auf einer Chipkarte.

Hier Abbildungen eines Dateisystems einfügen (Single sowie Multi Application)

Smartcards werden im ISO/IEC 7816 Standard definiert. Dies ist ein umfangreicher 15-teiliger Standard, der alle Eigenschaften von Chipkarten detailliert festlegt. In ISO/IEC 7816 werden physikalische Eigenschaften wie Abmessungen, elektrische Kontakte, elektrische Signale und Datenübertragungsprotokolle sowie -kommandos präzisiert. Darüber hinaus werden Kommunikationsprotokolle, Dateistruktur der Karten, Programmschnittstellen, Datenelemente, kryptografische Funktionen, Sicherheitsmechanismen und viele weitere Eigenschaften festgelegt [**15 Quellen zitieren?**]. Die kontaktlose Kommunikation über NFC sowie Kommunikationsprotokolle zur kontaktlosen Übertragung von Daten und zugehörige physikalische Eigenschaften wie die des elektromagnetischen Feldes werden hingegen im vierteiligen ISO/IEC 14443 Standard festgelegt [37–40]. Zusätzlich wurde bereits zu Beginn der 90er Jahre von den Kartengesellschaften Europay, Mastercard und Visa, der nach den Autoren benannte EMV-Sicherheitsstandard entwickelt. Dieser Standard wurde speziell für Kartenzahlungen eingeführt und soll dazu dienen eine grenzübergreifende einheitliche Schnittstelle für Kartenzahlungen bereitzustellen sowie die Sicherheit bei

Zahlungen zu erhöhen und Kartenmissbrauch zu verhindern [27, 28]. Der EMV Standard wird seit dem Jahr 1999 von einer eigenen Organisation, der EMVCo, verwaltet und weiterentwickelt. Am Ende des Jahres 2017 waren 63,7% aller weltweiten Transaktionen EMV-Transaktionen sowie 54,4% aller weltweit ausgestellten Karten waren mit dem EMV-Chip versehen [13].

Funktionsweise einer Zahlung mit einer kontaktlosen Smartcard

Die Kommunikation eines handelsüblichen Point-of-Sales (POS) Terminals mit einer Smartcard erfolgt in erster Linie über sogenannte Application Protocol Data Units (Apdus). Diese Datenblöcke sind in zwei unterschiedliche Arten unterteilt: Command-Apdus dienen zum Senden einer Instruktion, die ausgeführt werden soll, während Response-Apdus, die Antwort auf einen ausgeführten Befehl enthalten. Ein Command-Apdu tritt immer paarweise mit einem Response-Apdu auf [36]. Die Struktur eines Command-Apdus wird in Abbildung 2 beschrieben.

Hier Command Apdu Abbildung

Abbildung 2.3: Struktur eines Command-Apdus

Wie aus Abbildung 2 zu entnehmen ist, besteht ein Command-Apdu aus einem Header sowie einem Body, die die folgenden Elemente enthalten:

- Class: Die Art des Kommandos
- Instruction: Das Kommando selbst
- P1 und P2: Parameter für das Kommando (unterschiedlich je nach Instruktion)
- Lc: Die Länge der Daten
- Data: Die Nutzdaten
- Le: Die erwartete Länge des Response-Apdus

Nachdem vom POS-Terminal ein Command-Apdu an die Chipkarte gesendet wurde, wird von dieser erwartet, ein kompatibles Response-Apdu als Antwort zu senden . Die Struktur eines Response-Apdus wird in Abbildung 3 veranschaulicht.

Abbildung 2.4: Struktur eines Response-Apdus

Ein Response-Apdu besitzt die folgenden Elemente:

- Data: Die Antwortdaten auf das zuvor gesendete Kommando, die um mit ISO 7816 kompatibel zu sein, eine bestimmte Struktur aufweisen müssen. Diese Struktur ist entweder für das betreffende Kommando im Standard dokumentiert oder das Kommando ist TLV-enkodiert. Ein TLV ist eine Datenstruktur, die aus einem Tag(T), der den Typ der Daten angibt, einer Länge(L) der betreffenden Daten und den Daten selbst(V für Value) besteht.
- SW1 und SW2: Ein aus zwei Teilen bestehendes Statuswort, das den Status der Verarbeitung angibt (erfolgreich/Fehlercode)

Typischerweise wird bei einer Kartenzahlung über NFC zuerst die Applikation ausgewählt, die zur Zahlung verwendet werden soll. Sowohl das Terminal als auch die Karte können mehrere Zahlungsanwendungen unterstützen. Welche davon ausgewählt wird, hangt somit von der Priorität der

Anwendung ab. Um eine Zahlungsapplikation auszuwählen, wird ein SELECT APPLICATION Apdu an die Karte gesendet, das die ID der auszuwählenden Anwendung enthält. Die Karte antwortet auf dieses Kommando mit diversen Applikationsdaten wobei die ID der Anwendung in der Antwort ein weiteres Mal enthalten ist [15, 17].

Nach der Auswahl der Anwendung wird im Regelfall ein GET PROCESSION OPTIONS (GPO) Kommando an die Karte gesendet. Als Antwort auf diese Instruktion werden von der Chipkarte Informationen gesendet, die sich Application Interchange Profile (AIP) und Application File Locator (AFL) nennen. Das AIP dient in erster Linie zur Angabe von Informationen über die unterstützten Authentifikationsmethoden während der AFL angibt, wo die zahlungsanwendungsspezifischen Dateien zu finden sind [15, 17]. Die Daten des AFL sind vergleichbar mit Pfadangaben auf einem Computer.

Im nächsten Schritt können die vom AFL angegebenen Dateien gelesen und daraus Informationen wie die Kreditkartennummer sowie der Name des Inhabers gewonnen werden [17]. Wurden die Applikationsdaten erfolgreich gelesen, wird ein Authentifizierungsschritt durchgeführt, der die Daten auf der Karte verifiziert. Hierzu werden die Informationen des AIP verwendet, um festzustellen, welche Authentifizierungsmethoden die Chipkarte unterstützt. Die Methoden, die sowohl vom Terminal als auch von der Karte angewendet werden können, werden durchgeführt [30].

Nach der Authentifizierung überprüft das Terminal, ob die auszuführende Transaktion von der Smartcard erlaubt wird sowie ob die Karte gültig ist. Nach erfolgreicher Überprüfung wird falls notwendig der Schritt zur Besitzer-Authentifizierung eingeleitet, der häufig durch Eingabe einer persönlichen Identifikationsnummer (PIN) erfolgt [30].

Anschließend wird eine Risikoanalyse des Terminals ausgeführt. Diese dient zum Schutz vor Betrug und unrechtmäßiger Durchführung von Transaktionen und evaluiert, ob eine Transaktion offline ohne Authorisierung durch den Kartenaussteller oder online mit Authorisierung durchgeführt werden soll. Im Zuge der Risikoanalyse werden die zuvor abgeschlossenen Transaktionen derselben Karte betrachtet. Hierbei wird überprüft, ob die letzte durchgeführte Transaktion gemeinsam mit der im Moment durchgeführten ein gewisses Limit überschreitet [15]. Dieses Limit wird Floor Limit genannt und dient dazu, eine Grenze anzugeben, über welcher eine Authorisierung durch den Kartenaussteller beantragt werden muss. Dies dient der Verhinderung von Schäden durch das Bezahlen größerer Summen ohne Authorisierung. Transaktionen, deren Betrag sich unter dem Floor Limit befindet, können während der Risikoanalyse dennoch zufällig für eine Online-Authorisierung ausgewählt werden [15, 32]. Darüber hinaus wird kontrolliert, wann zuletzt eine Online-Authorisierung ausgeführt wurde [15].

Die Risikoanalyse liefert das Ergebnis, ob die Transaktion abgelehnt, online, oder offline durchgeführt werden soll. Basierend auf dem Resultat wird ein GENERATE APPLICATION CRYPTOGRAM (AC) Kommando an die Smartcard gesendet. Diese führt ihrerseits auf Basis der im Befehl gesendeten Informationen eine Risikoanalyse durch, auf welche Art und Weise die Transaktion fortgesetzt werden soll. Nach Abschluss dieses Schrittes wird ein Kryptogramm[1] generiert, das die Entscheidung der Karte über die Fortführung der Transaktion angibt. Nach Abstimmung mit der eigenen Entscheidung beendet das Terminal die Transaktion offline oder online [15, 30].

Anmerkung: Bei der Online-Authorisierung werden weitere Schritte durch den Kartenaussteller durchgeführt, die nicht im Rahmen dieser Erläuterung liegen. Nach Antwort des Kartenausstellers wird ein weiteres abschließendes GENERATE AC Kommando an die Karte übermittelt [30].

[1] Als Kryptogramm wird ein Hashwert von Transaktionsdaten bezeichnet, der zur Verifikation der Transaktion durch den Kartenaussteller genutzt werden kann [30].

2.3 NFC Sicherheitsrisiken und Gegenmaßnahmen

Obwohl die Übertragungsdistanz der Informationen über die NFC Technologie sehr gering ist, sorgt dieser Umstand nicht für eine sichere Datenübermittlung. In der Literatur sind bereits mehrere unterschiedliche Gefahren und Angriffsarten für NFC Kommunikation bekannt, die in diesem Kapitel erläutert werden sollen.

Yes-Card Attacke

Für EMV Smartcards existieren drei verschiedene Möglichkeiten, eine Datenauthentifizierung durchzuführen: statische, dynamische und kombinierte Datenauthentifizierung. Diese Mechanismen werden bei der Zahlung an einem POS-Terminal im Authentifizierungsschritt nach dem Lesen der Applikationsdaten durchgeführt [3, 15]. Bei der Ausführung der statischen Datenauthentifizierung (SDA) wird eine digitale Signatur, die vom Kartenaussteller verschlüsselt wird, zur Offline-Authentifizierung verwendet [5]. Die Signatur kann vom POS-Terminal zur Verifizierung der Daten verwendet werden. Diese Authentifizierungstechnik ist verwundbar gegenüber der sogenannten Yes-Card Attacke. Bei diesem Angriff werden die statischen Daten (die Signatur) der Smartcard kopiert und die kopierte Karte wird modifiziert, sodass sie jeden PIN akzeptiert. Dadurch können statisch signierte Offline-Transaktionen durchgeführt werden [3, 26].

Als Gegenmaßnahmen für die Yes-Card Attacke dienen die beiden Authentifizierungsmechanismen der dynamischen sowie der kombinierten Datenauthentifizierung [3]. Bei der DDA wird von der Karte selbst bei jedem Bezahlvorgang eine Signatur erstellt, die eindeutig ist, weil sie eine zufallsgenerierte Zahl des Terminals enthält [4]. Eine auf diese Art generiert Signatur kann nicht mehr gespeichert und in nachfolgenden Zahlungen verwendet werden, da sie bei jedem Zahlvorgang unterschiedlich ist. CDA ist eine Erweiterung der DDA und erzeugt zusätzlich eine dynamische Signatur, die mit dem im späteren Schritt erzeugten Kryptogramm der Karte zur Verifizierung an das Terminal gesendet wird [14].

Eavesdropping

Mithilfe einer Antenne ist es möglich, NFC Signale, die zwischen zwei NFC Geräten übertragen werden zu lesen bzw. zu modifizieren [23]. Geräte, die in der Lage sind, RFID Kommunikation abzuhören, können beispielsweise für Eavesdropping verwendet. Diese Art von Geräten ist darüber hinaus öffentlich zugänglich [34]. Die Durchführbarkeit von Eavesdropping hängt von unterschiedlichen Charakteristika, wie der Angriffsantenne, der Qualität des Empfängers oder der Signalstärke des aussendenden Gerätes ab [19].

Als Gegenmaßnahme zu Eavesdropping muss zwischen den kommunizierenden Geräten eine gesicherte Verbindung aufgebaut werden. Dies kann mithilfe von Verschlüsselungsmethoden durchgeführt werden [23].

Data Modification

Dieser Angriff beschreibt das Abfangen, Verändern und Weiterleiten der gefälschten Informationen von NFC Nachrichten. Data Modification ist sehr schwierig durchzuführen, weil das gefälschte Signal nach wie vor das richtige Format haben muss, um vom Empfänger akzeptiert zu werden [23]. Abgesehen davon ist diese Attacke sehr abhängig von der verwendeten Signalstärke des NFC Signals [19].

Um Data Modification zu verhindern, kann ein NFC Sender aus den nächstgelegenen Empfängergeräten das mit der höchsten Signalstärke auswählen, weil dieses mit hoher Wahrscheinlichkeit den beabsichtigten Empfänger darstellt. Darüber hinaus kann der Sender während der Datenübertragung überprüfen, ob weitere RF Signale entdeckt werden, die Daten aussenden. Dadurch kann

Data Modification entdeckt und verhindert werden [23].

Data Corruption

Im Gegensatz zur Data Modification wird bei der Data Corruption nicht versucht die Informationen abzufangen oder zu verändern. Data Corruption zielt darauf ab, eine NFC Verbindung zwischen zwei Geräten so zu stören, dass die übertragenen Daten für den Empfänger nutzlos werden bzw. die Verbindung selbst zu blocken. Dieser Angriff ist daher eine Art der Denial-of-Service (DOS) Attacke. Das Blocken oder Stören der Verbindung kann durch vom Angreifer ausgesendete Signale, die Rauschen in der ursprünglichen Verbindung erzeugen und diese damit unbrauchbar machen, erfolgen [23].

Data Corruption kann wie Data Modification gleichermaßen durch das Überprüfen auf weitere NFC Sendequellen entdeckt und verhindert werden.

Spoofing Grundsätzlich wird beim Spoofing die eigene Identität verschleiert bzw. eine falsche Identität vorgetäuscht. Übertragen auf die NFC Technologie könnte ein Angreifer einen NFC Tag, der ursprünglich einen anderen Zweck erfüllt, so programmieren, dass schädlicher Code auf dem Gerät, das versucht den Tag zu lesen, ausgeführt wird. Handelsübliche Smartphones bzw. NFC Lesegeräte sind häufig so konfiguriert, dass der auf einem NFC Tag vorhandene Code ohne zusätzliche Überprüfungen automatisch ausgeführt wird [23].

Um Spoofing zu verhindern, ist es notwendig, NFC Lesegeräte so zu konfigurieren, dass vor der Ausführung jeglichen gelesenen Codes eine Meldung für den Benutzer des Gerätes angezeigt wird [23].

Relay Attacke

Die Relay Attacke ist ein äußerst relevantes Sicherheitsrisiko vor allem im Bereich der kontaktlosen Zahlung mittels NFC. Aufgrund ihrer großen Relevanz für diese Arbeit wird die Relay Attacke detailliert in Kapitel 2.3 beschrieben.

2.4 Die Relay Attacke

Die Relay Attacke ist ein Angriff, der verwandt mit der Man-in-the-Middle Attacke ist und jegliche Sicherheitsmaßnahmen, die auf Applikationsebene implementiert werden, umgehen kann. Sichere Kommunikationskanäle sowie kryptographisch verschlüsselte Nachrichten bieten daher keinen Schutz vor dieser Art des Angriffs. Relay Attacken ermöglichen bei der Zahlung mittels NFC eine beliebig große Distanz zwischen Sender und Empfänger der Daten, wodurch die Sicherheit durch die kurze Übertragungsdistanz von NFC ebenfalls nicht mehr gegeben ist [3]. Es besteht daher bei der Durchführung einer Relay Attacke die Möglichkeit an einem POS Terminal mit einer Smartcard (real oder emuliert) zu zahlen, die sich tausende Kilometer entfernt befindet.

Dieser Angriff galt lange Zeit aufgrund physischer Limitationen des Kommunikationskanals sowie der zur Ausführung notwendigen speziellen Hardware als schwierig durchzuführen. Die Einführung NFC fähiger Mobilgeräte änderte diesen Umstand allerdings gravierend. Seit mehreren Jahren ist die Ausführung einer Relay Attacke mit jedem handelsüblichen Smartphone, das NFC unterstützt, möglich [44]. In der Literatur wurde die Durchführbarkeit dieses Angriffes vielfach bestätigt (siehe Abschnitt 5 - Related Work).

2.4.1 Man-in-the-Middle Angriffe

2.4.2 Prinzip der Relay Attacke

Vorgestellt wurde das Prinzip der Relay Attacke zum ersten Mal von John Conway im Jahr 1976 in dem Buch „On Numbers and Games"**[Buchzitat]**. Er beschreibt, wie es möglich ist, dass ein Schach Laie ohne Wissen über die Spielregeln, einen Großmeister im Spiel schlägt. Der Laie bzw. Angreifer spielt gleichzeitig gegen zwei Schach Meister, wobei er in beiden Partien unterschiedliche Farben einnimmt. Die Meister des Spiels wissen nichts voneinander und sind in dem Glauben, sie spielen nur gegen den Angreifer. Dieser leitet nun jeden Zug des einen Meisters an den anderen weiter, wodurch die beiden Experten des Spiels effektiv gegeneinander spielen **[Buchzitat]**.

Dieses Verfahren angewendet auf die Kommunikation mittels NFC wird folgendermaßen umgesetzt: Der Angreifer benötigt zwei NFC fähige Geräte, die in der Literatur auch "Ghost“und "Leech"genannt werden [24]. Der Ghost dient dazu eine falsche Chipkarte zu simulieren, während der Leech verwendet wird, um ein falsches POS Terminal dazustellen. Der Leech wird im Read/Write Modus in unmittelbarer Nähe der Smartcard bzw. des die Smartcard emulierenden Gerätes platziert. Mit dem Ghost wird nun versucht, mithilfe des Card Emulation Modus eine kontaktlose Zahlung an einem realen POS Terminal durchzuführen. Jegliche vom Terminal gesendeten Befehle, die vom Ghost empfangen werden, werden unverändert über einen sekundären Kommunikationskanal, der bereits vor der Attacke aufgebaut werden kann, an den Leech übertragen. Dieser übermittelt die Daten nun über NFC an das Opfer, welches daraufhin die passenden Antworten zu den Befehlen liefert. Diese werden umgekehrt an den Ghost und über diesen an das reale Terminal gesendet. Nachdem alle Befehle und Antworten nur weitergeleitet, aber nicht verändert werden, kann die Zahlung problemlos durchgeführt werden, als ob sie mit der echten Chipkarte ausgeführt worden wäre [24].

Abbildung 2.3 veranschaulicht das Prinzip der Relay Attacke.

Abbildung 2.5: Prinzip der Relay Attacke

Dieser Angriff eröffnet zahlreiche Möglichkeiten, Schaden anzurichten. Ein großes Sicherheitsrisiko, das diese Attacke nach sich zieht, ist, Zahlungen mit fremden Karten ohne das Wissen der Besitzer an POS Terminals durchzuführen. Darüber hinaus könnte sich ein Angreifer mithilfe dieser Technik Zugang zu für ihn nicht freigegebenen Bereichen verschaffen, indem die Relay Attacke verwendet wird, um eine NFC Sicherheitskontrolle für Zugangskarten zu umgehen [24].

2.4.3 Gegenmaßnahmen

Relay Attacken sind schwierig zu verhindern, da Sicherheitsmaßnahmen auf Applikationslevel keine Wirkung zeigen [3]. Verschlüsselte Nachrichten bzw. dynamisch generierte Daten werden durch den Ghost und den Leech ebenfalls weitergeleitet, wodurch eine Verifizierung immer dann erfolgreich ist, wenn sie es ohne die Relay Attacke auch wäre.

Gegenmaßnahmen können demnach in zwei Kategorien eingeteilt werden: Schutz der Karte (des Opfers) und Schutz des Systems selbst [3]. Die einfachste Art, um Schutz vor Relay Attacken zu gewährleisten, ist das Abschirmen der Karte gegenüber jeglicher RF Kommunikation. Dies kann beispielsweise mithilfe von RFID Hüllen oder Metallfolie erfolgen [3]. Bei die Karte emulierenden NFC Geräten, sollte daraus schlussfolgernd die NFC Funktion bei Nichtverwendung ausgeschaltet werden. Weiters können Relay Attacken durch sekundäre Authentifizierungsmechanismen wie biometrische Merkmale, PIN Codes oder Passwörter verhindert werden [3]. Diese würden allerdings viel Verantwortung auf den Benutzer übertragen und teilweise voraussetzen,

dass Detailwissen über die Transaktionen bekannt ist. Zusätzlich wird die angenehme Art und Weise, mit der Zahlungen mittels NFC abgewickelt werden können, zerstört werden [16].

Typischerweise wird für die Kommunikation zwischen Ghost und Leech zusätzliche Zeit benötigt, was bedeutet, dass die Transaktion insgesamt eine längere Zeitspanne in Anspruch nimmt. Theoretisch könnten Relay Attacken daher verhindert werden, indem für jede Karten-POS-Terminal-Kombination eine maximale Zeitspanne festgelegt wird, die die Transaktion dauern kann. Dies ist allerdings aufgrund der unzähligen unterschiedlichen Kartentypen nicht möglich. Eine allgemeine Obergrenze für die Zeitdauer einer Transaktion reicht im Normalfall nicht aus, um eine Relay Attacke zu verhindern [3].

Dass eine allgemeine Obergrenze nicht ausreicht, kann man folgendermaßen schließen: Unterschiedliche Chipkarten benötigen unterschiedliche Zeiten, um Transaktionen auszuführen. Eine allgemeine Obergrenze müsste daher die langsamste Karte in Betracht ziehen, um die maximale Zeit einer Transaktion festzulegen. Würde die Obergrenze auf Basis anderer Kriterien festgelegt, könnten valide Transaktionen mit langsamen Karten ansonsten verworfen werden. Dies bedeutet allerdings, dass Relay Attacken, deren insgesamte Transaktionsdauer geringer als die der langsamsten Karte sind, ohne Probleme durchgeführt werden können.

Ein Ansatz zur effektiven Verhinderung von Relay Attacken ist das sogenannte Distance Bounding. Bei diesem Verfahren wird davon ausgegangen, dass sowohl Karte als auch POS Terminal messen, wie lange eine initiale Nachricht (Secret) benötigt, um gesendet und empfangen zu werden. Aus dieser Zeit und der Übertragungsgeschwindigkeit kann die Entfernung der beiden Komponenenten bestimmt werden. Theoretisch könnte eine Relay Attacke dann nur mehr durchgeführt werden, wenn die eingesetzten Geräte Daten mit nahezu Lichtgeschwindigkeit übertragen könnten [3]. Die Sicherheit von Distance Bounding ist allerdings abhängig von der Übertragungsgeschwindigkeit und es hat sich gezeigt, dass NFC nicht geeignet dafür ist [16].

Wird die Smartcard mithilfe eines mobilen NFC Gerätes emuliert, können Ortsinformationen dabei behilflich sein, Relay Attacken zu unterbinden. Die Ortsinformationen können beispielsweise durch Ermittlung der Daten des nächstgelegen Funkmastens oder durch GPS Daten festgestellt werden. Durch die Integration der ortsbasierten Daten in die NFC Kommunikation können demnach Relay Attacken verhindert werden [16]. Dennoch gibt es Limitationen bei der Ermittlung des Ortes eines Gerätes und dessen Verwendung bei der Zahlung wie beispielsweise Ungenauigkeit der Messung oder die Nichtfreigabe der Ortsinformationen durch den Mobilfunkbetreiber [16].

In diesem Kapitel werden die theoretischen Grundlagen und alle in der Arbeit verwendeten und für das Verständnis relevante Begriffe erläutert. Kapitelnamen spezifizieren, anpassen an die Fragestellung der Arbeit.

Gerade im Bereich der Grundlagen wird viel Literatur zitiert – Details zum Zitieren finden Sie im Kapitel 4. Da keine Diplomarbeit so innovativ ist, dass sie nicht auf vorhandenes Wissen aufbaut und in ein entsprechendes Forschungsumfeld eingebettet ist, kommt an dieser Stelle der Literaturrecherche eine besondere Bedeutung zu. Als Daumenregel gilt, dass der aktuelle Stand der Wissenschaft in der Informatik üblicherweise durch Publikationen v.a. der letzten 2 – 4 Jahre repräsentiert wird.

2.5 Aktueller Stand der Technik

In diesem Kapitel wird ein Überblick über bereits existierende Lösungen für die Problemstellung bzw. verwandte Problemstellungen gegeben. Dabei ist eine Klassifizierung der existierenden Lösungen empfehlenswert. Eine Analyse der Lösungen, nach Kriterien sortiert, sollte insbesondere auch die Defizite der existierenden Lösungen erläutern und damit insbesondere auch eine Begründung liefern, warum diese Lösungen für die Problemstellung der Arbeit nicht herangezogen werden können.

2.5.1 Unterkapitel

Bei der Verwendung von Gliederungsebenen gibt es Folgendes zu beachten:

- Es sollten nicht mehr als 3 Gliederungstiefen nummeriert werden.
- Unterkapitel sind nur dann sinnvoll, wenn es auch mehrere Untergliederungen gibt. Ein Kapitel 2.1.1 sollte somit nur dann verwendet werden, wenn es auch 2.1.2 gibt.
- Oft ist es einfacher und besser verständlich, Aufzählungen als Text zu formulieren und somit weitere Gliederungsstufen zu vermeiden.

2.5.2 Abbildungen

Beschreibungen zu Abbildungen und Tabellen stehen unter dem Bild. Jede Abbildung muss im Fließtext referenziert werden. In LaTeXbesitzen Abbildungen typischerweise Labels, welche zum referenzieren verwendet werden. Zudem plaziert LaTeXdie Abbildungen an geeigneten Stellen, was meistens auch wünschenswert ist. Falls das nicht gewünscht wird, kann es durch Optionen beeinflusst werden.

Abbildung 2.6 verdeutlicht ...
(siehe Abbildung `\ref{<label>}`)

2.5.3 Tabellen

Jede Tabelle muss im Fließtext referenziertw werden. Für Tabellen gelten die selben Regeln, wie für Abbildungen (siehe dazu Abschnitt 2.5.2).

Eine Beispiel einer Tabelle ist in Tabelle 2.1 zu finden:

Bitte beachten Sie, dass Tabellen generell so einfach wie möglich gehalten werden sollen. Tabelle 2.1 dient unter anderem dazu Studierenden zu zeigen, wie Tabellen in LaTeX erstellt werden können und wie Farben verwendet werden.

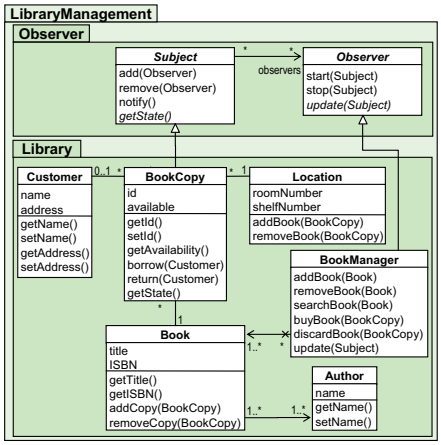

Abbildung 2.6: xxx (Quelle zitieren, wenn nicht selbst erstellt)

Linksbündig	Zentriert	Rechtsbündig
Zeile 1	xxx	xxx
Zeile 2	xxx	...
Zeile3	xxx	xxx
	xxx	xxx
xxx		

Tabelle 2.1: xxx (Quelle angeben)

3 Implementierung der Relay Attacke

Ziel dieses Kapitels ist die Beschreibung der praktischen Durchführung der NFC Relay Attacke. Es soll gezeigt werden, dass es mithilfe weniger Modifikationen am Android Betriebssystem möglich ist, die NFC Signale eines POS Terminals an ein zweites mobiles Gerät weiterzuleiten und auf diesem eine mobile Zahlung zu initiieren. Es wird dargestellt, auf welche Art und Weise ein Eingreifen in die Kommunikation der Zahlungsanwendung mit der NFC Schnittstelle möglich ist und wie die Antwortsignale der Zahlung an das weit entfernte POS Terminal zurückgesendet werden können.

Ebenfalls soll diskutiert werden, welche unterschiedlichen Komponenten zum Aufbau und zur Durchführung der Relay Attacke notwendig sind, wie diese implementiert wurden und welche Rolle sie bei der Durchführung der Attacke einnehmen. Auftretende Schwierigkeiten sowie getroffene Entscheidungen sollen ebenfalls dargelegt werden.

Am Ende dieses Kapitels soll eine funktionsfähige Proof-of-Concept Implementierung der Relay Attacke stehen, deren Umsetzung mit einfachen Methoden des Android Frameworks realisiert wird und die als Grundlage für weiterführende Forschungen dienen soll.

3.1 Aufbau der Implementierung

Zur Durchführung der Attacke wurden zwei handelsübliche, NFC-fähige Android Smartphones für die Rolle des Angreifers sowie des Opfers verwendet. Auf dem Angreifer-Gerät, einem HTC U11, war hierbei die Android Version 8.0.0 (API 26) vorhanden, auf dem Opfer-Gerät, einem Google Nexus 5X wurde eine modifizierte Version des Android Open Source Projektes mit der Version **XXX** installiert. Welche Modifikationen an diesem Betriebssystem vorgenommen wurden, wird in Kapitel **XXX** detailliert beschrieben.

Umgesetzt wurde die Relay Attacke durch die Entwicklung einer Proof-of-Concept Hostcard Emulation App, die eine handelsübliche mobile Payment Applikation wie beispielsweise Android Pay simulieren soll. Diese Anwendung wurde auf dem Opfer-Gerät installiert und als Ziel der Attacke sollte die HCE App zur Durchführung einer Zahlung und Weiterleitung der Daten an den Angreifer angeregt werden, ohne dass sich das Opfer in der Nähe eines POS Terminals befindet. Die Entwicklung der HCE App wird in Kapitel **XXX** dargelegt.

Die Kommunikation der Geräte wurde durch eine weitere mobile Anwendung, welche unabhängig von der HCE Applikation operiert, realisiert. Diese existiert sowohl auf dem Angreifer- als auch auf dem Opfer-Gerät und sorgt für eine Verbindung zwischen den Kommunikationspartnern über ein Wireless Local Area Network (WLAN).

Diese Anwendung kommuniziert darüber hinaus auch mit dem Android Betriebssystem und sorgt für die Ausführung der HCE App, sobald ein Signal vom Angreifer-Gerät empfangen wird. Darüber hinaus wird auch die Antwort der HCE App, die eigentlich an die NFC Schnittstelle gesendet wird, ebenfalls von der Kommunikationsanwendung abgefangen und an den Angreifer weitergeleitet. Mit welchen Mechanismen die unterschiedlichen Kommunikationswege umgesetzt wurden, wird in Kapitel **XXX** genauer ausgeführt.

Zur erfolgreichen Durchführung der Attacke sind Modifikationen am Android Betriebssystem notwendig. Diese sorgen dafür, dass die entwickelten Anwendungen die passenden Signale erhalten,

Abbildung 3.1: Übersicht der Relay Attacke

um korrekt operieren zu können. Bei der Durchführung der Attacke wird angenommen, dass eine modifizierte Version des Android Source Codes auf dem Opfergerät installiert werden kann. Die genaue Durchführung der Änderungen am Android Betriebssystem wird in Kapitel **XXX** beschrieben.

In folgender Abbildung ist eine Übersicht der Relay Attacke und der zur Durchführung entwickelten Komponenten gegeben.

In Abbildung **XXX** ist in erster Linie ein POS Terminal sowie ein Angreifer- und ein Opfer-Gerät dargestellt. Das POS Terminal wird durch eine Terminal Simulator Software auf dem Computer dargestellt. Über eine Universal Serial Bus (USB) Verbindung wird von einem NFC Reader die NFC Funktionalität zur korrekten Funktionsweise der Terminal Software bereitgestellt.

Bei Kontakt mit dem Angreifer wird vom POS Terminal eine kontaktlose Zahlung über NFC eingeleitet. Die Commmand Apdus werden hierbei direkt in der Kommunikationsanwendung auf dem Angreifer-Gerät empfangen. Diese nutzt nach dem Empfang eines Kommandos die WLAN Verbindung, um das Apdu an das Opfer weiterzuleiten. Die Kommunikationsanwendung, die auf dem Opfer-Gerät standardmäßig im Hintergrund aktiv ist, empfängt die Signale, woraufhin sie eine Kommunikation über das Betriebssystem mit der auf dem Opfer vorhandenen HCE Applikation startet. Die Rolle dieser HCE Applikation kann im Prinzip von jeder handelsüblichen Mobile Payment Anwendung übernommen werden. Diese Anwendung ist das eigentliche Ziel des Angriffes. Nachdem sie vom Betriebssystem gestartet wurde, wird eine mobile Zahlung in die Wege geleitet. Bei der Durchführung der Zahlung von der Zahlungsapplikation Response Apdus an die NFC Schnittstelle des Gerätes gesendet, nachdem die HCE Anwendung davon ausgeht, durch NFC aufgerufen worden zu sein. Das modifizierte Betriebssystem leitet nun diese Antworten an die Kommunikationsanwendung zurück, welche diese über WLAN wiederum dem Angreifer mitteilt. Vom Angreifer-Gerät werden die Apdus schlussendlich an das POS Terminal zurückgeleitet und die Relay Attacke wurde erfolgreich durchgeführt.

In den nachfolgenden Kapiteln wird ausgeführt, wie die einzelnen Komponenten im Detail umgesetzt wurden, um die beschriebene Funktionalität bereitzustellen.

3.2 Die Hostcard Emulation Anwendung

Das Android Framework stellt zur Implementierung einer Hostcard Emulation Anwendung die Klasse HostApduService zur Verfügung. Diese Android-Service Klasse sorgt für das Annehmen von Command Apdus, nachdem diese von der NFC Schnittstelle empfangen wurden. Um diese empfangenen Apdus verarbeiten zu können, muss die Klasse erweitert und die processCommandApdu Methode überschrieben werden. Welche Applikation zur Verarbeitung eines eintreffenden Command Apdus gestartet werden soll, wird vom Betriebssystem festgestellt. Dieses besitzt einen Routing Mechanismus, der auf den sogenannten Application IDs (AIDs) aufgebaut ist [11]. Das sind Identifier, die Zahlungsanwendungen eindeutig identifizieren und sie müssen bei der Installation einer Mobile Payment App von dieser beim Betriebssystem durch eine XML-Datei in der Anwendung registriert werden [11].

Die in dieser Arbeit dargestellte Proof-of-Concept Umsetzung einer HCE Applikation verwendet die AID einer Mastercard MAESTRO Debitkarte, die durch die Hexadezimal Darstellung A0000000043060 identifiziert wird. Gemeinsam mit den Standard Identifiern für Zahlungssysteme, die noch vor der eigentlichen Anwendung ausgewählt werden, wird die MAESTRO-AID in der aids.xml Datei festgelegt.

Abbildung 3.2: Festlegen der AIDs in der aids.xml Datei

Diese Datei wird bei der Registrierung der HostApduService Klasse in der AndroidManifest.xml Datei referenziert, um dem Betriebssystem mitzuteilen, welche AIDs von der HCE Anwendung verarbeitet werden können.

Wird eine Smartcard bzw. ein NFC fähiges Mobilgerät, auf dem eine Zahlungsanwendung vorhanden ist, in die Nähe eines POS Terminals gebracht, so wird von diesem das aus den Grundlagen bekannte SELECT PPSE Kommando gesendet. Nach der Antwort, welche AIDs verarbeitet werden können und dem SELECT AID Kommando wird die implementierte HCE Anwendung über die erweiterte HostApduService Klasse aufgerufen. Nach Auswahl der Applikation zur Verarbeitung der Zahlung werden, solange die NFC Verbindung nicht unterbrochen wird, alle nachfolgenden Apdus ebenfalls an diese Klasse weitergeleitet [11].

Um eine echte Zahlungsanwendung zu simulieren wurden in der implementierten App die ersten Response Apdus einer echten MAESTRO Debitkarte gespeichert. Als Reaktion auf ein ankommendes Command Apdu wird deshalb das passende Response Apdu an das Betriebssystem zurückgeliefert.

3.3 Die Kommunikation zwischen Angreifer und Opfer

Um die vom Angreifer-Gerät erhaltenen Apdus an das Opfer weiterleiten zu können, muss zwischen den beiden Geräten eine sekundäre Verbindung hergestellt werden. Moderne Smartphones besitzen für den Aufbau einer derartigen Kommunikation mehrere unterschiedliche Möglichkeiten:

- **Bluetooth**: Eine drahtlose Kommunikationsmöglichgeit zur Übertragung von Daten zwischen zwei Geräten. Seit Ende des Jahres 2016 ist der BLuetooth Standard in der Version 5 verfügbar [21]. Seit dieser Version ist mit Bluetooth eine Übertragungsdistanz von 40 Metern mit einer maximalen Geschwindigkeit von 2 Mbit/Sekunde möglich [31]. Um eine Verbindung über Bluetooth mit zwei Android Geräte aufzubauen, müssen diese bei ihrem ersten Verbindungsaufbau gekoppelt werden, was durch den Benutzer des Gerätes über ein Dialogfeld bestätigt werden muss [9].

- **Mobiles Internet**: Jedes Smartphone besitzt die Möglichkeit, eine mobile Internetverbindung herzustellen. Die Datenübertragungsgeschwindigkeit kann bei modernen Geräten ebenfalls mehrere hundert Mbit/Sekunde betragen [46].

- **WLAN**: Über ein Wireless Local Area Network kann eine lokale Drahtlosverbindung zwischen Geräten in näherer Umgebung hergestellt werden. Eine WLAN Verbindung wird typischerweise von einem Zugriffspunkt, wie einem WLAN Router realisiert, der die Verbindung zwischen den Geräten sowie meistens eine Verbindung zum Internet bereitstellt. Auf modernen Smartphones kann ein derartiger Zugriffspunkt auch über einen Hotspot erstellt werden. Die Übertragungsgeschwindigkeit sowie die -distanz hängen bei WLAN sehr stark von den verwendeten Geräten sowie Zugriffspunkten ab. Mehrere hunderte Mbit/Sekunde über Distanzen von 20 bis 500 Metern sind allerdings durchaus typische Größen. Befinden sich zwei Android Geräte im selben lokalen WLAN-Netzwerk ist für einen Verbindungsaufbau keine Benutzerinteraktion erforderlich.

- **Wi-Fi Direct (WifiP2P)**: Mobile Geräte, die Wi-Fi Direct unterstützen, sind in der Lage, eine Wireless Fidelity (Wi-Fi) Verbindung untereinander aufzubauen ohne einen zentralen

Zugangspunkt wie einen Hotspot oder Router zu benötigen. Mit Wi-Fi Direct sind Übertragungsdistanzen von 200 Metern mit Geschwindigkeiten von bis zu 250 Mbit/Sekunde möglich [1]. Beim ersten Aufbau einer Verbindung von zwei Android Geräten über Wi-Fi Direct muss diese wie bei Bluetooth vom Benutzer des Gerätes bestätigt werden.

Um eine Kommunikationsverbindung zwischen Angreifer und Opfer bereitzustellen, wurde in der implementierten Kommunikations App zunächst versucht, eine Verbindung zwischen den Geräten über das mobile Internet herzustellen. Um dies zu ermöglichen wurde auf dem Opfer-Gerät eine Server-Anwendung implementiert, die beim Starten der Kommunikationsanwendung einen Kommunikationsendpunkt (Socket) einrichtet. Über das TCP/IP Protokoll wird durch diesen auf eingehende Verbindungen gewartet. Konnte eine Verbindungsanfrage erfolgreich angenommen werden, wird zwischen den beiden Geräten eine TCP/IP Verbindung eingerichtet. Über diese können Nachrichten ausgetauscht werden. Auf dem Angreifer-Geräte wurde in Analogie zur Serverkomponente ein TCP/IP Client-Kommunikationsendpunkt erstellt, der versucht, eine Verbindung zum Server herzustellen. Dieser Versuch des Verbindungsaufbaus verlief allerdings erfolglos aufgrund der Tatsache, dass Smartphones eine private IP-Adresse im mobilen Internet zugewiesen bekommen, die von außen nicht erreichbar ist. Nachdem auch der Router, der die Internetverbindung für das Smartphone bereitstellt, nicht zugänglich ist, kann diese Möglichkeit nicht verwendet werden, um eine direkte drahtlose Verbindung zwischen zwei mobilen Geräten herzustellen [35].

Würde allerdings die Serveranwendung auf einem über das Internet zugänglichen Gerät gestartet werden und sowohl Angreifer als auch Opfer mit diesem Gerät verbunden werden, könnte das mobile Internet über diese dritte Instanz zum Austausch von Informationen genutzt werden.

Nachdem der Versuch, eine Verbindung über TCP/IP und das mobile Internet fehlschlug, wurde als nächste Verbindungsmöglichkeit Wi-Fi Direct herangezogen. Um eine derartige Verbindung aufzubauen, war es notwendig, einen BroadcastReceiver zu implementieren, der auf bestimmte Wi-Fi Direct Aktionen reagiert. Auf dem Angreifer-Gerät wurde danach eine sogenannte Peer Discovery durchgeführt, die als Ziel hatte, alle in der Nähe befindlichen Wi-Fi Direct Geräte zu finden und aufzulisten. Wurde die Liste von Geräten erfolgreich geladen, konnte eine Verbindung zum Opfer aufgebaut werden. Hierfür mussten die Verbindungsinformationen des Opfers bekannt sein, da ansonsten keine Möglichkeit bestand, die Geräte anhand der Verbindungsinformationen zu identifizieren. Wurde die Verbindung erfolgreich hergestellt, wurde die Applikation über den BroadcastReceiver und eine dafür vorgesehen Aktion benachrichtigt. Daraufhin konnten die Komponenten, die für die Verbindung über TCP/IP vorgesehen waren, mit den Verbindungsinformationen des Opfer-Gerätes wiederverwendet werden. Zu bemerken ist, dass beim ersten Erstellen der Verbindung, diese in einem Dialog auf dem Opfer-Gerät akzeptiert werden musste. Nach erfolgreicher Durchführung dieser Schritte war es möglich, Nachrichten zwischen dem Angreifer und dem Opfer zu versenden.

Diese Verbindung war allerdings relativ unzuverlässig. Ein Grund dafür war, dass das Wi-Fi Direct mehrmals in einen inaktiven Zustand wechselte. Ein Versuch, das Wi-Fi Direct aktiv zu halten, indem eine ständige Peer-Discovery in einem anderen Thread ausgeführt wurde, erzielte ebenfalls nicht die gewünschte Verbindungsqualität.

Nachdem auch nach mehreren Versuchen, keine durchgehende Wi-Fi Direct Verbindung über längeren Zeitraum hergestellt werden konnte, wurde schlussendlich die WLAN Technologie als Kommunikationskanal für die Relay Attacke ausgewählt. Diese verspricht eine zuverlässige, schnelle Verbindung auch über größere Distanzen im selben lokalen Netzwerk. Darüber hinaus, muss beim Verbindungsaufbau keine Benutzer-Interaktion erfolgen.

Für eine Kommunikation zwischen den Geräten über ein WLAN-Netzwerk konnten die Komponenten, wie sie für eine Implementierung über das mobile Internet anfangs entwickelt wurden,

Abbildung 3.3: Android NFC Host Card Emulation Architektur

ohne Änderungen übernommen werden, nachdem nur die IP-Adresse des Opfers geändert werden musste.

3.4 Interception der Android NFC Schnittstelle

Um zu verstehen, wie die Kommunikation mit der Android NFC Schnittstelle auf das Angreifer-Gerät umgelenkt wird, muss zunächst ein grundlegendes Verständnis der Host Card Emulation Funktionalität auf Android Smartphones geschaffen werden. Die Architektur ist in nachfolgender Abbildung veranschaulicht und wird im darunterstehenden Unterkapitel genauer beschrieben.

3.4.1 Die Android NFC Host Card Emulation Architektur

Im Android Betriebssystem finden sich 2 Pakete, die für die Funktionalität von Host Card Emulation von besonderer Bedeutung sind:

- **frameworks/base/core/java/android/nfc**: Beinhaltet die Komponenten, die vom Android Framework für die mobile App-Entwicklung bereitgestellt werden. Die Klassen in diesem Paket können von jedem Entwickler beim Entwickeln einer App zur Nutzung der NFC Funktionalität verwendet werden. Dieses Paket wird in der nachfolgenden Beschreibung als "Framework"bezeichnet.
- **packages/apps/Nfc**: Dieses Paket enthält die NFC Funktionalität auf Systemservice Ebene. Grundsätzlich stellen die Java Klassen eine Android Systemapplikation zur Verfügung, die die NFC Funktionalität bereitstellt. Die Klassen in diesem Paket werden von den Android Framework Komponenten über den Binder IPC Mechanismus angesprochen. Dieses Paket wird in der nachfolgenden Beschreibung als SSystemapplikation"bezeichnet.

Verfolgt man einen Top-Down Ansatz bei der Analyse der Host Card Emulation Funktionalität, befindet sich der Einstiegspunkt in der HostApduService Klasse, die sich im Frameworks-Paket befindet und somit in jeder beliebigen Android Applikation verwendet werden kann. Um HCE Funktionalität zur Verfügung stellen zu können, muss dieses Android Service von der App erweitert und im AndroidManifest.xml gemeinsam mit den unterstützten Application IDs registriert werden. Danach können Apdus über NFC von der Applikation in der processCommandApdu Methode empfangen werden.

Die Frage, die sich zunächst stellt ist, wie die vom NFC Controller empfangenen Apdus an die HostApduService Klasse weitergeleitet werden. Bei der Untersuchung der HostApduService Klasse stellt sich heraus, dass die processCommandApdu Methode innerhalb eines MessageHandlers aufgerufen wird. Dies ist eine Klasse des Android Binder IPC Mechanismus. Daraus lässt sich schließen, dass ebenfalls eine Instanz eines Messengers, der für die Kommunikation mit dem HostApduService verantwortlich ist, im Systemapplikation-Paket zu finden ist.

Diese Messenger Instanz befindet sich in der HostEmulationManager Klasse im Systemapplikation-Paket. Diese Klasse ist verantwortlich für die korrekte Weiterleitung sowohl von Command- als auch von Response-Apdus. Um diese Funktionalität bereitzustellen, besitzt die Klasse das bereits angesprochene Messenger-Objekt, um über dieses Command-Apdus an eine HostApduService-Instanz weiterzuleiten. Response-Apdus werden analog auf umgekehrtem Weg empfangen. Hierzu besitzt der HostEmulationManager ein MessageHandler-Objekt, das über die dazugehörige

Messenger-Instanz im HostApduService angesprochen wird. Darüber hinaus verfügt der HostEmulationManager eine Zustandsverwaltung für die HCE Funktionalität.

Nachdem auf einem Android-Gerät mehrere Applikationen vorhanden sein können, die eine HostApduService Komponente enthalten, muss das System eine Möglichkeit besitzen, die korrekte Instanz für ein ankommendes Command-Apdu auszuwählen. Um dies zu realisieren, besitzt der HostEmulationManager ein RegisteredAidCache-Objekt. Im Prinzip ist diese Komponente ein Speicherort, der registrierten AIDs die zugehörigen HostApduService-Instanzen zuordnet und diese verwaltet. Wird ein SELECT APDU registriert, wird daraufhin die passende HostApduService Instanz aus dem RegisteredAidCache geladen und alle nachfolgenden Apdus werden an dieses Service weitergeleitet.

Die Methoden des HostEmulationManagers zum Aktivieren der HCE Funktionalität sowie zur Weiterleitung von Command-Apdus werden von der sich im selben Paket befindlichen CardEmulationManager-Klasse aufgerufen. An diese Komponente werden ebenfalls die Response-Apdus zurückgeleitet. Der CardEmulationManager stellt eine zentrale Schnittstelle für unterschiedliche Services, die Card Emulation implementieren, zur Verfügung. In dieser Klasse findet die tatsächliche Verwaltung der HCE Services sowie der AIDs statt, die im HostEmulationManager ausgelesen werden.

Der nächste Schritt in der Implementierung führt von der CardEmulationManager-Komponente zur NfcService Klasse. Aus dieser äußerst umfangreichen Klasse ist zu erkennen, dass diese die zentrale Instanz der NFC Applikation darstellt. Sie ist verantwortlich für die Ausführung jeglicher NFC Funktionalität auf dem Gerät und zur Weiterleitung von NFC Signalen an die zur Verarbeitung verantwortlichen Komponenten. Das NfcService implementiert das DeviceHostListener Interface und besitzt eine Instanz der NativeNfcManager-Klasse die über das Java Native Interface (JNI) von der nativen Implementierung der NFC Schnittstelle angesprochen wird. Bei dieser Instanz registriert sich das NfcService selbst als Callback und kann auf diese Art und Weise Daten von der nativen NFC-Implementierung erhalten.

Über die native Implementierung werden die NFC Signale schlussendlich über die Hardware Abstraction Layer (HAL) Schicht und den Linux Kernel an die NFC Hardware weitergeleitet. Die Implementierung der nativen sowie der darunterliegenden Schichten liegt allerdings außerhalb des Rahmens dieser Diskussion und wird nicht weiter beleuchtet. Die Implementierung der Umleitung der NFC Signale erfolgt ohnehin in den Java-Komponenten der NFC Schnittstelle und wird im nächsten Kapitel erläutert.

3.4.2 Implementierung der Interception

Nachdem die Signale vom Angreifer-Gerät über die Kommunikationsanwendung zum Opfer-Gerät weitergeleitet wurden, musste zunächst die HCE Applikation aktiviert werden. Diese wird im Normalfall nur von der Systemservice NFC Anwendung gestartet, sobald ein Apdu von der NFC Schnittstellte empfangen wurde. Das Auswählen der passenden HCE App sowie das Starten dieser findet in der HostEmulationManager-Klasse in der Methode onHostEmulationData(byte[] data) statt. Es musste demnach eine Möglichkeit definiert werden, die Methoden der HostEmulationManager-Klasse aus der Kommunikationsanwendung aufzurufen. Um dies zu bewerkstelligen, wurde ein Standard Android BroadcastReceiver entwickelt, der mit der selbstdefinierten Action „HCE_COMMAND“in der HostEmulationManager-Klasse registriert wird. Hierbei wird an den BroadcastReceiver das HostEmulationManager-Objekt ebenfalls übergeben, um dessen Methoden ausführen zu können. Sobald ein Broadcast mit dieser Action gesendet wird, wird in der onReceive Methode des selbstgeschriebenen BroadcastReceivers die onHostEmulationData Methode aufgerufen. Die Apdu-Daten werden mit dem Broadcast übergeben. Gesendet wird ein entsprechender Broadcast in der Kommunikationsanwendung nach dem Empfangen von Apdu-Daten vom Angreifer. Diese Schritte ermöglichen das Weiterleiten der Command-Apdus zu einer beliebigen HCE Anwendung.

Abbildung 3.4: Weg eines NFC Command Apdus

Nach Implementierung des BroadcastReceivers sowie dessen Anbindung an die Android Systemkomponenten verläuft der Weg eines vom POS Terminal stammenden Command-Apdus wie in Abbildung **XXX** gezeigt.

Ausgehend vom POS Terminal wird das Command Apdu vom Angreifer Gerät durch die bereits beschriebenen unveränderten Systemkomponenten an das HostApduService in der Kommunikationanwendung gesendet. Daraufhin wird von diesem Service eine TCP/IP Verbindung über das lokale WLAN Netzwerk mit dem Opfer-Gerät aufgebaut. Konnte die Verbindung erfolgreich erstellt werden, wird das Command-Apdu über diese zur Kommunikationsanwendung im Opfer-Gerät gesendet. Beim Empfangen des Apdus wird ein Broadcast mit der Action „HCE_COMMAND“und den Apdu-Daten als Extra an das Android Betriebssystem gesendet. Dieses leitet den Broadcast an die Instanz des selbstdefinierten BroadcastReceivers in der modifizierten HostEmulationManager Klasse weiter, wodurch die Methode onHostEmulationData aufgerufen wird. Ab diesem Zeitpunkt nimmt das Command-Apdu denselben Weg zur HCE Anwendung, wie wenn es direkt von der NFC Schnittstelle und einem POS Terminal gestartet wird. Wird das APDU als SELECT APDU erkannt, wird aus dem RegisteredAidCache die entsprechende HostApduService Instanz zur Verarbeitung des Kommandos ausgewählt und die Nachricht an dieses Service weitergeleitet. Ansonsten wird das Command-Apdu an die zuvor ausgewählte HostApduService Instanz gesendet.

Nachdem das Command-Apdu von der HCE Applikation verarbeitet wurde, liefert diese ein entsprechendes Response-Apdu als Antwort. Dieses wird im Normalfall vom HostApduService wieder zurück an die HostEmulationManager-Komponente in der Android NFC Anwendung gesendet. Von dieser Stelle wird es danach über den CardEmulationManager und das NfcService an die Hardware weitergegeben, wo das Response-Apdu als NFC Antwort abgesendet wird. Nachdem sich das Opfer-Gerät physisch allerdings nicht in der Nähe eines POS Terminals befindet, würde das Response-Apdu einfach verloren gehen. Es wird daher ein Weg benötigt, das Response-Apdu wieder über die Kommunikationsanwendung zum Angreifer-Gerät und von diesem ans POS Terminal zurückzusenden.

Implementiert wurde dies analog in umgekehrter Art und Weise wie bei der Weiterleitung von Command-Apdus. Die HostEmulationManager-Klasse wurde so modifiziert, dass beim Empfangen eines Response-Apdus über den MessageHandler des entsprechenden HostApduServices, dieses Response-Apdu wiederum über einen Broadcast an das Android Betriebssystem gesendet wird. In der Kommunikationsanwendung wird bereits beim Initiieren der Verbindung mit dem Angreifer ein entsprechender BroadcastReceiver zur Bearbeitung von Response-Apdus registriert. Als Reaktion auf das Empfangen eines derartigen Broadcasts, der die Response-Apdu Daten als Extra enthält, werden diese Daten über die WLAN Verbindung an das Angreifer-Gerät zurückgesendet. Dieses leitet die Daten schlussendlich an das POS Terminal weiter.

Auf diese Art und Weise wurde durch nur zwei Modifikationen an der HostEmulationManager Klasse in der Android NFC Applikation erfolgreich eine Proof-of-Concept Relay Attacke durchgeführt.

Der Kommunikationsweg der vollständigen Relay-Attacke ist zur Veranschaulichung in der nachfolgenden Abbildung dargestellt.

Abbildung 3.5: Vollständiger Kommunikationsweg bei der Durchführung der Relay Attacke

In diesem Kapitel wird die eigentliche Problemlösung in einem oder mehreren Unterkapiteln ausgeführt. Die Strukturierung dieser Kapitel ist naturgemäß sehr stark von der konkre Aufgabenstellung abhängig. Der Name dieses Kapitels ist anzupassen, z.B. Umfeldbeschreibung – Fallbeispiel . . . , konkreter schreiben je nach Art Diplomarbeit/Fragestellung.

4 Hinweise zur Literatur

4.1 Literatursuche

Der Vollzugang zu einigen Publikationen ist nur intern aus dem TU-Netz möglich. Um auf möglichst viele Papers extern zugreifen zu können, wird von der TU Wien eine VPN-Zugangsmöglichkeit angeboten, diesen VPN-Zugang bitte gleich einrichten.

Besonders ergiebig sind folgende Search-Engines:
Microsoft Academic
ACM-Datenbank
Google Scholar

Wir empfehlen, vor Beginn Ihrer Arbeit einige Diplomarbeiten, die am INSO oder generell an der Fakultät für Informatik verfaßt wurden, zu Ihrem Themenbereich zu suchen und Aufbau, Schreibstil, Art der Abbildungen etc. durchzuschauen. Arbeiten finden Sie hier.

Weitere Datenbanken und Suchmaschinen:
Elektronische Zeitschriftenbibliothek der TU Wien
Scientific Literature Digital Library (CiteSeer)
Ingenta
INSPEC

Journals:
IEEE - Institute of Electrical and Electronics Engineers, Inc. - Library
Verlag Springer - Springer Link
Elsevier

Bibliotheken und Online-Kataloge:
Online-Kataloge des Österreichischen Bibliothekenverbundes
Online-Katalog der TU Wien (ALEPH)
Digital Bibliography & Library Project (DBLP) of University of Trier
The Collection of Computer Science Bibliographies

4.2 BibLatex

Biblatex bietet verschiedene Möglichkeiten an, um Literatur zu referenzieren. Die beiden häufigsten Befehle sind `\cite` und `\citeauthor`.

Beispiele wie referenziert werden kann:
fankhauser:2009:softwaretechnik-security beschreiben in [**fankhauser:2009:softwaretechnik-security**]
...
In [**schanes:2011:voip-fuzzer**] zeigen **schanes:2011:voip-fuzzer** wie ... Weitere Informationen können in [**oasis:2010:homepage**] von **oasis:2010:homepage** entnommen werden.

Wir empfehlen JabRef, um die Literaturdatenbank zu verwalten.

5 Algorithmen und Quellcode

5.1 Beispiele für Quellcode

Beispiel eines Quellcodes ist im Quellcode 5.1 zu finden.

```
1 //Start Program
2 System.out.println("Hello World!");
3 //End Program
```

Listing 5.1: Short code

5.2 Beispiele für Algorithmen

Algorithmus 5.1 dient als Beispiel.

input : A bitmap Im of size $w \times l$
output : A partition of the bitmap

```
1  special treatment of the first line;
2  for i ← 2 to l do
3      special treatment of the first element of line i;
4      for j ← 2 to w do
5          left ← FindCompress(Im[i, j − 1]);
6          up ← FindCompress(Im[i − 1, ]);
7          this ← FindCompress(Im[i, j]);
8          if left compatible with this then ;                    // O(left, this)==1
9
10             if left < this then Union(left,this);
11             ;
12             else Union(this,left);
13             ;
14         end
15         if up compatible with this then ;                      // O(up, this)==1
16
17             if up < this then Union(up,this);
18             ;
               // this is put under up to keep tree as flat as
                  possible
19             else Union(this,up);
               ;                                          // this linked to up
20         end
21     end
22     foreach element e of the line i do FindCompress(p);
23 end
```

Algorithmus 5.1 : Sample algorithm

6 Analyse der Attacke

Nach der erfolgreichen Durchführung der beschriebenen Relay Attacke, stellt sich die Frage,

6.1 Resultate

6.2 Anwendbarkeit

6.3 Folgen und Risiken

Die Resultate der Arbeit präsentieren und nach Möglichkeit aussagekräftige, eigenständige Abbildungen einbauen. Namen des Kapitels konkretisieren, an jeweilige Arbeit anpassen – Lösungsvorschlag/Implementierung im Titel des Kapitels benennen.

7 Zusammenfassung und Ausblick

Literatur

Wissenschaftliche Literatur

[3] Thomas Bocek u. a. “An NFC Relay Attack with Off-the-shelf Hardware and Software”. In: *10th IFIP WG 6.6 International Conference on Autonomous Infrastructure, Management, and Security, AIMS 2016* (2016). URL: https://link.springer.com/chapter/10.1007/978-3-319-39814-3_8 (besucht am 15. 01. 2018).

[7] Andrea Cuno. *Near Field Communication*. 2010. URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.395.8428&rep=rep1&type=pdf#page=11 (besucht am 19. 04. 2018).

[14] EMVCo. “Integrated Circuit Card Specifications for Payment Systems: Book 2 - Security and Key Management”. In: 2011. URL: https://www.emvco.com/wp-content/uploads/2017/05/EMV_v4.3_Book_2_Security_and_Key_Management_20120607061923900.pdf (besucht am 10. 05. 2018).

[15] EMVCo. “Integrated Circuit Card Specifications for Payment Systems: Book 3 - Application Specification”. In: 2011. URL: https://www.emvco.com/wp-content/uploads/2017/05/EMV_v4.3_Book_3_Application_Specification_20120607062110791.pdf (besucht am 29. 04. 2018).

[16] Lishoy Francis u. a. “Practical NFC Peer-to-Peer Relay Attack Using Mobile Phones”. In: *International Workshop on Radio Frequency Identification: Security and Privacy Issues* (2010). URL: https://link.springer.com/chapter/10.1007/978-3-642-16822-2_4 (besucht am 15. 01. 2018).

[19] Ernst Haselsteiner und Klemens Breitfuß. *Security in Near Field Communication (NFC)*. 2006. URL: https://s3.amazonaws.com/academia.edu.documents/8360228/002%20-%20security%20in%20nfc.pdf?AWSAccessKeyId=AKIAIWOWYYGZ2Y53UL3A&Expires=1516912912&Signature=sib36KcnIO0AAbNxqdEjxuId6TY%3D&response-content-disposition=inline%3B%20filename%3DSecurity_in_near_field_communication_NFC.pdf (besucht am 25. 01. 2018).

[24] Z. Kfir und A. Wool. “Picking Virtual Pockets using Relay Attacks on Contactless Smartcard”. In: *First International Conference on Security and Privacy for Emerging Areas in Communications Networks (SECURECOMM’05)* (2005). URL: http://ieeexplore.ieee.org/abstract/document/1607558/ (besucht am 15. 01. 2018).

[26] Gerald Madlmayr, Christian Kantner und Thomas Grechenig. “Near Field Communication”. In: *Secure Smart Embedded Devices, Platforms and Applications*. Hrsg. von Konstantinos Markantonakis und Keith Mayes. Springer New York, 2014.

[33] Wasim Raad, Tarek Sheltami und Mohammad Sallout. *A Smart Card Based Prepaid Electricity System*. Aug. 2007.

[34] Henning Siitonen Kortvedt und Stig Mjølsnes. “Eavesdropping Near Field Communication”. In: *The Norwegian Information Security Conference (NISK) 2009* (2009). URL: https://www.researchgate.net/publication/265976861_Eavesdropping_Near_Field_Communication (besucht am 10. 05. 2018).

[43] A. Supriya, S. Ramgopal und S. M. George. “Near field communication based system for health monitoring”. In: *2017 2nd IEEE International Conference on Recent Trends in Electronics, Information Communication Technology (RTEICT)*. 2017, S. 653–657. DOI: 10.1109/RTEICT.2017.8256678.

[44] José Vila und Ricardo J. Rodríguez. “Practical Experiences on NFC Relay Attacks with Android”. In: *International Workshop on Radio Frequency Identification: Security and Privacy Issues* (2015). URL: https://link.springer.com/chapter/10.1007/978-3-319-24837-0_6 (besucht am 15. 01. 2018).

[45] R. Want. “An introduction to RFID technology”. In: *IEEE Pervasive Computing* 5.1 (2006), S. 25–33. ISSN: 1536-1268. DOI: 10.1109/MPRV.2006.2.

Online-Referenzen

[1] Wi-Fi Alliance. *Wi-Fi Direct*. URL: https://www.wi-fi.org/discover-wi-fi/wi-fi-direct (besucht am 07. 07. 2018).

[2] *Android Interprocess Communication (IPC) with Messenger (Remote Bound Services)*. 2015. URL: http://codetheory.in/android-interprocess-communication-ipc-messenger-remote-bound-services/ (besucht am 17. 07. 2018).

[4] CardContact Software & System Consulting. *Dynamic Data Authentication*. URL: https://www.openscdp.org/scripts/tutorial/emv/dda.html (besucht am 10. 05. 2018).

[5] CardContact Software & System Consulting. *Static Data Authentication (SDA)*. URL: https://www.openscdp.org/scripts/tutorial/emv/SDA.html (besucht am 03. 05. 2018).

[6] Sony Corporation und Philips. *PHILIPS AND SONY ANNOUNCE STRATEGIC COOPERATION TO DEFINE NEXT GENERATION NEAR FIELD RADIO-FREQUENCY COMMUNICATIONS*. 2002. URL: https://www.sony.net/SonyInfo/News/Press_Archive/200209/02-0905E/ (besucht am 24. 01. 2018).

[8] Android Developers. *Architecture*. URL: https://source.android.com/devices/architecture/ (besucht am 17. 07. 2018).

[9] Android Developers. *Bluetooth overview*. URL: https://developer.android.com/guide/topics/connectivity/bluetooth (besucht am 07. 07. 2018).

[10] Android Developers. *Hardware Abstraction Layer (HAL)*. URL: https://source.android.com/devices/architecture/hal (besucht am 17. 07. 2018).

[11] Android Developers. *Host-based Card Emulation*. URL: https://developer.android.com/guide/topics/connectivity/nfc/hce.html (besucht am 25. 01. 2018).

[12] Android Developers. *Platform Architecture*. URL: https://developer.android.com/guide/platform/ (besucht am 17. 07. 2018).

[13] EMVCo. *EMVCo - Webseite*. URL: https://www.emvco.com/ (besucht am 28. 04. 2018).

[17] *Getting information from an EMV chip card with Java*. 2006. URL: https://blog.saush.com/2006/09/08/getting-information-from-an-emv-chip-card/ (besucht am 29. 04. 2018).

[18] Statista GmbH. *Global mobile OS market share in sales to end users from 1st quarter 2009 to 1st quarter 2018*. 2018. URL: https://www.statista.com/statistics/266136/global-market-share-held-by-smartphone-operating-systems/ (besucht am 16. 07. 2018).

[20] *History of Near Field Communication*. URL: http://nearfieldcommunication.org/history-nfc.html.

[21] Bluetooth SIG Inc. *Bluetooth 5 Now Available*. 2016. URL: https://www.bluetooth.com/news/pressreleases/2016/12/07/bluetooth-5-now-available (besucht am 07. 07. 2018).

[22] European Telecommunications Standards Institute. *ETSI TS 102 190: Near Field Communication (NFC) IP-1; Interface and Protocol (NFCIP-1)*. 2003. URL: http://www.etsi.org/deliver/etsi_ts/102100_102199/102190/01.01.01_60/ts_102190v010101p.pdf (besucht am 15. 04. 2018).

[23] Infosec Institute. *Near Field Communication (NFC) Technology, Vulnerabilities and Principal Attack Schema*. 2013. URL: http://resources.infosecinstitute.com/near-field-communication-nfc-technology-vulnerabilities-and-principal-attack-schema/ (besucht am 24. 01. 2018).

[25] Prof. Dr. Richard Lackes. *Gabler Wirtschaftslexikon: Chipkarte*. URL: https://wirtschaftslexikon.gabler.de/definition/chipkarte-27504#authors (besucht am 24. 04. 2018).

[27] Mastercard. *EMV Chip*. URL: https://www.mastercard.at/de-at/haendler/sicherheit-geschaeft/emv-chip.html (besucht am 26. 04. 2018).

[28] Österreichische Nationalbank. *Standardisierung und SEPA*. URL: https://www.oenb.at/Zahlungsverkehr/Kartenzahlungen/Standardisierung-und-SEPA.html (besucht am 26. 04. 2018).

[29] NearFieldCommunication.org. *NFC Forum*. URL: http://nearfieldcommunication.org/history-nfc.html (besucht am 15. 05. 2018).

[30] Jeroen Netten. *Step by step: How does a EMV contact card payment work?* 2016. URL: https://www.quora.com/Step-by-step-How-does-a-EMV-contact-card-payment-work (besucht am 29. 04. 2018).

[31] David Nield. *Bluetooth 5: everything you need to know*. 2016. URL: https://www.techradar.com/news/networking/bluetooth-5-everything-you-need-to-know-1323060 (besucht am 07. 07. 2018).

[32] *POS Terminal Risk Management Rules You Need to Know*. URL: http://blog.unibulmerchantservices.com/pos-terminal-risk-management-rules-you-need-to-know/ (besucht am 02. 05. 2018).

[35] *Socket Connection Over Internet*. URL: https://forum.xda-developers.com/showthread.php?t=1154801.

[36] International Organization for Standardisation (ISO). *ISO/IEC 7816-4:2013 Identification cards – Integrated circuit cards – Part 4: Organization, security and commands for interchange*. 2013. URL: https://www.iso.org/standard/54550.html (besucht am 28. 04. 2018).

[37] International Organization for Standardization. *ISO/IEC 14443-1:2016 Identification cards – Contactless integrated circuit cards – Proximity cards – Part 1: Physical characteristics*. URL: https://www.iso.org/standard/70170.html (besucht am 26. 04. 2018).

[38] International Organization for Standardization. *ISO/IEC 14443-2:2016 Identification cards – Contactless integrated circuit cards – Proximity cards – Part 2: Radio frequency power and signal interface*. URL: https://www.iso.org/standard/70170.html (besucht am 26. 04. 2018).

[39] International Organization for Standardization. *ISO/IEC 14443-3:2016 Identification cards – Contactless integrated circuit cards – Proximity cards – Part 3: Initialization and anticollision*. URL: https://www.iso.org/standard/70170.html (besucht am 26. 04. 2018).

[40] International Organization for Standardization. *ISO/IEC 14443-4:2016 Identification cards – Contactless integrated circuit cards – Proximity cards – Part 4: Transmission protocol*. URL: https://www.iso.org/standard/70170.html (besucht am 26. 04. 2018).

[41] International Organization for Standardization (ISO). *ISO/IEC 18092:2013: Information technology – Telecommunications and information exchange between systems – Near Field Communication – Interface and Protocol (NFCIP-1)*. 2013. URL: https://www.iso.org/standard/56692.html (besucht am 15. 04. 2018).

[42] *Structure of an Android Operating System*. 2016. URL: https://www.educba.com/structure-of-an-android-operating-system/ (besucht am 17. 07. 2018).

[46] *Wie schnell ist eigentlich LTE | 4G?*

A Anhang

Quellcode, Datenmodell, Fragebögen, ...