Man-in-the-Middle: Analyse des Datenverkehrs bei NFC-Zahlungen

Das Bezahlen mit kontaktlosen Kreditkarten liegt im Trend, insbesondere seit grosse Ladenketten wie Migros, Coop oder Valora diese Bezahlmöglichkeit unterstützen. Mit Google Wallet, Apple Pay und Tapit von Swisscom besteht vermehrt auch die Möglichkeit mittels Smartphone kontaktlose Zahlungen auszulösen. Wir haben die Daten, die zwischen einem Terminal und einer Karte bzw. einem Mobiltelefon ausgetauscht werden, bei einer echten Bezahlung aufgezeichnet. In diesem Artikel beschreiben wir dieses Protokoll und die Software, die nötig ist, um solche Daten aufzuzeichnen.

Christof Arnosti, Dominik Gruntz | dominik.gruntz@fhnw.ch

Immer mehr Schweizerinnen und Schweizer zahlen kontaktlos. Bei diesen Transaktionen wird die Karte an das Terminal gehalten. Bereits nach weniger als einer Sekunde ist der Vorgang abgeschlossen, da bei Beträgen unter 40 Franken keine PIN eingegeben werden muss. In den folgenden Kapiteln werden wir zeigen, wie eine Transaktion zwischen kontaktloser Kreditkarte und Terminal abläuft und insbesondere welche Daten dabei ausgetauscht werden. Dazu haben wir eine kleine Android-Applikation geschrieben, die es erlaubt, die über NFC übertragenen Daten zu protokollieren. Diese Applikation kann auch verwendet werden, um mit einem Relay-Angriff unerlaubte Zahlungen durchzuführen. In einem abschliessenden Kapitel gehen wir auf die Sicherheitsrisiken ein, die durch diese modernen Zahlungsmöglichkeiten entstehen.

Europay, MasterCard und Visa

Der Standard, der hinter dem Bezahlen mit modernen Kredit- und Bankkarten steht, heisst Europay, MasterCard and Visa (EMV). Er wird von der Firma EMVCo LLC betreut, die zu gleichen Teilen den sechs Kreditkartenunternehmen American Express, Discover, JCB, Mastercard, UnionPay und Visa gehört. Im Gegensatz zu früheren Kartentypen mit Magnetstreifen wird bei EMV auf der Karte ein Chip verwendet, der Berechnungen durchführen und Daten speichern kann. Zwischen dem Kartenleser (auch *Point of Sale*, POS) und der Karte (auch *Integrated Circuit Card*, ICC genannt) wird eine serielle Verbindung aufgebaut, über die der Leser Anfragen sendet, die von der Karte beantwortet werden. Diese in der EMV-Spezifikation definierte serielle Verbindung kann entweder über eine Kontaktfläche oder per NFC aufgebaut werden.

Implementiert wird der EMV-Standard von verschiedenen Firmen unter verschiedenen Namen. Im europäischen Markt sind auf der Karten-Seite *Visa Smart Debit Card* (VSDC) und *M/Chip* von Mastercard wichtig. Für die verschiedenen Verbindungstypen werden wiederum Marktnamen vergeben: *Chip&PIN* ist die gängige Bezeichnung für EMV-Transaktionen, deren Verwendungskonzept auf dem Einführen der Karte in einen Kartenleser und das Bestätigen der Karteninhaberanwesenheit durch eine PIN-Eingabe basiert. *PayPass* und *PayWave* sind die Markennamen von MasterCard und Visa für das kontaktlose bezahlen.

Grundsätzlich ist eine Bankkarte nichts weiter als eine SmartCard. SmartCards werden auch für diverse andere Aufgaben verwendet, zum Beispiel als SIM-Karten im Mobilfunk, als Zutrittskarten für Gebäude oder zum Speichern von kryptografischen Geheimnissen. Moderne SmartCards haben ein installiertes Betriebssystem wie z.B. JavaCard. Diese ressourcenschonende Java-Version enthält eine abgespeckte Standard-Library, die vor allem kryptografische Funktionen, Funktionen für die Kommunikation über die Chip-Schnittstelle und über NFC sowie Funktionen für spezielle Persistenz- und Transaktionsoperationen anbietet. Bei der Ausführung von Applets auf der SmartCard gilt, dass das Applet bei der Installation einmalig initialisiert wird und anschliessend alle nicht speziell im flüchtigen Speicher angelegten Felder automatisch persistiert werden. Bei einem Stromunterbruch – der bei jedem Entfernen der Karte aus dem Kartenslot oder aus dem NFC-Feld stattfindet – werden die Daten gespeichert.

SmartCard-Kommunikation

Das EMV-Protokoll basiert auf der standardisierten SmartCard-Kommunikation (ISO 7816 für kontaktbehaftete und ISO 14443 für kontaktlose Kommunikation). Das serielle Protokoll arbeitet mit Application Protocol Data Units (APDUs), wobei Command-APDUs vom Leser an die Karte gesendet werden, welche die Karte mit Response-APDUs beantwortet.

CLA	INS	P1	P2	Lc	Data	Le
Header				Body (Optional)		

Data	SW1	SW2
Body (Optional)	Trailer	

Abbildung 1: Command-APDU (oben), Response-APDU (unten)

Eine Command-APDU (Abb. 1 oben) besteht aus einem Header und einem Body. Im Header sind die Befehlsklasse (CLA), die Instruktion (INS) sowie zwei Parameter (P1, P2) vorhanden, wobei jedes der vier Felder ein Byte lang ist. Der Body ist von variabler Länge, und enthält drei Felder: Die zusammengehörenden Kommandolänge (Lc, ein Byte) und Kommandodaten (Data, von der durch Lc vorgegebenen Länge) sowie die erwartete maximale Länge der Antwortdaten (Le, ein Byte 00 bedeutet maximal 256 Bytes in der Antwort). Die Kommandolänge und der Data-Abschnitt zusammen sowie die erwartete Antwortlänge sind optional, so dass sich vier verschiedene Formate ergeben, die als Case 1 bis Case 4-Commands bezeichnet werden.

Eine Response-APDU (Abb. 1 unten) besteht aus einem optionalen Body sowie einem obligatorischen Trailer. Der Trailer enthält zwei Status-Bytes (SW1, SW2). Diese zeigen auf, ob das Kommando erfolgreich abgeschlossen wurde. Im Misserfolgsfall werden Details zum Fehler in diesen Feldern codiert, unter anderem die Information ob der nichtflüchtige Speicher auf der Karte verändert wurde. Im Body können beliebige Daten übertragen werden.

EMV-Protokoll

Eine EMV-Transaktion besteht aus vielen Teilschritten. Im Folgenden beschreiben wir jene Schritte eines normalen Bezahlvorgangs, die eine Kommunikation mit der Karte beinhalten:

- Application Selection
- Initiate Application Processing
- Read Application Data
- Offline Data Authentication

In jedem dieser Schritte werden eine oder mehrere Command-APDUs vom Terminal an die Karte geschickt und von dieser mit einer Response-APDU beantwortet. Die Antwort wird im Body der Response-APDU im Format *Basic Encoding Rules Tag-Length-Value* (BER-TLV, ITU-T X.690) abgelegt. TLV ist ein Format, in dem jeder Eintrag aus einem Tag (Bei EMV ein bis zwei Byte), welches das Datenformat sowie die Bedeutung der Daten beschreibt, seiner Länge sowie den eigentlichen Daten besteht. Diese Daten können wiederum im Format TLV codiert sein, d.h. die TLV-Codierung kann rekursiv verwendet werden.

Das Protokoll wird hier anhand der Anfragen und Antworten erklärt, die bei der Verwendung einer Prepaid-MasterCard als kontaktloses Zahlungsmittel bei einem Detailhändler aufgezeichnet worden sind (Abb. 2). Für jeden Schritt werden neben einer Erklärung jeweils die Anfrage als Hex-codierte Byte-Zeichenfolge und die Antwort als TLV-Decodierte Datenstruktur angegeben. Grosse Hilfen bei der Protokollanalyse sind die beiden Webseiten EMV-Lab [ELB], auf der ein Werkzeug zum decodieren von TLV-Datenstrukturen enthalten ist, sowie TVR-Decoder [TVR], ein Tool zur Decodierung von EMV-Datentypen.

Application Selection

Im ersten Schritt der Kommunikation wird durch das Terminal zuerst das *Payment System Environment* (PSE) Applet selektiert. Wenn ein Applet selektiert wird, dann ist dieses Programm für alle weiteren Anfragen zuständig, bis entweder ein anderes Applet selektiert oder die Karte vom Strom getrennt wird. Dieser Selektionsbefehl sieht immer gleich aus (Listing 1) und enthält neben dem Selektionsbefehl (00A40400) im Body die Application-ID des PSE-Applets (2PAY.SYS.DDF01, wenn diese ID als ASCII-Zeichenkette interpretiert wird). Das PSE-Applet antwortet auf den Selektionsbefehl mit einer Liste von vorhandenen EMV-Applets. Pro Eintrag sind mindestens eine Application-ID, eine Priorität sowie ein sprechender Name enthalten. Im Beispiel ist bloss der Eintrag für ein MasterCard-Applet enthalten.

Mit einer zweiten Command-APDU wird nun das gewünschte EMV-Applet selektiert (Listing 2). Falls das PSE-Applet mehrere EMV-Applets zur Wahl anbietet, so erfolgt die Wahl des zu verwen-

Abbildung 2: Der Bezahlvorgang beim Detailhändler ist erfolgreich, als letzter Log-Eintrag wird die kryptografische Bestätigung durch die Karte angezeigt.

```
Anfrage:
00A40400 0E 325041592E5359532E4444463031 00

Antwort:
6F  File Control Information (FCI) Template
|  84  Dedicated File (DF) Name : Name des aufge-
|  |      rufenen Applet, stimmt mit der Anfrage überein
|  |  325041592E5359532E4444463031
|  A5  FCI Proprietary Template
|  |  BF0C  FCI Issuer Discretionary Data
|  |  |  61  Application Template
|  |  |  |  4F  Application Identifier (AID) – card
|  |  |  |  |  A0000000041010
|  |  |  |  87  Application Priority Indicator
|  |  |  |  |  1
|  |  |  |  50  Application Label
|  |  |  |  |  M a s t e r C a r d
```

Listing 1: Selektion des Payment System Environment Applets

denden EMV-Applets entweder automatisch über die Priorität oder durch den Benutzer am Terminal. Neben den Daten, die bereits in der ersten Antwort zurückgegeben worden sind, wird bei dieser zweiten Antwort zusätzlich noch die auf der Karte gespeicherte Sprachpriorisierung zurückgeliefert.

Initiate Application Processing

Im zweiten Schritt *Initiate Application Processing* wird die eigentliche Transaktion mit dem selektierten EMV-Applet gestartet. Das Terminal frägt dazu zuerst die Prozessoptionen der Karte ab (Listing 3). Falls die Karte im letzten Schritt in den *File Control Information* (FCI) bereits Optionen definiert hat, die vom Terminal abgefragt werden sollen, so werden diese Daten als Body der Anfrage an die Karte gesendet, andernfalls (wie in unserem Fall) wird der Platzhalter 8300 gesendet.

Die Antwort enthält das *Application Interchange Profile* (AIP) und den *Application File Locator* (AFL). In ersterem sind in einer Bitmaske codiert die Fähigkeiten der EMV-Implementierung auf der Karte enthalten; in unserem Fall, dass die Karte *Cardholder Verification* und *Terminal Risiko-Management* unterstützt, sowohl mittels Chip als auch über den Magnetstreifen verwendet werden kann, und was für Datenverifizierungsmodi implementiert sind. *Terminal Risiko-Management* bedeutet, dass das Terminal Einschränkungen bei der Transaktion machen darf, z.B. einen Maximalbetrag festlegen oder bei zufällig ausgewählten Transaktionen eine Online-Verifizierung erzwingen darf. Die *Cardholder Verification* ist der Vorgang, dass die Karte bei der Transaktion bestimmen kann, wie der Kartenbesitzer seine Anwesenheit bestätigen muss, z.B. per Unterschrift oder PIN-Eingabe. Im *Application File Locator* sind in einer EMV-spezifischen Kodierung die Speicherorte angegeben, an der die für die Transaktion benötigten Daten zu finden sind.

```
Anfrage: 00A40400 07 A0000000041010 00

Antwort:
6F  File Control Information (FCI) Template
|  84  Dedicated File (DF) Name
|  |  A0000000041010
|  A5  FCI Proprietary Template
|  |  50  Application Label
|  |  |  M a s t e r C a r d
|  |  87  Application Priority Indicator
|  |  |  1
|  |  5F2D  Language Preference
|  |  |  e n d e f r i t
```

Listing 2: Selektion des EMV Applets

Read Application Data

Im Schritt *Read Application Data* werden diejenigen Datensätze ausgelesen, die im AFL angegeben sind. Dieser datenaufwändige Schritt wird in Listing 4 nur auszugsweise dargestellt. Für das Auslesen der Datensätze werden mehrere Anfragen nach dem Schema 00B2XXYY00 durchgeführt, wobei das Kommando B2 als Read Record interpretiert wird. Die beiden Bytes XX und YY bezeichnen den zu lesenden Datensatz.

Die Antworten sind wiederum TLV-Codiert. Die erste Hierarchieebene wird mit dem Tag 70 *(EMV Proprietary Template)* markiert, darunter finden sich die einzelnen Datensätze. Zur einfacheren Lesbarkeit werden die einzelnen Antworten in Listing 4 zu einer einzigen zusammengefasst.

In den von der Karte zurückgelieferten Daten sind zwei Public Keys zu finden: Der eine gehört dem Kartenherausgeber, der andere gehört zu einem Schlüsselpaar, das auf der Karte selbst abgelegt ist. Das Zertifikat des auf der Karte selbst gespeicherten Schlüssels ist mit dem Herausgeberschlüssel signiert, um ihn zu beglaubigen. Neben diesen Daten werden jedoch auch die Kreditkartennummer und das Ablaufdatum ausgegeben, ohne dass wir irgendwelche Tricks anwenden müssen. Das heisst, diese Angaben können über die NFC Schnittstelle ohne Probleme ausgelesen werden. In der *Card Risk Management Data Object List 1* ist hinterlegt, welche Daten vom Terminal an die Karte übertragen werden müssen, damit diese die Transaktion bestätigen kann. Mit der *Cardholder Verification Method* und den *Issuer Action Codes* werden die durch die Karte erlaubten bzw. erforderten Transaktionssicherheitsmechanismen beschrieben.

Offline Data Authentication

Die Daten, die mit der *Card Risk Management Data Object List 1* durch die Karte angefordert werden, werden vom Terminal zusammengestellt

```
Anfrage: 80A80000 02 8300 00

Antwort:
77  Response Message Template Format 2
|  82  Application Interchange Profile (AIP)
|  |  1980
|  94  Application File Locator (AFL)
|  |  08010100100101011004040018010100200 10200
```

Listing 3: Initiate Application Processing

```
Anfrage: 00B2XXYY 00

Antwort:
70 EMV Proprietary Template
  5A Application Primary Account Number (PAN) : Kreditkartennummer
    XXXXXXXXXXXX6317
  5F24 Application Expiration Date : YYMMDD, Kreditkartenablaufdatum
    160430
  8C Card Risk Management Data Object List 1 (CDOL1) : TLV-Tags von Daten, die zum Ezeugen des Kryptogramms
                                                       vom Terminal an die Karte gesendet werden müssen
    9F02069F03069F1A0295055F2A029A039C019F37049F35019F45029F4C089F3403
  8E Cardholder Verification Method (CVM) List : Cardholder Verification Method, in diesem Fall ausschliesslich
                                                 Online-PIN (Überprüfung der PIN durch einen Server des Kreditkartenanbieters.
                                                 Andere Möglichkeiten wären z.B. Unterschrift oder keine Verifikation nötig)
    000000000000000042031E031F03
  9F0D Issuer Action Code – Default : beschreibt die erlaubten Operationen bei einem nicht-online-fähigen Terminal
    F450840000
  9F0F Issuer Action Code – Online : beschreibt die erlaubten Operationen bei einem online-fähigen Terminal
    F470848000
  9F32 Issuer Public Key Exponent / 92 Issuer Public Key Remainder / 90 Issuer Public Key Certificate :
                                                                      Public Key und Zertifikat des Kartenherausgebers
    3 / AA159AD2B21FE1196403B51EC1 … / 474BCDC46F0B12709F6A1DD4 …
  9F47 Integrated Circuit Card (ICC) Public Key Exponent / 9F48 Integrated Circuit Card (ICC) Public Key
  Remainder / 9F46 Integrated Circuit Card (ICC) Public Key Certificate : Public Key und Zertifikat des auf der Karte
                                                                          verwendeten Schlüssels
    3 / 6A78A538D815D79F3C05 / B24B6B1AC2F0046A6…
```

Listing 4: Read Application Data

und an die Karte übertragen. Diese Daten (Tabelle 1) beinhalten Informationen über das Terminal, kryptografische Daten zur Transaktion (Nonce) sowie Informationen zur Transaktion.

Zuerst wird durch das Terminal anhand der *Issuer Action Code List*, der *Cardholder Verification Method List*, Transaktionseigenschaften (z.B. sind bei NFC-Karten in der Schweiz Zahlungen unter CHF 40 ohne PIN-Eingabe möglich, höhere Beträge erfordern jedoch eine PIN) sowie von im Terminal gespeicherten Action Codes und den Terminal-Fähigkeiten (z.B. Onlinefähigkeit, Vorhandensein eines PIN-Eingabegerätes) entschieden, ob und wie die Anwesenheit des Kartenhalters bestätigt werden muss. Die gewählte Prüfmethode wird als *Cardholder Verification Result* codiert und muss von der Karte bestätigt werden. Die eigentliche Verifizierung der Anwesenheit findet dann erst später statt.

80AE5000	SmartCard-Befehl (proprietär), Parameter P1 (hier: 50) gibt den Typ der gewünschten Bestätigung an (Offline-Signatur, kombinierte dynamische Datenauthentifizierung und Kryptogramm-Generierung)
000000000095	Transaktionsbetrag (0.95)
000000000000	Transaktionsgebühren (0.00)
0756	Ländercode (CH)
0000000000	Resultat der Kartenüberprüfung (erfolgreich)
0756	Währungscode (CHF)
141021	Zahlungsdatum (21.10.2014)
00	Transaktionstyp (Einkauf)
4CF7ED0B	Nonce (Number Once, eine Zufallszahl für die Kryptografie)
25	Terminal-Type(kundenbedientes Offline-Terminal mit Online-Fähigkeiten)
0000	Data Authentication Code (für ältere Sicherheitsprotokolle verwendet)
0000000000000000	dynamische Nr. der Karte (immer 0)
1F0302	Cardholder Verification Result, gewählte Methode (Verzicht auf Verifizierung)

Tabelle 1: Anfrage für Offline Data Authentication

In der Antwort der Karte (Listing 5) sind im *Cryptogram Information Data* Informationen über den Typ des mitgelieferten Kryptogramm enthalten (AAC: Transaktion abgelehnt, TC: Transaktion offline bestätigt, ARQC: Online-Autorisierung benötigt). Diese bestimmen zusammen mit der Terminal-Entscheidung den weiteren Ablauf der Transaktion. In unserem Fall ist dies ein Request für die Online-Autorisierung ohne weitere Kundeninteraktion.

Ebenfalls in der Antwort enthalten sind ein Transaktionszähler, der von der Karte nachgeführt wird, und die Issuer Application Data. Letztere werden unverändert an den Kartenherausgeber weitergeleitet. Der Transaktionszähler ist wohl rein informativer Natur und scheint nicht für die Validierung der Zahlung verwendet zu werden.

Die signierten Applikationsdaten sind verschlüsselt und können vom Terminal zur Bestätigung der Daten anhand des Public Keys der Karte überprüft werden. Mit den Informationen, die

```
77 Response Message Template Format 2
  9F27 Cryptogram Information Data
    80
  9F36 Application Transaction Counter (ATC)
    011D
  9F4B Signed Dynamic Application Data :
       enthält das von der Karte erzeugte Kryptogramm
    31F640F15C3E5434978…
  9F10 Issuer Application Data
    0210A0400322300000000000000000000FF
```

Listing 5: Antwort der Offline Data Authentication

dem Terminal nun vorliegen, kann die Zahlung abgeschlossen werden. Das Terminal nimmt die *Cardholder Verification* vor und führt den Online-Teil der Transaktion aus, bei der auch die signierten Applikationsdaten an den Kartenherausgeber übertragen werden.

Im Fall von NFC-Karten ist vorgeschrieben, dass eine mögliche PIN-Überprüfung bei Beträgen über CHF 40 oder € 25 online durch den Herausgeber vorgenommen werden muss. Dies ist nötig, da bei einer Offline-Überprüfung durch die Karte während dem Eingeben der PIN weiterhin eine Verbindung zur Karte bestehen müsste oder die PIN bei einer zweiten Berührung durch die Karte überprüft werden müsste.

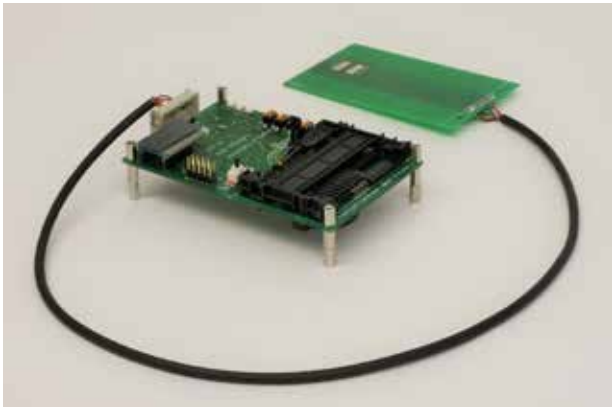

Abbildung 3: Der Smart Card Detective.
Foto von http://www.smartcarddetective.com

Man-in-the-Middle

Für das Aufzeichnen der übertragenen Daten haben wir ein Man-in-the-Middle-System verwendet. Bereits bei Kontakt-Kreditkarten kamen Angreifer auf die Idee, Man-in-the-Middle-Angriffe zwischen der Karte und dem Terminal durchzuführen. Dazu wird spezielle Hardware verwendet, die zwischen dem Kreditkarteninterface und der Karte eingesetzt wird. Darauf befindet sich typischerweise eine serielle Schnittstelle zum Auslesen und Manipulieren des Datenverkehrs oder ein Microcontroller, der dies direkt auf der Hardware erledigt. Abbildung 3 zeigt eine Open-Source-Implementierung eines solchen Man-in-the-Middle Gerätes, den Smart Card Detective, der von Omar Choudary im Rahmen seiner Master-Thesis entwickelt wurde [OSC10].

Mit dem Aufkommen der kontaktlosen Karten sowie dem Bezahlen mittels Smartphone hat sich diese Situation geändert, spezielle Hardware für die Kreditkarten-Schnittstelle ist nicht mehr nötig: Seit der Version 4.4 (KitKat) unterstützt Android Host Card Emulation (HCE), mit der eine NFC-Karte emuliert werden kann [HCE13]. HCE ermöglicht das Implementieren eines Service, der die APDUs eines NFC-Terminals entgegennimmt und beantwortet.

Die Idee hinter dem Man-in-the-Middle-System ist einfach (Abb. 4): Anstatt dass direkt die Karte an den Kartenleser gehalten wird, wird ein Android-Smartphone mit einer Proxy-Software verwendet. Der darauf implementierte Service („Terminal to Wireless“) reagiert auf die Daten, die vom NFC-Leser gesendet werden, und schickt sie über das Netzwerk an ein zweites NFC-Handy, das wiederum an eine NFC-Kreditkarte gehalten wird. Eine auf dem zweiten Handy installierte Anwendung („Wireless to Card“) empfängt diese Anfragen und sendet sie an die Karte. Die Antworten der Karte werden daraufhin über diese beiden Applikationen wieder zurück an das Terminal übermittelt. Damit ist es einerseits möglich, ohne physisch anwesende Karte zu bezahlen, andererseits ist es auch möglich, den Datenverkehr für den Bezahlvorgang aufzuzeichnen und zu manipulieren. Das wirklich Problematische daran ist, dass auf diese Art eine fremde NFC-Kreditkarte angesprochen werden kann, die sich unter Umständen weit entfernt vom Terminal befindet.

Bereits vor Android 4.4 gab es bei Android-Telefonen mit modifiziertem Betriebssystem die Möglichkeit NFC-Karten zu emulieren [FHMM11]. Während bei uns erste Versuche für Man-in-the-Middle-Angriffe mit modifiziertem Betriebssystem im Jahr 2013 fehlschlugen, gelang dies anderen Forschungsteams. Einige Studierende an der *University of Texas* hielten ihre erfolgreiche Transaktion in einem Video fest [UTC12].

Neu an unserer Software ist, dass im Gegensatz zu unseren früheren Versuchen und den Applikationen von anderen Teams standardmässig durch Android angebotene Schnittstellen verwendet werden, so dass der Angriff auf aktuellen Android-Smartphones durch die blosse Installation der entwickelten Software möglich ist. Sie bietet die Möglichkeit, zwischen dem Terminal und der Karte mitzulesen, um das in der Praxis verwendete EMV-Protokoll zu dokumentieren und zu analysieren (Listing 6). Durch einfache Anpassungen an der entwickelten Software könnte der Datenverkehr auch manipuliert werden.

Eine Einschränkung bei der Implementierung einer solchen Proxy-Software ist, dass bereits vor der Installation der Software die Application IDs der emulierten Karten in XML-Dateien festgeschrieben werden. Das Android-Betriebssystem leitet dann die APDUs, die zur entsprechenden Software gehören, an diese weiter. Falls mehrere Apps die gleichen AIDs beanspruchen, dann kümmert sich das Betriebssystem um die Konfliktbehebung, z.B. indem beim User nachgefragt wird, welche App jetzt zuständig ist. Durch diese Mechanismen ist es nicht möglich, einen allgemeingültigen Proxy für alle Karten zu schreiben, sondern die einzelnen Kartentypen müssen vor der Installation festgelegt werden.

Mit dem Versuchsaufbau in Abbildung 4 gelingt es uns, einen Man-in-the-Middle-Angriff in

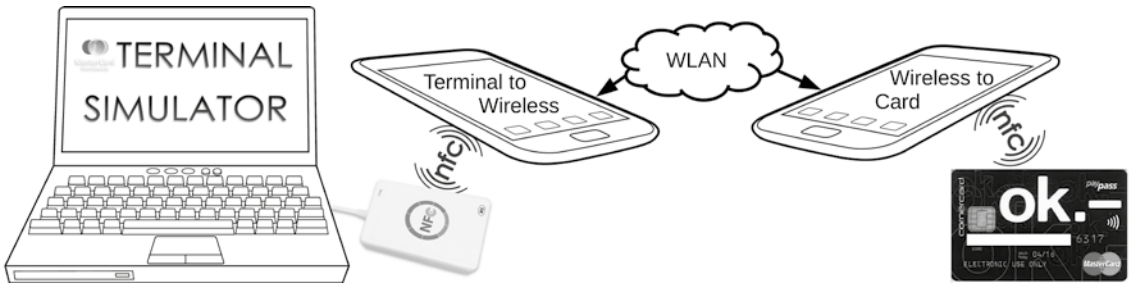

Abbildung 4: Versuchsaufbau zum Testen der Implementierung

der Praxis umzusetzen. Verwendet wird ein LG Nexus 4 und ein Sony Z1 compact; auf beiden Geräten ist Android 4.4.4 installiert. Als Terminal wird im Versuchsaufbau ein Windows-Computer mit der Software *Terminal-Simulator* von MasterCard [MCW] und einem ACR122 NFC-Kartenleser verwendet.

Eine beobachtete Schwierigkeit bei der Transaktion ist die Stabilität der Verbindung zwischen dem „Wireless to Card"-Smartphone und der daran gehaltenen Karte. Während erste Versuche fehlschlugen, konnte durch Trial-and-Error eine Kombination aus einer PrePaid-MasterCard und dem Sony Z1 compact als stabilste Variante ermittelt werden. Bei anderen Karten und Smartphones entsteht oft das Problem, dass die Verbindung zwischen dem „Wireless to Card"-Smartphone und der Karte abbricht, sobald auf der Karte kryptografische Operationen ausgeführt werden. Aufgrund der höheren Leistungs- und damit Stromanforderung der Karten in diesem Moment, gehen wir davon aus, dass die Speisung der Karte über Induktion das Problem ist.

In einem Feldversuch bei einem Detailhändler konnte nachgewiesen werden, dass das Vorgehen nicht nur unter Laborbedingungen funktioniert. Die im Geschäft zwischen dem *Point of Sale Terminal* und der Kreditkarte übertragenen Daten stimmten – bis auf die für jede Transaktion einmaligen kryptografischen Daten – mit den im Versuchsaufbau ermittelten überein.

Der im EMV-Abschnitt dieses Artikels enthaltene Datenmitschnitt ist aus diesen Transaktionen entstanden.

Auslesen von persönlichen Daten

Während der Transaktion werden im Schritt *Read Application Data* diverse Informationen auf der Karte durch das Terminal ausgelesen. Darunter befinden sich mindestens die *Primary Account Number* (PAN, die Kreditkartennummer) und das Ablaufdatum. In manchen Fällen wird zusätzlich noch der Name des Besitzers übertragen, wobei dieser nicht auf allen Karten gespeichert ist. Das Auslesen dieser Daten ist nicht an eine Transaktion gebunden – sobald also das richtige Applet ausgewählt ist, können diese Daten von der Karte gelesen werden.

Damit liegen einem Angreifer, der unabhängig von einer Transaktion eine NFC-Verbindung mit einer Kreditkarte aufbauen kann, genügend Daten für eine Online-Bestellung vor, falls der Händler keine weiteren Sicherheitsfeatures implementiert. Von den Kartenanbietern werden zwei weitere Sicherheitsfeatures angeboten: das Überprüfen des *Card Verification Code* (CVC oder auch CVV für *Card Verification Value*) und ein Online-Passwort (3-D). Einige Online-Händler entscheiden sich jedoch dazu, diese Sicherheitsfeatures nicht zu implementieren, um den Kunden den Einkauf möglichst hürdenlos zu gestalten. Damit tragen die Händler allerdings selbst das Risiko für Kreditkartenmissbrauch. Nur durch den Einsatz eines zusätzlichen Sicherheitsfeatures wird das Risiko vom Händler auf die Bank bzw. den Kreditkartenbesitzer abgewälzt.

Der CVC ist eine Nummer, die bei den meisten Kredit- und Prepaidkarten auf das Unterschriftenfeld gedruckt ist. Diese Nummer (bei MasterCard als CVC2 bekannt, bei Visa als CVV2) ist bloss dem Kreditinstitut bekannt und darf von Händlern nicht zwischengespeichert oder bearbeitet werden. Dies soll ermöglichen, dass bei jeder neuen Bestellung eine Überprüfung stattfindet, ob die Karte physisch vorliegt und somit dieser Code abgelesen werden kann – eine durchaus fragwürdige Verifikation, da ein dreistelliger, optisch lesbarer statischer Code bei einem Diebstahl keine grosse Hürde darstellt. Einzelne Händler wie z.B. Amazon, die Kreditkartendaten von Kunden zwischenspeichern und ein 1-Click-payment ermöglichen, implementieren diese Methode nicht, damit die Kunden ohne Bedenkzeit und ohne Aufwand eine Bezahlung tätigen können.

Mit *3-D Secure* steht Online-Händlern eine weitere Möglichkeit zum Schutz vor Kreditkartenmissbrauch zur Verfügung. Bei diesem Verfahren, das in Europa vor allem unter den Namen *MasterCard SecureCode* und *Verified by Visa* bekannt ist, wird vom Käufer bei einer Onlinetransaktion zusätzlich auf einer zur Kreditkartenfirma gehörenden Website ein Passwort eingegeben. Auch hier gilt, dass viele Onlinehändler auf dieses Sicherheitsmerkmal verzichten, um den Kunden beim Bezahlen möglichst wenige Hürden in den Weg zu legen.

```
public class TerminalToWirelessProxyServiceSimplified extends HostApduService {
   private Socket socket;
   private ObjectInputStream ois;
   private ObjectOutputStream oos;

   @Override
   public byte[] processCommandApdu(final byte[] requestApdu, Bundle extras) {
      new Thread(new Runnable() { /* Not thread-safe in this implementation */
         public void run() {
            if (!isConnected()) { connect(); }
            if (isConnected()) {
               try {
                  /* transmit data to the server, and send response to the terminal. */
                  oos.writeObject(requestApdu);
                  sendResponseApdu((byte[]) ois.readObject());
               } catch (Exception ex) { /* ... */ }
            }
         }
      }).start();

      /* If null is returned here, the function sendResponseApdu(byte[]) can be used to answer
         the request. Here this happens in a separate thread to allow networking. */
      return null;
   }

   @Override
   public void onDeactivated(int reason) {
      disconnect();
   }
   /* ... */
}
```

Listing 6: Ausschnitt aus dem Terminal-to-Wireless-Service des Man-in-the-Middle-Systems (Host Card Emulation Service)

Da bei Online-Bezahlungen der Name auf der Kreditkarte nicht überprüft wird und die beiden von den Kartenausgebern angebotenen Sicherheitsmassnahmen bei Händlern oft nicht implementiert werden, reicht zum Bezahlen oftmals die PAN sowie das Ablaufdatum; zwei Eigenschaften der Karte also, die per NFC auslesbar sind und die bei jeder Transaktion von der Karte an das Terminal übertragen werden. In der „Wireless to Card"-Komponente des Man-in-the-Middle-Angriffs werden die Kartennummer und das Ablaufdatum nach Abschluss der Transaktion in einem Popup dargestellt.

Angriffsvektoren

War es bisher für das Auslesen der Kartendaten nötig, physischen Zugriff auf die Karte zu erhalten, sei es per Skimming oder einer kurzen Entwendung, so ist es jetzt möglich die Daten aus einigen Zentimeter Entfernung auszulesen, z.B. im Gedränge am Bahnhof. Durch diese Vereinfachung des Datendiebstahls in Kombination mit dem Verhalten vieler Online-Händler, die zur Vereinfachung des Bezahlvorgangs auf zusätzliche Sicherheiten verzichten, ist mit einem Anstieg des digitalen Missbrauchs von Kreditkarten zu rechnen. Deshalb wird es auch in Zukunft wichtig sein, die Abrechnung auf verdächtige Zahlungen zu überprüfen – auch oder gerade weil die Haftung für missbräuchliche Zahlungen bei verantwortungsvollem Umgang mit der Karte beim Händler oder beim Kreditkarteninstitut liegt.

War bei der kontaktbehafteten Bezahlung für einen Man-in-the-Middle-Angriff das Einführen eines Gerätes in den Kartenslot eines Terminals notwendig, so kann dieser nun auch mit zwei Smartphones ausgeführt werden. Durch das Aufkommen der Möglichkeit, direkt mit Smartphones zu bezahlen, ist dieses Vorgehen absolut unauffällig. Bereits bevor diese Möglichkeit durch Apple Pay grössere Bekanntheit erlangte, wurden wir zwar von Mitarbeitern beim Detailhandel angesprochen. Diese zeigten sich allerdings eher hilfsbereit beim Versuch, mit dem Handy zu bezahlen, als misstrauisch. Da für kleine Beträge bei der kontaktlosen Zahlung keine PIN erforderlich ist, kann so eine unerwünschte Abbuchung vorgenommen werden, indem ein Angreifer die APDUs vom Terminal über das Internet an ein Smartphone weiterleitet, das mit der Kreditkarte eines ahnungslosen Opfers kommuniziert.

Der Online-Teil der Transaktion, der erst nach dem oben beschriebenen Kontakt zwischen Terminal und Karte ausgeführt wird, kann auch verzögert ausgeführt werden. Im Artikel „Harvesting High Value Foreign Currency Transactions from EMV Contactless Credit Cards without the PIN" [EAF14] beschreiben die Autoren von der School of Computing Science der Newcastle University zwei Sicherheitslücken: Einerseits wird aufgrund eines Implementations- oder Provisionierungsfehlers bei VISA-Karten aus dem vereinigten Königreich bei Transaktionen in Fremdwährungen das Limit für Transaktionen ohne PIN-Eingabe nicht über-

prüft. Dadurch können Beträge bis zu 999 999.99 in der entsprechenden Währung ohne Autorisierung ausgegeben werden. Andererseits beschreiben sie auch das Vorgehen, dass ein Smartphone ein Terminal simuliert und Transaktionen mit Kreditkarten durchführt. Die auf dem Smartphone gespeicherten Transaktionsdaten können anschliessend zu einem beliebigen Zeitpunkt über das Internet mit einem gültigen Händler-Account bei einer Bank platziert werden, um so Geld von den Kreditkarten auf das eigene Konto zu übertragen. Zur Schadensbegrenzung ist bei einigen Kreditkarten ein „Offline-Limit" eingebaut, d.h. nach einer bestimmten Anzahl Transaktionen oder einem bestimmten Betrag ist eine Online-Buchung erforderlich. Auch dieser Angriff ist nur möglich, da bei kleinen (oder durch den Implementierungsfehler bei VISA UK auch grossen) Beträgen keine PIN-Eingabe nötig ist.

Der Nachteil der kontaktlosen Kreditkarte gegenüber der kontaktbehafteten ist es also, dass unbemerkt mit dieser kommuniziert werden kann und nicht immer eine Transaktionsbestätigung durch den Kartenbesitzer notwendig ist. Wenn die Karte jedoch in einem Mobiltelefon gespeichert wird, so ist der Zugriff über die NFC-Schnittstelle nur möglich wenn der Bildschirm aktiviert, eine entsprechende Applikation gestartet worden ist oder gar (wie bei ApplePay) das Auslesen der Kartendaten durch eine zusätzliche Autorisierung (wie ein Fingerprint) freigegeben worden ist. Während also der Zugriff über die kontaktlose Schnittstelle durch Smartphones besser geschützt ist, so besteht bei diesen das zusätzliche Risiko von Malware. Bei einer reinen Software-Lösung kann eine entsprechend programmierte Schadsoftware den privaten Schlüssel der Karte auslesen und versenden. Damit ist es dann möglich, mittels eines anderen Smartphones eine Kopie der simulierten Kreditkarte zu erstellen, um Zahlungen mit den erbeuteten Daten auszulösen.

Eine z.B. bei Google Wallet verwendete Technik zum Schutz des privaten Schlüssel ist es, ein Secure-Element auf dem Telefon zu verwendet. Dieses Secure-Element funktioniert dann gleich wie eine SmartCard, führt also eigene kryptografische Operationen durch und beantwortet APDUs selbständig über die NFC-Schnittstelle des Smartphones, wenn der Bildschirm bzw. die Applikation aktiviert ist. Michael Roland hat eine Relay-Attacke beschrieben, in der – statt über die NFC-Schnittstelle auf eine NFC-Kreditkarte – direkt von der Smartphone-Software aus auf das Secure Element eines Android-Geräts zugegriffen wird [ROL13]. Damit ist der Vorteil verloren, dass der Zugriff auf die Karte nur durch eine Userinteraktion am Smartphone hergestellt werden kann, die Malware kann jederzeit per Netzwerk aktiviert werden.

Fazit

Mit der Möglichkeit kontaktlos bezahlen zu können haben sich gegenüber der kontaktbehafteten Bezahlung per Karte neue Sicherheitsprobleme geöffnet. Risiken, die bereits mit Kontaktkreditkarten bestehen, können dank Kommunikation aus kurzen Distanzen auch ohne Entwenden von Karten oder manipulierte Terminals ausgeführt werden. Dies in Kombination mit der Möglichkeit, kleine Beträge ohne PIN-Eingabe oder Unterschrift bezahlen zu können, führt zu einem gesteigerten Risiko des Kreditkartenmissbrauchs.

Durch die Zahlmöglichkeit mit dem Smartphone, und damit der Ablösung von kontaktlosen Kreditkarten, ist das Risiko gebannt, dass ständig per NFC auf die Karte zugegriffen werden kann: Erst durch Aktivieren des Bildschirms oder sogar einer Benutzerauthentifizierung auf dem Smartphone wird die Zahlung ausgelöst. Dank spezieller Protokollimplementierungen können auch persönliche Daten besser vor manipulierten Terminals geschützt werden. Dafür existiert das Risiko, dass Smartphone-Malware einen anderen Zugangskanal zu der Karte hat: Bei ungeschützter Speicherung des privaten Schlüssels auf dem Smartphone ist es möglich, direkt eine Kopie der Karte anzulegen. Bei einem Secure-Element, das die privaten Schlüssel auf dem Smartphone sicher speichert, besteht implementationsabhängig das Risiko, dass aus dem Betriebssystem mit diesem kommuniziert wird, und so die APDUs eines Terminals über das Internet weitergeleitet direkt an dieses gesendet werden können.

Referenzen

[EAF14] M. Emms, B. Arief, L. Freitas, J. Hannon, A. van Moorsel: Harvesting High Value Foreign Currency Transactions from EMV Contactless Credit Cards without the PIN. http://homepages.cs.ncl.ac.uk/budi.arief/home.formal/Papers/CCS2014.pdf

[ELB] EMV Lab. http://emvlab.org

[FHMM11] L. Francis, G. Hancke, K. Mayes, K. Markantonakis: Practical Relay Attack on Contactless transactions by Using NFC Mobile Phones. https://eprint.iacr.org/2011/618.pdf

[HCE13] Android Developer Documentation: Host Card Emulation. http://developer.android.com/guide/topics/connectivity/nfc/hce.html

[MCW] Master Card Worldwide: Terminal Simulator. https://www.terminalsimulator.com/

[OSC10] Omar S. Choudary: The Smart Card Detective, a hand-held EMV interceptor. http://www.cl.cam.ac.uk/~osc22/docs/mphil_acs_osc22.pdf

[ROL13] M. Roland, J. Langer, and J. Scharinger: Applying Relay Attacks to Google Wallet. In: Proceedings of the 5th International Workshop on Near Field Communication (NFC 2013), pp. 1–6, Zurich, Switzerland, Feb. 2013.

[TVR] TVR-Decoder: Will Currie http://tvr-decoder.appspot.com/t/home

[UTC12] University of Texas at Austin IEEE Communications Society Student Chapter: NFC Proxy. http://www.youtube.com/watch?v=Yjfc60LGjik