ISO/IEC-14443-4 Weiterleitung über Android-Smartphones

Studienarbeit

Humboldt-Universität zu Berlin
Mathematisch-Naturwissenschaftliche Fakultät
Institut für Informatik

eingereicht von:	Kai Warncke
geboren am:	18. Mai 1983
in:	Schwerin
Betreuer:	Dr. Wolf Müller
eingereicht am:	19. Juni 2014

Inhaltsverzeichnis

1 Einleitung

Chipkarten, auch *Smartcard* oder *Integrated Circuit Card* (ICC) genannt, erfreuen sich einer zunehmenden Beliebtheit. Es handelt sich dabei um kleine Plastikkarten, welche einen Chip beinhalten. Die Funktion dieses Chips reicht dabei von der einfachen Speicherung von Daten, bishin zu deren Verarbeitung. Sie verfügt über einen Prozessor, nicht flüchtigen und flüchtigen Speicher, bringt ein eigenes Betriebssystem, sowie passende Programme mit. Bei der Electronic Cash (EC) Karte oder der Krankenversicherungskarte handelt es sich zum Beispiel um eine solche Chipkarte. Für die Kommunikation mit der Chipkarte gibt es sowohl ein kontaktbehaftetes Verfahren, als auch ein kontaktloses.
Kontaktbehaftete Chipkarten verfügen über markante, an die Oberfläche der Plastikhülle geführte, Kontaktflächen (siehe Abb. 1, links). Über die Leitungen, unter den jeweiligen Kontaktflächen, wird der Chip mit Strom und einem Takt vom Lesegerät versorgt. Der Takt ist besonders für die Synchronisation der Kommunikation, zwischen Lesegerät und Chipkarte, wichtig.
Die Kommunikationsschnittstelle kontaktloser Chipkarten ist hingegen komplett im Inneren das Plastikkarte untergebracht. Stromversorgung und Kommunikation werden dabei durch das Verfahren der *Radio-Frequency Identification* ermöglicht.

1.1 Radio-Frequency Identification

Radio-Frequency Identification (RFID) bezeichnet eine Technologie zur kontaktlosen Datenübertragung mittels Radiowellen. Ein RFID-System besteht mindestens aus zwei Komponenten.

- Ein RFID-*Tag* ist ein kontaktlos ansteuerbarer Chip.
- Ein RFID-*Leser* kann mit RFID-Tags kommunizieren.

Ein RFID-Tag besteht im Allgemeinen aus einer, zu einer Spule gewickelten, Antenne und einem integrierten Schaltkreis (Chip) (siehe Abb. 1, rechts). Abgesehen von der spezifischen Funktionalität der verwendeten Chips, unterscheiden sich RFID-Tags auch in der Art ihrer Stromversorgung (siehe Tabelle 1). So lassen sich RFID-Tags, abhängig von ihrer Stromquelle, in die Klassen *aktiv*, *semi-passiv* und *passiv* einteilen. Aktive Tags verfügen dabei über eine eigene Stromquelle, wie zum Beispiel einer Batterie. Sie sind in der Lage eine Verbindung zu einem RFID-Leser oder einem anderen aktiven Tag zu initialisieren. Ebenso wie aktive Tags, verfügen auch semi-passive Tags über eine eigene Stromquelle,

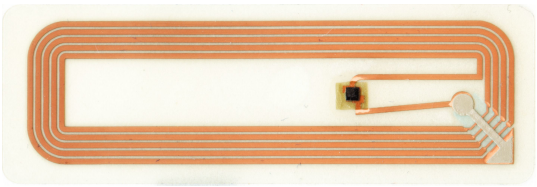

Abbildung 1: Links: kontaktbehaftete Chipkarte mit sichtbaren Kontaktflächen[1]; Rechts: Innenansicht einer kontaktlosen Chipkarte[2] (Chip und gewickelte Antenne)

können jedoch nicht eigenständig einen Verbindungsaufbau initialisieren. Passive Tags verfügen über keine eigene Stromquelle und verbleiben inaktiv, bis sie von einem RFID-Leser aktiviert werden. Durch einen RFID-Leser wird ein elektromagnetisches Feld erzeugt. Sobald ein passives Tag in dieses Feld kommt, bezieht es, die für den Betreib notwendige Energie über Induktion aus dem Feld des Lesers. Tags mit integrierten Stromquellen ermöglichen es, eine Verbindung über eine größere Entfernung aufzubauen, als es mit passiven Tags möglich wäre. Aktive Tags haben jedoch auch höhere Produktionskosten, weniger kompakte Fertigungsgrößen, sowie eine durch die Batterie begrenzte Lebensdauer. Aus diesen Gründen sind heute passive Tags am weitesten verbreitet. Man findet sie häufig als Diebstahlsicherung in Etiketten des Einzelhandels, Kreditkarten, Ticketsystemen, Wegfahrsperren von Automobilen, sowie zunehmend in Personaldokumenten.

Tag Typ	**Passiv**	**Semi-Passiv**	**Aktiv**
Stromquelle	Induktion	Batterie	Batterie
Kommunikation	nur Antworten	nur Antworten	Antworten & Initialisieren
max. Reichweite	10 m	>100 m	>100 m
relative Kosten	am günstigsten	weniger günstig	am teuersten
Beispiele für Anwendungsbereiche	elektronische Schlösser	automatisierte Mautstationen	Überwachung von Frachtcontainern

Tabelle 1: Vergleich passiver, semi-passiver und aktiver Tags.[Wei]

RFID-Systeme arbeiten in bestimmten Frequenzbereichen zwischen 120 kHz und 10.6 GHz (siehe Tabelle 2). Die physikalischen Eigenschaften der Radiofrequenzen begünstigen bestimmte Anwendungsszenarien. So ist eine geringe Reichweite der Datenübertragung, für das Auslesen von Kreditkartendaten oder das Öffnen einer Tür, durchaus wünschenswert. Beim passieren einer automatischen Mautstation ist ein notwendiger Abstand von weniger

[1] Quelle: `http://upload.wikimedia.org/wikipedia/de/a/a9/Chipkarte.jpg`
[2] Quelle: `http://upload.wikimedia.org/wikipedia/commons/6/6f/Transponder2.jpg`

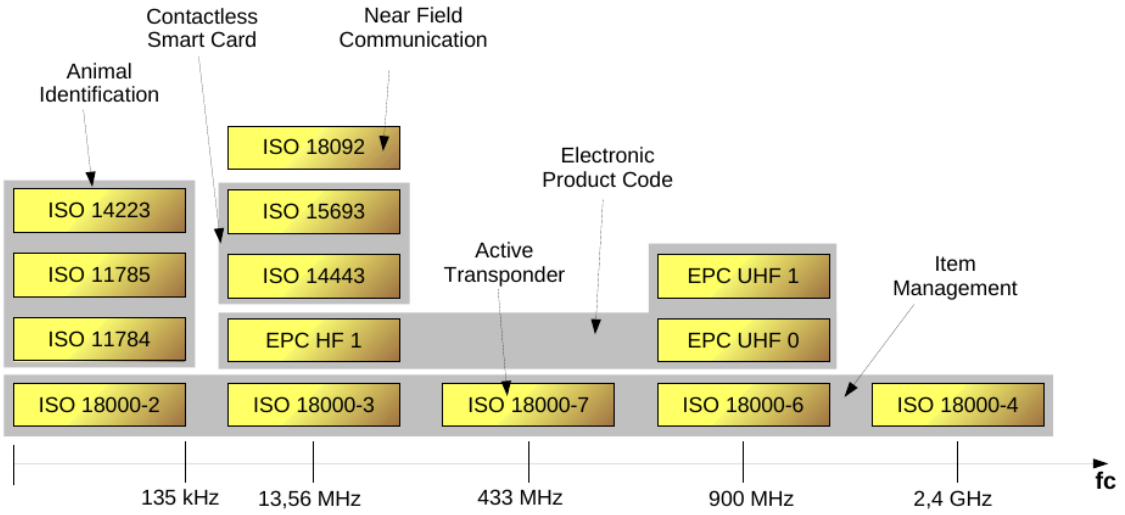

Abbildung 2: Verschiedene RFID Standards in den jeweiligen Frequenzbändern [Wei10, S. 4].

als 20 cm zwischen Tag und Leser eher unpraktisch. Für letzteren Anwendungsfall bieten sich höhere Frequenzen an, welche sowohl eine größere Reichweite, als auch eine größere Datenübertragungsrate bieten.

Typ	**Frequenzband**	**Reichweite (passiv)**
Low Frequency (LF)	120 kHz - 140 kHz	10 - 20 cm
High Frequency (HF)	13.56 MHz	10 - 20 cm
Ultra-High Frequency (UHF)	868 - 928 MHz	3 m
Microwave	2.45 & 5.8 GHz	3 m
Ultra-Wide Band (UWB)	3.1 - 10.6 GHz	10 m

Tabelle 2: RFID-Frequenzbänder und die maximale Reichweite zum lesen eines passiven Tags. [Wei]

Die Vielfälltigkeit von RFID-Systemen begünstigt ihre schnelle Verbreitung in verschiedensten Anwendungsbereichen. Heute existiert eine Vielzahl internationaler Standards für bestimmte RFID-Systeme (Abb. 2). Der Standard, mit der Bezeichnung ISO/IEC-14443, für kontaktlose Chipkarten ist ein solcher und soll im Folgenden näher betrachtet werden.

1.2 Kontaktlose Chipkarten nach ISO/IEC-14443

RFID-Tags, in Form von kontaktlosen Chipkarten, finden sich mittlerweile in vielen Bereichen des täglichen Lebens. Kreditkartenanbieter sind zum Beispiel dazu übergegangen,

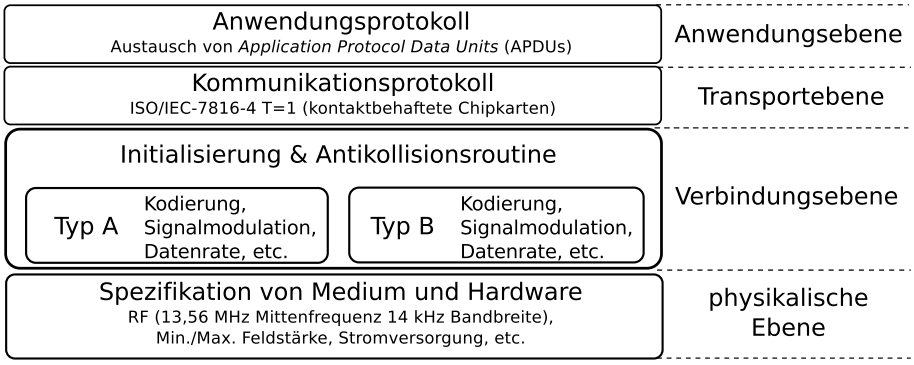

Abbildung 3: ISO/IEC-14443 Schichtmodell

Kreditkarten mit RFID-Tags auszustatten. Diese Tags enthalten, die für einen Zahlungsvorgang notwendigen, Kreditkarteninformationen und sollen eine möglichst unkomplizierte Übertragung dieser ermöglichen. Auch offizielle Personaldokumente, wie Ausweise und Reisepässe, werden mit Chips ausgestattet, welche zum Beipiel biometrische Informationen des Besitzers enthalten können.
Kontaktlose Chipkarten werden auch für elektronische Schließmechanismen verwendet. Dabei erfüllt ein RFID-Leser die Aufgabe eines Schlosses und die Chipkarte die Rolle des Schlüssels. Die Feststellung der Zugangsberechtigung kann dabei beliebig kompliziert erfolgen. Oft reicht jedoch bereits die Existenz einer zugangsberechtigten Karte im Feld des Lesers aus, um eine Tür zu öffnen.
Im Standard ISO-14443 [isoa], [isob], [isoc] und [isod] sind solche kontaktlosen Chipkarten, unter der Bezeichnung *Proximity Integrated Circuit Card* (PICC), spezifiziert. Die korrespondierenden Leser werden in diesem Zusammenhang als *Proximity Coupling Device* (PCD) bezeichnet. PICC und PCD verwenden ein Frequenzband um 13,56 MHz ($\pm$7 kHz). Da die Verbindungsanfragen eines PCD von allen PICC's in seiner Reichweite beantwortet werden, wird zusätzlich ein Auswahlverfahren notwendig. Zu diesem Zweck sieht der Standard einen *anticollision loop* vor. Dabei handelt es sich um einen Algorithmus, der anhand des jeweiligen *Unique Identifiers* (UID) die sich im Feld befindenden Chipkarten unterscheidet. Die UID einer Chipkarte ist 4, 7 oder 10 Byte lang. Dabei kann es sich um eine feste, vom Hersteller der Karte vorgegebene, oder eine von der Chipkarte zufällig generierte UID handeln. Nach der erfolgreichen Abarbeitung des Algorithmus verbleibt genau eine Kombination aus PICC und PCD, über deren Verbindung im weiteren Verlauf die Kommunikation stattfindet. Der Standard sieht zu diesem Zweck zwei verschiedene Verbindungsmodi vor. Sie werden als *Typ A* und *Typ B* bezeichnet. Die Modi unterscheiden sich in Hinblick auf Signalmodulation, die verwendete Kodierung, sowie ihre Initialisierung. Sobald eine Verbindung zwischen PICC und PCD mit einem der beiden Modi hergestellt wurde, setzt darauf das gleiche Kommunikationsprotokoll auf. Das verwendete Kommunikationsprotokoll basiert auf dem kontaktbehafteter Chipkarten[1]. Die Kommunikation findet auf Applikationsebene, durch den wechselseitigen Austausch

[1] Spezifiziert in ISO/IEC-7816-4 (Protokoll T=1).

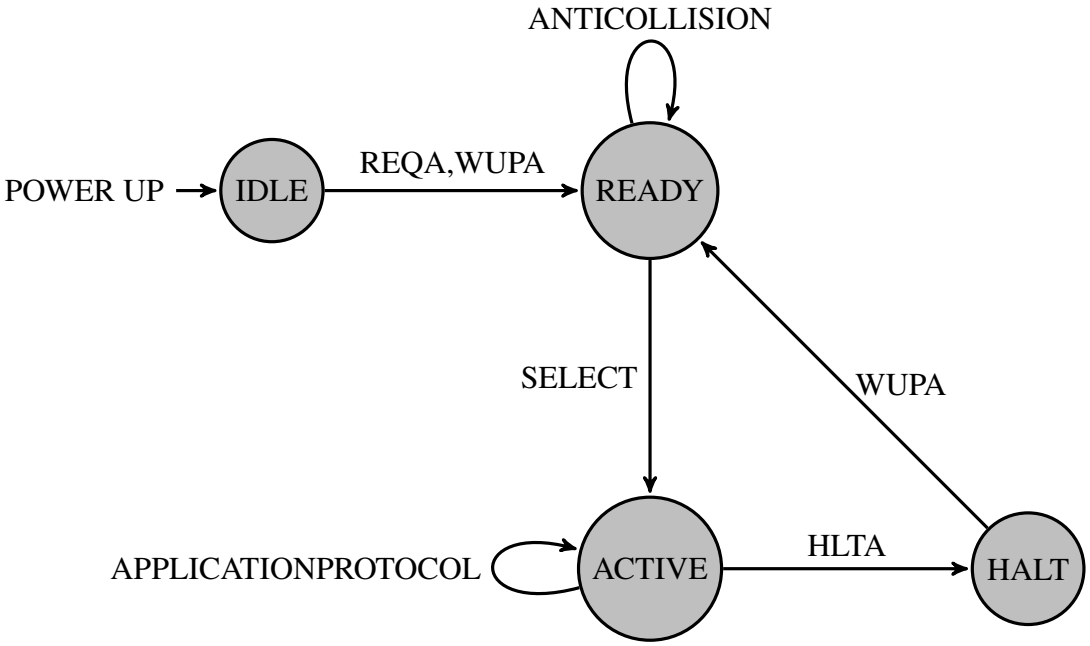

Abbildung 4: Zustandsautomat für eine Typ A Verbindung nach ISO/IEC-14443.

von *Application Protocol Data Units* (APDUs), statt. Wie lange ein PCD dabei auf eine APDU der PICC wartet wird beim Verbindungsaufbau festgelegt. Laut Standard beträgt die Wartezeit, die *Frame Wait Time* (FWT), 4833 μs. Sollte es der PICC, zum Beispiel aufgrund von fehlerhaft empfangenen Daten, nicht möglich sein in dieser Zeit mit dem Senden der Antwort zu beginnen, kann eine Verlängerung der Wartezeit verlangt werden. Durch das Senden eines *Request for a Waiting Time Extension* kann die Wartezeit dabei maximal um 4949 *ms* verlängert werden. Wie oft man die Wartezeit verlängert ist durch den Standard nicht limitiert. Auf die sicherheitsrelevanten Folgen, der potentiell beliebig langen Verzögerungen, wird in Abschnitt 2 näher eingegangen.

Die Initialisierungs- und Antikollisionsroutine lässt sich durch einen Zustandgraphen (siehe Abb. 4) beschreiben. Um eine Typ A Verbindung zwischen PCD und PICC zu etablieren wird zunächst die PICC in das Feld des PCD gebracht. Sie wird durch das Feld mit Strom versorgt und in den Zustand IDLE versetzt. In diesem Zustand ist die Karte in der Lage Signale zu demodulieren und REQA, sowie WAKE-UP Kommandos (WUPA) zu erkennen. Das PCD sendet, in regelmäßigen Abständen, ein REQA-Kommando an alle PICCs in seinem Feld. Jede PICC im Feld antwortet auf das REQA-Kommando des PCD mit einer *Answer to Request* (ATQA) und wechselt in den Zustand READY. Wenn sich mehr als eine PICC im Feld des Lesers befindet, kommt es, bei der zeitgleichen Übermittlung der ATQA-Kommandos, zur Kollision und die Antikollisionsroutine wird ausgeführt, um eine PICC auszuwählen. Zu diesem Zweck sendet das PCD ein SELECT Kommando, in dem der Parameter *Number of Valid Bits* (NVB) und eine Bitmaske enthalten sind. Dabei gibt NVB die Anzahl der, in der Bitmaske enthaltenen, Bits an. Diese Bits werden dann von jeder PICC mit ihrer UID verglichen. Stimmt die Bitmaske mit der UID überein, antwortet

die PICC mit dem Rest ihrer UID. Da die UID 4, 7 oder 10 Byte lang sein kann, sind unter Umständen bis zu drei Durchläufe der Antikollisionsroutine notwendig. Am Ende der Antikollisionsroutine sendet das PCD ein SELECT Kommando, mit der vollständigen UID der PICC, mit der die Verbindung hergestellt werden soll. Die PICC bestätigt ihre Auswahl mit dem senden des *SELECT Acknowledge* (SAK) Kommandos und wechselt in den Zustand ACTIVE. PCD und PICC sind damit verbunden und können mit der Abarbeitung des Kommunikationsprotokolls beginnen. Sollte es sich bei dem Kommunikationprotokoll um das in ISO/IEC-14443-4 angegebene handeln, würde das PCD zunächst ein *Request Answer to Select* (RATS) an die PICC senden. Die PICC antwortet daraufhin mit den von ihr unterstützten Einstellungen, in Form eines *Answer to Select* (ATS) Kommandos. Sofern die PICC *Protocol Parameter Selection* (PPS) unterstützt, kann das PCD dies verwenden um bestimmte Parameter für das Kommunikationsprotokoll zu setzen.

1.3 Near Field Communication

Neben Bluetooth und Wireless LAN wird in neueren Mobiltelefonen, sogenannten Smartphones, auch immer öfter eine Schnittstelle für *Near Field Communication* (NFC) verbaut. Ein Smartphone mit einer NFC-Schnittstelle ermöglicht es dem Nutzer mit bestimmten RFID-Systemen zu interagieren. NFC greift dazu verschiedene RFID-Kommunikationsstandards, wie zum Beispiel ISO/IEC-14443 oder ISO/IEC-15693, auf und erweitert diese um zusätzliche Funktionalität.
Die NFC-Technologie wurde zunächst von ECMA[2], in Form von ECMA-340 (NFCIP-1) und ECMA-352 (NFCIP-2), standardisiert. Später wurden diese Standards auch von der *International Standardization Organisation* (ISO/IEC[3]), in Form von ISO/IEC-18092 sowie ISO/IEC-21481, aufgenommen. Unabhängig von den Organisationen ECMA und ISO/IEC wird die Weiterentwicklung der verwendeten Protokolle und Datenstrukturen sowie die Zertifizierung von NFC konformen Geräten und Anwendungen vom NFC-Forum[4] vorran getrieben.

NFC kann unter anderem zur Kommunikation mit kontaktlosen Chipkarten verwendet werden. Die Datenübertragung findet dabei im 13.56 MHz Band statt und ist, mit bis zu 424 kBit/s (NFCIP-1), eher für kleinere Datenmengen ausgelegt. Typische Anwendungsgebiete sind elektronische Ticket-Systeme, wie zum Beispiel *touch & travel* von der Deutschen Bahn, Bezahlsysteme, wie zum Beispiel *mpass* von den Telekommunikationsdienstleistern O_2,Vodafon und der Deutschen Telekom oder *girogo* von der Deutschen Kreditwirtschaft. Um diese und weitere Anwendungsszenarien für sich zu erschließen wurden dem NFC-Standard weitere Verbindungsmodi hinzugefügt. Der NFC-Standard

[2] http://www.ecma-international.org
[3] http://www.iso.org
[4] http://www.nfc-forum.org

spezifiziert drei verschiedene Verbindungsmodi:

- Der *Schreib/Lese*-Modus stellt eine klassische Verbindung zwischen einem aktiven NFC-Gerät und einem passiven Tag dar.
  (Aktiv $\leftrightarrow$ Passiv)
- Der *Peer-to-Peer*-Modus beschreibt eine direkte Verbindung zwischen zwei aktiven NFC-Geräten. Zum Beispiel könnten zwei NFC-fähige Smartphones diesen Modus nutzen, um einen Eintrag im Telefonbuch auszutauschen.
  (Aktiv $\leftrightarrow$ Aktiv)
- Im *Kartenemulation*-Modus emuliert das aktive NFC-Gerät einen passiven Tag. In diesem Modus verbleibt es passiv, bis es in die unmittelbare Umgebung eines Lesegeräts kommt.
  (Passiv $\leftrightarrow$ Aktiv)

Während der *Schreib/Lese*-Modus in dieser Form auch in den RFID-Standards zu finden ist, stellen die anderen beiden Modi, *Peer-to-Peer* (P2P) und *Kartenemulation*, zwei Neuerungen im NFC-Standard dar. Der P2P-Modus wird aufgrund der Datenübertragungsrate eher für das *Pairing* zweier aktiver NFC-Geräte oder den Austausch kleiner Datenmengen verwendet. *Pairing* sei in diesem Zusammenhang als Aushandlung einer Verbindung über ein breitbandigeres Medium zu verstehen. So kann der P2P-Modus genutzt werden, um alle notwendigen Informationen für eine WLAN oder Bluetooth-Verbindung zu übertragen. Der eigentliche Datentransfer wird dann über die breitbandigere Verbindung stattfinden.
Ein, für diese Arbeit besonders wichtiger, Verbindungsmodus ist der Kartenemulationsmodus. Bei der *Kartenemulation* (KE) erzeugt das NFC-Gerät kein eigenes Feld. Es verhält sich wie ein RFID-Tag und wartet auf das Feld eines Lesegerätes. Einem Smartphone wäre es somit möglich, sowohl als PICC, als auch als PCD zu fungieren. Damit kann das Smartphone sowohl mit kontaktlosen Chipkarten, als auch mit Lesegeräten kommunizieren. Die KE kann dabei auf zwei Arten erfolgen. Zum einen kann sie durch Software, zum anderen durch dedizierte Hardware, einem sogenannten *Secure Element* (SE), realisiert werden. Ein *Secure Element* ist ein Chip, wie er auch in Chipkarten zu finden ist. Es verfügt über einen gesicherten Speicherbereich, eine gesicherte Ausführungsumgebung und Hardware für bestimmte kryptografische Funktionen. Ein SE könnte im NFC-Chipsatz oder in Form einer *Universal Integrated Circuit Card* in das Smartphone integriert sein. Letzteres ist als *Subscriber Identity Module* (SIM) des Mobilfunkanbieters bekannt. Gegen eine KE auf Hardwarebasis sprechen laut Roland ([Rol12] Abbschnitt 3.1) folgende drei Gründe:

- Die Hersteller von NFC-Chipsätzen und Mobilfunkanbieter beanspruchen die Kontrolle über das verbaute SE.
- Konkurierende Hersteller könnten den Zugriff auf das SE auf ihre eigenen Produkte beschränken, um ihre Position zu stärken.

- Der Aufwand, mehrere sicherheitsrelevante Anwendungen in einem SE unterzubringen, wobei nachweislich weder die Funktionalität, noch die Sicherheit beeinträchtigt werden darf, verursacht hohe Kosten.

Als Alternative wird in seiner Arbeit die Kartenemulation durch Software (Soft-KE) angeführt. Bei der Soft-KE kann eine Anwendung auf dem Smartphone zum Beispiel eine ISO/IEC-14443 konforme Chipkarte emulieren. Zu diesem Zweck registriert sich die Anwendung beim Betriebsystem des Smartphones, für entsprechende NFC-*Events*. Wenn nun das Smartphone in das Feld eines Lesegerätes kommt, benachrichtigt der NFC-Controller das Betriebsystem, welches dann die empfangenen Daten an die Anwendung weiterreicht. Die Anwendung kann dann ihrerseits Daten auf diesem Weg zurückschicken.
Vorteile dieser Art der Kartenemulation sieht Roland vor allem darin, dass keine spezielle Hardware für die Soft-KE benötigt wird. Demzufolge ergeben sich auch nicht die Probleme der hardwarebasierten KE. Da bei der Soft-KE kein Zugriff auf spezialisierte Hardware erforderlich ist, begünstigt das die Entwicklung von Anwendungen für NFC-fähige Mobiltelefone. Diese Anwendungen können zur Kommunikation mit bereits bestehenden RFID-Systemen, zum Beispiel für mobiles Bezahlen, verwendet werden.
Die mangelnde Unterstützung des P2P-Modus seitens der im Einsatz befindlichen RFID-Leser verhindert weitgehend seine Verwendung. Um trotzdem eine Verbindung zwischen einem NFC-fähigen Mobiltelefon und einem RFID-Leser herzustellen, kann die Soft-KE verwendet werden. Da das Mobiltelefon durch die Kartenemulation die Rolle eines passiven Tags übernimmt, muss der RFID-Leser lediglich zum Schreib-/Lesemodus in der Lage sein um Daten auszutauschen.
Kartenemulation auf Softwarebasis bringt jedoch auch einen entscheidenden Nachteil, gegenüber einer hardwarebasierten Lösung, mit sich. Das Fehlen eines Secure Elements, macht das sichere Aufbewahren von sensiblen Daten sehr viel schwerer. Sicherheitsrelevante Informationen, wie zum Beispiel privates Schlüsselmaterial, liegen nun im Dateisystem des Betriebssystems statt im geschützten Speicherbereich des SE. Dadurch ergeben sich eine Vielzahl neuer Angriffvektoren, was wiederum höhere Anforderungen an die Sicherheit des gesamten Betriebssystems zur Folge hat.

1.4 Sicherheitsprobleme

RFID-Systeme sind vielfälltig und für sehr viele Anwendunsszenarien verfügbar. In all diesen Szenarien geht es um die Speicherung oder die Verarbeitung von Daten. Dabei gilt es zusätzlich zur reinen Funktionalität, notwendige Sicherheitsanforderungen zu erfüllen. Um zu beurteilen ob bestimmte Sicherheitsanforderungen erfüllt werden können, ist es notwendig grundlegende Angriffsmöglichkeiten auf RFID-Systeme zu kennen. Eine mögliche Klassifikation von Angriffen auf die Kommunikation in einem RFID-System wurde von Mitrokotsa et al. [MRT10] gegeben.

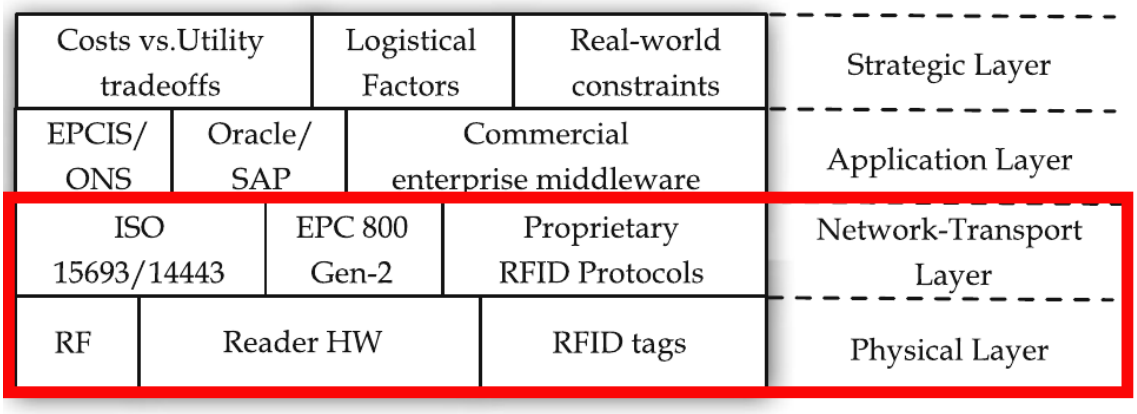

Abbildung 5: Aufteilung der RFID-Kommunikation in Schichten. Für diese Arbeit relevante Schichten sind rot hervorgehoben. Quelle: [MRT10] (S. 492)

Zunächst wird die Kommunikation von einem RFID-System in Form von verschiedenen Schichten kategorisiert (siehe Abb. 5). Auf Grundlage dessen werden Angriffe der Schicht zugeordnet, in der sie stattfinden. Im Folgenden wird näher auf die physikalische Schicht, sowie die Netzwerkschicht eingegangen, da der in Abschnitt 2 beschriebene Weiterleitungsangriff dort einzuordnen ist. Angriffe auf die Anwendungsschicht (*Application Layer*), zum Beispiel in Form von *Buffer Overflows*, sind genauso möglich. Da RFID-Systeme im Allgemeinen nur ein Teil eines größeren Sicherheitskonzeptes sind, können auch Angriffe auf die restliche Infrastruktur, wie zum Beispiel *Social Engineering*[5], ebenfalls in einer Schicht zusammengefasst werden. Auf diese sogenannte strategische Schicht (*Strategic Layer*) wird in dieser Arbeit jedoch ebenso wie auf die Anwendungsschicht nicht näher eingegangen.
In der physikalischen Schicht (*Physical Layer*) befinden sich Angriffe, welche die Manipulation oder Störung des verwendeten Frequenzbandes oder der Leser-/Tag-Hardware zum Ziel haben. Dabei können Tags zum Beispiel permanent oder zeitweilig ausgeschaltet werden. Tags sind empfindlich gegenüber elektromagnetischen Feldern. Bringt man einen Tag in ein besonders starkes Feld, ist es möglich den Chip durch eine Spannungsspitze zu zerstören. Um die Kommunikation zwischen Tag und Leser nur für eine bestimmte Zeit zu verhindern kann der Tag, durch das Radiosignal störende Materialien, abgeschirmt werden. Abhängig von der verwendeten Radiofrequenz reicht dazu unter Umständen schon das umwickeln mit Alufolie. Eine andere Möglichkeit stellt das Fluten des Frequenzbandes mit Störsignalen, das sogenannte *Active Jamming*, dar. Nutzsignale können durch Tag und Leser nicht von den Störsignalen unterschieden werden. Das führt dazu, dass keine Verbindung aufgebaut werden kann.

[5] Social Engineering beschreibt in diesem Zusammenhang Angriffe, auf die menschliche Komponente von Sicherheitssystemen.

Der Netzwerkschicht (*Network Layer*) werden alle Angriffe zugeordnet, welche auf die eigentliche Kommunikation zwischen Tag und Leser ausgerichtet sind. Dabei ist das Ziel einen der Kommunikationspartner zu ersetzen ohne das es der legitimen anderen Seite auffällt. Ein Angriff stellt zum Beispiel das sogenannte *Cloning* dar. Dabei wird ein originalgetreues Abbild des zu ersetzenden Tags erzeugt. Dazu wird Zugriff auf alle im Original enthaltenen Daten benötigt, sowie ein beschreibbarer Tag, dessen *Unique Identifier* (siehe Abschnitt 1.2) frei wählbar ist. Jedoch stellt gerade der Zugriff auf alle Daten häufig ein Problem dar. In vielen Fällen verfügen Tags über einen geschützen Speicherbereich, um sicherheitsrelevante Daten vor dem Auslesen zu schützen. Auch wenn der direkte Zugriff auf Daten in gesicherten Speicherbereichen nicht möglich ist, finden sich mitunter andere Wege daran zu gelangen. Karsten Nohl, Henryk Plötz und David Evans haben in ihren Arbeiten, [NE08] und [Plö08] den proprietären Verschlüsselungalgorithmus eines Tags vom Typ *Mifare Classic* extrahiert, indem sie die einzelenen Schichten des Chips abgeschliffen und die freigelegten Schaltkreisstrukturen untersucht haben . Durch die so gewonnenen Informationen über die verwendeten kryptographischen Komponenten war es ihnen möglich den geheimen Schlüssel von beliebigen Tags vorraus zu berechnen. Die Kenntnis des, im geschützten Speicherbereich liegenden, geheimen Schlüssels ermöglicht schließlich das *Cloning*, da alle Informationen des Originals bekannt sind. Da Tags dieser Art häufig dazu verwendet werden um die Zugangsberechtigung zu sicherheitsrelevanten Bereichen zu überprüfen, ist die Anfälligkeit für *Cloning*-Angriffe besonders kritisch. Das zeigt, dass der Zugriff auf Daten, welche nicht auslesbar sind, trotzdem möglich ist, jedoch erheblich größeren Aufwand erfordert.
Da der Standard ISO/IEC-14443 (Abschnitt 1.2) keinerlei Maßnahmen zum Schutz vor Angriffen beinhaltet, wird er oft nur als Basis für proprietäre RFID-Systeme verwendet. So basieren die von Nohl und Evans untersuchten Mifare Karten zwar auf dem Standard für kontaktlosen Chipkarten (ISO/IEC-14443), verwenden jedoch proprietäre Datenstrukturen und Sicherheitsmechanismen.
Sicherheitsmechanismen, für ISO/IEC-14443 basierte Chipkarten, werden auf Anwendungsebene implementiert. Nachdem eine standardkonforme Verbindung hergestellt wurde, werden bestimmte Sicherheitsprotokolle verwendet, um zum Beispiel die restliche Kommunikation zu verschlüsseln. Diese Sicherheitsprotokolle dienen dem Zweck, Sicherheitsbedürfnisse in bestimmten Anwendungsszenarien zu erfüllen, ohne die Vorteile kontaktloser Chipkarten, wie Robustheit und Komfort, aufzugeben. Kryptographische Verfahren, Protokolle zum Schlüsselaustausch oder zusätzliche Faktoren zur Authentisierung, wie zum Beispiel Passworteingaben, können den Komfort kontaktloser Chipkarten zu Gunsten zusätzlicher Sicherheit reduzieren. Gerade auf zusätzliche Authentisierungsfaktoren wird aus diesem Grund häufig verzichtet. Der automatische Charakter der Kommunikation zwischen Tag und Leser soll dadurch erhalten bleiben. Genau diese Eigenschaft, sowie kontaktlose Kommunikation und die beliebig langen Signallaufzeiten (siehe Abschnitt 1.2) ermöglichen Weiterleitungsangriffe.

2 Weiterleitung auf ISO/IEC-14443-4-Ebene

2.1 Szenario

Eine klassische Beschreibung eines Weiterleitungsszenarios stellt das *Grand Master Chess Problem* [Con76] dar (siehe Abb. 7). Darin wird beschrieben, wie es einer Person, auch ohne Kenntnis der Schachregeln, möglich ist, gegen einen Schachgroßmeister zu gewinnen oder zumindest ein Remis zu erreichen. Dies wird erreicht, indem der Spieler zeitgleich mit zwei unterschiedlichen Schachgroßmeistern jeweils eine Schachpartie beginnt. Dabei hat der Spieler bei einem Spiel die schwarzen und im anderen die weißen Figuren. Die beiden Schachgroßmeister wissen nichts voneinander. Der Spieler beginnt nun damit jeden Zug eines Schachgroßmeisters an den jeweiligen anderen weiterzuleiten. In Wirklichkeit spielen also die zwei Schachgroßmeister gegeneinander. In dieser Situation ist es für den Spieler unnötig die einzelnen Züge zu verstehen, um gegen einen der Schachgroßmeister zumindest ein Remis zu erreichen.
Eine Adaption dieses Angriffs, für *Challenge-Response* basierte Sicherheitsprotokolle, wurde in [DGB06] unter der Bezeichnung *Mafia Fraud* vorgestellt. Demzufolge ist der Angreifer in der Lage entsprechende Sicherheitsprotokolle zu umgehen, indem er eine Weiterleitung zwischen zwei legitimen Entitäten einrichtet. Dem Angreifer werden von einer Entität Aufgaben gestellt (*Challenge*), durch deren korrekte Beantwortung (*Response*) er sich ihr gegenüber legitimiert. Der Angreifer erhält die korrekten Antworten, indem er die Fragen der einen, an die jeweils andere Entität weiterleitet und von dieser beantworten lässt. Wie beim *Grand Master Chess Problem* wissen die beiden Entitäten nichts voneinander. Das Sicherheitsprotokoll wird auf diese Weise korrekt befolgt, und trotzdem hat sich der Angreifer am Ende gegenüber allen Beteiligten legitimiert. Der Angreifer benötigt dazu keinerlei Wissen über das zugrundeliegende Protokoll, die weitergeleiteten

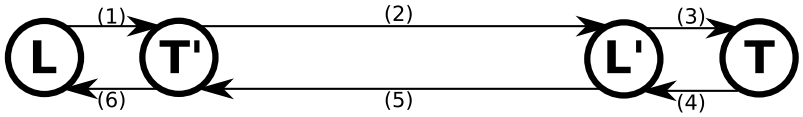

Abbildung 6: *Mafia Fraud*: Leser **L** und Tag **T** kommunizieren unwissentlich indirekt über **L'** und **T'** miteinander. **L'** und **T'** stehen unter Kontrolle des Angreifers. Ziel: **L** hält **T'** für **T** und **T** hält **L'** für **L**.

Daten, möglicherweise verwendete kryptographische Verfahren oder Schlüssel. Es muss ihm lediglich möglich sein, für die gesamte Dauer der Abarbeitung des Protokolls, eine Weiterleitung zu etablieren.
In vielen Weiterleitungsszenarien ist es wünschenswert, dass Leser und Emulator einen möglichst unauffälligen Formfaktor aufweisen. Für Angriffe im öffentlichen Raum bieten sich in diesem Zusammenhang Mobiltelefone an.
In den vergangenen Jahren hat die Integration von NFC-zertifizierter Hardware in han-

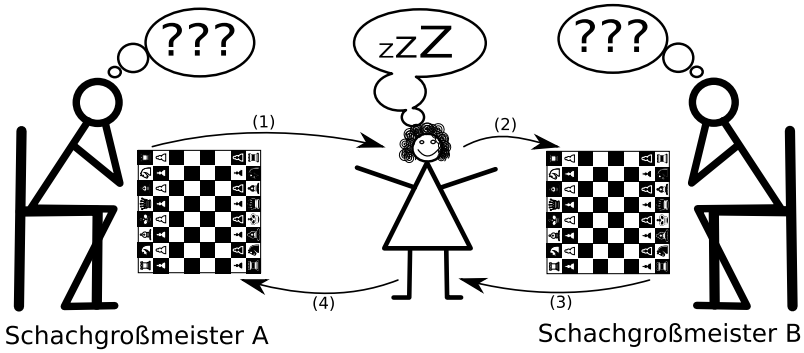

Abbildung 7: Grandmaster Chess: Da beim Schach stets der Spieler mit den weißen Figuren (hier A) beginnt, kann das Mädchen seine Züge **(1)** für ihre Partie gegen B verwenden **(2)**. Im Gegenzug kann sie B's Züge **(3)** in der Partie mit A verwenden **(4)**.

delsüblichen Mobiltelefonen stetig zugenommen. Zusätzliche Schnittstellen zu Wireless LAN, Bluetooth- oder GSM-Netzen[1] in denselben Geräten ermöglichen es die NFC-Kommunikation über diese weiter zu leiten. Die begrenzte Reichweite bestimmter NFC-Kommunikation kann dadurch stark erweitert werden.
Betrachten wir im Folgenden eine Weiterleitung der Kommunikation zwischen ISO/IEC-14443 kompatiblen Kommunikationspartnern, in Form von Tag und Lesegerät, über NFC-fähige Mobiltelefone.
Für eine Weiterleitung über zwei Mobiltelefone (*A* und *B*) ist es notwendig, dass zumindest eines der Geräte in der Lage ist einen Tag zu emulieren. Unter der Annahme, dass *A* über einen solchen Kartenemulationsmodus verfügt, kann im einfachsten Fall von einem Aufbau wie in Abbildung 8 ausgegangen werden. Die Mobiltelefone etablieren zunächst eine Verbindung über eine beliebige Schnittstelle (Bluetooth, WLAN, etc.). Im Anschluss daran wird *B*, im Schreib-/Lesemodus, in unmittelbare Nähe zur Chipkarte gebracht. *A* wird im Kartenemulationsmodus in das Feld des Lesegerätes gebracht. Das Lesegerät erkennt *A* und beginnt mit der Kommunikation. Diese Kommunikation wird von *A* über *B* an die Chipkarte weitergeleitet. Die Antwort der Chipkarte wird dann, auf dem umgekehrten Weg, zurück an den Leser übertragen. Setzt man eine stabile Datenverbindung zwischen *B* und *A* vorraus, ist die Weiterleitung von den Kommunikationspartnern nur anhand einer erhöhten Signallaufzeit wahrnehmbar.

[1] Bei dem *Global System for Mobile Communications* (GSM) handelt es sich um einen weit verbreiteten Standard für Mobilfunknetze.

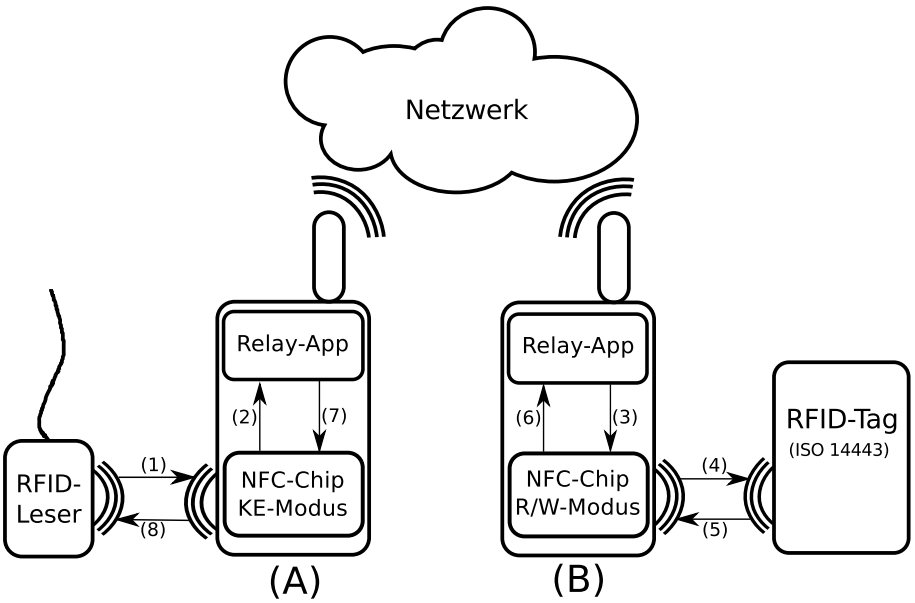

Abbildung 8: **(A)** und **(B)** stellen jeweils eine lokale NFC-Verbindung zu Tag bzw. Leser her. Über eine Netzwerkverbindung (W-Lan, Bluetooth, etc.) zwischen **(A)** und **(B)** werden dann APDUs weitergeleitet.

Für eine direkte Weiterleitung limitiert also die Entfernung von *A* und *B* zueinander, über die eine stabile Datenverbindung erzeugt werden kann, die Reichweite der Weiterleitung. Wenn also *A* und *B* über das Internet miteinander verbunden sind, können Chipkarte und Lesegerät beliebig weit voneinander entfernt sein.

2.2 Bezug zu bisherigen Arbeiten

Studien zu Weiterleitungsangriffen auf RFID-Systeme wurden bereits mehrfach durchgeführt. Gerhard Hancke hat in seiner Arbeit [Han05] eine Implementation des Weiterleitungsangriffs vorgestellt. Ziel der Arbeit war es, die Kommunikation zwischen einer kontaktlosen Chipkarte und einem Lesegerät, über eine Distanz von 50 m, weiterzuleiten. Für den Angriff verwendete er spezielle Hardware, mit der Bezeichnung *Proxy* und *Mole*, an beiden Enden der Verbindung. Der *Proxy* gab sich dem Lesegerät gegenüber als Chipkarte aus. Der *Mole* hatte die Rolle eines Lesegerätes und die Aufgabe mit der legitimen Chipkarte zu kommunizieren. *Proxy* und *Mole* waren in der Lage, via *Ultra High Frequency* (UHF) Radiowellen, miteinander zu kommunizieren. Für den gesamten Zeitraum der Weiterleitung wurde das *High Frequency* (HF) Radiosignal von Chipkarte und Lesegerät, durch *Proxy* und *Mole*, analog moduliert/demoduliert. Die dadurch erhaltene binäre '0' oder '1' wurde dann über die UHF-Verbindung weitergeleitet.
Diese Form der Weiterleitung umfasst die gesamte Kommunikation zwischen den angegriffenen Geräten. Jedes einzelne im Radiosignal kodierte Bit wird weitergeleitet. Das die

Weiterleitung bereits auf der Verbindungsebene (siehe Abb. 3) ansetzt hat einen Vorteil gegenüber anderen Verfahren. Es ist dadurch möglich alle übertragenen Daten, einschliesslich des UID, weiterzuleiten oder zu manipulieren. Um das zu erreichen ist es jedoch notwendig, die im Standard festgelegten Signallaufzeiten für Initialisierung und Antikollision einzuhalten. Die fest vorgeschreibenen maximalen Signallaufzeiten während der Antikollisionsroutine sorgen dafür, dass die Weiterleitung nicht über beliebig lange Strecken erfolgen kann. Hancke konnte in seiner Arbeit nachweisen, dass eine Entfernung von 50 m auf diese Weise überbrückt werden kann.
Eine alternative Vorgehensweise wäre es, die unteren Ebenen des ISO/IEC-14443 Protokolls, bis einschliesslich Antikollision und Initialisierung, an beiden Enden der Verbindung lokal abzuarbeiten und im Anschluss daran die Anwendungsdaten (APDUs) weiterzuleiten (siehe Abb. 8). Durch die lokale Abarbeitung der Initialisierungs- und Antikollisionsroutine werden die vorgeschriebenen maximalen Signallaufzeiten nicht überschritten. Eine Weiterleitung auf Anwendungsebene wurde in einer Arbeit von Francis et al. [FHMM10] mit Mobiltelefonen durchgeführt. Darin geht es um die Weiterleitung einer P2P Verbindung zwischen zwei Mobiltelefonen. Der Angriff wurde dabei durch zwei manipulierte, NFC-fähige, Mobiltelefone (*Proxys*) ermöglicht, die ihrerseits über Bluetooth [Spe01] verbunden waren. Francis et al. haben in ihrer Arbeit frei verfügbare Java APIs und handelsübliche Mobiltelefone verwendet, um die für die Weiterleitung verwendeten Proxys vorzubereiten.
Im Rahmen dieser Arbeit wurde ebenfalls ein Angriff auf Anwendungsebene durchgeführt. Wie in der Arbeit von Francis et al. wurden zwei handelsübliche Mobiltelefone als Proxys verwendet. Das Ziel des Angriffs war die Weiterleitung der Kommunikation zwischen einem Lesegerät und einem Tag. Die Proxys wurden indirekt, über einen Server, miteinander verbunden. Der Server hatte die Aufgabe die Daten zwischen den Proxys weiterzuleiten und zu protokollieren. Desweiteren konnten durch eine serverseitige Verzögerung der Weiterleitung beliebig lange Signallaufzeiten erzeugt werden.

2.3 Implementationsbeschreibung

In dieser Arbeit werden zwei Smartphones, mit dem Betriebssystem Android, für die Weiterleitung von ISO/IEC-14443-4 basierter Kommunikation verwendet. Bei Android handelt es sich um ein, maßgeblich von Google entwickeltes, Open Source Betriebssystem für mobile Geräte. Durch das zur Verfügung stehende *Software Development Kit*[2] (SDK) ist es möglich Programme für diese Plattform zu entwickeln. Ein weiterer Vorteil, dieses offenen Betriebssystems, ist die Verfügbarkeit verschiedenster *Distributionen* auf Basis von Android. Dadurch ist man nicht von Betriebssystemupdates des jeweiligen Herstellers abhängig. Man kann auf eine möglichst aktuelle Portierung Dritter zurückgreifen oder selber gewünschte Funktionalitäten hinzufügen.

[2] Das SDK ist unter `http://developer.android.com` erhältlich.

Der Versuchsaufbau besteht aus den in Abb. 8 zu sehenden Komponenten und einem Relay-Server. Auf jedem der beiden Smartphones läuft eine Anwendung, welche dafür sorgt, dass die Daten der NFC-Schnittstelle an den Server weitergeleitet werden und umgekehrt. Die Aufgabe des Servers ist es, die Daten zwischen den Smartphones weiterzuleiten und zu protokollieren.
Als RFID-Leser wird das Modell „cyber *Jack*® RFID basis" der Firma *ReinerSCT*[3] verwendet. Der zu lesende RFID-Tag war zum einen ein elektronischer Reisepass (ePass), zum anderen ein neuer Personalausweis (nPA) der Bundesrepublik Deutschland. Bei den verwendeten Smartphones handelt es sich um ein *Galaxy Nexus* (GT-I9250) sowie ein *Galaxy S3* (GT-I9300) der Firma Samsung. Beide Geräte verwenden den gleichen NFC-Controller (NXP PN544). Der NFC-Controller unterstützt die Verbindungsmodi Peer-to-Peer, Schreib-/Lese-Modus und Kartenemulation. Da es mit der vorinstallierten Android Version 4.0.4 nicht möglich war den Kartenemulationsmodus zu nutzen, musste, für wenigstens eines der Geräte, eine alternative Android Distribution verwendet werden. *CyanogenMod*[4] ist eine solche Distribution auf Android Basis und wurde aufgrund der integrierten Unterstützung des Kartenemulationsmodus, in der Version 9.1, für das Samsung Galaxy S3 benutzt.

Für die Weiterleitung wurde sowohl für das Nexus, als auch das S3 eine Applikation (App) geschrieben. Das Nexus hatte die Aufgabe als Lesegerät mit einem Tag zu kommunizieren. Das S3 hatte die Aufgabe im Kartenemulationsmodus mit einem Lesegerät zu kommunizieren. Sowohl die Reader-App des Nexus, als auch die Emulator-App des S3 stellen eine TCP/IP-Verbindung zum Relay-Server her. Im Anschluss daran wird der zu lesende Tag (ePass, nPA) ins Feld des Nexus gebracht und das S3 auf den „cyber *Jack*® RFID basis"-Leser gelegt. Daraufhin kann damit begonnen werden, über den RFID-Leser Daten an den Tag zu schicken. Im folgenden soll näher darauf eingegangen werden, welche Schritte unter Android zur Nutzung der NFC-Schnittstelle notwendig sind.

Das Betriebssystem Android erlaubt nur eventbasierten Zugriff auf die NFC-Schnittstelle. Eine App muss sich beim Betriebssystem für bestimmte NFC-Events (ACTION_NDEF_-DISCOVERED, ACTION_TECH_DISCOVERED, ACTION_TAG_DISCOVERED), durch einen Eintrag in ihrem Manifest, registrieren. Die App *nfcreader* des Nexus kann ihre MainActivity, durch folgenden Eintrag in ihr Manifest, für das Event ACTION_TECH_DISCOVERED registrieren.

```
//AndroidManifest.xml
<activity
   android:name="com.example.nfcreader.MainActivity"
   android:label="@string/app_name"
   android:launchMode="singleTop" >
   <intent-filter>
      <action android:name="android.nfc.action.ACTION_TECH_DISCOVERED"/>
```

[3] http://www.reiner-sct.com
[4] http://www.cyanogenmod.org

```
        <category android:name="android.intent.category.DEFAULT"/>
      </intent-filter>
      <meta-data android:name="android.nfc.action.TAG_DISCOVERED"
          android:resource="@xml/filter_nfc"/>
   </activity>
```

Listing 1: Die MainActivity der nfcreader-App registriert sich für ein NFC-Event.

Die Events können zusätzlich nach Tag-Technologien (MifareClassic, IsoDep, Ndef, u.a.) gefiltert werden. Der folgende Ausschnitt, aus der Datei filter_nfc.xml (Listing 2), zeigt zum Beispiel, dass sich die nfcreader-App nur für die Tag-Technologien IsoDep, NfcA und NfcB registriert. NfcA, NfcB und IsoDep sind Implementationen der *TagTechnology*-Klasse und stellen eine Schnittstelle zu ISO/IEC-14443-3 Typ A, ISO/IEC-14443-3 Typ B und ISO/IEC-14443-4 konformen Tags dar. Die *TagTechnology*-Klasse stellt lediglich elementare Funktionen (`connect()`,`close()`,`isConnected()`,`getTag()`) zum Verbindungsmanagement bereit. Technologiespezifische Funktionen wie zum Beispiel `transcieve()`, `getAtqa()`, `getBlockCount()` oder `getMaxTranscieveLength()` werden dann von den jeweiligen implementierenden Klassen bereitgestellt.

```
   //filter_nfc.xml
   <resources>
      <tech-list>
         <tech>android.nfc.tech.IsoDep</tech>
         <tech>android.nfc.tech.NfcA</tech>
         <tech>android.nfc.tech.NfcB</tech>
      </tech-list>
   </resources>
```

Listing 2: Liste der für die nfcreader-App interessanten Tag-Typen.

Der auf dem S3 verwendete CyanogenMod ergänzt die Liste der *TagTechnology*-Implementationen um zwei weitere, mit der Bezeichnung IsoPcdA und IsoPcdB. Diese stellen Schnittstellen zur Kommunikation mit ISO/IEC-14443-4 konformen Lesegeräten dar. IsoPcdA und IsoPcdB erweitern die Klasse `BasicTagTechnology`, welche Funktionen zur Verbindungsverwaltung (`connect()`, `close()`, u.a.) und Datenübertragung (`transcieve()`) beinhaltet.

```
abstract class BasicTagTechnology implements TagTechnology {
     public boolean isConnected() {...}
     public void connect() throws IOException {...}
     public void reconnect() throws IOException {...}
     public void close() throws IOException {...}
    byte[] transceive(byte[] data, boolean raw) throws IOException {...}
}
```

Listing 3: Interface der BasicTagTechnology Klasse. IsoPcdA und IsoPcdB erweitern die Klasse unter anderem um eine `public transcieve()`-Funktion.

Neben der Registrierung für bestimmte Events und den Vorkehrungen zum Filtern dieser, muss ebenfalls berücksichtigt werden, dass sich durchaus mehrere Apps für das selbe Event registrieren lassen können. Haben sich für ein NFC-Event mehrere Applikationen registriert, muss der Benutzer manuell eine Auswahl treffen. Um eine manuelle Nutzerinteraktion im Konfliktfall zu verhindern, kann das *Foreground Dispatch System* (Listing 4) durch eine Applikation verwendet werden. Diese Applikation wird dann, in Bezug auf das NFC-Event, gegenüber allen anderen priorisiert.

```
public class MainActivity extends Activity {

   public void onCreate(Bundle savedInstanceState) {
    pendingIntent = PendingIntent.getActivity(this, 0, new Intent(this,
              getClass()).addFlags(Intent.FLAG_ACTIVITY_SINGLE_TOP), 0);
    filters = new IntentFilter[] { new IntentFilter(
              NfcAdapter.ACTION_TECH_DISCOVERED) };
    techLists = new String[][] { { "android.nfc.tech.NfcA",
        "android.nfc.tech.NfcB", "android.nfc.tech.IsoDep"} };
  }

   public void onResume() {
    super.onResume();
    if (adapter != null) {
      adapter.enableForegroundDispatch(this, pendingIntent, filters,
                techLists);
    }
  }

   public void onPause() {
    super.onPause();
    if (adapter != null) {
      adapter.disableForegroundDispatch(this);
    }
  }
}
```

Listing 4: Für den *Foreground Dispatch* werden die zuvor eingerichteten Event- und Technologie-Filter der jeweiligen Applikation verwendet. Der entsprechende Eintrag in `onResume()` und `onPause()` sorgt dafür, dass nur eine aktuell ausgeführte App priorisiert werden kann.

Auf diese Weise kann sichergestellt werden, dass die gewünschten NFC-Events an die

richtige App weitergeleitet werden. Sobald ein entsprechendes Event stattfindet, erhält die App über die `onNewIntent()`-Funktion ihrer MainActivity[5] Zugriff auf das Tag-Objekt.

```
public void onNewIntent(Intent intent) {

   Tag intag = intent.getParcelableExtra(NfcAdapter.EXTRA_TAG);
   [...]
   IsoDep mf = IsoDep.get(intag);
   [...]
}
```

Listing 5: Zugriff auf ein IsoDep kompatiblen Tag über die `onNewIntent()`-Funktion

Die App kann dann, mit der `getTechList()`-Funktion des Tag-Objekts, ermitteln welche Technologie (NfcA, IsoDep, etc.) von dem Tag unterstützt wird. Über ein Objekt der gewählten Technologie findet dann jede weitere Kommunikation statt.

Für die Emulator-App des S3 gibt es zusätzlich ein paar weitere Punkte zu beachten. Da weder IsoPcdA, noch IsoPcdB oder die BasicTagTechnology-Klasse Teil des offiziellen Android SDK sind, muss die App gegen die CyanogenMod-Quellen gebaut werden. Eine andere Möglichkeit ist die Verwendung einer *Wrapper*-Klasse, mit dem offiziellen Android SDK. Eine Wrapper-Klasse[6] fungiert in diesem Zusammenhang, als Platzhalter für die nicht vorhandene IsoPcdA- oder IsoPcdB-Klasse.

```
public class TagWrapper implements TagTechnology {

    private Method isConnected;
    private Method connect;
    private Method reconnect;
    private Method close;
    private Method getMaxTransceiveLength;
    private Method transceive;

    private Tag tag;
    private Object tagTech;

    public TagWrapper(Tag tag, String tech) {
           this.tag = tag;

           Class<?> cls = Class.forName(tech);
           Method get = cls.getMethod("get", Tag.class);
           tagTech = get.invoke(null, tag);
```

[5] Eine App kann aus verschiedenen Komponenten bestehen. Unter anderem aus sogenannten Activities. Weitere Informationen finden sich unter `http://developer.android.com/guide/index.html`.

[6] Wrapper ist Englisch für Hülle oder Umschlag.

```
            isConnected = cls.getMethod("isConnected");
            connect = cls.getMethod("connect");
            reconnect = cls.getMethod("reconnect");
            close = cls.getMethod("close");
            getMaxTransceiveLength = cls.getMethod("getMaxTransceiveLength");
            transceive = cls.getMethod("transceive", byte[].class);
    }
      public void connect() throws IOException {...}
      public byte[] transceive(byte[] data) throws IOException {...}
      [...]
}
```

Listing 6: Auszug aus einer Wrapper-Klasse für IsoPcdA/IsoPcdB[7].

Um zum Beispiel ein IsoPcdA-Objekt zu erhalten, wird zur Laufzeit dem Konstruktor der TagWrapper-Klasse sowohl das Tag-Objekt, als auch der String "android.nfc.tech.IsoPcdA" übergeben. Falls eine, zum String passende, Klasse und deren Funktionen in der Laufzeitumgebung gefunden wurden, werden diese über die *public*-Funktionen der TagWrapper-Klasse zur Verfügung gestellt. Anstelle eines IsoPcdA-Objekts, kann also zunächst ein TagWrapper-Objekt verwendet werden. Dadurch muss die Tag-Technologie "android.nfc.tech.-IsoPcdA" lediglich zur Laufzeit bekannt sein.
Über das IsoPcdA-Objekt kann eine Verbindung mit dem Leser hergestellt (`connect()`) werden. Die APDU-Kommandos des Lesers werden nicht automatisch an die Applikation weitergereicht. Daten vom Leser können ausschliesslich als Rückgabewert der `transceive()`-Funktion empfangen werden. Das bedeutet, dass zu Beginn einer Verbindung zunächst Daten an den Leser geschickt werden müssen, um das erste APDU-Kommando des Lesers zu erhalten.

```
byte[] readerAPDU = transceive(new byte[]{(byte) 0x90, 0x00});
```

Listing 7: Nach dem Senden des Hexdezimalwertes "9000" enthält `readerAPDU` das erste APDU-Kommando des Lesers.

Die `readerAPDU` wird dann an den Tag weitergeleitet, welcher wiederum mit einer `tagAPDU` antwortet. Im folgenden findet die Kommunikation mit dem Leser, auf der Seite der Emulator-App, über den folgenden Aufruf statt.

```
byte[] readerAPDU = transceive(tagAPDU);
```

[7] Quelle: `https://github.com/nelenkov/virtual-pki-card/blob/master/se-emulator/src/org/nick/se/emulator/TagWrapper.java`

3 Evaluation

Um die Weiterleitung zu testen, wurden zwei unterschiedliche Szenarien betrachtet. Im ersten Szenario sollte ein ePass über eine große Entfernung ausgelesen werden. Dabei beschränkt sich das Auslesen auf den Teil der Daten, welche ausschliesslich durch die *Basic Access Control* (BAC) geschützt sind. BAC soll das ungewollte Auslesen des ePass verhindern. Zu diesem Zweck wird die *maschinenlesbare Zone* (MRZ) des Dokuments gedruckt, welche dem Leser bekannt sein muss, um Zugriff auf die Daten zu bekommen. Das soll sicher stellen, dass ein Angreifer physischen Zugriff auf das Dokument haben muss, da die MRZ nur optisch erfasst werden kann. Bei der MRZ handelt es sich jedoch um ein statisches Merkmal. Es verliert seinen sicherheitsrelevanten Nutzen, sobald es einmal bekannt geworden ist. Nähere Informationen, zu Spezifikation und Limitierung des BAC-Protokolls, können den technischen Richtlinien des Bundesamt für Sicherheit in der Informationstechnik (BSI) [BSI12] entnommen werden. Im Kontext dieses Testszenarios gilt die MRZ als bekannt. Das Programm *cyberflex-shell*[1] von Henryk Plötz kommuniziert über das Lesegerät mit dem ePass. Es sorgt dafür, dass korrekte APDU-Kommandos an den ePass geschickt werden und gibt die übermittelten Daten aus. Da im Verlauf der Abarbeitung des BAC-Protokolls der ePass einige kryptographische Berechnungen durchzuführen hat, musste die Reader-App länger auf eine Antwort warten. Das führte dazu, dass es zu einer Zeitüberschreitung (engl. Timeout) kam, wodurch die Verbindung zum ePass beendet wurde. Die Lösung bestand darin den Timeout mit der Funktion `public void setTimeout (int timeout)` des IsoDep-Objektes zu erhöhen. Der ePass konnte somit ausgelesen werden. In einem weiteren Schritt sollte demonstriert werden, dass die Weiterleitung über sehr große Entfernungen durchgeführt werden kann. Da sich eine größere Entfernung zwischen Tag und Leser bei einer Weiterleitung lediglich durch eine erhöhte Signallaufzeiten äußert, wurde diese durch eine zusätzliche Verzögerung im Relay-Server simuliert. Zwischen dem Empfang einer APDU durch den Relay-Server und ihrer Weiterleitung, wurde zusätzliche 5 Sekunden gewartet. Da die Daten für beide Richtungen der wechselseitigen Kommunikation verzögert wurden, musste die cyberflex-shell mehr als 10 Sekunden auf eine Antwort vom ePass warten. Trotz der erhöhten Signallaufzeit wurde der ePass problemlos ausgelesen.

Im zweiten Szenario wurde die Kommunikation zwischen dem *neuen Personalausweis* (nPA) und der offiziellen *AusweisApp*[2] (Version 1.1) des BSI weitergeleitet. Auch in diesem Fall wurde sich auf bestimmte Daten beschränkt. Für bestimmte Anwendungen des nPA

[1] Quelle: `https://github.com/henryk/cyberflex-shell`

[2] Quelle:`https://www.ausweisapp.bund.de/pweb/cms/links/download.jsp`

ist die Möglichkeit *Extended Length APDUs* zu Senden und zu Empfangen notwendig. Da weder das Nexus, noch das S3 dazu in der Lage sind, konnte nur Kommunikation weitergeleitet werden, die über APDUs normaler Länge verläuft. Die Änderung der *persönlichen Identifikationsnummer* (PIN) des nPA über die AusweisApp erfordert keine Extended Length APDUs und diente daher als weiterzuleitende Beispielanwendung. Die Weiterleitung funktionierte ebenso wie beim ePass problemlos. Um festzustellen ob eine erhöhte Signallaufzeit erkannt wird, und die Kommunikation daraufhin abgebrochen wird, wurde die APDU-Übertragung wie im ersten Szenario verzögert. Trotzdem es sich bei der PIN-Änderung um eine Anwendung handelt die Nutzereingaben erfordert, sind Beschränkungen der Signallaufzeit durchaus möglich. So könnte zum Beispiel die Kommunikation vor und nach der Nutzereingabe dazu verwendet werden um erhöhte Signallaufzeiten zu ermitteln. Es stellte sich herraus, dass die Verzögerung der APDU-Übertragung keinen Einfluss auf die Beispielanwendung hatte. Die PIN-Änderung konnte, auch mit der Verzögerung, problemlos durchgeführt werden.

In beiden Szenarien konnte eine, für Tag (ePass, nPA) und Leser (cyberflex-shell, AusweisAPP) transparente, Weiterleitung erfolgreich durchgeführt werden. Die Erhöhung der Signallaufzeit zeigt, dass eine solche Weiterleitung, praktisch über beliebige Entfernungen, durchgeführt werden kann. Bei 5 Sekunden Signallaufzeit zwischen Tag und Leser, sowie einer Übertragung in Lichtgeschwindigkeit, ergibt sich eine maximale Entfernung von ca. 1500000 km.[3] Diese Entfernung entspricht in etwa dem 235-fachen des Erdradius.[4]

[3] Lichtgeschwindigkeit = 299792458 $\frac{m}{s}$

[4] Der Erdradius beträgt ca. 6371 km.

4 Fazit

In dieser Arbeit wurde gezeigt, dass eine Weiterleitung von ISO/IEC-14443-4 basierter Kommunikation, mit handelsüblichen Mobiltelefonen, über eine sehr große Entfernung, eingerichtet werden kann. Die Verwendung von Android Mobiltelefonen ermöglichte jedoch lediglich die Weiterleitung von Anwendungsdaten (APDUs). Verbindungsparameter und UID werden zwischen Mobiltelefon und Tag, bzw. zwischen Mobiltelefon und Leser, ohne Beteiligung der Applikation ausgetauscht. Auf diese kann daher kein Einfluss genommen werden. Bei jedem Verbindungsaufbau über die NFC-Schnittstelle wird, vom NFC-Chipsatz, eine zufällig generierte UID übermittelt. Das hat zur Folge, dass Weiterleitungen unmöglich gemacht werden, bei denen die UID des Mobiltelefons einen bestimmten Wert haben muss. Wie in der Arbeit von Hancke [Han05] zu sehen, kann diese Einschränkung umgangen werden, indem die Weiterleitung bereits auf Verbindungsebene etabliert wird. Basierend auf einer freiwilligen Selbstverpflichtung der Hersteller von NFC-Chipsätzen ist das Festlegen einer UID durch den Nutzer jedoch nicht ohne weiteres möglich. Es zeigte sich also keine prinzipielle Schwäche des Weiterleitungsangriffes, sondern vielmehr ein Mangel an günstiger geeigneter Hardware. Entsprechende Hardware vorrausgesetzt, währe auch eine Weiterleitung anderer Tag-Formate möglich.
Um eine Weiterleitung auf Anwendungsebene zu bemerken, ist es nötig auf die Signallaufzeiten zu achten. In einem ersten Ansatz könnte ein Grenzwert für die Signallaufzeit verwendet werden. Der Leser würde die Zeit zwischen dem Senden eines Kommandos und dem Empfang der Antwort messen und beim Überschreiten des Grenzwertes die Verbindung beenden. Jedoch ist ein statischer Grenzwert für die Signallaufzeit nur schwer festzulegen und in vielen Anwendungsbereichen unzureichend. Aus der Verwundbarkeit von RFID-Systemen, gegenüber Weiterleitungsangriffen und der Unzulänglichkeit statischer Grenzwerte für die Signallaufzeit, entwickelten sich komplexere *Distance Bounding*-Protokolle. In solchen Protokollen wird, durch einen über mehrere Runden stattfindenden Datenaustausch, ermittelt ob sich ein Tag nah genug am Leser befindet, bevor mit der Übertragung der Anwendungsdaten begonnen wird. Aus der Arbeit von Kim und Avoine [KA09] geht hervor, dass sich die Wahrscheinlichkeit eine Weiterleitung zu etablieren mit jeder weiteren Runde halbieren lässt. Mit steigender Rundenzahl steigt jedoch auch der Zeitbedarf für die Protokollabarbeitung. Um *Distance Bounding*-Protokolle zu verwenden müssen sowohl Tags, als auch Leser dies, in den untersten Protokollschichten, unterstützen. Eine dahingehende Anpassung bestehender RFID-Systeme ist sehr kostenintensiv und daher in näherer Zukunft nicht zu erwarten.
Es bleibt somit festzuhalten, dass sich Tag und Leser in einem RFID-System keinesfalls in unmittelbarer Nähe zueinander befinden müssen, um miteinander zu kommunizieren.

RFID-Systeme, die sich ausschliesslich auf diese Annahme verlassen, sind gefährdet durch Weiterleitungsangriffe kompromittiert zu werden.

Literaturverzeichnis

[BSI12] BUNDESAMT FÜR SICHERHEIT IN DER INFORMATIONSTECHNIK: Technical Guideline Advanced Security Mechanisms for Machine Readable Travel Documents - Part 1. Version: März 2012. https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Publications/TechGuidelines/TR03110/TR-03110_v2.1_P1pdf.pdf. 2012 (BSI TR-03110-1). – Technische Richtlinie

[Con76] CONWAY, John H.: On numbers and games. In: *London-New York* (1976)

[DGB06] DESMEDT, Yvo ; GOUTIER, Claude ; BENGIO, Samy: Special uses and abuses of the Fiat-Shamir passport protocol. In: *Advances in Cryptology—CRYPTO'87* Springer, 2006, S. 21–39

[FHMM10] FRANCIS, Lishoy ; HANCKE, Gerhard ; MAYES, Keith ; MARKANTONAKIS, Konstantinos: Practical NFC peer-to-peer relay attack using mobile phones. In: *Radio Frequency Identification: Security and Privacy Issues*. Springer, 2010, S. 35–49

[Han05] HANCKE, Gerhard P.: A practical relay attack on ISO 14443 proximity cards. In: *Technical report, University of Cambridge Computer Laboratory* (2005)

[isoa] INTERNATIONAL ORGANISATION FOR STANDARDIZATION: Identification cards - Contactless integrated circuit(s) cards - Proximity cards - Part 1: Physical characteristics (ISO/IEC-14443-1). – ISO/IEC Standard

[isob] INTERNATIONAL ORGANISATION FOR STANDARDIZATION: Identification cards - Contactless integrated circuit(s) cards - Proximity cards - Part 2: Radio frequency power and signal interface (ISO/IEC-14443-2). – ISO/IEC Standard

[isoc] INTERNATIONAL ORGANISATION FOR STANDARDIZATION: Identification cards - contactless integrated circuit(s) cards - proximity cards - part 3: Initialization and anticollision (ISO/IEC-14443-3). – ISO/IEC Standard

[isod] INTERNATIONAL ORGANISATION FOR STANDARDIZATION: Identification cards - contactless integrated circuit(s) cards - proximity cards - part 4: Transmission protocol (ISO/IEC-14443-4). – ISO/IEC Standard

[KA09] KIM, Chong H. ; AVOINE, Gildas: RFID distance bounding protocol with mixed challenges to prevent relay attacks. In: *Cryptology and Network Security*. Springer, 2009, S. 119–133

[MRT10] MITROKOTSA, Aikaterini ; RIEBACK, MelanieR. ; TANENBAUM, AndrewS.: Classifying RFID attacks and defenses. In: *Information Systems Frontiers* 12 (2010), Nr. 5, 491-505. http://dx.doi.org/10.1007/s10796-009-9210-z. – DOI 10.1007/s10796–009–9210–z. – ISSN 1387–3326

[NE08] NOHL, Karsten ; EVANS, David: Reverse-Engineering a Cryptographic RFID Tag. In: *USENIX Security Symposium*, 2008, S. 185–194

[Plö08] PLÖTZ, Henryk: Mifare Classic-Eine Analyse der Implementierung. In: *Diplomarbeit, Humboldt-Universität zu Berlin* (2008)

[Rol12] ROLAND, Michael: Software card emulation in NFC-enabled mobile phones: Great advantage or security nightmare. In: *Fourth International Workshop on Security and Privacy in Spontaneous Interaction and Mobile Phone Use (IWSSI/SPMU 2012)*, 2012, S. 6

[Spe01] SPECIFICATION, Bluetooth: Version 1.1. In: *Order* (2001), Nr. 242016-001

[Wei] WEIS, Stephen A.: *RFID (Radio Frequency Identification): Principles and Applications*. http://www.eecs.harvard.edu/cs199r/readings/rfid-article.pdf, . – [Online; Stand: 21.01.2014]

[Wei10] WEISS, Michael: *Performing Relay Attacks on ISO 14443 Contactless Smart Cards using NFC Mobile Equipment*, Technische Universität München, Master's thesis, Mai 2010. http://www.sec.in.tum.de/assets/studentwork/finished/Weiss2010.pdf. – [Online; Stand: 28.01.2014]