Diplomarbeit

Mobiler Chipkartenleser für den neuen Personalausweis

Sicherheitsanalyse und Erweiterung des „Systems nPA“

Frank Morgner

10. Mai 2012

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

Erstgutachter Prof. Dr. rer. nat. Jens-Peter Redlich
Zweitgutachter Dr. rer. nat. Dennis Kügler
Betreuer Dr. rer. nat. Wolf Müller

Zusammenfassung

Der neue Personalausweis (nPA) besitzt einen kontaktlosen Chip, der sensitive Daten über den Ausweisinhaber speichert und die qualifizierete elektronische Signatur (QES) erlaubt. Ist ein Missbrauch dieser Funktionen möglich, ist eine Vielzahl von Ausweisinhabern betroffen. In dieser Arbeit wird untersucht, ob und wie ein Chipkartenleser im Allgemeinen und ein als Leser genutztes Smartphone im Speziellen gegen möglichen Missbrauch schützen können. Dazu wird ein Sicherheitsmodell des „Systems nPA“ aus den Dokumenten des Bundesamtes für Sicherheit in der Informationstechnik hergeleitet. Verschiedene Angriffe werden skizziert, die eine Bedrohung der Privatsphäre und der informationellen Selbstbestimmung des Ausweisinhabers sowie der sicheren Signaturerstellung darstellen können. Es werden geeignete Gegenmaßnahmen für das System nPA vorgeschlagen, um diesen Problemen wirkungsvoll zu begegnen. Bei dem Entwurf des mobilen Chipkartenlesers für den nPA (CAT-M) als vertrauenswürdiger Mittler werden sinnvolle Erweiterungen der Technische Richtlinie TR-03119 vorgeschlagen. Die praktische Umsetzbarkeit des CAT-M wird prototypisch auf dem Openmoko Neo FreeRunner gezeigt. Die Verwendung eines Smartphones zur sicheren und anwenderfreundlichen Nutzerinteraktion kann eine preiswerte Alternative zu klassischen Chipkartenlesern darstellen.

Inhaltsverzeichnis

Abkürzungen **xi**

1. Einführung **1**
1.1. Ziel der Arbeit . . . 1
1.2. Aufbau der Arbeit . . . 2

2. Problemanalyse **3**
2.1. Die elektronische Nutzung des neuen Personalausweises . . . 3
2.1.1. Extended Access Control . . . 6
2.1.2. Lesegeräteklassen nach BSI TR-03119 1.2 . . . 15
2.1.3. Nutzung der eID-Anwendung . . . 20
2.1.4. Nutzung der Signaturanwendung . . . 26
2.2. Sicherheitsmodell des Systems nPA . . . 29
2.2.1. Rekonstruktion des Sicherheitsmodells . . . 29
2.2.2. Diskussion und Modifikation des Sicherheitsmodells . . . 33
2.3. Angreifermodelle . . . 35
2.3.1. Malware . . . 35
2.3.2. Nearby Attacker . . . 36
2.3.3. Skimmer . . . 37
2.3.4. Man-in-the-Middle Hardware . . . 40
2.4. Bedrohungsszenarien . . . 40
2.4.1. Missbrauch des elektronischen Identitätsnachweises . . . 43
2.4.2. Tracing . . . 46
2.4.3. Privilege Escalation . . . 52
2.4.4. Missbrauch der Signaturfunktion . . . 54
2.5. Mitigation . . . 58
2.5.1. Distance Bounding . . . 58
2.5.2. Signatur-Nutzung mit Medienbruch . . . 59
2.5.3. Terminalauthentisierung zur Erkennung von Manipulationen . . . 60
2.5.4. Selektive Deaktivierung des nPA . . . 61
2.5.5. Schutz von CAN und MRZ . . . 62
2.5.6. Sicheres Chipkartendisplay . . . 62
2.5.7. Entflechtung von eID- und eSign-Anwendung . . . 63
2.5.8. „Man-in-the-Middle Defence“ . . . 64

3. Mobiler Chipkartenleser für den neuen Personalausweis **67**
3.1. nPA-Nutzung am Mobiltelefon in anderen Arbeiten 68
3.2. Erarbeitung des mobilen Chipkartenlesers für den neuen Personalausweis . . 70
3.3. Neue und angepasste Module für BSI TR-03119 1.2 71
3.3.1. PIN-Pad mit PACE-Unterstützung 71
3.3.2. Speicherung der Kartenzugriffsnummer im Chipkartenleser 74
3.3.3. Display: Vollständige und übersichtliche Darstellung 74
3.3.4. Standardisierte Hardware-Treiber 76
3.3.5. Prüfungen als Kartenemulator 80
3.3.6. Zeitaktualisierungsdienst 82
3.3.7. Autarkes PIN-Management und temporäre PIN 83
3.3.8. Sichere Ausführungsumgebung 84

4. Umsetzung eines mobilen Kartenlesers mit PACE-Funktion **89**
4.1. Kryptographische Umsetzung: OpenPACE 90
4.1.1. Aufbau und Funktionsweise 94
4.1.2. Besonderheiten und Schwierigkeiten 95
4.2. Schnittstelle zum nPA als Chipkarte 96
4.2.1. Smartphone-Gerätetreiber 97
4.2.2. OpenSC für PACE-fähige Leser 98
4.2.3. nPA Smart Card Library 99
4.3. Vom Smartphone zum Chipkartenleser 100
4.3.1. CCID-Emulator für den nPA 104
4.4. nPA-Kartenemulator 111

5. Diskussion und Ausblick **117**
5.1. Nicht „die gesamte Technik ist sicher“ 117
5.2. Schutzmaßnahmen des mobilen Chipkartenlesers 118
5.3. Openmoko als mobiler Chipkartenleser 120
5.4. Konklusion für den nPA und seine Infrastruktur 123

A. Mögliche Kryptographische Bibliotheken zur Umsetzung der EAC **125**

B. Mögliche Middleware zum Chipkartenleser **129**

C. EF.CardSecurity **131**

Literatur **137**

Abbildungsverzeichnis

2.1. Der neue Personalausweis . . . 3
2.2. Zusammenspiel von Local und Remote Terminal bei Online-Authentisierung 7
2.3. Nutzerauthentisierung per eID-PIN unter Berücksichtigung von Fehleingaben 9
2.4. Der nPA mit einigen sichtbaren Merkmalen . . . 10
2.5. Phasen der Terminalauthentisierung (TA) aus Sicht des Terminals . . . 12
2.6. Beispiele für zertifizierte Basisleser . . . 15
2.7. Die bisher einzigen zertifizierten Standard- und Komfortleser . . . 16
2.8. Nutzung der AusweisApp zum elektronischen Identitätsnachweis im Internet mit einem Basisleser . . . 21
2.9. Kommunikationspartner und -kanäle bei der eID-Nutzung . . . 22
2.10. Interaktion zwischen nPA, Bürger-PC und Dienstanbieter bei eID-Nutzung . 24
2.11. Prozesse zur Aktivierung des nPA als SSEE . . . 27
2.12. Prozesse zur Nutzung des nPA als SSEE . . . 28
2.13. Befehl zur Entschlüsselung der AusweisApp 1.3 . . . 34
2.14. Datensatz der AusweisApp zum Komfortleser REINER SCT cyberJack RFID komfort aus der Datei `siqTerminalsInfo.cfg` . . . 38
2.15. Erkennung von zertifizierten Chipkartenlesern bei der AusweisApp . . . 39
2.16. Elektronische Nutzung des nPA an einem öffentlichen Terminal als offizieller Anwendungsfall . . . 41
2.17. Nachdem die Daten des nPA erfolgreich ausgelesen wurden, werden sie auf der Website des KBA angezeigt. . . . 44
2.18. Relay-Angriff zum unberechtigten Ermitteln der Ausweisdaten . . . 45
2.19. Zeitverlauf beim Brute-Force-Angriff auf die CAN des nPA . . . 48
2.20. Tracing eines nPA mittels seiner CAN . . . 50
2.21. Relay-Angriff zum unberechtigten Terminieren und Aktivieren des nPA als SSEE 53
2.22. Skimming eines Standardlesers (CAT-S), Komfortlesers (CAT-K) oder öffentlichen Terminals ermöglicht es einem Angreifer volle Kontrolle beider Authentisierungsfaktoren bei der Nutzung des nPA zu erlangen. . . . 54
2.23. Verteiltes und integriertes Terminal . . . 55
2.24. Privilege Escalation . . . 63
2.25. „Man-in-the-Middle Defence“ per NFC . . . 65

3.1. Der mobile Chipkartenleser bei der Nutzung an einem PC . . . 71
3.2. Darstellung der Verbindungsdetails einer eID-Nutzung . . . 75
3.3. Vorschlag zur Darstellung der Verbindungsdetails einer eID-Nutzung auf dem Openmoko Neo FreeRunner . . . 76

3.4. Der mobile Chipkartenleser bei der Überprüfung eines externen Terminals . 81

4.1. Openmoko Neo FreeRunner mit „integriertem“ drahtlosen Kartenleser und -emulator 91
4.2. pyPACE macht OpenPACE mittels SWIG in anderen Programmiersprachen verfügbar 93
4.3. Die PC/SC-Middleware und das OpenCard-Framework (OCF) zum Zugriff auf Chipkartenleser 96
4.4. Aufbau der nPA Smart Card Library (`libnpa`) 99
4.5. Zusammenspiel von `npa-tool` und der libnpa-Module zum nPA und Secure Messaging (SM) 101
4.6. Umsetzung des PIN-Management und der Zeitaktualisierung 102
4.7. Beispiel einer physikalischen Topologie des USB zwischen USB-Host und mehreren Peripheriegeräten („Functions“) 103
4.8. Softwareschichten des CCID-Emulators auf dem Openmoko Neo FreeRunner 104
4.9. Initialisierung von USB-Controller und GadgetFS für den CCID-Emulator . . 105
4.10. Logische Organisation der Kommunikation zwischen USB-Host und CCID-Emulator 106
4.11. Aufbau des `PC_to_RDR_Secure`-Kommandos mit Erweiterung um PACE . . . 109
4.12. Umsetzung des mobilen Chipkartenlesers 110
4.13. Kartenidentifikationsprozess 113
4.14. Umsetzung des nPA-Emulators 114

5.1. Beispiel einer minimalen lokalen Client-Software 124

Tabellenverzeichnis

2.1. Ausgewählte Dateien der Biometrieanwendung 4
2.2. Übersicht der Chipkartenleser-Kategorien 17
2.3. Eingabedaten für `GetReaderPACECapabilities` und `EstablishPACEChannel` . . 18
2.4. Preise aktueller Chipkartenlesegeräte 20
2.5. Ausgewählte Terminaltypen und ihre möglichen Berechtigungen 25
2.6. Laufzeitübersicht für PACE mit `npa-tool` mittels `ltrace` 49

3.1. Kommandos eines Chipkartenlesers 78

4.1. Kryptographische Primitiven für den neuen Personalausweis 92
4.2. `libnpa` und OpenPACE im Vergleich zu anderen Implementierungen 100
4.3. USB Smartcard Klassendeskriptor des CCID-Emulators 107
4.4. PIN-Funktionen des CCID-Emulators mit PACE-Erweiterung 108
4.5. Unterstützte Kommandos des CCID-Emulators 108

5.1. Vergleich des CAT-M mit anderen Chipkartenlesern für den nPA 120

Abkürzungen

AA Aktive Authentisierung („Active Authentication“, ICAO Doc 9303 Pt. 3 Vol. 2)

APDU Application Protocol Data Unit (ISO/IEC 7816-4)

ATR Answer to Response

BAC Basic Access Control (ICAO Doc 9303 Pt. 3 Vol. 2)

CAN Kartenzugriffsnummer („Card Access Number“, BSI TR-03110 2.05; BSI TR-03127)

CAT-B Basis-Chipkartenleser (Minimalanforderungen nach BSI TR-03119 1.2, S. 8–9)

CAT-K Komfort-Chipkartenleser (Minimalanforderungen nach BSI TR-03119 1.2, S. 9)

CAT-M Mobiler Chipkartenleser für den nPA (Siehe „Mobiler Chipkartenleser für den neuen Personalausweis“, S. 67)

CAT-S Standard-Chipkartenleser (Minimalanforderungen nach BSI TR-03119 1.2, S. 9)

CA Chipauthentisierung („Chip Authentication“, BSI TR-03110 2.05)

CHAT Card Holder Authorization Template

CV-Zertifikat Kartenverifizierbares Zertifikat („Card Verifiable Certificate“ BSI TR-03110 2.05; BSI TR-03128)

CVCA Country Verifying Certification Authority

DVCA Document Verifying Certification Authority

EAC-PKI Extended Access Control Public Key Infrastructure (BSI TR-03110 2.05; BSI TR-03128)

EAC Extended Access Control (BSI TR-03110 2.05)

eID-PIN Geheimzahl für die eID-Anwendung (BSI TR-03110 2.05; BSI TR-03127)

FBZ Fehlbedienungszähler („Retry Counter“ BSI TR-03110 2.05; BSI TR-03127)

libnpa nPA Smart Card Library (Morgner und Oepen 2011)

MRTD Maschinenlesbares Reisedokument („Machine Readable Travel Document“, ICAO Doc 9303 Pt. 3 Vol. 1)

MRZ Maschinenlesbare Zone („Machine Readable Zone“, ICAO Doc 9303 Pt. 3 Vol. 1)

MTM Mobile Trusted Module

NFC Near Field Communication wird in dieser Arbeit auch gebraucht, wenn es um die Kommunikation zum nPA geht. Tatsächlich ist NFC ein Standard für sich, der aber ISO/IEC 14443-4 mehr oder weniger enthält. D. h. Geräte nach ISO/IEC 14443-4 (z. B. der nPA) sind kompatibel zu Geräten nach NFC.

nPA Neuer Personalausweis

OTG Universal Serial Bus (USB) On-the-Go

PACE Password Authenticated Connection Establishment (BSI TR-03110 2.05)

PAOS Reversed HTTP binding for SOAP

PUK PIN Unblocking Key (BSI TR-03110 2.05; BSI TR-03127)

QES Qualifizierte Elektronische Signatur (SigG)

R-APDU Response Application Protocol Data Unit (ISO/IEC 7816-4)

RI Restricted Identification (BSI TR-03110 2.05)

RPC Remote Procedure Call

RZ Rücksetzzähler („Usage Counter“ BSI TR-03110 2.05; BSI TR-03127)

SAML Security Assertion Markup Language

SM Secure Messaging (ISO/IEC 7816-4; BSI TR-03110 2.05)

SOAP Simple Object Access Protocol

SSEE Sichere Signaturerstellungseinheit (BSI TR-03117)

SSL/TLS Secure Sockets Layer bzw. Transport Layer Security

TA Terminalauthentisierung („Terminal Authentication“, BSI TR-03110 2.05)

TLS-PSK Transport Layer Security mit Preshard Secret

TPM Trusted Platform Module

USB Universal Serial Bus

VfB Vergabestelle für Berechtigungszertifikate

VMM Virtual Machine Monitor

ZDA Zertifizierungsdienstanbieter (BSI TR-03127)

1. Einführung

Der neue Personalausweis (nPA) soll sichere Behördengänge, Geschäftsabschlüsse und Authentisierungen im Internet erlauben. Dazu verbirgt sich im Innern des Ausweises ein Chip, der kontaktlos mit einem Lesegerät kommunizieren kann. Er speichert sensitive Daten des Ausweisinhabers und ermöglicht diesem den elektronischen Identitätsnachweis und das verbindliche Unterschreiben digitaler Dokumente. Daher muss sichergestellt werden, dass nur berechtigte Parteien die Funktionen des nPA nutzen können. Aufgrund der damit verbundenen Risiken ist die Skepsis gegenüber dem nPA jedoch groß.[1]

Der nPA wird indes oft als sehr sichere Chipkarte bezeichnet.[2] Beispielsweise bei der Authentisierung mit dem nPA im Internet sind aber viele andere Komponenten involviert, insbesondere auch der PC des Ausweisinhabers. Es ist allgemein bekannt, dass Schadsoftware auf einem PC, mit welchem ein Online-Dienst genutzt wird, Transaktionen manipulieren oder gar selbst auslösen kann.[3] Sollte dies auch beim nPA möglich sein, könnte ein sicheres Lesegerät vor unerlaubten Zugriffen schützen. Ein hochwertiger Leser stellt jedoch für den Ausweisinhaber eine nicht zu vernachlässigende Investition dar. Auf der anderen Seite verbreiten sich immer mehr Smartphones, welche die drahtlose Kommunikation zum nPA ermöglichen. Einem solchen Gerät wird vertraut; man gibt sein Mobiltelefon selten aus der Hand, hat es immer dabei und es besitzt für den Anwender stets das gleiche Interface.

1.1. Ziel der Arbeit

In dieser Arbeit wird versucht zu klären, ob ein Smartphone[4] ähnlich wie ein Chipkartenleser fungieren kann, um an einem PC – oder allgemein an einem Terminal – als externes Gerät die sichere nPA-Nutzung zu ermöglichen. Vor welchen Bedrohungen ein herkömmlicher Chipkartenleser überhaupt schützen *kann*, muss dabei erst untersucht werden. Vermutlich lässt ein Smartphone mehr und andere Prüfungen zu, wegen seines höheren Funktionsumfangs. Es besitzt genau dadurch aber sicherlich auch andere Angriffsvektoren als ein Kartenleser. Daher muss dargelegt werden, wie und ob sich die (impliziten) Funktionen eines normalen Kartenlesers durch einen möglichen mobilen Chipkartenleser für den nPA realisieren lassen.

[1] ARIS 2010; Unisys 2011.

[2] Beispielsweise die IT-Beauftragte der Bundesregierung, Cornelia Rogall-Grothe, ist dieser Ansicht: „Es [der nPA] ist die sicherste Technik, die es gibt." (Welchering 2010)

[3] Bezogen auf den nPA werden dafür Beispiele in Abschnitt 2.4 erläutert.

[4] Definition nach Welte 2008, 2010, siehe „Mobiler Chipkartenleser für den neuen Personalausweis", S. 67

1.2. Aufbau der Arbeit

Diese Arbeit ist in vier weitere Teile gegliedert. In Kapitel 2 werden Bedrohungsszenarien erarbeitet, denen der nPA bzw. der Ausweisinhaber ausgesetzt ist. Dazu müssen zunächst die genutzte Infrastruktur mit den verwendeten Protokollen und Komponenten vorgestellt werden. Dies bildet gleichzeitig die terminologische Grundlage für die nachfolgenden Betrachtungen. Die Untersuchungen orientieren sich an den gegebenen Spezifikationen statt an bestimmten Implementierungen. Daraus lassen sich *prinzipielle* Probleme ableiten, die den nPA und seine Infrastruktur unabhängig von bestimmten Produkten betreffen.

In Kapitel 3 wird der mobile Chipkartenleser für den nPA (CAT-M) vorgestellt. Die Eigenschaften bzw. Funktionen dieser Leserklasse werden in verschiedenen Modulen definiert. Grundlage bildet hier BSI TR-03119 1.2, welche bereits einige Module spezifiziert, die bei den anderen Lesegeräteklassen zum Einsatz kommen. Für jedes neue Modul wird zugleich gezeigt, ob es sich nach dem aktuellen Forschungsstand auch in ein Smartphone integrieren lässt.

Die Machbarkeit wesentlicher Funktionen für einen CAT-M wird in Kapitel 4 demonstriert. Ein Smartphone wird hier um die Funktion eines Chipkartenlesers erweitert, der die sichere PIN-Eingabe für den nPA unterstützt. Weiterhin werden auch Funktionen wie das PIN-Management oder die Überprüfung von öffentlichen Terminals implementiert. Dabei wird auch ein Emulator für einen nPA vorgestellt, der die Echtheit eines Terminals prüfen kann. Die Möglichkeit zur Realisierung in verschiedenen portablen und quelloffenen Softwarebibliotheken wird diskutiert und letztlich umgesetzt.

In Kapitel 5 werden die Ergebnisse dieser Arbeit kurz dargestellt und diskutiert. So wird ein Versuch unternommen, das Risiko bzw. die Reichweite der ermittelten Bedrohungsszenarien einzuschätzen. Potenzial und Grenzen der hier vorgestellten Umsetzung einiger Aspekte des CAT-M sollen herausgestellt werden. Diese Diskussion benennt dabei auch mögliche Ansatzpunkte für zukünftige Arbeiten, um die verbliebenen Herausforderungen des sicheren mobilen Kartenlesers zu bewältigen.

2. Problemanalyse

2.1. Die elektronische Nutzung des neuen Personalausweises

Der nPA ist ein Ausweisdokument, welches neben der Identifizierung per Sichtprüfung auch eine elektronische Nutzung erlaubt.[1] Der nPA liegt im td-1-Format[2] vor und enthält einen kontaktlosen Chip nach ISO/IEC 14443-4.[3] Die physikalischen Sicherheitseigenschaften (z. B. Hologrammbilder), die die Echtheit des Ausweises garantieren sollen, sind nicht Gegenstand dieser Arbeit. Hier soll allein die elektronische Funktionalität des Ausweises behandelt werden.

Die Datenübertragung sowie die Stromversorgung des Ausweises erfolgt per induktiver Kopplung. Dementsprechend verfügt der Ausweis über keine autarke Energieversorgung. Das Datenformat der Kommunikation zwischen Lesegerät und nPA ist definiert im Rahmen des mehrteiligen ISO-Standards 7816.[4]

Funktionale und sicherheitsrelevante Anforderungen an den nPA wurden im „CARD"-Schutzprofil (BSI-CC-PP-0061) definiert. Als primäre Schutzgüter werden hier nicht nur die Vertraulichkeit, Integrität und Authentizität der Nutzerdaten definiert, die auf dem Ausweis

[1] BSI TR-03127, S. 8.

[2] ICAO Doc 9303 Pt. 3 Vol. 2.

[3] Im Folgenden wird gelegentlich der Begriff „Near Field Communication (NFC)" gebraucht, wenn es um die Kommunikation zum nPA geht. Tatsächlich ist NFC ein Standard für sich, der aber ISO/IEC 14443-4 mehr oder weniger enthält. D. h. Geräte nach ISO/IEC 14443-4 (z. B. der nPA) sind kompatibel zu Geräten nach NFC.

[4] Siehe insbesondere ISO/IEC 7816-8; ISO/IEC 7816-3; ISO/IEC 7816-4.

(a) Vorderseite (b) Rückseite

Abbildung 2.1.: Der neue Personalausweis (BMI 2010a)

Datei	Inhalt
DG1	Maschinenlesbare Zone, wie auf dem Ausweiskörper aufgedruckt (siehe „Password Authenticated Connection Establishment“, S. 7)
DG2	Digitales Gesichtsbild, identisch mit dem aufgedruckten Bild
DG3	Zwei Fingerabdrücke, falls vom Ausweisinhaber gewünscht, oder andernfalls zufällige Daten
EF.SOD	Signaturen der Datengruppen DG1, DG2 und DG3

Tabelle 2.1.: Ausgewählte Dateien der Biometrieanwendung nach BSI TR-03127, S. 11

gespeichert oder an den Dienstanbieter übermittelt werden.[5] Gleichzeitig soll das Tracing eines Ausweises explizit unterbunden werden, sofern weder Kartenzugriffsnummer (CAN), Maschinenlesbare Zone (MRZ), Geheimzahl für die eID-Anwendung (eID-PIN) noch PIN Unblocking Key (PUK) bekannt sind.[6] Hierzu ist eine vertrauliche und integritätsgesicherte Kommunikation zwischen authentischen Kommunikationspartnern notwendig.[7] Die Nutzerdaten sollen nur autorisierten Parteien zugänglich sein.

Zur elektronischen Nutzung bietet der nPA drei Funktionen an:[8]

eID-Anwendung (eID-Funktion) Die eID-Anwendung dient der Identifizierung und Authentisierung gegenüber dritten Parteien, also nicht-hoheitlichen Stellen. Möglich ist z. B. eine Nutzung im Rahmen von E-Government- und E-Business-Diensten.[9] Der Ausweisinhaber kann mithilfe der eID-Anwendung selektiv Eigenschaften seiner Identität preisgeben. Der elektronische Identitätsnachweis bietet eine Authentisierung ausschließlich für den Zeitpunkt der Nutzung.

Signaturanwendung (eSign-Funktion) Mit der Signaturanwendung kann eine Qualifizierte Elektronische Signatur (QES) nach SigG erstellt werden.[10] Anders als bei bisher verwendeten Signaturkarten wird der nPA erst nach der Auslieferung für die QES aktiviert und Signaturschlüssel generiert. Rechtlich ist die QES der handschriftlichen Unterschrift gleichgestellt sofern dies nicht durch ein Gesetz eingeschränkt wird.[11] Die QES liefert – anders als der elektronische Identitätsnachweis – den dauerhaften Beweis einer Willenserklärung.[12]

Biometrieanwendung (ePassport-Funktion) Die Biometrieanwendung speichert einige der durch ICAO Doc 9303 Pt. 3 Vol. 1 definierten Datengruppen zur hoheitlichen Nutzung (siehe Tabelle 2.1). Diese Daten sind nach der Produktion des Ausweises nicht mehr

[5] BSI-CC-PP-0061, S. 16–17.
[6] Siehe „Tracing“, S. 46.
[7] BSI-CC-PP-0061, S. 17–18.
[8] BSI TR-03127, S. 11–14.
[9] BSI TR-03127, S. 13.
[10] BSI TR-03127, S. 14.
[11] SigG, § 6; beispielsweise ist die handschriftliche Unterschrift bei einer Heirat oder dem Testament notwendig und kann nicht durch eine elektronische ersetzt werden.
[12] „Kartenleser für den neuen deutschen Personalausweis“, S. 765.

veränderbar. Das Auslesen der Daten ist ausschließlich von authentisierten Inspektionsterminals möglich,[13] z. B. zur Ergänzung der visuellen Identifikation durch einen Polizei- oder Grenzbeamten. Obwohl die verwendeten Mechanismen zur Nutzung der Biometrieanwendung nahezu identisch sind mit denen der eID- bzw. Signaturanwendung, werden sie in dieser Arbeit nicht im Detail beleuchtet.

Von diesen drei Funktionen können die eID-Anwendung und die Signaturanwendung im privaten Bereich gebraucht werden. Die eID-Anwendung ist explizit für die Nutzung im Internet vorgesehen.[14]

Mit der Authentifikation per eID-Anwendung sind bereits jetzt die SCHUFA-Auskunft im Internet,[15] die Einsicht in das Punktekonto beim Kraftfahrt-Bundesamt, die Abfrage von Kindergeldleistungen oder dem Rentenkonto sowie verschiedene Bürgerdienste möglich.[16] Die bisher vergebenen Berechtigungszertifikate lassen außerdem darauf schließen, dass die Authentifikation und Identifikation in Online-Netzwerken, Identitätsprüfung bei Online-Petitionen und De-Mail-Diensten, die An- und Abmeldung in sicherheitsrelevanten Gebäudebereichen, der Vertragsabschluss von Domaindienstleistungen sowie die Altersverifikation bei Erotikdiensten geplant ist.[17] Weitere Anwendungen im Bereich E-Business schließen Finanzdienstleistungen bei Banken und Versicherungen oder Online-Handel ein, z. B. die Kontoeröffnung oder den Versicherungsabschluss (insbesondere auch von Lebensversicherungen).[18] Wie die Identifikation gegenüber anderen Nutzern eines Online-Netzwerkes oder die Anwendung gegenüber eines Identifikationsdienstleisters[19] genau aussehen wird, ist noch unklar, da nur die von der Vergabestelle für Berechtigungszertifikate (VfB) zugelassenen Stellen die Daten des nPA auswerten dürfen.[20] An Dritte dürfen die Ausweisdaten nicht weitergegeben werden.

Die praktische Nutzung der Signaturanwendung ist im Gegensatz zur eID-Anwendung derzeit noch nicht möglich. Zum einen ist noch kein Komfortleser[21] mit dem dafür notwendigen Berechtigungszertifikat ausgestattet.[22] Zum anderen gibt es trotz großem Interesse an der Umsetzung der QES mit dem nPA[23] noch keinen Zertifizierungsdienstanbieter (ZDA), der von offizieller Seite eine Freigabe erhalten hat. Anwendungszenarien zur Nutzung der Signaturanwendung dürften aber denen bisher genutzter Signaturkarten entsprechen.

Die Extended Access Control (EAC) schützt die sensiblen Daten der eID- und Signaturanwendung vor unberechtigtem Zugriff.[24] Bevor darauf eingegangen wird, wie sich beide Anwendungen aus Sicht des Ausweisinhabers und aus technischer Sicht gestalten, wird zunächst

[13] BSI TR-03110 2.05, S. 21.
[14] BSI TR-03127, S. 25.
[15] Ein separates Login mittels Benutzernamen und Passwort ist zusätzlich notwendig, um die SCHUFA-Auskunft mit dem nPA zu erhalten.
[16] Borchers 2011a.
[17] VfB 2012.
[18] VfB 2011, 2012.
[19] Cybits AG 2011.
[20] BSI TR-03127, S. 29.
[21] Siehe „Lesegeräteklassen nach BSI TR-03119 1.2“, S. 15.
[22] Das Berechtigungszertifikat muss bei dem einzigen bisher erhältlichen Komfortleser, Reiner SCT cyberJack RFID komfort, noch nachgeladen werden (TÜV NORD 2011).
[23] Schwarz 2011a.
[24] BSI TR-03110 2.05, S. 9.

das EAC-Protokoll betrachtet.

2.1.1. Extended Access Control

Sensible dokumenten- und personenspezifische Daten werden durch die EAC geschützt. Sie soll einerseits sicherstellen, dass sich der nPA in physischer Nähe zum Lesegerät befindet.[25] Andererseits soll dabei auch geprüft werden, dass das Terminal bzw. der Dienstanbieter auch berechtigt ist, auf die geforderten Daten zuzugreifen. Ferner soll die Echtheit des Ausweisdokuments bestätigt werden. Im Rahmen der privaten Nutzung der eID- und Signaturanwendung wird stets die EAC Version 2 nach dem Schema der „General Authentication Procedure" genutzt.[26] Sie besteht aus folgenden Schritten:[27]

1. Password Authenticated Connection Establishment (PACE)
2. Terminalauthentisierung (TA) Version 2
3. Passive Authentisierung
4. Chipauthentisierung (CA) Version 2

Zu beachten ist hier, dass dieses Verfahren inkompatibel ist zur elektronischen Prüfung eines maschinenlesbaren Reisedokuments (MRTD), welches bisher bei Grenzkontrollen eingesetzt wurde. Obwohl PACE mittlerweile für MRTDs standardisiert wurde,[28] schreibt BSI TR-03110 2.05 bei geforderter Konformität zu ICAO Doc 9303 Pt. 3 Vol. 2 die Nutzung von EAC Version 1 mit Basic Access Control (BAC) vor.[29] Im Folgenden wird mit EAC stets EAC Version 2 nach dem Schema der „General Authentication Procedure" gemeint sein. Für weitere Authentifizierungsschemata, die alternativ für die Biometrieanwendung genutzt werden können, siehe BSI TR-03110 2.05, S. 21, 104–106.

Gerade im Rahmen der Online-Authentisierung ist die Unterscheidung zwischen „Local" und „Remote Terminal"[30] wichtig (siehe Abbildung 2.2). Das Remote Terminal weist in der TA seine Berechtigungen nach, Daten aus dem nPA zu lesen oder Funktionen anzusprechen. Das Local Terminal interagiert mit dem Ausweisinhaber. Hier werden die angeforderten Berechtigungen des Remote Terminal angezeigt. Auch die Authentisierung des Ausweisinhabers gegenüber dem nPA erfolgt am Local Terminal.

Eine strikte Trennung muss nicht in jedem Fall vorliegen. Beispielsweise beim Signieren[31] findet sowohl ein Teil der Nutzerintaraktion als auch der Nachweis über die notwendigen Berechtigungen zur Nutzung der Signaturanwendung am bzw. durch den Komfortleser statt. Das *integrierte Terminal*[32] vereint Local und Remote Terminal in einer Instanz. Ich werde im Folgenden „Terminal" kontextabhängig für Local und Remote Terminal gebrauchen. Im Rahmen von PACE ist damit das Local Terminal gemeint. Für die anschließenden Operationen, insbesondere TA und CA, wird damit das Remote Terminal bezeichnet.

[25] BSI TR-03110 2.05, S. 9.
[26] BSI TR-03110 2.05, S. 22.
[27] BSI TR-03110 2.05, S. 25.
[28] ICAO 2010a.
[29] BSI TR-03110 2.05, S. 9.
[30] BSI TR-03110 2.05, S. 26.
[31] Siehe „Nutzung der Signaturanwendung", S. 26.
[32] BSI TR-03112, S. 32–33.

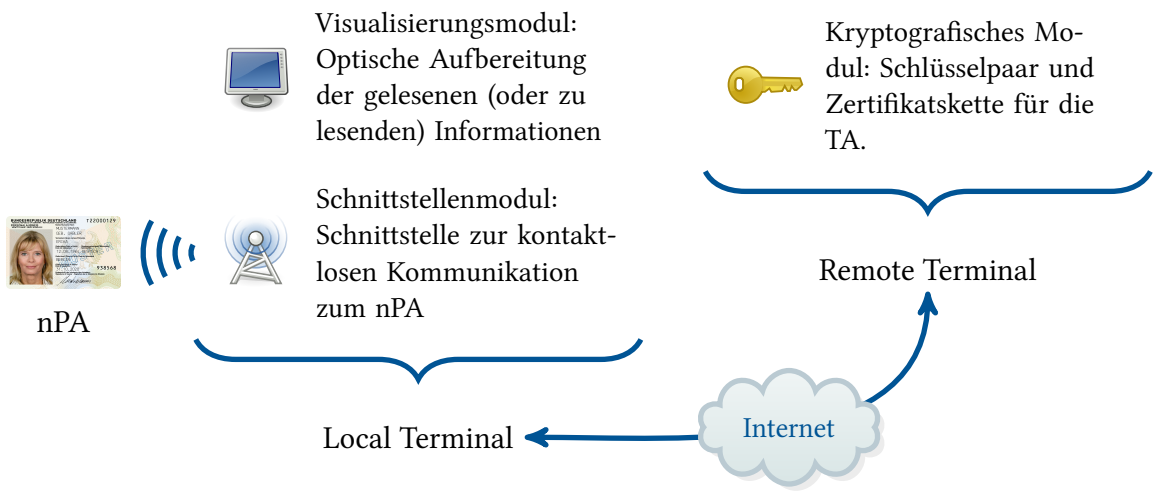


Abbildung 2.2.: Zusammenspiel von Local und Remote Terminal bei Online-Authentisierung

Password Authenticated Connection Establishment

PACE dient der passwortauthentisierten Schlüsselaushandlung und erreicht damit drei Ziele. Erstens kann sich der Besitzer gegenüber dem Ausweis authentifizieren. Zweitens bestimmt der Besitzer willentlich über die Wahl und Eingabe des Geheimnisses in welchem Kontext und zu welchem Zweck der Ausweis ausgelesen werden darf.[33] Drittens sichert das im Anschluss gestartete Secure Messaging (SM) die Kommunikation über die Luftschnittstelle.

PACE soll die bei MRTDs übliche BAC ersetzen.[34] Letztere wird beispielsweise beim deutschen elektronischen Reisepass eingesetzt. Sitzungsschlüssel für die sichere Übertragung der im Chip enthaltenen Daten werden bei BAC ausschließlich von der MRZ geschützt.[35] Diese besitzt theoretisch maximal 73 Bit Entropie. Tatsächlich verringert sich der ohnehin niedrige Wert dadurch, dass in vielen Ländern manche Bestandteile der MRZ nicht zufällig vergeben werden.[36] So wird die MRZ beim deutschen elektronischen Reisepass im Mittel mit 50 Bit Entropie angegeben.[37]

Bei PACE hingegen wird ein Sitzungsschlüssel mit höherer Entropie abgeleitet. Außerdem erlaubt es die Nutzung mehrerer Passworte. Folgende Geheimnisse sind im Rahmen von PACE spezifiziert:[38]

Geheimzahl für die eID-Anwendung (eID-PIN) Wie der Name es andeutet, erlaubt die

[33] BSI TR-03110 2.05, S. 26; BSI TR-03127, S. 22.
[34] BSI TR-03110 2.05, S. 9, 12.
[35] Tatsächlich tauschen MRTD und Terminal Zufallszahlen aus, von denen die Sitzungsschlüssel abgeleitet werden. Es sind diese Zufallszahlen, die mit einem Schlüssel, der von der MRZ abgeleitet wurde, geschützt sind. Einem Angreifer, der den Sitzungsschlüssel erhalten möchte, genügt also die Berechnung der Zufallszahlen, d. h. das Ermitteln der MRZ.
[36] Chothia und Smirnov 2010, S. 25; Bender 2010b, S. 70–72.
[37] Bender 2010a, S. 71.
[38] BSI TR-03110 2.05, S. 24–25.

meist sechsstellige numerische[39] eID-PIN den Zugriff auf die eID-Anwendung.[40] Sie soll ausschließlich dem Besitzer des nPA bekannt sein und wird daher zur Authentisierung des Besitzers gegenüber der Chipkarte verwendet. Der Besitzer des Ausweises erhält nach Auslieferung des Dokuments mit dem PIN-Brief die Transport-PIN. Sie ist nicht zur operationellen Nutzung der eID-Anwendung geeignet.[41] Mit der Transport-PIN wird die sechsstellige eID-PIN eingerichtet und die eID-Anwendung aktiviert.

Die eID-PIN ist mit einem Fehlbedienungszähler (FBZ) versehen, der nach erfolgreichen Eingaben zurückgesetzt wird. Nach zwei Fehlversuchen wird PACE mit der eID-PIN unterbrochen („suspended“) und muss durch einen erfolgreichen Protokolldurchlauf von PACE mit CAN wieder freigegeben werden. Wird auch der dritte und letzte Versuch verwirkt, ist PACE mit der eID-PIN blockiert („blocked“). Nun kann nur durch Eingabe der PUK der FBZ zurück gesetzt werden. Das Ablaufdiagramm[42] 2.3 illustriert das Zusammenspiel von eID-PIN, CAN und PUK im Rahmen der Authentisierung per PACE.

Es gibt zwei Möglichkeiten zum Neusetzen oder Wechseln der eID-PIN. Der Ausweisinhaber kann entweder mittels entsprechender Client-Software (z. B. der AusweisApp) durch Eingabe der alten eID-PIN die Änderung veranlassen. Er kann aber auch die eID-PIN in der Ausweisbehörde neu setzen, wenn er seine eID-PIN vergessen hat. Das Terminal muss in diesem Fall mit entsprechenden Rechten ausgestattet sein.[43]

PIN Unblocking Key (PUK) Die PUK kann höchstens zehnmal genutzt werden, um den FBZ der eID-PIN und der Signatur-PIN[44] zurück zu setzen. Nach jedem erfolgreichen Einsatz der PUK wird deren Rücksetzzähler (RZ) inkrementiert. Die zehnstellige PUK wird bei der Herstellung des Ausweises zufällig erzeugt und ist Bestandteil des PIN-Briefs.

Maschinenlesbare Zone (MRZ) Die Verwendung und Konstruktion der MRZ entspricht der Definition nach ICAO Doc 9303 Pt. 3 Vol. 1, S. V-6–V-9. Für PACE werden allerdings lediglich die Ausweisnummer, das Geburtsdatum des Ausweisinhabers sowie das Ablaufdatum des nPA verwendet.[45] Die MRZ ist ausschließlich zur Verwendung an hoheitlichen Terminals vorgesehen und besitzt keinen Fehlbedienungszähler.

Kartenzugriffsnummer (CAN) Die zufällig generierte CAN ist ebenso wie die MRZ auf den Kartenkörper gedruckt. BSI TR-03110 2.05, S. 24 erlaubt explizit auch die Möglichkeit zur dynamischen Darstellung einer wechselnden CAN auf einem Kartendisplay. Mit der

[39] BSI TR-03110 2.05, S. 93–94 gibt für die einzelnen Zeichen der eID-PIN eine Untermenge von ISO/IEC 88591 vor. Die Begrenzung auf kurze Zahlen ist durch das PIN-Pad-Layout der Kartenleser oder durch die AusweisApp vorgegeben. Tatsächlich erlaubt der Ausweis auch die andere Zeichen wie mit dem `npa-tool` überprüft werden konnte (siehe „nPA Smart Card Library“, S. 99).

[40] Siehe „Nutzung der eID-Anwendung“, S. 20.

[41] BSI TR-03127, S. 17.

[42] Datenflussdiagramm nach DIN 66001

[43] Siehe „Terminalauthentisierung und Extended Access Control Public Key Infrastructure“, S. 12.

[44] Siehe „Nutzung der Signaturanwendung“, S. 26.

[45] BSI TR-03110 2.05, S. 54.

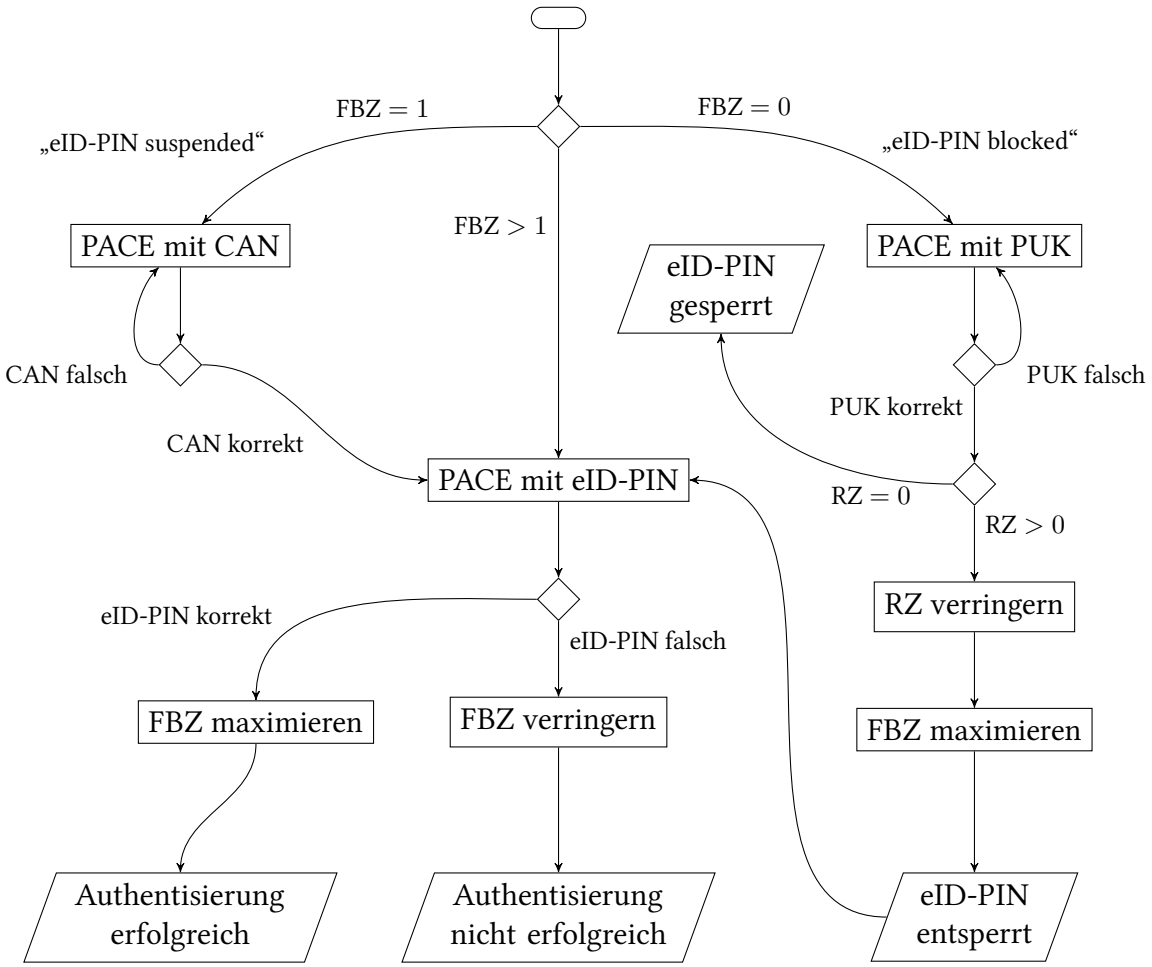


Abbildung 2.3.: Nutzerauthentisierung per eID-PIN unter Berücksichtigung von Fehleingaben. Abhängig vom Fehlbedienungszähler (FBZ) muss die Geheimzahl für die eID-Anwendung (eID-PIN) vor der Nutzung freigeschaltet werden. Nach zwei Fehleingaben (FBZ $= 1$) muss die eID-PIN mit der Kartenzugriffsnummer (CAN) freigeschaltet werden („resume“), nach drei Fehleingaben (FBZ $= 0$) mit der PUK („unblock“). Bei jedem erfolgreichen Gebrauch der PUK wird dessen Rücksetzzähler (RZ) verringert und der FBZ der eID-PIN maximiert (FBZ $:= 3$). Auch eine erfolgreiche Authentisierung per eID-PIN setzt den FBZ zurück.

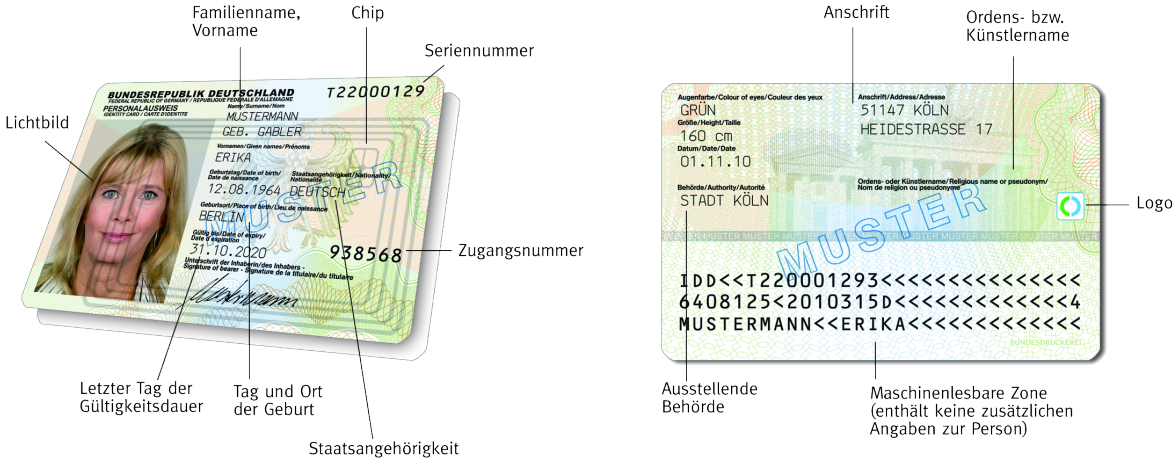


Abbildung 2.4.: Der nPA mit einigen sichtbaren Merkmalen u. a. MRZ und CAN (BMI 2010a)

CAN darf bei entsprechenden Terminalberechtigungen auf alle Anwendungen des nPA zugegriffen werden.[46] Im Normalfall kommt sie allerdings beim Freigeben der eID-PIN nach zwei Fehleingaben sowie im Rahmen der Signatur zum Einsatz.[47] Die CAN selbst ist nicht durch einen Fehlbedienungszähler geschützt.[48]

Es sind diese Geheimnisse, welche nPA und dem Ausweisinhaber bekannt sind, und auf deren Grundlage die Schlüsseletablierung nach PACE vollzogen wird.[49] Der Ablauf von PACE kann in vier Phasen zerlegt werden.

1. Zur *Initialisierung mittels geteiltem Geheimnis* π (eID-PIN, PUK, MRZ oder CAN) erzeugt der nPA eine Nonce r und verschlüsselt sie mit einem von π abgeleiteten Schlüssel. Das Terminal kann nach Eingabe von π durch den Ausweisinhaber die Nonce wieder herstellen.
2. Für die *Erzeugung ephemeraler Domänenparameter* $\widetilde{D}$ bilden r und die statischen Domänenparameter D des nPA die Grundlage. BSI TR-03110 2.05, S. 57 definiert für den nPA das „DH Generic Mapping“ und „ECDH Generic Mapping“. Beide Verfahren nutzen einen Diffie-Hellman-Schlüsselaustausch mit zufällig gewählten Schlüsseln sowie die Verknüpfung mit r, um $\widetilde{D}$ durch Modifikation des Erzeugers bzw. Basispunkts von D zu gewinnen.
3. Die *Schlüsselerzeugung* nutzt einen (weiteren) Diffie-Hellman-Schlüsselaustausch in $\widetilde{D}$. Von dem so ermittelten geteilten Schlüssel K werden K_{Enc} und K_{Mac} zur Verschlüsselung bzw. Integritätssicherung abgeleitet.
4. Der *Authentizitätsnachweis* wird über den soeben etablierten Schlüssel K_{Mac} geführt. Beide Parteien bilden mit ihm eine Prüfsumme über den ephemeralen öffentlichen

[46] BSI TR-03110 2.05, S. 24; Obwohl ein Terminal zwar prinzipiell mit dem Recht „CAN allowed“ auf die Signaturanwendung zugreifen darf, sollen keine Terminalzertifikate ausgestellt werden, die gleichzeitig die Signaturerstellung erlauben (BSI-CC-PP-0061, S. 6).
[47] Siehe „Nutzung der Signaturanwendung“, S. 26.
[48] BSI TR-03110 2.05, S. 24; BSI TR-03127, S. 17.
[49] BSI TR-03110 2.05, S. 32–34.

Schlüssel der Gegenseite.[50] Nach beiderseitiger Verifizierung der Prüfsumme kann mit K_{Enc} und K_{Mac} ein gesicherter Kanal zwischen nPA und Terminal aufgebaut werden.

Nach PACE kommunizieren nPA und Terminal gesichert per SM mit den soeben abgeleiteten Sitzungsschlüsseln. Beide Parteien haben sich gegenseitig über das geteilte Geheimnis authentifiziert.

Die vom Remote Terminal angeforderten Berechtigungen sollen vor der Eingabe des Geheimnisses dem Ausweisinhaber angezeigt werden. Im sogenannten Card Holder Authorization Template (CHAT) sind diese Zugriffsrechte für den Ausweis kodiert und können über das Local Terminal vom Ausweisinhaber weiter eingeschränkt werden.[51] PACE wird mit dem CHAT initialisiert, so dass der nPA sicherstellen kann, dass nach der EAC nur diese (eingeschränkten) Funktionen bzw. Daten für den Dienstanbieter nutzbar sind. Dem Ausweisinhaber werden zudem die Informationen der Zertifikatsbeschreibung des Dienstanbieters angezeigt.[52] Hier enthalten sind insbesondere der Name und die Anschrift des Dienstes sowie der Zweck des Auslesevorgangs.

Analysen zur kryptographischen Sicherheit von PACE werden in Bender, Fischlin und Kügler 2009 und Dagdelen und Fischlin 2010 gegeben. Das Real-or-Random-Sicherheitsmodell zur Evaluation Passwort-basierter Schlüsseletablierung („Password Authenticated Key Exchange“, PAKE) dient dafür als Grundlage.[53] In Dagdelen und Fischlin 2010 wird PACE nach BSI TR-03110 2.02 betrachtet. Auch die derzeit ausgegebenen Ausweise sind konform zu dieser Version der *Technical Guideline TR-03110*.[54] Dagdelen und Fischlin resümieren für PACE folgende Sicherheitseigenschaften:[55]

Passive Sicherheit Unter kryptographischen Annahmen ist es einem Angreifer durch bloßes Abhören nicht möglich, die etablierten Schlüssel oder gar das verwendete geteilte Geheimnis zu erfahren. Die Sitzungsschlüssel besitzen das Niveau von kryptographischen Schlüsseln (und nicht etwa das des lediglich „Passwort-sicheren“ geteilten Geheimnisses).

Aktive Sicherheit Auch wenn der Angreifer aktiv Nachrichten sendet oder modifiziert, ist die Sicherheit der Sitzungsschlüssel nur durch das Auffinden des geteilten Geheimnisses gefährdet. Je PACE-Durchlauf ist hierbei ein Rateversuch möglich („Online-Dictionary-Attack“). Das Ableiten des geteilten Geheimnisses anhand eines aufgezeichneten Protokolldurchlaufs („Offline-Dictionary-Attack“) ist nicht möglich.

Vorwärts-Sicherheit Selbst wenn nach einem PACE-Durchlauf das geteilte Geheimnis dem Angreifer bekannt wird, bleiben die etablierten Sitzungsschlüssel kryptographisch sicher („Forward Security“).

[50] Bis einschließlich BSI TR-03110 2.01 wurden auch die ephemeralen Domänenparameter $\widetilde{G}$ in die Prüfsumme mit einbezogen. Um Tracing-Angriffen vorzubeugen, änderte sich dies mit BSI TR-03110 2.02. Seitdem wird lediglich der zum Protokolldurchlauf gehörende Object Identifier zusätzlich in den Authentizitätsnachweis aufgenommen.

[51] BSI TR-03110 2.05, S. 18.

[52] BSI TR-03127, S. 26.

[53] Bender, Fischlin und Kügler 2009, S. 35–37; Dagdelen und Fischlin 2010, S. 22–23.

[54] BSI-CC-PP-0061, S. 5.

[55] Dagdelen und Fischlin 2010, S. 7–11.

1. Übermittlung der Zertifikatskette bis hin zum Terminalzertifikat des Dienstanbieters zur Validierung durch den nPA.
2. Generierung ephemeraler Schlüssel in den Domänenparametern der Chipauthentisierung (CA)
3. Nachweis über die Authentizität des Terminalzertifikats per Challenge-Response-Verfahren

Abbildung 2.5.: Phasen der Terminalauthentisierung (TA) aus Sicht des Terminals

Vorausgesetzt werden bestimmte Sicherheitsanforderungen bzgl. der Hardware, Personalisierung und der kryptographischen Mechanismen.[56] Die kryptographischen Mechanismen unterliegen kontinuierlichen Bewertung durch das BSI. Die Annahmen bzgl. der Hardwaresicherheit wurden zwar bereits infrage gestellt.[57] In der vorliegenden Arbeit wird aber lediglich auf Annahmen der Personalisierung eingegangen.

Der Ausweisinhaber soll seine eID-PIN echt zufällig wählen und geheim halten. Insbesondere soll er nicht sein eigenes Geburtsdatum wählen sowie kein Geheimnis, das er bereits in einem anderen Kontext verwendet. Das Geheimnis soll auch nur an dafür bestimmten Stellen preisgegeben werden. Diese Forderungen an den Ausweisinhaber werden unter anderem gesetzlich durch PAuswG, § 27 fixiert.

Terminalauthentisierung und Extended Access Control Public Key Infrastructure

Bei der TA weist das Terminal bzw. der Dienstanbieter seine Berechtigungen gegenüber dem nPA nach. Der Dienstanbieter ist dafür in die Extended Access Control Public Key Infrastructure (EAC-PKI) integriert.[58] Die EAC-PKI ist dreistufig gegliedert in Country Verifying Certification Authority (CVCA), Document Verifying Certification Authority (DVCA) und Terminals. Die CVCA stellt den nationalen Vertrauensanker der Infrastruktur dar. Die DVCA ist eine organisatorische Einheit, die zusammengehörende Terminals betreut und ihnen Zertifikate ausstellt. Die Berechtigungen eines Terminals dürfen nicht über die Berechtigungen der übergeordneten DVCA hinausgehen.

Die VfB veranlasst die Ausgabe von Zertifikaten an Dienstanbieter. Sie prüft die Angaben des Dienstanbieters, sein Datenschutzkonzept und die Notwendigkeit der von ihm geforderten Berechtigungen für den Auslesezweck.[59] Nur wenn diese zur Realisierung des Anwendungsszenarios absolut notwendig sind, wird dem Dienstanbieter ein Zertifikat ausgestellt.[60] Weil die Berechtigungen des Dienstes im Zertifikat selbst vermerkt sind, genügt zu ihrem Nachweis der private Schlüssel zum Zertifikat. Diese Berechtigungen werden vor der TA während PACE vom Ausweisinhaber auf das für ihn akzeptable Maß begrenzt.[61]

Damit der nPA überprüfen kann, ob das Terminal den privaten Schlüssel zu einem validen Terminal-Zertifikat besitzt, muss dem Chip zunächst eine Zertifikatskette übermittelt werden.

[56] Dagdelen und Fischlin 2010, S. 79–83.
[57] Plötz 2009.
[58] BSI TR-03110 2.05, S. 13–20; BSI TR-03128, S. 16–37.
[59] BSI TR-03128, S. 43–47.
[60] PAuswG, § 21.
[61] Siehe „Password Authenticated Connection Establishment“, S. 11.

Sie beginnt mit einem CVCA-Link- oder DVCA-Zertifikat passend zum Vertrauensanker des nPA und endet mit dem eigentlichen Terminal-Zertifikat. Durch sukzessive Prüfung der einzeln in der Kette enthaltenen Zertifikate kann der nPA sicherstellen, dass auch das Terminal-Zertifikat von einer vertrauenswürdigen Instanz ausgestellt wurde.

Ein expliziter Rückruf von Terminalzertifikaten über Widerruflisten ist nicht möglich.[62] Stattdessen wurde versucht über den begrenzten Gewährleistungszeitraum („validity period“[63]) von zwei Tagen[64] bzw. einem Monat[65] den Missbrauch von kompromittierten Terminals einzuschränken.[66] Für ein Signaturterminal gilt diese Beschränkung allerdings nicht; das Zertifikat eines Komfortlesers ist (selbst nach einer möglichen Kompromittierung) drei Jahre gültig.[67]

Mit einem Challenge-Response-Verfahren prüft der nPA, dass das Terminal im Besitz des privaten Schlüssels zu dem übermittelten Terminal-Zertifikat ist. Nachdem die Zertifikatskette vom Chip geprüft wurde, sendet er eine zufällige Nonce an das Terminal. Das Terminal generiert seinerseits ein ephemerales Schlüsselpaar in den Domänenparametern des Chips, welches später in der CA verwendet wird. Das Terminal signiert einen Hash gebildet aus Nonce, öffentlichem PACE-Schlüssel des Chips und öffentlichem CA-Schlüssel des Terminals sowie eventueller Zusatzdaten[68] mit seinem privaten Terminalschlüssel.[69] Der Chip kann diese Signatur mit dem extrahierten öffentlichen Schlüssel aus dem Terminalzertifikat verifizieren. Damit ist nicht nur die Authentizität des Terminals gewährleistet, sondern es wurden auch PACE, TA und CA aneinander gebunden. Damit ist es einem Angreifer nicht möglich, verschiedene Teile unabhängiger EAC-Durchläufe miteinander zu verknüpfen.

Chipauthentisierung und Passive Authentisierung

Nachdem sich bei einer erfolgreichen TA das Terminal gegenüber dem Chip authentisiert hat, weist letzterer über die CA seine Authentizität nach. Für den Nachweis besitzt der nPA ein statisches Schlüsselpaar. Der öffentliche Schlüssel und die Domänenparameter für die CA sind im Rahmen einer Dokumenten-PKI digital signiert.[70] Der Dienstanbieter ist nach der TA berechtigt, das `EF.CardSecurity` auszulesen, welches eine Signatur dieser Daten enthält.[71] Die Signatur kann nun vom Terminal mithilfe des Zertifikats der Wurzelinstanz der Dokumenten-PKI sowie Rückruflisten geprüft werden. Mit dieser passiven Authentisierung weist der nPA

[62] BSI TR-03128, S. 56, 59, 62.
[63] BSI TR-03110 2.05, S. 15.
[64] Lesen der eID-Daten oder Pseudonym
[65] Alters- oder Wohnortverifikation, Nachladen der QES
[66] BSI TR-03128, S. 59.
[67] BSI TR-03128, S. 52.
[68] Die sogenannten „Authenticated Auxiliary Data“ werden z. B im Zusammenhang mit der Alters- oder Wohnortverifikation benötigt (BSI TR-03110 2.05, S. 65–66). Im Falle der Altersverifikation wäre dies ein Referenzdatum, zu dem der Chip nach erfolgreicher EAC angibt, ob der Ausweisinhaber älter oder jünger ist. Bei der Wohnortverifikation handelt es sich um den Gemeindeschlüssel (oder einen Teil davon), den der Chip nach der EAC mit dem Wohnort des Ausweisinhabers vergleicht.
[69] Genauer gesagt werden die öffentlichen Schlüssel vor dem Signieren komprimiert, um so Daten fester Länge zu erhalten (BSI TR-03110 2.05, S. 54).
[70] BSI TR-03127, S. 28; ICAO Doc 9303 Pt. 3 Vol. 1.
[71] BSI TR-03110 2.05, S. 48.

dem Terminal nach, dass sein CA-Schlüssel von einer vertrauenswürdigen Instanz signiert wurde und es sich damit um einen echten Ausweis handelt.

Die eigentliche CA verwendet ähnliche Verfahren wie PACE:[72]

1. Für die *Berechnung der Sitzungsschlüssel* nutzt der nPA sein statisches CA-Schlüsselpaar und das Terminal sein während der TA generiertes ephemerales CA-Schlüsselpaar.[73] Mit ihrem geheimen und dem öffentlichen Schlüssel der Gegenseite wird ein geteiltes Geheimnis K berechnet. Von K und einem vom nPA zufällig generiertem Seed werden die Sitzungsschlüssel K_{Enc} und K_{Mac} abgeleitet.
2. Der gegenseitige *Authentizitätsnachweis* wird wie bei PACE[74] mit dem öffentlichen Schlüssel der Gegenseite über den Schlüssel K_{Mac} geführt. Nach beiderseitiger Prüfung kann mit K_{Enc} und K_{Mac} ein gesicherter Kanal zwischen nPA und Terminal aufgebaut werden, der den zuvor durch PACE etablierten Kanal ersetzt.

Der nPA hat damit die Kenntnis seines privaten statischen CA-Schlüssels nachgewiesen.

Weil ein Tracing des nPA explizit unterbunden werden soll,[75] ist es notwendig, dass der Nachweis über den privaten Schlüssel bzw. das Schlüsselpaar selbst kein Ausweis-eindeutiges Merkmal ist. Daher teilen sich mehrere Ausweise das selbe Schlüsselpaar. Angenommen die Produktion der Ausweise verteilt sich gleichmäßig über ein Jahr, es gibt stets nur genau einen Document Signer und das Anonymity Set soll eine Größe von 1,5 Millionen haben, dann würde sich *mindestens* ein vierteljährlicher Wechsel des CA-Schlüssels ergeben.[76] Dem Ausweis liegt außerdem der öffentliche Schlüssel des Document Signers bei. Die „Private Key Usage“ ist bei aktuellen Ausweisen auf 1,5 Jahre festgelegt,[77] d. h. derselbe CA-Schlüssel dürfte *höchstens* über diesen Zeitraum an mehrere Ausweise vergeben werden.

Wie auch PACE wurde die CA als Ersatz eines Verfahrens zur Authentizitätsprüfung von MRTDs konzipiert. Die aktive Authentisierung (AA) wird z. B. beim deutschen elektronischen Reisepass eingesetzt und weist die Authentizität des Dokuments mit einem Challenge-Response-Verfahren nach. Das Terminal sendet hierbei eine zufällige Challenge, die der ePass signiert. Ein böswilliges Terminal könnte aber die signierten Daten auch mit einer Semantik belegen.[78] Hier könnten Zeitpunkt und Ort des Auslesevorgangs kodiert sein. Das Terminal könnte dann die vom Ausweis signierten Daten nutzen, um diesen wiederzuerkennen (Tracing) und ein Aufenthaltsprofil des Ausweisinhabers erstellen. Verschlimmert wird das Problem noch dadurch, dass die AA auch von nicht autorisierten Terminals durchgeführt werden kann.

Das Gegenteil liegt bei der beim nPA genutzten CA Version 2 vor. Sie steht nur autorisierten Terminals zur Verfügung, d. h. nach der TA. Zudem werden nicht beliebige Daten mit dem privaten Schlüssel des nPA signiert. Es wird ein Schlüsselaustauschverfahren durchgeführt und der Nachweis über die Authentizität des privaten CA-Schlüssels des nPA erfolgt über die etablierten Sitzungsschlüssel. Die Authentisierungsdaten sind ephemeral, und können dem

[72] BSI TR-03110 2.05, S. 34–36; BSI TR-03127, S. 10.

[73] Siehe „Terminalauthentisierung und Extended Access Control Public Key Infrastructure“, S. 13.

[74] Siehe „Password Authenticated Connection Establishment“, S. 10.

[75] BSI TR-03110 2.05, S. 12.

[76] Insgesamt 60 Mio. Ausweise (VfB 2011) sind je 10 Jahre gültig. Das heißt 1,5 Mio. Ausweise werden nach $\frac{10\text{a}}{60\text{Mio.}} \cdot 1,5\text{Mio.} \approx \frac{1}{4}\text{a}$ produziert und mit dem gleichen CA-Schlüssel versehen.

[77] Siehe „EF.CardSecurity“, S. 131.

[78] BSI TR-03110 2.05, S. 109.

(a) HID Global OMNIKEY 5321 V2 (b) SCM SDI011 (c) ACS ACR1281U

Abbildung 2.6.: Beispiele für zertifizierte Basisleser (Quellen: HID Global, SCM Microsystems, ACS)

Ausweis (-inhaber) nicht über verschiedene Sitzungen hinweg zugeordnet werden.

2.1.2. Lesegeräteklassen nach BSI TR-03119 1.2

Zur heimischen Nutzung des nPA ist zwingend ein Lesegerät notwendig, das die Kommunikation zum Ausweis erlaubt. In BSI TR-03119 1.2 werden mit Hinblick auf die unterschiedlichen Anwendungszenarien drei Profile von Lesegeräteklassen definiert. An Basis-, Standard- und Komfortleser werden verschiedene funktionale und sicherheitsrelevante Anforderungen gestellt. Als Zertifizierungsgrundlage wird die Technische Richtlinie genutzt.[79]

In BSI TR-03119 1.2, S. 5 heißt es:

> Die wichtigste Anforderung an einen Chipkartenleser ist der fehlerfreie, störungsfreie und zuverlässige Betrieb sowie die Unversehrtheit der Chipkarten. Dazu sind bei kontaktlosen und kontaktbehafteten Chipkartenlesern weitere Anforderungen und Funktionen zur Sicherstellung der Interoperabilität notwendig. Ebenso muss die Informationssicherheit berücksichtigt werden, um die Vertraulichkeit und die Integrität der Abläufe und der Kommunikation gewährleisten zu können.

Die Anforderungen werden dabei in mehrere Module gegliedert. Basis-, Standard- und Komfortleser müssen verschiedene Untermengen aller möglichen Module erfüllen, weitere sind optional. In Tabelle 2.2 wird eine Übersicht über die Leserklassen und ihre Anforderungen gegeben. Ich werde im Folgenden die Begriffe Basis-, Standard- und Komfortleser äquivalent benutzen zu Basis-Chipkartenleser (CAT-B), Standard-Chipkartenleser (CAT-S) und Komfort-Chipkartenleser (CAT-K), welche in BSI TR-03119 1.2 definiert werden. Gemeint ist damit das jeweilige Profil an Mindestanforderungen ohne optionale Zusätze.

Zu jedem der Module werden ebenso Prüfvorschriften gegeben.[80] Die Konformitätsprüfung wird von unabhängigen Prüfstellen, die vom BSI anerkannt sind, durchgeführt. Die abschließenden Prüfberichte sind Grundlage für die Zertifizierung durch das BSI und online einsehbar.[81] Fast alle aktuellen Kartenleser wurden bisher nur nach BSI TR-03119 1.1 bestätigt.

[79] BSI TR-03119 1.2, S. 5.
[80] BSI TR-03119 1.2, S. 23–30.
[81] BSI 2011.

(a) REINER SCT cyberJack RFID standard (b) REINER SCT cyberJack RFID komfort

Abbildung 2.7.: Die bisher einzigen zertifizierten Standard- und Komfortleser

Im hoheitlichen und behördlichen Einsatz dürfen anders als im privaten Bereich ausschließlich zertifizierte Produkte verwendet werden.[82]

Ein Basisleser muss im Wesentlichen den Datenaustausch zwischen dem nPA und dem jeweiligen Anwendungsserver im Internet bereitstellen.[83] Als mögliche Anwendungen werden e-Government Dienste, Altersverifikation, eTicketing, Identitätsnachweis beim Onlineshopping und Postident-Ersatz genannt.[84] Ein Standardleser bietet die Nutzung der eID-Funktion mit erhöhten Sicherheitsanforderungen.[85] Dafür besitzt dieser insbesondere die Möglichkeit zur sicheren PIN-Eingabe. Ein Komfortleser bietet darüber hinaus ein Display und die Möglichkeit zur kontaktbasierten und kontaktlosen QES nach SigG.

Bei der Realisierung der einzelnen Funktionsmodule soll auf ein ausgewogenes Verhältnis von Kosten und Nutzen geachtet werden.[86] Je nach Anwendungszweck kann auf ein PIN-Pad oder Display am Chipkartenleser verzichtet werden. Auch anwenderfreundlich soll die Nutzung der Geräte sein. Die Installation soll für Laien verständlich sein und ein technischer Support angeboten werden. Weiterhin gehört zu den Grundlagenanforderungen, dass PIN-Pad und Display – sofern vorhanden – wirksam vor Manipulationen geschützt werden oder diese durch den Inhaber leicht erkannt werden können. Leuchtdioden zur Anzeige der Funktionsbereitschaft und des sicheren Modus (SM) besitzen als Nutzerschnittstellen einen eigenen Abschnitt, genauso wie Anmerkungen zum optimalen PIN-Pad-Layout.[87] Einige für den sicheren Betrieb wichtigen Funktionsmodule werden im Folgenden näher betrachtet.

[82] BSI TR-03119 1.2, S. 7.
[83] BSI TR-03119 1.2, S. 9.
[84] BSI TR-03119 1.2, S. 8.
[85] BSI TR-03119 1.2, S. 9.
[86] BSI TR-03119 1.2, S. 11.
[87] BSI TR-03119 1.2, S. 12–13.

Module	CAT-B	CAT-S	CAT-K
Schnittstelle zum Host-Rechner	✓	✓	✓
kontaktlose Schnittstelle	✓	✓	✓
kontaktbehaftete Schnittstelle			✓
PIN-Pad (sichere PIN-Eingabe) mit PACE-Unterstützung		✓	✓
Display (mindestens 32 alphanumerische Zeichen)			✓
Qualifizierte Signatur mit kontaktbehafteten Karten			✓
Qualifizierte Signatur mit kontaktlosen Karten (z.B. nPA)			✓
Firmwareupdate		✓	✓

Tabelle 2.2.: Übersicht der Chipkartenleser-Kategorien nach BSI TR-03119 1.2, S. 8. Module, die hier für Basis-Chipkartenleser (CAT-B), Standard-Chipkartenleser (CAT-S) oder Komfort-Chipkartenleser (CAT-K) nicht als verpflichtend gekennzeichnet sind (✓), sind optional.

Das verbreitetste Framework zur Abstraktion von Chipkartenlesegeräten ist der PC/SC-Standard. Im Rahmen des Moduls *Schnittstelle zum Host-Rechner*[88] werden die Kommandos `GetReaderPACECapabilities` und `EstablishPACEChannel` definiert, die vom Gerätetreiber eines Standard- oder Komfortlesers bereitgestellt werden müssen.[89] Die Chipkartenapplikation kann über diese PC/SC-Funktionen nun die notwendigen Daten zur sicheren Authentisierung an das Lesegerät senden. Der Ausweisinhaber gibt die eID-PIN am Lesegerät ein und letzterer baut daraufhin den PACE-Kanal auf. Die Eingangsdaten von `EstablishPACEChannel` umfassen Informationen über den Dienstanbieter und seiner angefragten und vom Ausweisinhaber eingeschränkten Zugriffsrechte. Außerdem kann die Chipkartenapplikation dem Leser bereits eine CAN vorgeben, um dem Ausweisinhaber die manuelle Eingabe zweier Geheimnisse in einigen Fällen zu ersparen.[90] Das kürzlich erschienene Dokument PC/SC Pt. 10 amd. 1 definiert zusätzlich zu den Vorgaben des BSI das Kommando `DestroyPACEChannel` zum Terminieren eines sicheren Kanals.[91]

Neben den Kommandos zum Auslösen von PACE wird außerdem die Abfrage des Hersteller- und Produktnamen sowie der Firmware- und Treiberversion definiert (`GetReaderInfo`).[92] Dazu werden ähnlich wie in PC/SC Pt. 3 Pseudo-APDUs genutzt, die mit dem Class-Byte `0xFF` beginnen. Die Informationen aus `GetReaderInfo` wird von der AusweisApp zur Identifikation des Lesers genutzt. Für Chipkartenleser, die keine sichere PIN-Eingabe für PACE unterstützen, genügt die Bereitstellung der PC/SC-Schnittstelle ohne die Erweiterungen aus BSI TR-03119 1.2 bzw. PC/SC Pt. 10 amd. 1.

Über die besprochene PC/SC-Schnittstelle wird der Chipkartenapplikation der Zugriff auf

[88] BSI TR-03119 1.2, S. 31–36.

[89] BSI TR-03119 1.2, S. 14–15.

[90] Die Eingabe von CAN *und* eID-PIN ist notwendig, wenn letztere zweimal falsch eingegeben wurde. Die Eingabe sowohl von CAN und Signatur-PIN ist für die Erstellung einer QES notwendig.

[91] Einige Chipkartenleser schließen einen PACE-Kanal implizit nach einer Transaktion, so dass die Umsetzung von `DestroyPACEChannel` nicht zwingend vorgeschrieben ist (PC/SC Pt. 10 amd. 1, S. 6,8)

[92] BSI TR-03119 1.2, S. 14–15.

GetReaderPACECapabilities	
Name	Beschreibung
BitMap	Chipkartenleser unterstützt… • sichere PIN-Eingabe und PACE (Bit 7, 0x40) • eID, ggf. mit Anzeige von eID-Daten (Bit 6, 0x20) • QES mit kontaktlosen Karten (Bit 5, 0x10)

EstablishPACEChannel	
Name	Beschreibung
PinID	Art des Passworts für PACE, also MRZ, CAN, eID-PIN oder PUK
CHAT	Vom Ausweisinhaber eingeschränkter CHAT
PIN	Das Passwort selbst (optional, nur CAN oder MRZ)
CertificateDescription	Zertifikatsbeschreibung des Dienstanbieters (nur für eID)

Tabelle 2.3.: Eingabedaten für GetReaderPACECapabilities und EstablishPACEChannel (ohne Längeninformationen)

den Kartenleser erlaubt. Das Betriebssystem stellt hierfür einen Dienst zur Verfügung, der entsprechende Aufrufe an den Gerätetreiber des Chipkartenlesers weitergibt. Der Treiber wiederum übernimmt die Kommunikation zum Hardware-Interface. In BSI TR-03119 1.1, S. 36 werden hier Universal Serial Bus (USB), RS-232 oder Bluetooth als Beispiele genannt. Von Morgner u. a. werden für die mobile Nutzung des nPA auch andere Möglichkeiten wie die Anbindung per GSM/UMTS oder Netzwerkverbindung vorgeschlagen.[93] Die Beschränkung, dass das Interface zum Host-System „geprüft und qualifiziert werden“[94] soll, wird in der neuen Technischen Richtlinie weder als Funktions- noch Prüfmodul genannt. Daher wird hier davon ausgegangen, dass eine entsprechende Prüfung nicht Gegenstand der Zertifizierung des BSI ist.

Das Modul *PIN-Pad mit PACE-Unterstützung*[95] spezifiziert die Eingabe von Passworten direkt an der Tastatur des Chipkartenlesers. Der Chipkartenleser muss für dieses Modul das PACE-Verfahren und SM gemäß *Technical Guideline TR-03110* implementieren.[96] Zur Erzeugung der ephemeralen PACE-Schlüssel muss ein zertifizierter Pseudozufallszahlengenerator[97] verwendet werden. Unmittelbar nach der Verwendung sollen Passworte und Schlüsselmaterial aus dem internen Speicher des Lesers gelöscht werden. Das Modul schließt seit dem Erscheinen von BSI TR-03119 1.2 auch Filterregeln von Verbindungen ausgehend vom Host-Rechner und Manipulationsschutz des PIN-Pads ein. Die Filterregeln sollen sicherstellen, dass PACE mit eID-PIN oder PUK ausschließlich durch die Passwort-Eingabe an dem PIN-Pad des Lesers erfolgt. Die Qualität des Manipulationsschutzes orientiert sich an den Vorgaben aus dem

[93] Morgner u. a. 2011, S. 232.
[94] BSI TR-03119 1.1, S. 36.
[95] BSI TR-03119 1.2, S. 17–18.
[96] Siehe „Password Authenticated Connection Establishment“, S. 7.
[97] Klasse K3, (BSI AIS 20)

Signaturrecht.[98]

Manipulationssicherheit[99] ist ebenfalls für das Modul *Display*[100] erforderlich. Die Anzeige muss in mindestens zwei Zeilen mit je sechzehn Zeichen möglich sein. Standardisierte Anzeigetexte und ein blinkender Cursor bei erwarteten Nutzereingaben sollen zur einheitlichen Nutzerinteraktion bei verschiedenen Modellen von Standard- oder Komfortlesern beitragen. Sicherheitsrelevant ist die Anzeige von Berechtigtem und Berechtigungen bei der eID-Nutzung. Dazu müssen aus der Zertifikatsbeschreibung des Dienstanbieters und dem CHAT die Daten extrahiert werden und auf dem Display dargestellt werden. Der Ausweisinhaber bestätigt die Verbindungsdetails am Lesegerät.

Das Modul *QES mit kontaktbehafteten Karten*[101] entspricht den Vorgaben aus der Signaturverordnung. Die *QES mit kontaktlosen Karten gemäß TR-03117*[102] erfordert die Implementierung der vollständigen EAC. Der Schlüssel, der den Komfortleser bei der TA als bestätigtes Signaturterminal ausweist, muss im Rahmen der Berechtigungs-PKI signiert sein und in einem zertifizierten Sicherheitsmodul[103] gespeichert werden. Die auf das Sicherheitsmodul zugreifende Firmware soll ebenfalls sicher sein.[104] Mittels der PC/SC-Kommandos `PIN_VERIFY` und `PIN_MODIFY`[105] wird die Nutzerauthentisierung mittels Signatur-PIN sowie die Änderung der Signatur-PIN ausgelöst. Filterregeln des Chipkartenlesers sollen die Eingabe der Signatur-PIN am PIN-Pad erzwingen. Nach erfolgreicher EAC und Authentisierung des Ausweisinhabers wird nach BSI TR-03117 eine unverschlüsselte Application Protocol Data Unit (APDU) mit den zu signierenden Daten an den Leser gegeben. Dieser schützt die APDU mit SM und sendet sie an den nPA. Die Antwort wird vom Komfortleser entschlüsselt und die signierten Daten zurück an die Chipkartenapplikation geleitet.

Das Modul *Firmware-Update*[106] regelt die Aktualisierung der im Chipkartenleser verwendeten Software. Der Chipkartenleser soll mittels kryptographischer Verfahren die Vollständigkeit und Integrität eines Updates sicherstellen. Weiterhin dürfen nur autorisierte Personen oder Systeme Veränderungen am Chipkartenleser vornehmen. Die spätere Verwendung des Chipkartenlesers bestimmt das hierfür erforderliche Sicherheitsniveau.[107] Ein Beispiel für ein hohes Sicherheitsniveau wäre das Nachladen des Terminalschlüssels für die Signatur beim REINER SCT cyberJack RFID komfort.[108]

Chipkartenleser, die für eine bestimmte Klasse zertifiziert wurden, können über die obligatorischen Module hinaus weitere implementieren. Der Standardleser REINER SCT cyberJack RFID standard bietet beispielsweise zusätzlich die QES mit kontaktbehafteten Karten sowie

[98] EAL3 bzw. EAL3+ für Chipkartenleser mit QES-Unterstützung (BSI TR-03119 1.2, S. 18, 37)
[99] EAL3 bzw. EAL3+ für Chipkartenleser mit QES-Unterstützung (BSI TR-03119 1.2, S. 24, 37)
[100] BSI TR-03119 1.2, S. 18–20.
[101] BSI TR-03119 1.2, S. 20.
[102] BSI TR-03119 1.2, S. 20–22.
[103] EAL4+ nach CEN/ISSS 2009, Ausnahmen sind möglich (BSI TR-03119 1.2, S. 24)
[104] Empfohlen wird nach BSI-CC-PP-0064 mindestens mit EAL3, bei implementierten Importmechanismus für neue Schlüssel EAL4+ (BSI TR-03119 1.2, S. 24–25).
[105] PC/SC Pt. 10.
[106] BSI TR-03119 1.2, S. 22.
[107] EAL3 bzw. EAL3+ für Chipkartenleser mit QES-Unterstützung, falls eine sicherheitsrelevante Änderung vorgenommen wird (BSI TR-03119 1.2, S. 25, 37)
[108] TÜV NORD 2011.

	Bezeichnung	Klasse	Preis
	REINER SCT cyberJack RFID komfort (IT-Sicherheitskit komfort)	Komfortleser	159,90 €
	REINER SCT cyberJack RFID standard (IT-Sicherheitskit standard)	Standardleser	68,50 €
	REINER SCT cyberJack RFID basis (IT-Sicherheitskit basis)	Basisleser	34,90 €

Tabelle 2.4.: Preise aktueller Chipkartenlesegeräte (ohne Förderung, REINER SCT 2011).

ein Display an (siehe Abbildung 2.7a).

Mit Mitteln aus dem Konjunkturpaket II versucht die Bundesregierung, „eine signifikante Anzahl an Lesegeräten für die Nutzung elektronischer Chipkarten verfügbar“[109] zu machen. In sogenannten „IT-Sicherheitskits“ wurden bereits zahlreiche kostenlose Basisleser verteilt.[110] Die Ausgabe von vergünstigten Standard- oder Komfortlesern ist ebenfalls angelaufen. Insgesamt bis zu 1.237.000 Basisleser und 230.000 Standard- und Komfortlesern werden auf diese Weise gefördert.[111] Einen Überblick über die Kosten beim Kauf eines Chipkartenlesers wird in Tabelle 2.4 gegeben. Hier zeigt sich, dass die „Optimierung von Kosten und Nutzen“[112] bei einfachen Lesern dazu führt, dass z. B. auf ein Anzeigemodul oder ein sicheres PIN-Pad verzichtet wird. Standard- oder Komfortleser erfordern größere Investitionen des Ausweisinhabers, schützen aber Passworte und Zugriff auf den Ausweis besser als ein Basisleser.

2.1.3. Nutzung der eID-Anwendung

Auch wenn gegenüber dem Ausweis lediglich die EAC benötigt wird, um ihn z. B. für eine Online-Authentisierung zu nutzen, gestaltet sich die Ausweisnutzung im Internet recht komplex. Der Ausweisinhaber muss zunächst den Ausweis mithilfe der Transport-PIN aktivieren. Dies stellt die erste Nutzung der eID-Anwendung dar, um sich per PACE gegenüber dem Ausweis zu authentisieren. Ohne nun TA oder CA durchführen zu müssen, wird es dem *nicht authentisierten Terminal*[113] („Unauthenticated Terminal“[114]) gestattet, die eID-PIN initial zu setzen. Neben diesem einmaligen Vorgang kann auch der Wechsel der aktuellen eID-PIN und das Rücksetzen des FBZ der eID-PIN ebenfalls mit einem nicht authentisierten Terminal durchgeführt werden (siehe Tabelle 2.5).

Im Umgang mit dem nPA wird eine Client-Software nach BSI TR-03112 benötigt. Sie steuert den Chipkartenleser nach BSI TR-03119 1.2 an und vermittelt so zwischen Dienstanbieter, nPA und Ausweisinhaber. Die Einheit aus Chipkartenlesegerät und Computer bzw. Client-Software

[109] BMI 2010b.
[110] MoOe10
[111] BMI 2010c.
[112] BSI TR-03119 1.2, S. 11.
[113] BSI TR-03127, S. 24–25.
[114] BSI TR-03110 2.05, S. 27–28.

Abbildung 2.8.: Nutzung der AusweisApp zum elektronischen Identitätsnachweis im Internet mit einem Basisleser (BMI 2010a).

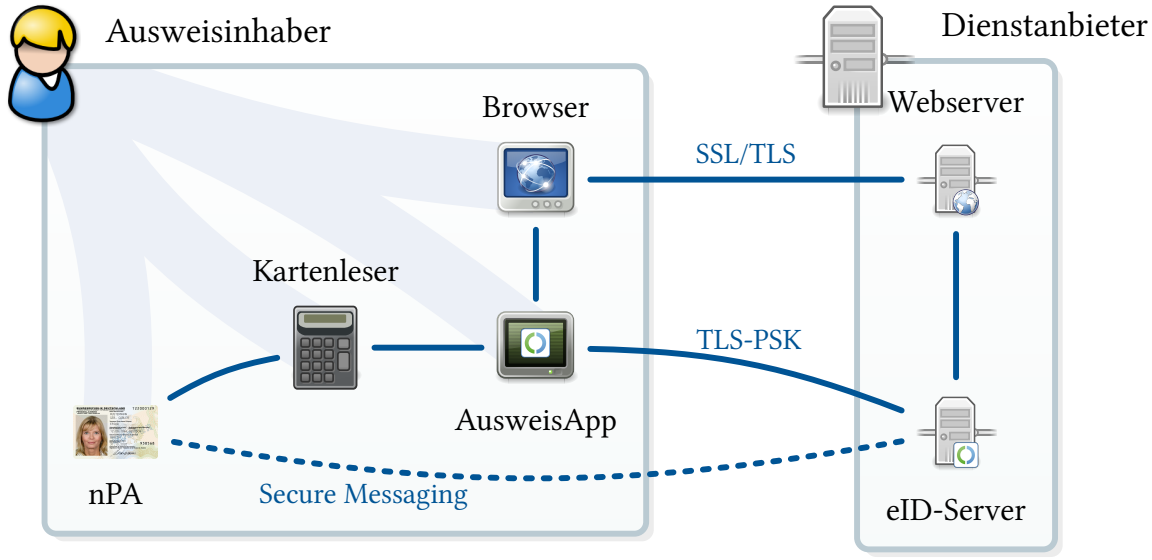


Abbildung 2.9.: Kommunikationspartner und -kanäle bei der eID-Nutzung. Der Ausweisinhaber baut mit seinem Webbrowser eine SSL/TLS-Verbindung zum Webserver des Dienstanbieters auf. Über diesen gesicherten Kanal wird die Kommunikation zwischen Client-Software und eID-Server initialisiert. Der eID-Server liest die geschützten Daten nach erfolgreicher EAC per SM aus dem nPA.

des Ausweisinhabers bilden das Local Terminal, der Dienstanbieter das Remote Terminal.[115]

Für die Online-Authentisierung nimmt der Ausweisinhaber über seinen Webbrowser Kontakt zum Dienstanbieter auf. Diese Verbindung ist per Secure Sockets Layer bzw. Transport Layer Security (SSL/TLS) gesichert. Der Webserver des Dienstanbieters leitet die Verbindung zumeist zu einem eID-Server[116] um. Letzterer ist in die Berechtigungs-PKI integriert und besitzt insbesondere das für die TA notwendige Berechtigungszertifikat samt privatem Schlüssel. Er kann vom Dienstanbieter selbst betrieben werden oder es kann sich hierbei um einen beauftragten Dienst handeln. Die Verbindung von der Client-Software zum eID-Server wird ebenfalls mit SSL/TLS gesichert. Dabei wird vom Webserver des Dienstanbieters ein zufälliges Geheimnis erzeugt, das sowohl dem Webbrowser des Ausweisinhabers als auch dem eID-Server sicher übermittelt wird. Die gesicherte Kommunikation zwischen Client-Software und eID-Server entsteht im Modus TLS-PSK und wird damit über dieses Geheimnis („Preshared Secret“) an den ersten sicheren Kanal zwischen Browser und Webserver gebunden.

Der eID-Server führt nun Remote Procedure Calls (RPCs) durch, um bei der Client-Software die notwendigen Operationen zur Authentisierung zu initiieren.[117] Das dazu genutzte Nachrichtenformat Security Assertion Markup Language (SAML) wird mit dem Reversed HTTP binding for SOAP (PAOS)[118] genutzt. Authentisierung und Authentifizierung mit dem nPA per

[115] Zur Unterscheidung zwischen Local und Remote Terminal siehe auch „Extended Access Control“, S. 6.
[116] BSI TR-03130.
[117] BSI TR-03130, S. 49–76.
[118] BSI TR-03130, S. 54.

SAML nutzt also das XML-basierte Simple Object Access Protocol (SOAP). Weil die Client-Software aber durchaus auch von einer Firewall oder einem NAT aus geschützt operieren kann, ist das direkte Senden von RPCs des eID-Servers unter Umständen nicht möglich. Dieses Problem umgeht PAOS, indem die Client-Software die Verbindung aufrechterhält und sich den RPC sozusagen „abholt". Die Umsetzung der RPCs per Webservices ist ebenfalls spezifiziert,[119] wird im Moment aber von keinem dem Autor bekannten Dienst genutzt.

In der Client-Software wird zunächst die Nutzerauthentisierung ausgelöst. D. h. dem Ausweisinhaber werden die Details zum Dienstanbieter und Auslesevorgang angezeigt und er wird zur Eingabe der eID-PIN aufgefordert. Damit baut die Client-Software bzw. der genutzte Chipkartenleser den PACE-Kanal zum nPA auf. Die TA und CA werden vom eID-Server durchgeführt. Nachdem die EAC erfolgreich abgeschlossen wurde, besteht ein sicherer Ende-zu-Ende-Kanal zwischen eID-Server und nPA. Die Ausweisdaten bzw. -funktionen können ausschließlich vom eID-Server entsprechend seiner Berechtigungen genutzt werden. Eine Übersicht der bei der eID-Nutzung involvierten Kommunikationspartner und -kanäle ist in Abbildung 2.9 gegeben. Die wesentlichen Ablaufschritte sind in Abbildung 2.10 dargestellt.

Über die Dokumenten- und Berechtigungs-PKI wird die Verbindung zwischen eID-Server und nPA abgesichert. Beide werden nach öffentlichen Standards bewertet, geprüft und zertifiziert. Die SSL/TLS-PKI hingegen zeigte in der Vergangenheit des öfteren Schwachstellen.[120] Um diese Probleme zu umgehen, wird der Hash des SSL-Zertifikats des Dienstanbieters im Berechtigungszertifikat gespeichert. In der TA prüft der nPA die Signatur des Berechtigungszertifikats (und damit den Hash des SSL-Zertifikats). Die Client-Software soll nun den aus dem Berechtigungszertifikat vorgegebenen Hash mit dem des SSL-Zertifikats des Dienstanbieters vergleichen.[121] Die unabhängige Absicherung der Authentizität des Webservers über die Berechtigungs-PKI verhindert insbesondere einen Man-in-the-Middle-Angriff mit einem anderen (z. B. ungültigen oder gefälschten) SSL/TLS-Zertifikat.[122] Die Bindung des SSL-Kanals an die EAC wurde mit BSI TR-03110 2.04 eingeführt. Es ist eine offene Forschungsfrage, ob dies zu Problemen mit den im Feld befindlichen Ausweisen führen kann, welche nach BSI TR-03110 2.02 zertifiziert sind.

Nachdem die EAC zwischen nPA und eID-Server erfolgreich abgeschlossen wurde, ist die Authentizität beider Kommunikationspartner sichergestellt. Sie etablieren eine vertrauliche und integritätsgeschützte Kommunikation per SM. Der eID-Server darf nun die vom Ausweisinhaber autorisierten Operationen durchführen, d. h. entsprechende Ausweisdaten auslesen oder Ausweisfunktionen aufrufen. Das Ergebnis dieses Auslesevorgangs wird dem Dienstanbieter über eine sichere Verbindung vom eID-Server mitgeteilt.

Die Nutzung der eID-Anwendung ist damit abgeschlossen. Der auf ihrer Grundlage entstandene sichere SSL/TLS-Kanal zum Dienstanbieter kann nun aber auch für weitere Operationen genutzt werden, die ausschließlich dem rechtmäßigen Ausweisinhaber möglich sein sollen.[123]

[119] BSI TR-03130, S. 19–46.
[120] Bachfeld 2011; Soghoian und Stamm 2010; Stevens u. a. 2009; VASCO Data Security International, Inc. 2011.
[121] BSI TR-03127, S. 30; BSI TR-03110 2.05, S. 87.
[122] Vgl. Dietrich, Rossow und Pohlmann 2010, S. 8–9.
[123] In „Die elektronische Nutzung des neuen Personalausweises", S. 5 werden eine Reihe von Beispielen genannt, die mit der Nutzung der eID-Anwendung umgesetzt wurden oder deren Umsetzung geplant ist.

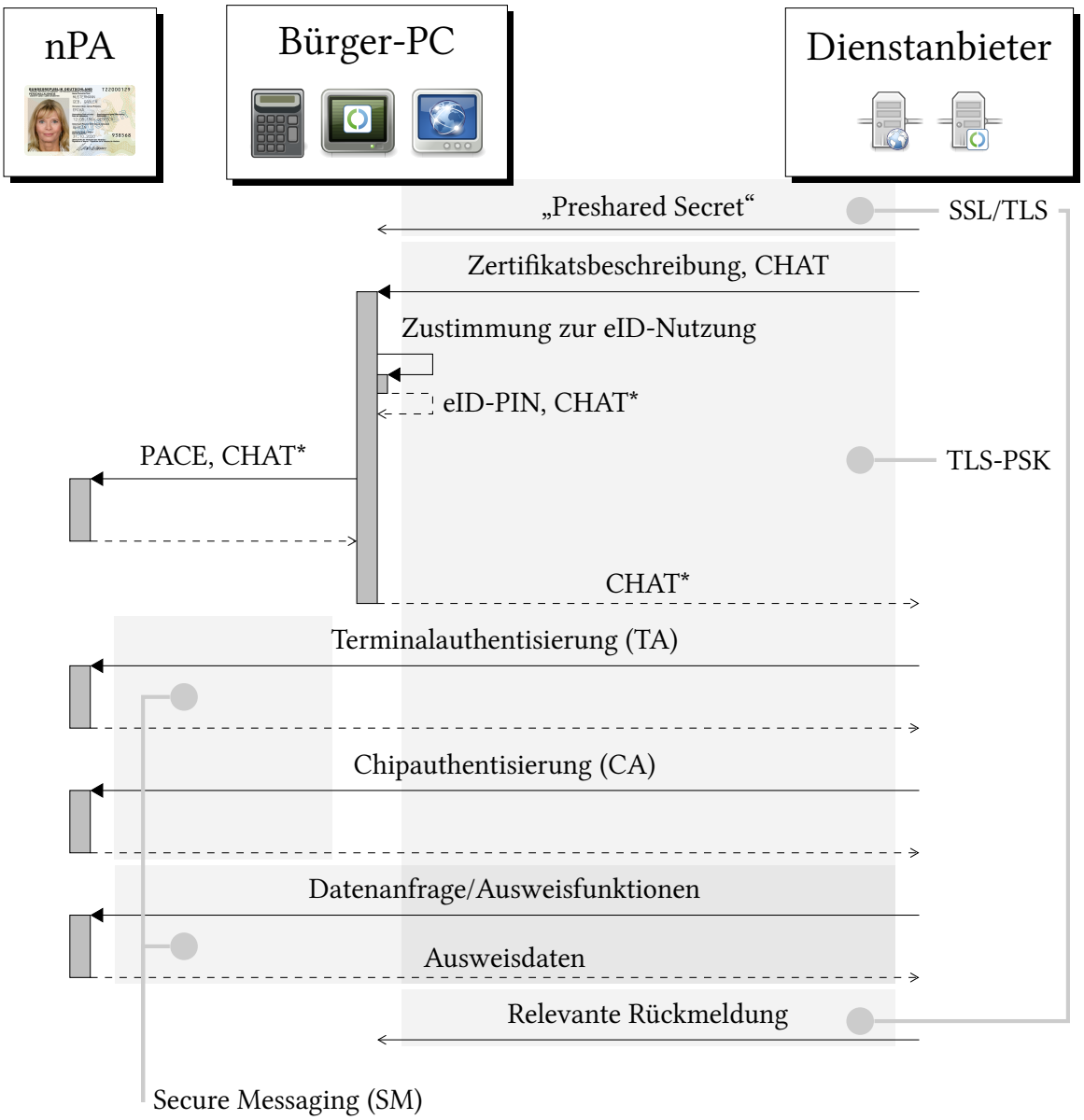


Abbildung 2.10.: Interaktion zwischen nPA, Bürger-PC und Dienstanbieter bei eID-Nutzung. Nachdem die Verbindung zwischen Bürger-PC und Dienstanbieter per TLS-PSK hergestellt wurde, stimmt der Ausweisinhaber dem Auslesen durch Eingabe der eID-PIN zu. Dabei kann der Nutzer die angeforderten Berechtigungen einschränken. Der modifizierte CHAT* wird an nPA und Dienstanbieter übermittelt. Nach erfolgreicher EAC (PACE, TA, Passive Authentisierung, CA) darf der Dienstanbieter die Ausweisdaten lesen.

Terminaltyp	PACE-Passwort	Mögliche Terminalrechte
Authentisierungsterminal	eID-PIN; CAN (falls Recht „CAN allowed“ nachgewiesen)	Lese-/Schreibzugriff auf die Datengruppen der eID-Anwendung gemäß authentisierten Rechten Spezielle Rechte: • Erzeugung eines Signaturschlüsselpaares • eID-PIN setzen, eID-Anwendung An-/Ausschalten • Pseudonym • Altersverifikation • Wohnortabfrage • Priveligiertes Terminal (Nutzung Chip-eindeutiger CA-Schlüssel)
Bestätigtes Signaturterminal	CAN	• Erzeugung qualifizierter Signaturen mit zusätzlicher Eingabe der Signatur-PIN • Setzen einer neuen Signatur-PIN mit zusätzlicher Eingabe der alten Signatur-PIN
	eID-PIN	• Anlegen der Signatur-PIN • Terminieren des Schlüssels für qualifizierte Signaturen und der Signatur-PIN
	PUK	Zurücksetzen des Fehlbedienungszähler (FBZ) der Signatur-PIN
Nicht authentisiertes Terminal	eID-PIN	Setzen einer neuen eID-PIN
	PUK	Zurücksetzen des Fehlbedienungszähler (FBZ) der eID-PIN

Tabelle 2.5.: Ausgewählte Terminaltypen und ihre möglichen Berechtigungen (vgl. BSI TR-03127, S. 21)

2.1.4. Nutzung der Signaturanwendung

Der nPA könnte die Verbreitung der QES verbessern, da mit dem Ausweis jeder Ausweisinhaber zugleich auch eine sichere Signaturerstellungseinheit (SSEE) besitzt. Zusätzlich wird teilweise[124] ein als Signaturterminal bestätigter Komfortleser benötigt.

Im Folgenden wird zunächst der technische Vorgang der Initialisierung, Aktivierung, operationellen Nutzung und Außerbetriebsetzung des nPA als SSEE beschrieben. Danach werden weitere technische Maßnahmen beschrieben, die für die Prüfung einer QES von Bedeutung sind. In der Literatur werden die Begriffe „Signaturanwendungskomponente“[125] und „Signaturterminal“[126] verwendet. Zur Differenzierung von der Signaturanwendung auf dem nPA wird hier lediglich zwischen „Client-Software“ und „Komfortleser“ unterschieden. Komfortleser sind stets auch zur Verwendung als Signaturterminal zertifiziert[127] und die AusweisApp als derzeit einzig öffentlich verfügbare Client-Software ist zur Zertifizierung als Teil der Signaturanwendungskomponente angemeldet.[128]

Initialisierung als SSEE

Der nPA ist bei seiner Auslieferung als SSEE initialisiert.[129] eID- und Signaturanwendung sowie die notwendigen Daten für TA und CA sind vorhanden. Der nPA besitzt aber keinen Signaturschlüssel und die Nutzung der Signaturanwendung ist damit zunächst nicht möglich.

Aktivierung als SSEE

Die Aktivierung als SSEE erfolgt in zwei Schritten. Der Ausweisinhaber muss seine operationellen Authentisierungsdaten, die Signatur-PIN, und der ausstellende ZDA muss das Signaturschlüsselpaar auf dem nPA einrichten (siehe Abbildung 2.11).[130]

Die Signatur-PIN ist ein sechsstelliges numerisches Geheimnis.[131] Um sie zu setzen, muss die eID-Anwendung für den operationellen Betrieb vorbereitet sein.[132] Der Ausweisinhaber benötigt einen Komfortleser, der einen vollständigen EAC-Durchlauf (PACE mit der eID-PIN, Authentisierung als Signaturterminal) durchführt. Die Client-Software wählt über den gesicherten Kanal eines Komfortlesers die Signaturanwendung des nPA aus. Die neue Signatur-PIN setzt der Ausweisinhaber anschließend direkt am Komfortleser.

Um den Ausweis mit einem Signaturschlüsselpaar zu aktivieren, werden sowohl eID- und Signaturanwendung genutzt. Die Authentisierung des Ausweisinhabers und des ZDA

[124] Prozesse, die einen Komfortleser benötigen (oder eben nicht), werden unten beschrieben.
[125] Die Signaturanwendungskomponente umfasst Hard- und Softwareprodukte, die zur Erstellung oder Verifikation von einer QES bestimmt sind (BSI TR-03117, S. 6)
[126] Das Signaturterminal ist der Teil der Signaturanwendungskomponente, die mit der TA gegenüber dem nPA sein Recht zur Signaturerstellung nachweist (BSI TR-03117, S. 6).
[127] Siehe „Lesegeräteklassen nach BSI TR-03119 1.2“, S. 16.
[128] OpenLimit SignCubes AG 2010.
[129] BSI TR-03117, S. 22.
[130] BSI TR-03117, S. 23–25.
[131] BSI TR-03117, S. 42.
[132] Siehe „Nutzung der eID-Anwendung“, S. 20.

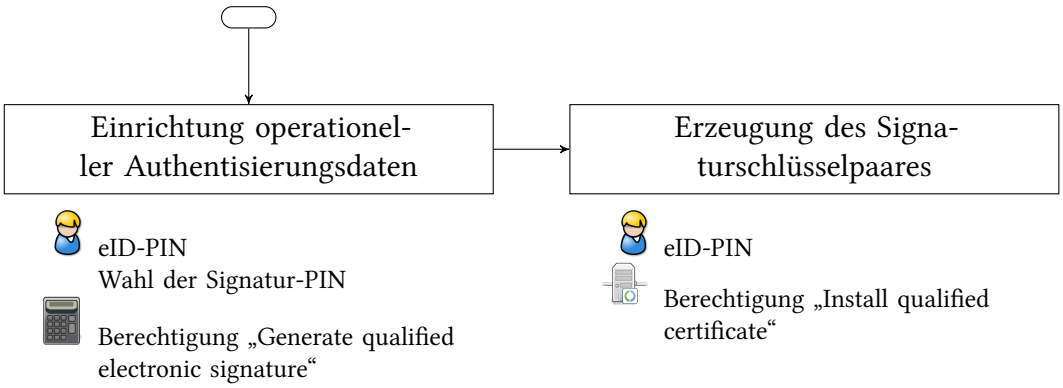


Abbildung 2.11.: Prozesse zur Aktivierung des nPA als SSEE

geschieht mittels eID-Anwendung per EAC mit eID-PIN,[133] also nicht notwendigerweise mit einem Komfortleser. Den entstehenden sicheren Kanal zwischen nPA und ZDA nutzt letzterer, um den Ausweisinhaber zu identifizieren. Anschließend wird im selben Kanal die Signaturanwendung durch den ZDA ausgewählt und die Signaturschlüsselerzeugung auf dem nPA ausgelöst.[134] Das Terminalzertifikat des ZDA besitzt neben den Befugnissen zum elektronischen Identitätsnachweis auch die Berechtigung zur Installation von qualifizierten Signaturzertifikaten.[135]

Operationelle Nutzung

Ist der nPA als SSEE aktiviert, ist die entsprechende operationelle Nutzung möglich. Neben der eigentlichen Signaturerstellung ist auch der Wechsel oder das Entsperren der Signatur-PIN ausführbar.

Der Wechsel der Signatur-PIN erfolgt analog zu deren Aktivierung (siehe oben), die alte Signatur-PIN muss jedoch vor dem Wechsel zur Bestätigung am Komfortleser eingegeben werden. Das Entsperren der Signatur-PIN nach drei Fehleingaben wird über denselben Mechanismus realisiert wie bei der eID-PIN (siehe Abbildung 2.3). Es genügt ein nicht authentisiertes Terminal, an dem PACE mit PUK durchgeführt wird, um den FBZ der Signatur-PIN zurückzusetzen.

Die Erstellung der QES kann dagegen nur mit dem Zugriffsrecht „Generate qualified electronic signature“[136] durchgeführt werden und erfordert daher einen Komfortleser. Diese Berechtigung wird wieder über das Terminalzertifikat nachgewiesen. Für PACE im Rahmen der EAC wird hier die CAN als Geheimnis verwendet. Sie darf von der Client-Software dauerhaft gespeichert werden.[137] Trotzdem soll mit der CAN sichergestellt werden, dass

[133] Zur Nutzung der eID-Anwendung siehe „Nutzung der eID-Anwendung“, S. 20

[134] „Die nach SigG zulässige Erzeugung durch technische Komponenten für Zertifizierungsdienste und der anschließende Import des Signaturschlüssels in die eCard soll nicht unterstützt werden“ (BSI TR-03117, S. 21).

[135] „Install certificate“ und „Install qualified certificate“ (BSI TR-03110 2.05, S. 90).

[136] BSI TR-03110 2.05, S. 91.

[137] BSI TR-03117, S. 25.

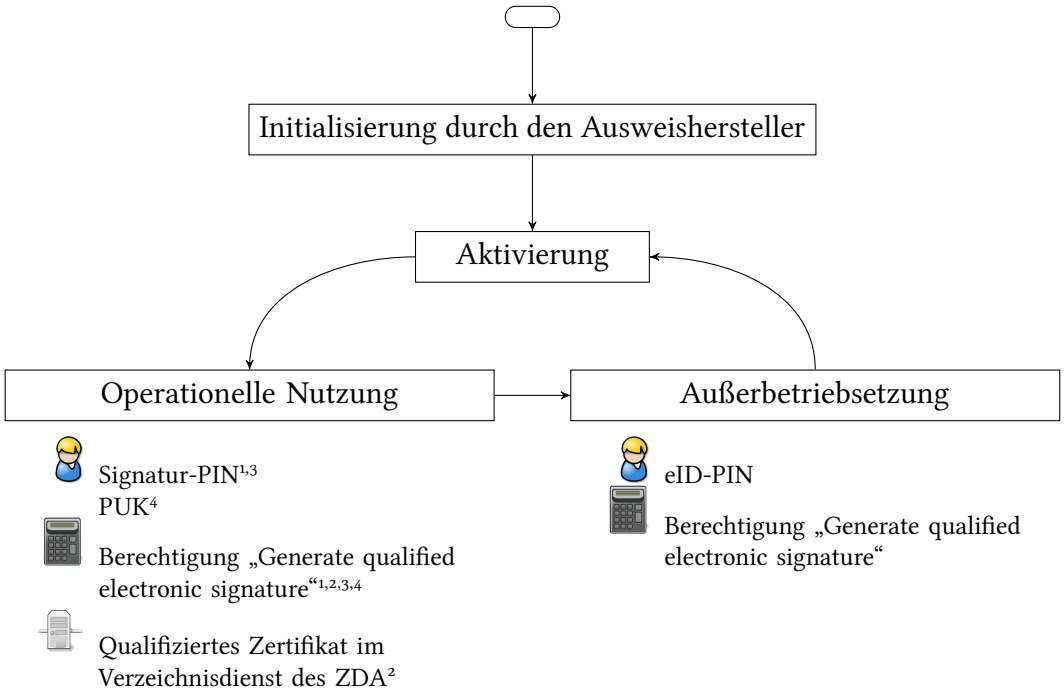


Abbildung 2.12.: Prozesse zur Nutzung des nPA als SSEE. Die operationelle Nutzung enthält die Signaturerstellung ([1]) und Signaturprüfung ([2]) sowie das Ändern ([3]) und Entsperren der Signatur-PIN ([4]).

der Ausweis auch physisch am Leser präsent ist.[138] Nach der EAC besteht zwischen nPA und Komfortleser ein sicherer Kanal, der nach Auswahl der Signaturanwendung für die Authentisierung des Ausweisinhabers verwendet wird. Der Ausweisinhaber muss nun am Komfortleser die Signatur-PIN eingeben. Ist diese korrekt, nimmt der nPA nun ein Kommando zum Signieren entgegen. Die Client-Software formatiert eine solche APDU mit dem Hash des zu signierenden Dokuments und sendet es durch den Komfortleser an den nPA. Der Komfortleser schützt die APDU dabei per SM.

Zur Prüfung einer bestimmten QES kann die Client-Software verwendet werden. Diese prüft,[139]

- auf welche Daten sich die Signatur bezieht,
- ob die signierten Daten unverändert sind,
- welchem Signaturschlüssel-Inhaber die Signatur zuzuordnen ist,
- welche Inhalte das qualifizierte Zertifikat, auf dem die Signatur beruht, und zugehörige qualifizierte Attribut-Zertifikate aufweisen und
- zu welchem Ergebnis die Nachprüfung des ZDA über die Zuordnung des Zertifikat-Inhabers zum Signaturschlüssel geführt hat.

Dazu ist es erforderlich, dass die Client-Software Zugriff auf das qualifizierte Zertifikat im Verzeichnisdienst des ZDA besitzt. Erst mit der Veröffentlichung im Verzeichnisdienst ist die QES auch für andere überprüfbar. Nur wenn alle erforderlichen Prüfungen des ZDA positiv abgeschlossen wurden und die Signatur überprüfbar ist, handelt es sich bei einer Signatur auch tatsächlich um eine QES.

Außerbetriebsetzung

Die Außerbetriebnahme läuft nach demselben Schema ab wie die Aktivierung als SSEE durch Setzen der Signatur-PIN.[140] Ein Komfortleser weist bei der EAC seine Berechtigung als „Generate qualified electronic signature“ nach. PACE wird dabei ebenfalls mit der eID-PIN durchgeführt. Die Client-Software wählt daraufhin die Signaturanwendung aus und sendet über den Komfortleser Kommandos zur Löschung der Signaturschlüssel und Signatur-PIN. Die Signatur-PIN selbst wird für die Außerbetriebnahme als SSEE nicht benötigt.

2.2. Sicherheitsmodell des Systems nPA

Im Folgenden wird zunächst ein ausgewählter Teil des Sicherheitsmodell des Systems nPA aus den gegebenen Vorschriften und bisherigen Überlegungen rekonstruiert. Anschließend wird sich mit diesem Modell kritisch auseinandergesetzt und einige Modifikationen vorgestellt.

2.2.1. Rekonstruktion des Sicherheitsmodells

Ziele des Systems sind:

[138] Es wird angenommen, dass die CAN nur dann vom Ausweis abzulesen ist. Dies ist bei einer gespeicherten CAN nicht notwendigerweise der Fall.

[139] SigG, § 17.

[140] BSI TR-03117, S. 28–29.

- Operationen sollen nur mit Autorisierung und Kenntnis des Ausweisinhaber statt finden. D. h. eID- und Signaturanwendung dürfen nicht unbemerkt genutzt oder aktiviert werden.
- Operationen sollen nur mit Parteien und im Umfang ausgeführt werden, denen der Ausweisinhaber zugestimmt hat.
- Die im Ausweis gespeicherten Daten sollen nur von berechtigten Parteien gelesen oder geschrieben werden können.[141]
- Die Verfolgung eines Ausweises soll nicht durch bloßes Ansprechen der kontaktlosen Schnittstelle möglich sein.[142]

Die folgenden Modellierungen werden dabei in Beziehung zu den Bedrohungen (*T*) betrachtet, denen sie begegnen sollen. Unterschieden wird hier zwischen den Komponenten nPA, Client-Software (*API*), Chipkartenleser (*IFD*) und Dienstanbeiter bzw. ZDA (*SP)*.

nPA.Integrity, nPA.Authenticity, nPA.Confidentiality Der nPA muss die Integrität, Authentizität und Vertraulichkeit der gespeicherten oder zu speichernden Ausweisdaten sicherstellen.[143] Dazu müssen nicht autorisierte Modifikationen oder Zugriffe verhindert werden. Die Ausweisdaten dürfen erst im sicheren Kanal nach EAC gelesen werden.

Durch die vertrauliche Übermittlung der Ausweisdaten ist ein Angreifer nicht in der Lage, diese durch Abhören der (kontaktlosen) Verbindung zwischen nPA und einem legitimen Terminal zu ermitteln (*T.Eavesdropping*[144]).[145]

nPA.Card-Holder Der Karteninhaber soll die eID-PIN, PUK Signatur-PIN geheimhalten. CAN oder MRZ dürfen Personen oder Parteien preisgeben werden die definitiv vertrauenswürdig sind.[146]

SP.TA Berechtigte Terminals werden im Rahmen der EAC-PKI autorisiert. Dienstanbieter sollen ihre Credentials und Authentisierungsgeheimnisse sicher speichern.

Durch die letztgenannten Maßnahmen soll Skimming (*T.Skimming*[147]) verhindert werden.[148] Ein Angreifer ist weder in der Lage sich als legitimes Terminal auszugeben (SP.TA), noch kann er die Kommunikation mit einem legitimen Terminal belauschen, um die Ausweisdaten zu ermitteln (nPA.Integrity, nPA.Authenticity, nPA.Confidentiality). Da der Ausweisinhaber seine Geheimnisse nicht preisgibt (nPA.Card-Holder) und angenommen wird, dass dem Angreifer keines der Authentisierungsgeheimnisse bekannt sind, kann er nicht anstelle des Inhabers die Ausweisdaten auslesen. Dabei wird die Annahme getroffen, dass dem Angreifer weder MRZ, CAN noch eID-PIN oder PUK vorliegen.[149]

[141] BSI-CC-PP-0061, S. 14.
[142] BSI-CC-PP-0061, S. 15.
[143] BSI-CC-PP-0061, S. 31–32.
[144] BSI-CC-PP-0061, S. 23.
[145] BSI-CC-PP-0061, S. 41.
[146] BSI-CC-PP-0061, S. 38.
[147] BSI-CC-PP-0061, S. 23.
[148] BSI-CC-PP-0061, S. 40–41.
[149] BSI-CC-PP-0061, S. 22–23, 104.

nPA.Card_Tracing Der nPA muss eine Identifizierung verhindern, wenn der Angreifer keines der Authentisierungsgeheimnisse kennt.

Ein Tracing des Ausweises (*T.Card_Tracing*[150]) wird durch die entsprechende Maßnahmen des nPA (nPA.Card_Tracing) und der Geheimhaltungspflicht der Authentisierungsgeheimnisse verhindert (nPA.Card-Holder).[151]

API.End-User Der Ausweisinhaber ist geschult im Umgang mit dem nPA und prüft regelmäßig die Integrität der Client-Software.[152]

API.Interfaces Die Kommunikation zwischen Client-Software, Chipkartenleser, Webbrowser und eID-Server muss syntaktisch korrekt sein. Die Nachrichten dürfen nicht als Programmcode ausgeführt werden.

API.Integrity, API.Update, API.Browser-Plugin Der Nutzer soll die Integrität der Client-Software beim Erhalt prüfen können. Die Authentizität soll bei einem Update der Client-Software sichergestellt werden.[153] Browser-Plugin und Client-Software werden zueinander passend installiert.

API.CurrentDate Das Betriebssystem stellt der Client-Software eine relativ genaue Zeitquelle zur Verfügung, um den Gültigkeitszeitraum von Zertifikaten prüfen zu können.

Das Einfügen von Malware in die Client-Software oder ein entsprechendes Nachladen durch ein Update (*T.Malware*[154]) soll durch spezifikationsgemäße, authentische oder integere Kommunikation sichergestellt werden (API.Interfaces, API.Integrity, API.Update, API.Browser-Plugin, API.End-User, API.CurrentDate).

API.Platform Die Client-Software ist auf einem System installiert, zu dem der Zugang beschränkt oder unter ständiger Kontrolle des Nutzers ist. Das System soll nicht unwissentlich manipulierbar. Es wird angenommen, dass das System nicht die Vertraulichkeit oder Integrität der eID-PIN gefährdet.[155]

API.Password Die Client-Software soll die Authentisierungsgeheimnisse des Nutzers ausschließlich für ihren vorgesehenen Zweck verwenden und nach Verwendung überschreiben.[156]

IFD.API Ein Chipkartenleser mit sicherer PIN-Eingabe und sicherer Anzeige soll weder die Authentisierungsgeheimnisse kompromittieren noch die angeforderten Zugriffsrechte manipulieren.[157]

[150] BSI-CC-PP-0061, S. 23.
[151] BSI-CC-PP-0061, S. 40–41.
[152] BSI-CC-PP-0066-V2, S. 17, 21.
[153] BSI-CC-PP-0066-V2, S. 20.
[154] BSI-CC-PP-0066-V2, S. 15, 23–24.
[155] BSI-CC-PP-0066-V2, S. 17, 21.
[156] BSI-CC-PP-0066-V2, S. 20.
[157] BSI-CC-PP-0066-V2, S. 21.

IFD.PACE Ein am PIN-Pad eingegebenes Geheimnis verlässt den Kartenleser nicht.[158] Es ist nach der Verwendung unmittelbar zu löschen. Zur Schlüsselerzeugung bei PACE wird ein zertifizierter Zufallszahlengenerator verwendet.

IFD.Manipulationsschutz PIN-Pad und Display müssen entsprechend den Vorgaben des Signaturrechts so gestaltet werden, dass Manipulationen wirksam verhindert oder erkannt werden.[159]

IFD.Filter PACE wird ausschließlich durch den Leser durchgeführt.[160] eID-PIN oder PUK müssen am PIN-Pad eingegeben werden. Der Kartenleser erlaubt ausschließlich folgende Kombinationen von Terminaltypen und Passworten:

- nicht authentisiertes Terminal mit CAN, PUK oder eID-PIN
- Authentisierungsterminal mit CAN oder eID-PIN
- Signaturterminal mit CAN, eID-PIN oder PUK (falls QES mit kontaktlosen Karten implementiert wird)

Die Vertraulichkeit der Authentisierungsgeheimnisse soll gewährleistet bleiben, wenn ein Angreifer Zugriff auf die oben spezifizierten Schnittstellen (siehe API.Interfaces) besitzt und kein PACE-fähiges Lesegerät verwendet wird (*T.eID-PIN*).[161] Schutzmaßnahmen müssen auf Ebene der Client-Software, des Betriebssystems und des Chipkartenlesers getroffen werden (API.Platform, API.Password, IFD.API).[162]

Falls ein PACE-fähiger Leser verwendet wird, so schützt er Vertraulichkeit der Authentisierungsgeheimnisse durch sichere und ausschließliche Nutzung des PIN-Pads (IFD.PACE, IFD.Manipulationsschutz, IFD.Filter).

API.SecureDisplay Die Client-Software soll Informationen zum Dienstanbieter und dessen (eingeschränkten) Zugriffsrechte ungehindert und vollständig anzeigen können.[163] Außerdem sollen die Ergebnisse der Plausibilitätsprüfung bzgl. SSL/TLS und der CV-Zertifikate sowie der TA angezeigt werden.

API.FinalAccessRights Die Client-Software soll die ungehinderte und unveränderte Beschränkung der Zugriffsrechte des Dienstanbieters erlauben.[164]

IFD.Display Das Display eines Kartenlesers besitzt mindestens zwei Zeilen mit jeweils 16 Zeichen.[165] Es muss die authentische Anzeige von Berechtigtem und Berechtigungen zum elektronischen Identitätsnachweis gewährleisten.[166]

[158] BSI TR-03119 1.2, S. 17.
[159] BSI TR-03119 1.2, S. 19.
[160] BSI TR-03119 1.2, S. 19.
[161] BSI-CC-PP-0066-V2, S. 15.
[162] BSI-CC-PP-0066-V2, S. 23–24.
[163] BSI-CC-PP-0066-V2, S. 20–21.
[164] BSI-CC-PP-0066-V2, S. 21.
[165] BSI TR-03119 1.2, S. 18.
[166] BSI TR-03119 1.2, S. 19.

Eine Manipulation der Zugriffsrechte (*T.FinalAccessRights*[167]) oder eine veränderte Anzeige der Informationen eines elektronischen Identitätsnachweises (*T.ManDisplaying*[168]) sollen durch ein vertrauenswürdiges Betriebssystem (API.Platform) und entsprechende Maßnahmen der Client-Software (API.SecureDisplay, API.FinalAccessRights) und des Chipkartenlesers (IFD.API) verhindert werden. Unterstützend zeigt der Kartenleser einige dieser Informationen an (IFD.Display).

IFD.eSign Der Chipkartenleser ist für die QES vorbereitet und enthält ein entsprechendes Signaturterminalzertifikat, das im Rahmen der EAC-PKI signiert ist.[169] Der private Schlüssel wird sicher im Leser gespeichert. Der Chipkartenleser implementiert die vollständige EAC, um seine Echtheit und die des nPA zu bestätigen.

IFD.Signatur-PIN Die Signatur-PIN wird nur verschlüsselt an die Chipkarte gesendet und nach Gebrauch gelöscht.[170] Die Nutzung der Signatur-PIN soll ausschließlich über das PIN-Pad des Kartenlesers möglich sein. Entsprechende Kommandos vom Host-System sind zu filtern.[171]

Die Vertraulichkeit der Signatur-PIN (*T.Signatur-PIN*) wird von dem Ausweisinhaber verlangt (nPA.Card-Holder) und vom Chipkartenleser sichergestellt (IFD.eSign, IFD.Signatur-PIN, IFD.Manipulationsschutz).

SP.eSign Der ZDA soll sich schriftlich oder durch eine QES bestätigen lassen, ob der Ausweisinhaber bei der Erstellung der Signaturschlüssel alleiniger Besitzer des nPA war.[172]

Die missbräuchliche Initialisierung der Signaturfunktion (*T.eSign*) soll durch Maßnahmen des ZDA sicherstellt werden (SP.eSign).

2.2.2. Diskussion und Modifikation des Sicherheitsmodells

Social Engineering[173] eignet sich für einen Angreifer, um Wissen über das Passwort zu erlangen.[174] Demonstriert wurde dies mittels einer Website, bei der ein Fenster erschien, welches der AusweisApp nachempfunden war und nach der Eingabe der eID-PIN verlangt hat.[175] Dies zeigt bereits, dass die Täuschung des Ausweisinhabers nie ganz ausgeschlossen werden kann. Trotzdem soll hier einfaches Phishing nach den Authentisierungsgeheimnissen nicht möglich sein und API.End-User in vollem Umgang gelten.

Der initiale Bezug der Client-Software soll beim Erhalt prüfbar sein auf Integrität (API.Integrity). Der Download der AusweisApp auf `https://www.ausweisapp.bund.de` ist gesichert durch

[167] BSI-CC-PP-0066-V2, S. 16.
[168] BSI-CC-PP-0066-V2, S. 15–16.
[169] BSI TR-03119 1.2, S. 20.
[170] BSI TR-03119 1.2, S. 21.
[171] BSI TR-03119 1.2, S. 22.
[172] SigV, §5 Abs. 2.
[173] Für eine Vielzahl verschiedener Beispiele, siehe Street 2010.
[174] Morgner und Oepen 2010.
[175] Schejbal 2011b.

```
openssl aes-128-ecb -d -K ac5370f9265e5e6d9ca4ff03495198fb -in /opt/
   olsc/AusweisApp/ext/aa_classes.jar -out aa_classes.dec.jar
```

Abbildung 2.13.: Befehl zur Entschlüsselung der AusweisApp 1.3. Im Security Workshop 2011 der Humboldt-Universität zu Berlin konnte von Michael Gehring herausgefunden werden, mit welchem statischen Geheimnis die Klassenbibliotheken der AusweisApp verschlüsselt werden (der Schlüssel hat sich in neueren Versionen geändert). Diese könnten nun von einem Angreifer dekompiliert und manipuliert werden, um die AusweisApp mit Schadcode auszustatten.

SSL/TLS. Ob das beigefügte „Integrity Tool“ authentisch ist, kann jedoch schwer ohne weiteres geprüft werden. Das SSL-Zertifikat dieser Website wurde von Comodo CA Limited ausgestellt – einer Root-CA, die vor kurzer Zeit selbst unautorisierte Zertifikate erzeugt und ausgestellt hat.[176] Die Webseite der AusweisApp schien zwar nicht betroffen gewesen zu sein, trotzdem verdeutlicht dies die Anfälligkeit durch Phishing. Hier wird dennoch davon ausgegangen, dass ein Benutzer nur eine authentische und unveränderte Version der Client-Software verwendet, obwohl Manipulationen an der AusweisApp prinzipiell möglich sind (siehe Abbildung 2.13).

Bei komplexen Programmen sind Implementierungsfehler nicht selten – so auch beim Update-Mechanismus früher Versionen der AusweisApp. Eine Signatur des Updates sollte sicherstellen, dass es integer ist und aus vertrauenswürdiger Quelle stammt (API.Integrity, API.Update). Die Signatur-Prüfung war jedoch fehlerhaft implementiert, so dass gar eine Infizierung mit Schadsoftware drohte.[177] Um der Sache willen, wird hier aber davon ausgegangen, dass die Client-Software frei von jeglichen Implementierungsfehlern ist. Die Evaluierung des Programms im Rahmen einer Zertifizierung nach BSI-CC-PP-0066-V2 kann dazu beitragen, diesem Ziel nahe zu kommen.

Beim Betrieb der Client-Software werden Annahmen über das Betriebssystem bzw. die darauf installierte Software gemacht. Im wesentlichen wird angenommen bzw. verlangt, dass das System in buchstäblich allen Belangen sicher ist (API.Interfaces, API.CurrentDate, API.Platform). Diese Annahmen sind auf die üblicherweise verwendeten Desktop-Systeme *nicht* zutreffend. Besonders ein online genutztes System wie das des nPA ist Angriffen von Schadsoftware ausgeliefert.

Als Beispiel sei hier der Fehler in einem Browser-Plug-in genannt, das einem entfernten Angreifer ohne Installation von Malware unter bestimmten Umständen den uneingeschränkten Zugriff auf einen Chipkartenleser und damit auf den nPA erlaubte.[178] Außerdem ein online genutztes System wie das des nPA (mit der eID) in besonderem Maße Angriffen von Schadsoftware ausgeliefert. Auch die vorgeschriebenen technischen und organisatorischen Maßnahmen (z. B. Firewall, Virenscanner oder Software-Updates)[179] genügen nicht, um eine

[176] Bachfeld 2011.
[177] Schejbal 2010.
[178] Schejbal 2011a.
[179] BSI 2010b.

geeignete Sicherheit zu erreichen.[180]

Abhängig von den Systemrechten, mit denen die Schadsoftware oder ein ausgenutztes fehlerhaftes Programm arbeitet, können neue Prozesse gestartet, laufende Prozesse manipuliert, Eingaben oder Ausgaben gelesen und verändert werden. Damit sind aber unter Umständen *alle* Sicherungsmaßnahmen der Client-Software wirkungslos (API.Integrity, API.Update, API.Browser-Plugin, API.Interfaces, API.CurrentDate, API.FinalAccessRights, API.Password, API.SecureDisplay). Hier wird davon ausgegangen, dass Malware ein veritables Problem ist und sie die korrekte Ausführung der Client-Software verhindern kann.

Unter Skimming (*T.Skimming*) wird nach BSI-CC-PP-0061 die Fälschung eines rechtmäßigen Authentisierungs- oder Signaturterminal verstanden. Dabei wird die Annahme getroffen, dass dem Angreifer weder MRZ, CAN noch eID-PIN oder PUK vorliegen.[181] Diese Annahme ist nach meiner Auffassung jedoch nicht immer gerechtfertigt. Denn das Mittel der Täuschung wird beim Skimming ja gerade dazu eingesetzt, um neben den Zugriff auf eine Chipkarte auch die nötigen Authentisierungsgeheimnisse zu erlangen. Der Begriff des Skimmings wird hier also erweitert zu einer Bedrohung, bei der der Angreifer die Authentisierungsgeheimnisse kennt, die vom Ausweisinhaber preisgegeben werden (*T.Skimming**).

2.3. Angreifermodelle

2.3.1. Malware

Die Kompromittierung des Computers des Ausweisinhabers mit Malware kann nicht nur über die Client-Software geschehen (*T.Malware*), von der hier angenommen wird dass sie sicher ist. Viele andere Angriffspfade können dazu führen, dass Schadsoftware ausgeführt wird. In anderen Veröffentlichungen wurde verdeutlicht, dass Schadsoftware sowohl als Keylogger fungieren kann[182] als auch Zugriff auf den in einen Leser eingelegten Ausweis zulässt[183].

Entsprechend der Diskussion von API.Platform[184] muss die Bedrohung durch Schadsoftware in Betracht gezogen werden. Ein Angreifer, der die Malware kontrolliert, besitzt mindestens folgende Fähigkeiten:

- Passives Beobachten von Ein- und Ausgaben des Computers, insbesondere auch PIN-Eingaben bei Verwendung des Basislesers und eID- oder eSign-Transaktionen des Nutzers
- Aktive Nutzung der Internetverbindung
- Aktive Nutzung angeschlossener Geräte, insbesondere auch des Chipkartenlesers, z. B. über die PC/SC-Schnittstelle
- Aktives Starten von Programmen, insbesondere auch manipulierte Varianten des Browsers oder der Client-Software.[185]
- Manipulation legitimer Transaktionen mit dem nPA, die nicht geschützt sind

[180] Warentest 2012.
[181] BSI-CC-PP-0061, S. 22–23, 104.
[182] R. Stumpf 2010.
[183] erdgeist 2010; Moser und Schröder 2010.
[184] Siehe „Diskussion und Modifikation des Sicherheitsmodells“, S. 34.
[185] Siehe Abbildung 2.13.

- Angreifer besitzt Zugriff auf den nPA über einen langen (möglicherweise unterbrochenen) Zeitraum

Abhängig davon ob der Ausweisinhaber einen Basisleser oder einen PACE-fähigen Leser verwendet, müssen die erlangten Zugriffsrechte und Authentisierungsgeheimnisse des Angreifers unterschieden werden:

Verwendung eines Basislesers

- Angreifer besitzt ungefilterten Zugriff auf den nPA
- Der Angreifer erfährt die eID-PIN

Verwendung eines PACE-fähigen Lesers

- Angreifer besitzt gefilterten Zugriff auf den nPA, insbesondere kann der Angreifer nicht unabhängig vom Ausweisinhaber PACE durchführen (IFD.Filter).

Obwohl auch Phishing nach den Authentisierungsgeheimnissen selbst bei Verwendung eines PACE-fähigen Lesegeräts möglich wäre, gehen wir hier von einem sehr gut geschulten Anwender aus (API.End-User).[186] Daher gibt der Ausweisinhaber die eID-PIN ausschließlich am PIN-Pad des Lesers ein. Die Möglichkeit zur Speicherung der CAN – auch bei Verwendung eines PACE-fähigen Lesers – wird von der aktuellen AusweisApp nicht realisiert, daher besitzt Malware in diesem Modell (zunächst)[187] kein Wissen darüber.

2.3.2. Nearby Attacker

Ein Angreifer, der sich in der Nähe des nPAs befindet, kann uneingeschränkt drahtlos mit dem Ausweis kommunizieren. Aktives Auslesen wurde für Chipkarten mit geringerer Leistungsaufnahme als dem nPA bei einer Reichweite von 55 Zentimetern demonstriert.[188] Die praktisch mögliche maximale Auslesedistanz wird für den nPA auf weniger als 25 Zentimeter geschätzt.[189] Die Kommunikation zum nPA ist außerdem möglich, wenn sich der Ausweis in einer Hosentasche befindet oder auf dem Tresen eines Hotels verweilt, weil Textilien oder Holz den nPA nicht ausreichend abschirmen. Ein Angreifer, der sich in räumlicher Nähe zum Ausweis befindet und damit kommunizieren kann (Nearby Attacker), besitzt folgende Fähigkeiten:

- Aktive Nutzung einer eigenen Internetverbindung
- Angreifer besitzt ungefilterten Zugriff auf den nPA
- Angreifer besitzt Zugriff auf den nPA über einen kurzen Zeitraum

Auch die auf den Ausweis gedruckten Authentisierungsgeheimnisse sind unter Umständen für den Angreifbar einsehbar.

[186] Siehe „Diskussion und Modifikation des Sicherheitsmodells“, S. 33.
[187] Siehe „Tracing per CAN“, S. 47.
[188] *ISO/IEC 14443 Eavesdropping and Activation Distance* 2008, S. 20.
[189] *ISO/IEC 14443 Eavesdropping and Activation Distance* 2008, S. 22.

Sichtkontakt zum nPA

- Angreifer erfährt die CAN und MRZ

2.3.3. Skimmer

Unter Skimming wird die Fälschung eines rechtmäßigen Authentisierungs- oder Signaturterminal verstanden.[190] Für den Ausweisinhaber ist nicht erkenntlich, dass es sich um ein boshaftes Terminal handelt, so dass er sowohl Zugriff auf den nPA erlaubt als auch seine Geheimnisse eingibt. Im Rahmen des Sicherheitsmodell sind damit effektiv alle Maßnahmen des Chipkartenlesers unwirksam (z. B. IFD.API, IFD.PACE, IFD.Display, IFD.eSign, IFD.Signatur-PIN).

Skimming-Angriffe können durchaus preiswert umgesetzt werden[191] und an öffentlichen Terminals viele potenzielle Opfer in Mitleidenschaft ziehen. Die Gefahr, die vom Skimming ausgeht, ist offensichtlich. Was also wird dagegen getan?

Änderungsterminals der Ausweisbehörden oder Authentisierungsterminals in kommerziellem Umfeld stehen während der Öffnungszeiten meist unter Beobachtung durch Mitarbeiter, weshalb eine Manipulation der Geräte in dieser Zeit schwierig ist. Nichtsdestotrotz können auch Mitarbeiter oder Einbrecher versuchen, Chipkartenlesegeräte zu manipulieren.[192] Für den Ausweisinhaber ist es schwierig Manipulationen oder sogar einfache Veränderungen festzustellen, da er mit den Geräten i. d. R. nicht täglich zu tun hat und die Terminals nicht einheitlich gestaltet sind. Mit entsprechender Verbreitung und Gewinnaussicht könnte Skimming für den nPA ein ähnliches gravierendes Problem werden wie aktuell die Manipulation von Geldautomaten.[193]

Von Standard- oder Komfortleser existieren immerhin Referenzbilder der entsprechenden Modelle vom Hersteller. Trotzdem kann das Design bzw. Aussehen eines Chipkartenlesers wohl kaum als Sicherheitsmerkmal bezeichnet werden. Marken- und Produktpiraterie gehört zu den häufigsten Betrugsfällen.[194] Auch das Logo des neuen Personalausweises, das nach BSI TR-03119 1.2 zertifizierte Leser ausweisen soll,[195] ist sehr leicht fälschbar. REINER SCT liefert die Modelle cyberJack RFID standard und cyberJack RFID komfort mit Hologrammaufklebern aus. Diese erhöhen zwar den Aufwand für einen Fälscher, beseitigen das Problem aber nicht. Die Absicherung von Vertriebswegen wäre eine Möglichkeit, beim Kauf eines Lesers sicherzustellen, dass es sich um einen echten handelt. Dies ist jedoch in Zeiten vom Online-Handel nur schwer umsetzbar.

Softwareseitig prüft die AusweisApp die Informationen des Chipkartenlesers auf Konsistenz mit der ihr vorliegenden zertifizierten Datenbank über bestätigte Leser. Bei Unstimmigkeiten wird dem Nutzer eine Warnung angezeigt (siehe Abbildung 2.15a). Ein beispielhafter Datensatz

[190] Manipulationen eines echten Lesers sollen nach IFD.Manipulationsschutz nicht möglich sein (siehe „Lesegeräteklassen nach BSI TR-03119 1.2“, S. 16.). Im folgenden beziehe ich mich daher auf Skimming durch Fälschung von Standard- oder Komfortlesern statt durch deren Manipulation.

[191] Adida u. a. 2009.

[192] Der Betrug an Geldautomaten ist nicht neu. Auch EC-Kartenlesegeräte an Kassensystemen wurden jüngst (wieder) per Skimming angegriffen (Adomeit 2011; Soboll 2011).

[193] BKA 2011.

[194] BMF 2011.

[195] Siehe Abbildung 2.15a.

```
[REINER SCT cyberJack RFID komfort]
OpticalSignalUnit = true
AcousticSignalUnit = true
ActiveAntenna = true
DisplayLines = 2
DisplayColumns = 16
DisplayVirtualLines = 0
DisplayVirtualColumns = 0
KeysCount = 16
BioSensorType = 0
Protocol = "urn:iso:std:iso-iec:7816:-3:tech:protocols:T-equals-0;
   urn:iso:std:iso-iec:7816:-3:tech:protocols:T-equals-1"
VendorName = "REINER SCT"
ProductName = "cyberJack RFID komfort"
Firmware = 1.0
Driver = 3.99.5
```

Abbildung 2.14.: Datensatz der AusweisApp zum Komfortleser REINER SCT cyberJack RFID komfort aus der Datei `siqTerminalsInfo.cfg`

findet sich in Abbildung 2.14. Der PC/SC-Name des Chipkartenlesers wird verglichen mit dem in den eckigen Klammern angegebenem Identifikator des Datensatzes. Hersteller- und Produktbezeichnung sowie Firmware- und Treiberversion werden vom Chipkartenleser per Pseudo-APDUs gemäß BSI TR-03119 1.2 abgefragt[196] und mit den Werten der Felder `VendorName`, `ProductName`, `Firmware` und `Driver` verglichen.

Dass ein gefälschter Chipkartenleser auf die Pseudo-APDUs mit beliebigen Daten (insbesondere mit denen eines zertifizierten Chipkartenlesers) antworten kann, ist offensichtlich. Der PC/SC-Name kann bei manchen Systemen auch vom Leser selbst vorgegeben werden.[197] In jedem Fall kann der Treiber des Lesers den Namen festlegen. Ein gefälschter Leser könnte einen entsprechenden Treiber mitliefern. Speziell wird der `ccid-emulator`[198] mit entsprechender Konfiguration von der AusweisApp als REINER SCT cyberJack RFID komfort erkannt inklusive der Möglichkeit zur eID- und eSign-Nutzung (siehe Abbildung 2.15b).[199] Die Prüfung der AusweisApp ist also nicht sehr wirkungsvoll.

Der Angreifer besitzt beim Skimming mindestens folgende Fähigkeiten:

- Aktive Nutzung einer Internetverbindung, z. B. mit einem integrierten UMTS-Modul
- Angreifer besitzt ungefilterten Zugriff auf den nPA
- Angreifer kann CAN und MRZ optisch vom Kartenkörper ablesen, z. B. wenn der Ausweisinhaber seinen nPA in den Kartenslot steckt

Mindestens drei verschiedene Varianten des Skimmings sind denkbar.

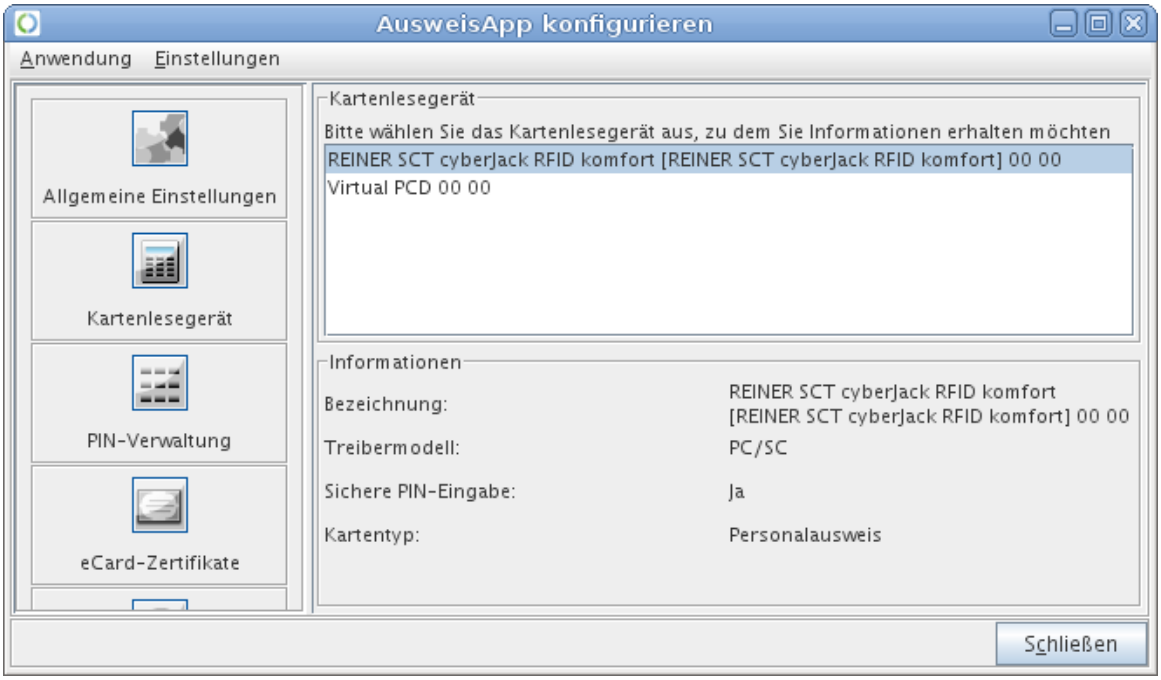
[196] Siehe „Lesegeräteklassen nach BSI TR-03119 1.2“, S. 16.

[197] Unter Linux und MacOS X ermittelt der `libccid`-Treiber für USB-CCID-Chipkartenleser den PC/SC-Namen unter anderem aus dem Feld `iInterface` des vom Chipkartenleser vorgegebenen USB-Deskriptor (siehe „USB Smartcard Klassendeskriptor des CCID-Emulators“, S. 107.

[198] Siehe „CCID-Emulator für den nPA“, S. 104.

[199] Morgner und Oepen 2011.

(a) Der virtuelle Smartcardleser (Virtual PCD) wird als nicht zertifiziert erkannt. Der Ausweisinhaber wird vor dessen Nutzung gewarnt.



(b) Die AusweisApp erkennt den `ccid-emulator` als REINER SCT cyberJack RFID komfort. Die Gerätekennung und -bezeichnung wurde dem Komfortleser angepasst. Die PACE-Funktionalität ist im Emulator implementiert.

Abbildung 2.15.: Erkennung von zertifizierten Chipkartenlesern bei der AusweisApp

Skimming eines öffentlichen Terminals

- Angreifer besitzt die eID-PIN
- Angreifer besitzt Zugriff auf den nPA über einen kurzen Zeitraum

Skimming eines Komfortlesers

- Angreifer besitzt die eID-PIN und Signatur-PIN
- Angreifer besitzt Zugriff auf den nPA über einen langen (möglicherweise unterbrochenen) Zeitraum

Skimming eines anderen PACE-fähigen Lesers

- Angreifer besitzt die eID-PIN
- Angreifer besitzt Zugriff auf den nPA über einen langen (möglicherweise unterbrochenen) Zeitraum

2.3.4. Man-in-the-Middle Hardware

Ein Hardware-Modul zwischen Computer und Chipkartenleser kann den unverschlüsselten Datenverkehr lesen oder manipulieren.[200] Das Modul ist schnell angebracht, erfordert allerdings physischen Zugriff zum Arbeitsplatz des Ausweisinhabers. Da der Zugang in diesem Modell aber beschränkt oder unter ständiger Kontrolle des Nutzers ist, können nur dem Nutzer bekannte Personen diesen Angriff durchführen. Der Angreifer besitzt folgende Fähigkeiten:

- Passives Beobachten der Transaktionen zum Chipkartenleser
- Aktive Nutzung einer Internetverbindung, z. B. mit einem integrierten UMTS-Modul
- Aktive Nutzung des angeschlossen Chipkartenlesers
- Manipulation legitimer Transaktionen mit dem nPA, die nicht geschützt sind
- Angreifer besitzt gefilterten Zugriff (IFD.Filter) auf den nPA über einen langen (möglicherweise unterbrochenen) Zeitraum

2.4. Bedrohungsszenarien

Bereits erste Arbeiten, die sich nicht allein mit dem nPA, sondern mit der gesamten Infrastruktur zu dessen Nutzung beschäftigten, haben einige Probleme identifiziert. In Dietrich 2008 wurde beispielsweise darauf hingewiesen, dass nach BSI TR-03110 1.10 ein Man-in-the-Middle-Angriff mit einem kompromittierten SSL-Zertifikat genutzt werden könnte, um eine boshafte Client-Software (implementiert als Java-Applet) zu laden.[201] In BSI TR-03110 2.04 wurde die Bindung des SSL-Zertifikats an das Terminalzertifikat des Dienstanbieters eingeführt,

[200] Morgner und Oepen 2010.
[201] Dietrich 2008, S. 53–55.

Abbildung 2.16.: Elektronische Nutzung des nPA an einem öffentlichen Terminal als offizieller Anwendungsfall (BMI 2010a). Für den Ausweisinhaber ist die Echtheit eines Terminals bzw. das Skimming kaum erkennbar.

die diesen Angriff verhindert.[202] Die Prüfung des Terminalzertifikats, die der nPA während der EAC durchführt, stellt implizit auch die Vertrauenswürdigkeit des SSL-Zertifikats sicher. Die Client-Software muss dabei allerdings sicherstellen, dass das verwendete SSL-Zertifikat tatsächlich auch zu dem vermerkten Hash passt.

In BSI TR-03119 1.1 waren die vorgeschriebenen Schutzmaßnahmen durch Standard- oder Komfortlesern nicht ausreichend, um Relay-Angriffe zu verhindern.[203] Mit einem PACE-fähigen Leser des Anwendungstests konnten sowohl die eID- als auch Signaturanwendung missbraucht werden. Der Test-Leser akzeptierte beliebig vorgegebene Passworte, insbesondere auch die eID-PIN, welche eigentlich am PIN-Pad eingegeben werden sollte. Mit BSI TR-03119 1.2 wurde die vorgeschriebenen Schutzmaßnahmen ausgebaut[204] und Filterregeln in die Konformitätsprüfung aufgenommen.

Eine neuere Studie[205] kommt zu dem Ergebnis, „dass die aktuelle Spezifikation des Kommunikationsmodells für die eID-Funktion keine Schwachstellen aufweist. Es ist notwendig, dass Implementierungen auf die Einhaltung der sicheren Konzepte genau geprüft werden." Die Studie benennt dabei vier „Restrisiken", die also dem Wort nach ein *prinzipiell* nicht zu beseitigendes Risiko darstellen.

Erstens kann ein Basisleser nicht vor dem Ausspähen der eID-PIN am Bürger-PC schützen.[206] Auf mögliche Angriffe, die dieses Problem ausnutzten, werde ich in diesem Kapitel eingehen. Insbesondere die Größe eines möglichen Schadens, auf die Dietrich, Rossow und Pohlmann nicht eingehen, spielt bei dieser Betrachtung eine wichtige Rolle. Außerdem ist es fraglich, ob man hier von einem „Restrisiko" sprechen kann, wenn man die Verwendung des Basislesers schlicht vermeiden könnte.

Zweitens werden unter dem allgemeinen Begriff „Awareness" bestimmte Kenntnisse des Nutzers vorausgesetzt, die Risiken minimieren sollen.[207] Dazu gehören die Unterscheidung zwischen eID- und Signaturanwendung oder Basis-, Standard- und Komfortleser. Außerdem wird vom Nutzer verlangt, dass er Antiviren-Software, Firewall und automatische Updates installiert, um einen Befall von Schadsoftware zu vermeiden. Zudem ist es wichtig, dass auch die Maßnahmen bei einem Verlust oder bemerkten Missbrauch hinreichend bekannt sind. Ich werde hier zeigen, dass Skimming verschiedene Angriffe ermöglicht – trotz aller Vorsichtsmaßnahmen des Ausweisinhabers.

Zum Dritten wird bemerkt, dass es technisch möglich ist, das Sperrmerkmal des Ausweises zum Tracing zu verwenden.[208] Die Zugriffsberechtigung *Restricted Identification* ist eigentlich gedacht zur Generierung eines Dienst- und Nutzer-spezifischen Pseudonyms.[209] Genau dieses Protokoll wird aber zur Prüfung auf Revozierung genutzt. Obwohl es beispielsweise lediglich das Ziel eines Dienstanbieters sein kann, eine Altersverifikation durchzuführen, wird es ihm

[202] Das Feld `commCertificates` der Zertifikatsbeschreibung enthält Hash-Werte der zugelassenen SSL-Zertifikate des Dienstanbieters (BSI TR-03110 2.04, S. 88). Ein Hash der Zertifikatsbeschreibung ist wiederum als Certificate Extension in dem vom nPA geprüften Terminalzertifikat enthalten (BSI TR-03127, S. 29–30).

[203] Morgner und Oepen 2010.

[204] Weiterhin bestehende und in Teilen komplexe Probleme werden im Rest dieses Kapitels betrachtet.

[205] Dietrich, Rossow und Pohlmann 2010, S. 9.

[206] Dietrich, Rossow und Pohlmann 2010, S. 4–5.

[207] Dietrich, Rossow und Pohlmann 2010, S. 6.

[208] Dietrich, Rossow und Pohlmann 2010, S. 7–8.

[209] BSI TR-03110 2.05, S. 38, 90.

zum Zweck der Gültigkeitsprüfung gestattet, das eindeutige Sperrmerkmal zu lesen.[210] Nach Einschätzung von Dietrich, Rossow und Pohlmann stellt dies aber kein großes Risiko dar, weil ein Missbrauch dieser Daten gegen geltendes Recht verstößt. Unabhängig von dieser Behauptung, werde ich in „Tracing", S. 46 weitere Merkmale diskutieren, die ebenfalls zum Tracing verwendet werden können.

Als letztes Restrisiko werden mögliche Fehler bei der Umsetzung entsprechender Software genannt.[211] Ihnen könne mit der Zertifizierung von Implementierungen begegnet werden. Darüber hinaus wird aber festgestellt, „dass die aktuelle Spezifikation des Kommunikationsmodells für die eID-Funktion keine Schwachstellen aufweist".[212] Ich werde in den folgenden Abschnitten verschiedene Angriffe demonstrieren, die die kryptographischen Sicherungen komplett umgehen. Außerdem erwächst daraus das Problem der Privilege Escalation, das zu schwerwiegenden Konsequenzen für den Ausweisinhaber führen kann.

2.4.1. Missbrauch des elektronischen Identitätsnachweises

Die Zwei-Faktor-Authentisierung mit dem nPA soll sicherer sein als die Authentisierung mit einem Passwort, einem einzelnen Merkmal.[213] Selbst wenn es einem Angreifer gelingen würde, die eID-Anwendung zu missbrauchen, könnten nur berechtigte Dienstanbieter die Ausweisdaten lesen, nicht aber ein Angreifer. „Unter keinen Umständen" sei es möglich, „Einblick in die persönlichen Daten des Ausweis-Inhabers bekommen, da sie verschlüsselt übermittelt" würden.[214]

Ich behaupte, dass sich auch der elektronische Identitätsnachweis relativ einfach missbrauchen lässt und ein Angreifer unberechtigterweise die Ausweisdaten ermitteln kann. Mindestens zwei verschiedene Angriffe sind denkbar. Erstens könnte ein Angreifer den Identitätsnachweis eigenständig und möglicherweise ohne Wissen des Ausweisinhabers durchführen. Zweitens könnte ein legitimer Identitätsnachweises des Ausweisinhabers manipuliert werden, um ihn zu missbrauchen.

Mit einer Internetverbindung, dem Wissen über die eID-PIN und vollen Zugriff auf den nPA kann ein also Angreifer beliebige Online-Dienste für den nPA nutzen (Malware mit Basisleser, Skimmer). Damit ist nicht nur beispielsweise der Missbrauch zum Login bei Webdiensten, bei Kontoeröffnungen oder zur Bestellung beim Online-Shopping möglich.[215]

Beim Online-Versandhandel müssen dem Käufer die Liefer- und Rechnungsaddresse zur Bestätigung angezeigt werden.[216] Dies bedeutet, dass ein Angreifer die vom berechtigten Dienst ausgelesenen Daten über die Anzeige auf der Website erfährt, ohne dass er im Sinne der EAC dazu berechtigt wäre (SP.TA). Dieser Angriff ist derzeit mittels der Einsicht in das Punktekonto beim Kraftfahrt-Bundesamt möglich.[217] Zu diesem Zweck werden aus

[210] BSI TR-03110 2.05, S. 90.
[211] Dietrich, Rossow und Pohlmann 2010, S. 8–9.
[212] Dietrich, Rossow und Pohlmann 2010, S. 9.
[213] BSI 2010b.
[214] Kossel 2010.
[215] Siehe „Die elektronische Nutzung des neuen Personalausweises", S. 5.
[216] EGBGB, Art. 246, § 2.
[217] KBA 2011.

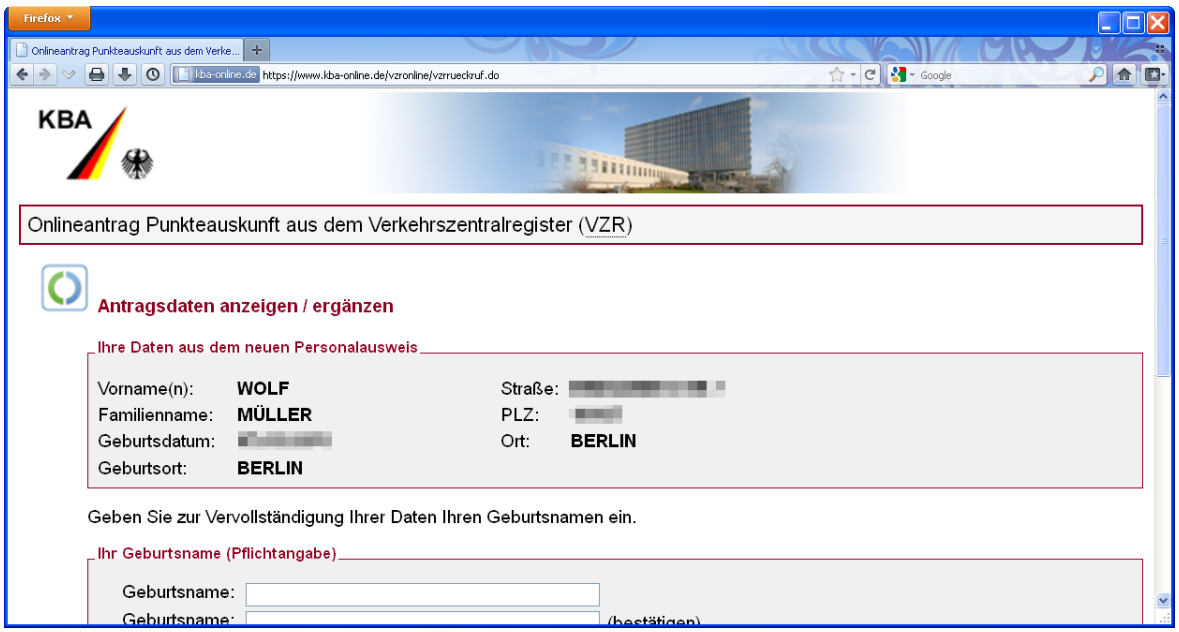


Abbildung 2.17.: Nachdem die Daten des nPA erfolgreich ausgelesen wurden, werden sie auf der Website des KBA angezeigt (KBA 2011).

dem Ausweis Vor- und Familienname, Anschrift, Geburtsdatum, -ort und -name ausgelesen und dem Aufrufenden angezeigt (siehe Abbildung 2.17).[218] Über diesen „Seitenkanal“ wird also die Vertraulichkeit der ausgetauschten Daten zwischen nPA und Dienstanbieter (nPA.Confidentiality) nicht kryptografisch gebrochen sondern lediglich umgangen. Der Angreifer kann sich zwar nicht sicher sein, dass die ihm angezeigten Daten korrekt und authentisch sind (vgl. nPA.Authenticity, nPA.Integrity). Aber in diesem Modell wird angenommen, dass der Dienstanbieter sich rechtskonform verhält und keine Modifikationen der Daten vor der Anzeige auf der Website vornimmt.

Aufwendiger als die missbräuchliche Nutzung von Online-Diensten ist die Nutzung des fremden Ausweises an einem (öffentlichen) Terminal. Wenn z. B. der Ausweis zur An- und Abmeldung in sicherheitsrelevanten Gebäudebereichen[219] genutzt werden soll, muss das Terminal den Ausweis auch per NFC erkennen können. Dies kann ebenfalls mit einem Relay-Angriff umgesetzt werden, wie in „Privilege Escalation“, S. 52 gezeigt wird.

Kennt ein Angreifer nicht die eID-PIN, kann er versuchen, den Ausweisinhaber zu täuschen, um ihn zur Authentisierung einer ungewollten Transaktion zu bewegen. Besonders wenn der Ausweisinhaber ausschließlich ein sicheres PACE-fähiges Lesegerät verwendet, könnte der Angreifer (Malware mit PACE-fähigem Leser, Man-in-the-Middle Hardware) die Daten manipulieren, nachdem sie mit der Client-Software angezeigt wurden. Der Angreifer baut selbstständig eine Verbindung zu einem (anderen) Dienstanbieter auf und leitet dies dann an den PACE-fähigen Leser weiter (statt der legitimen Transaktion). Nachdem der Ausweisinhaber – im Glauben eine legitime Transaktion zu autorisieren – die eID-PIN am Leser eingegeben

[218] Der eigentliche Punktestand wird dem Ausweisinhaber per Post zugeschickt.
[219] Siehe „Die elektronische Nutzung des neuen Personalausweises“, S. 5.

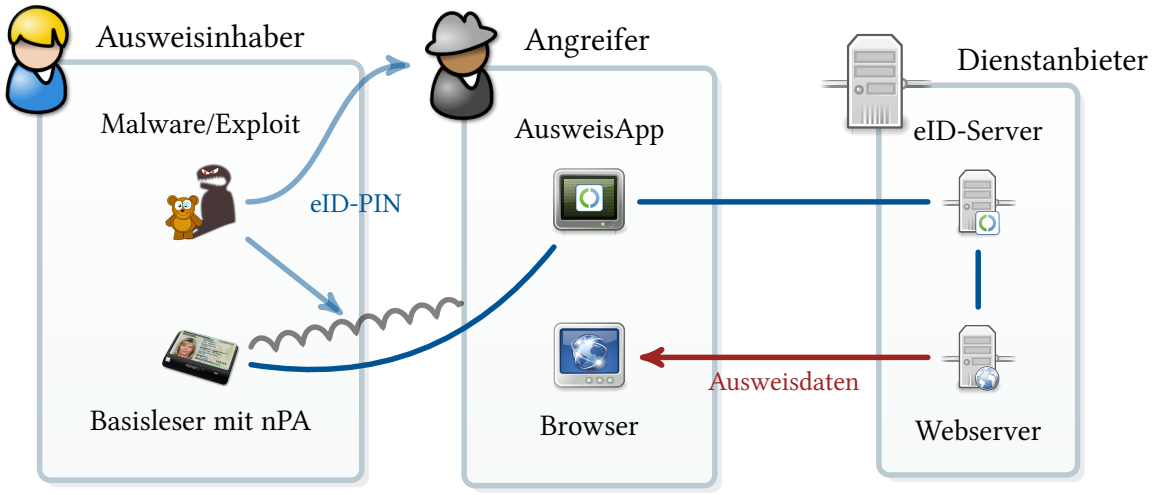


Abbildung 2.18.: Relay-Angriff zum unberechtigten Ermitteln der Ausweisdaten. Der Basisleser mit nPA wird virtuell an den PC des Angreifers gesteckt. Nach erfolgreichem Lesen der Ausweisdaten werden diese dem Aufrufenden (dem Angreifer) zur Bestätigung angezeigt.

hat, wird der Identitätsdiebstahl ohne weiteres Zutun des Ausweisinhabers oder seiner Client-Software durchgeführt.

Sofern der vertrauenswürdige PACE-fähige Leser ein Display besitzt (IFD.Display), könnte der Ausweisinhaber den Betrug erkennen.[220] Aber bei einem geforderten Display von zwei mal sechzehn Zeichen reicht dies „nicht aus, um die im Zertifikat enthaltenen Daten umfassend und übersichtlich darzustellen. Daher kann das Leserdisplay in der Praxis maximal zur nochmaligen, stichprobenartigen Überprüfung“[221] dienen. Zudem muss, vorgegeben von den im Feld befindlichen eID-Servern, der Authentisierungsvorgang innerhalb von etwa ein bis zwei Minuten abgeschlossen sein. Der Ausweisinhaber hätte also auch nur wenig Zeit, die Angaben des Dienstanbieters eingehend zu prüfen.

„Der Ausweisinhaber ist für Schäden, die durch den Missbrauch entstehen, nur bei Verletzung seiner Sorgfaltspflicht verantwortlich.“[222] Pflichten des Ausweisinhaber schließen die Nutzung von Firewall, Virenscanner, System- bzw. Programm-Updates, getrennter Aufbewahrung von eID-PIN und nPA und Sperrung bei Verlust des Ausweises ein.[223] „Die Leitlinie der Risikoverteilung lautet, dass der Getäuschte das Risiko [...] trägt.“[224]

[220] Siehe „Lesegeräteklassen nach BSI TR-03119 1.2“, S. 15.
[221] „Kartenleser für den neuen deutschen Personalausweis“, S. 763.
[222] Borges 2011, S. 275.
[223] Borges 2011, S. 281–282.
[224] Borges 2011, S. 276.

2.4.2. Tracing

Unter Tracing wird nach BSI-CC-PP-0061 die eindeutige Identifikation durch ein entferntes Terminal verstanden. Zunächst sei darauf hinweisen, dass bereits eine Möglichkeit zum Tracing beschrieben wurde. In „Bedrohungsszenarien“, S. 42 wurde bemerkt, dass das Sperrmerkmal eines nPA ein dienst- und kartenspezifisches Merkmal ist. Damit erhält ein lediglich zum Altersnachweis berechtigter Dienstanbieter ein Mittel zur Wiedererkennung des Ausweises. Die zweckentfremdete Auswertung dieses Merkmals ist den Dienstanbietern untersagt.

Auch die zweckfremde Nutzung von Informationen, die aus der EAC abgeleitet werden, dürfte dem Dienstanbieter nicht erlaubt sein. Im `EF.CardSecurity` sind unter anderem der öffentliche CA-Schlüssel des nPA[225] und die „Usage Period“ des Document Signers[226] gespeichert. Beide erlauben eine Eingrenzung des Produktionsszeitraums des Ausweises auf einen Zeitraum von höchstens vier Monaten.[227] Allerdings ist das Anonymity-Set dennoch relativ groß,[228] weshalb sich diese Merkmale für das Tracing nur in Verbindung mit anderen Daten lohnt. Beide Informationen sind im `EF.CardSecurity` erst nach der TA, d. h. für Terminals die wahrscheinlich auch das Sperrmerkmal prüfen, lesbar. Der öffentliche CA-Schlüssel erlaubt also keine genauere Eingrenzung als das Sperrmerkmal. Aber anders als das Sperrmerkmal ist das `EF.CardSecurity` eines bestimmten Ausweises für verschiedene Dienste identisch und auch für einen Angreifer lesbar, der nach PACE die Kommunikation zum Dienstanbieter lesen kann[229] (Malware, Nearby Attacker, Skimmer, Man-in-the-Middle Hardware).

Im Zusammenhang mit der Echtheitsprüfung muss aber auch eine bisher noch nicht genutzte Funktion für einen möglichen Tracing-Angriff betrachtet werden. Das Sperrmerkmal wird nach der EAC vom Ausweis an den Dienst übertragen. Die Korrektheit des Sperrmerkmals basiert allein darauf, dass nur vertrauenswürdige nPAs die CA durchführen können. Wäre ein Angreifer aber in der Lage, einen privaten CA-Schlüssel aus einem der Ausweise zu extrahieren, könnte ein emulierter Ausweis die EAC vollständig korrekt durchführen und im Anschluss ein beliebiges Sperrmerkmal wählen.[230] Mit der BSI TR-03110 2.04 wurde daher ein Chip-eindeutiger signierter Schlüssel eingeführt. Im Gegensatz zum Sperrmerkmal ist dieses Merkmal für alle Dienste gleich und ermöglicht damit die Wiedererkennung des Ausweises auch bei anderen Dienstanbietern. *Privilegierte Terminals* (Inspektionsterminals

[225] Siehe „EF.CardSecurity“, S. 131, Bytes 239–372.

[226] Siehe „EF.CardSecurity“, S. 131, Bytes 1044–1048.

[227] Die Usage Period des Schlüssels vom Document Signer umfasst vier Monate, vgl. Fußnote 226.

[228] Insgesamt 60 Mio. Ausweise (VfB 2011) sind je 10 Jahre gültig. Das heißt, dass bei absoluter Gleichverteilung in vier Monaten $\frac{60\text{Mio.}}{10 \cdot 12\text{m}} \cdot 4\text{m} = 2\text{Mio.}$ Ausweise produziert werden. Der CA-Schlüssel ist unabhängig von dem Schlüssel des Document Signers und grenzt das Anonymity-Set weiter ein. Die Regeln zum Umgang mit dem CA-Schlüssel sind nicht öffentlich, ein Rechenbeispiel ist in „Chipauthentisierung und Passive Authentisierung“, S. 14 gegeben. Da beide Schlüssel unabhängig voneinander sind, erlauben sie gemeinsam eine stärkere Eingrenzung der Menge möglicher Ausweise.

[229] Die SM geschützte Kommunikation vom nPA zum Dienstanbieter (nPA.Confidentiality, nPA.Authenticity) wird erst nach der vollständiger EAC gestartet. Das `EF.CardSecurity` wird nach der TA ausgelesen, wobei nur die Luftschnittstelle zum Ausweis geschützt ist (nach PACE). Damit ist SP.TA zwar notwendig, um den nPA in den entsprechenden Zustand zu versetzen – nicht aber um das `EF.CardSecurity` (mit-) zu lesen. Z. B. nach PACE mit CAN kann ein Angreifer die Kommunikation zu einem echten Dienstanbieter aufbauen. Nachdem dieser das `EF.CardSecurity` nach erfolgreicher TA liest, erhält es auch der Angreifer.

[230] Vgl. Plötz 2009.

und Authentisierungsterminals mit entsprechender Berechtigung) dürfen das Merkmal prüfen (siehe Tabelle 2.5). Dem Dienstanbieter sollen auch hier die schon erwähnten rechtlichen Regelungen einen solchen Missbrauch verbieten. Ich will daher im Folgenden Tracing-Angriffe behandeln, die keine Terminal-Berechtigungen zum Zugriff auf den nPA benötigen.

Tracing per elektronischem Identitätsnachweis

Eine Möglichkeit des Tracing leitet sich direkt aus den Beobachtungen zu „Missbrauch des elektronischen Identitätsnachweises“, S. 43 ab. Besitzt ein Angreifer gleichzeitig die eID-PIN und Zugriff auf den nPA (Malware mit Basisleser, Skimmer), kann er den elektronischen Identitätsnachweis im Namen des Ausweisinhabers durchführen. Da die gelesenen Ausweisdaten von einigen Dienstanbietern dem Aufrufenden angezeigt werden, kann der Angreifer auf diese Weise zum Namen, der Adresse und anderen Ausweisdaten des Inhabers gelangen. Über diese Daten ist der Ausweisinhaber eindeutig identifizierbar. Da die Daten im Ausweis recht statischer Natur sind, kann der Angriff zu einem späteren Zeitpunkt wiederholt werden.[231]

Tracing per CAN

Auch die Eindeutigkeit des Passwortes identifizieren einen nPA. Die änderbare eID-PIN ist in diesem Falle weniger problematisch. Für Angreifer interessanter ist die CAN. Zum einen kann sie optisch vom Kartenkörper gelesen (Nearby Attacker mit Sichtkontakt zum nPA) oder gestohlen werden (Malware mit Basisleser, Skimmer eines PACE-fähigen Lesegeräts). Zum anderen kann ein Angreifer mit genügend langem Zugriff auf den Ausweis sie per Brute-Force-Angriff ermitteln (Malware, Man-in-the-Middle Hardware). Der nPA implementiert keine Gegenmaßnahmen zum Schutz der CAN, wie z. B. eine Zeitverzögerung nach fehlgeschlagenen Authentisierungen.[232] Zwar sollen Geheimnisse nach BSI TR-03110 2.05, S. 24 gegen Brute-Force-Angriffe geschützt werden. Aber in BSI-CC-PP-0061 wird die CAN nicht als Geheimnis („represent secret“), sondern nur als „restricted revealable“ klassifiziert.[233]

Die in Morgner und Oepen 2010 ermittelten sechs Tage zur Ermittlung der CAN konnten mit dem `npa-tool`[234] und einem Basisleser auf durchschnittlich 4,2 Tage verkürzt werden (siehe Abbildung 2.19). Diese durchschnittliche Laufzeit stellt dabei etwa die untere Grenze dar, weil sie zum Großteil durch die Geschwindigkeit des nPA vorgegeben ist (siehe Tabelle 2.6). In BSI-CC-PP-0061 wird der Aufwand zum Erraten der CAN als „beyond high“ eingestuft.[235] Das Ergebnis des Brute-Force-Angriffs widerspricht dieser Aussage.

Die Nutzung einer eigenen PACE-Implementierung bei einem PACE-fähigen Lesegerät ist für einen Angreifer seit BSI TR-03119 1.2 nicht mehr möglich. Ein Standard- oder Komfortleser kann aber den Brute-Force-Angriff auf die CAN nicht verhindern. Im Gegenteil, die PACE-Funktionalität des Lesers kann missbraucht werden, um die CAN zu ermitteln. Mit dem `npa-tool` lassen sich nacheinander auch direkt beliebige CANs vom PC an den Leser senden,

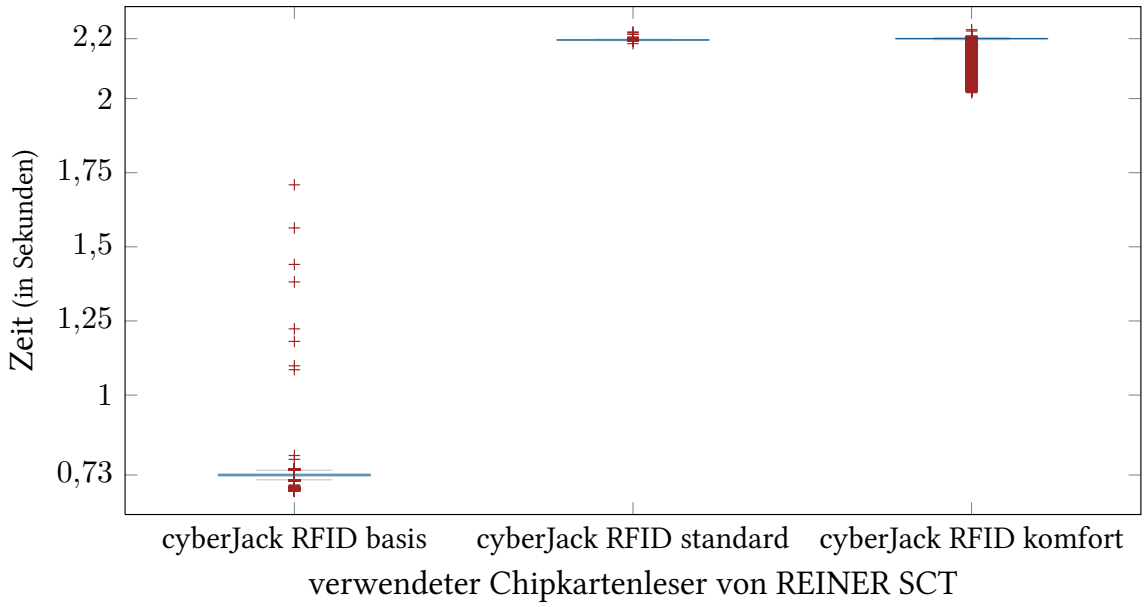
[231] In einigen Fällen könnte sich die Anschrift ändern (BSI TR-03127, S. 12). Diese sollten aber Ausnahmen darstellen und werden hier vernachlässigt.

[232] Siehe „Password Authenticated Connection Establishment“, S. 7.

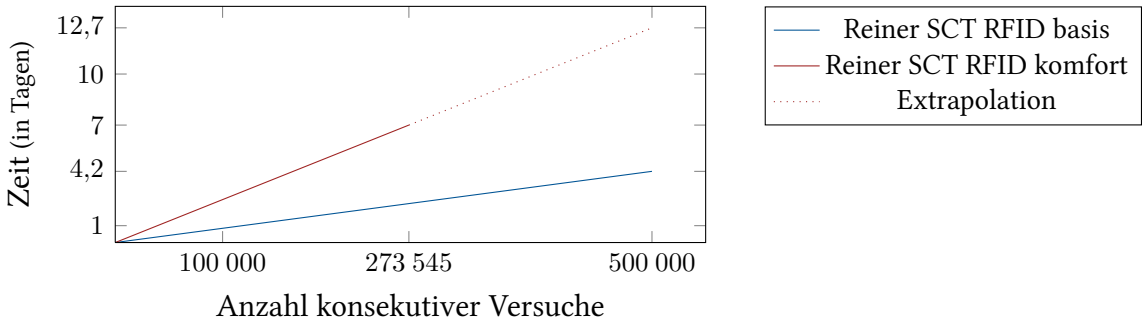
[233] BSI-CC-PP-0061, S. 23,63,65.

[234] Siehe „nPA Smart Card Library“, S. 99.

[235] BSI-CC-PP-0061, S. 62.



(a) Die Komfort- und Standardleser von Reiner SCT benötigen für einen PACE-Durchlauf mit durchschnittlich 2,2 Sekunden etwa gleich viel Zeit (273 545 bzw. 119 999 Versuche, Standardabweichung 7,3 ms bzw. 0,9 ms). Dahingegen benötigt die PACE-Implementierung des `npa-tool` mit dem Basisleser Reiner SCT cyberJack RFID basis durchschnittlich 0,73 Sekunden (500 000 Versuche, Standardabweichung 6,4 ms).

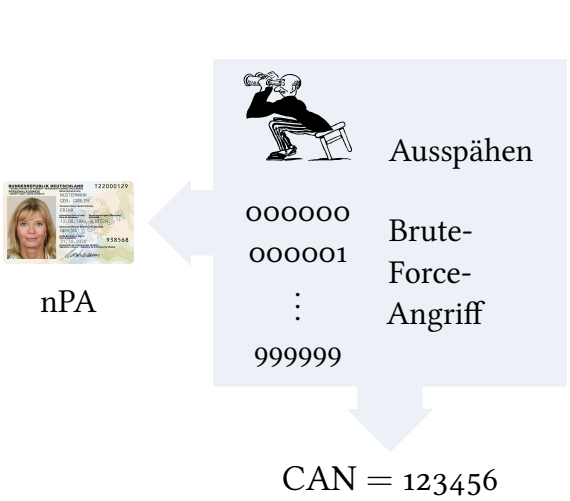


(b) Der nPA oder aktuelle PACE-fähige Leser implementiert keine Zeitverzögerung bei fehlgeschlagenen Authentisierungsversuchen mit CAN. Ein Brute-Force-Angriff benötigt durchschnittlich 4,2 Tage mit einem Basisleser und 12,7 Tage mit einem Standard- oder Komfortleser von Reiner SCT.

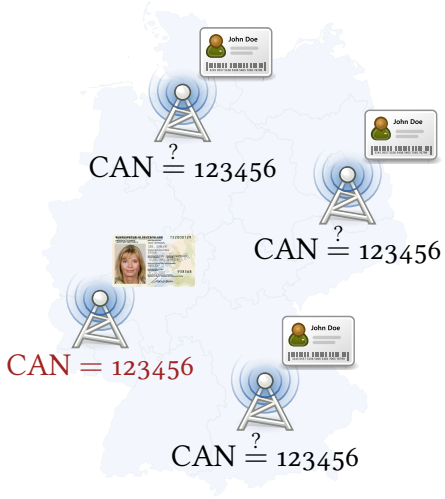
Abbildung 2.19.: Zeitverlauf beim Brute-Force-Angriff auf die CAN des nPA. Mit `npa-tool --can --break` kann dafür sowohl die PACE-Funktion eines entsprechenden Chipkartenlesers als auch eine eigenständige Umsetzung des Protokolls genutzt werden.

Anteil in %	Aufrufe	Bibliotheksfunktion	
89,73	5	`sc_transmit_apdu`	I/O Chipkarte
4,07	1	`sc_reset`	
2,63	1	`sc_connect_card`	
2,03	1	`sc_disconnect_card`	
0,55	1	`sc_read_binary`	
0,46	1	`sc_select_file`	
0,03	3	`sc_detect_card_presence`	
0,05	1	`PACE_STEP3A_map_dp`	Kryptografische Operationen
0,02	1	`PACE_STEP3A_generate_mapping_data`	
0,02	1	`PACE_STEP3B_compute_ephemeral_key`	
0,02	1	`PACE_STEP3B_generate_ephemeral_key`	
0,00	1	`parse_ef_card_access`	
0,00	1	`PACE_STEP3D_verify_authentication_token`	
0,00	1	`PACE_STEP3D_compute_authentication_token`	
0,00	1	`PACE_STEP2_dec_nonce`	
0,00	1	`PACE_STEP3C_derive_keys`	
0,39	140	Verschiedene weitere Funktionen	

Tabelle 2.6.: Laufzeitübersicht für PACE mit `npa-tool` mittels `ltrace`. Bei dem verwendeten PC wird die meiste Zeit auf Antworten der Chipkarte gewartet, OpenPACE bzw. das `npa-tool` arbeiten also sehr effizient.



(a) Durch Ablesen oder Ausprobieren per PACE gelangt ein Angreifer zur CAN, die dem nPA zugeordnet ist.



(b) PACE mit CAN kann an verteilten Stationen beliebig durchgeführt werden. Ist eine Authentisierung mit der gegebenen CAN erfolgreich, ist der nPA damit fast sicher wieder erkannt.

Abbildung 2.20.: Tracing eines nPA mittels seiner CAN

die dieser dann für PACE gegenüber dem nPA benutzt. PACE mit vorgegebener CAN zum Entsperren der eID-PIN oder zur Nutzung der Signaturfunktion ist ein expliziter Anwendungsfall beider Lesertypen.[236] Das entsprechende Interface zum Chipkartenleser wird von BSI TR-03119 1.2 vorgegeben,[237] der Missbrauch zum Ermitteln der CAN aber nicht verhindert. Die Lesegeräte Reiner SCT cyberJack RFID standard bzw. komfort benötigen für einen PACE-Durchlauf gut zwei Sekunden. Der Angreifer muss bei einem PACE-fähigen Leser also durchschnittlich etwa dreimal soviel Zeit einplanen wie für den Brute-Force mit einem Basisleser, d. h. knapp zwei Wochen (siehe Abbildung 2.19b).

Die CAN wird bei der Herstellung des Ausweises zufällig erzeugt und besteht aus sechs Ziffern, d. h. es gibt eine Million mögliche CANs. Bei geschätzten 60 Millionen Ausweisinhabern gibt es nur eine geringe Anzahl von Ausweisen mit gleicher CAN. Bei absoluter Gleichverteilung aller möglichen Kombinationen gäbe es etwa 60 nPAs je CAN. Um einen nPA wieder zu erkennen, muss ein Angreifer nur einen PACE-Durchlauf mit der ihm bekannten CAN starten. Ist der Vorgang erfolgreich und bezieht man räumliche Lokalität der Nutzung oder die Daten des `EF.CardSecurity`[238] in der Betrachtung hinzu, ist die Identifizierung des gesuchten Ausweisinhabers fast sicher (siehe Abbildung 2.20).

Anders als bei der Nutzung der eID-PIN hilft in diesem Falle nicht das Ändern des Geheimnisses. Eine Änderung der CAN ist schlicht nicht vorgesehen.[239]

[236] BSI TR-03119 1.2, S. 18; BSI TR-03117, S. 25.
[237] Siehe „Lesegeräteklassen nach BSI TR-03119 1.2“, S. 15.
[238] Siehe „Tracing“, S. 46.
[239] BSI TR-03127, S. 15.

Tracing per MRZ

Statt der CAN könnte man zum Tracing auch die MRZ verwenden. Sie ist – anders als die CAN – eineindeutig einem Ausweis zugeordnet. Nutzt man die MRZ als PACE-Passwort, so sind lediglich die Ausweisnummer, das Geburtsdatum des Inhabers und das Ablaufdatum relevant.[240] Bereits bei der Diskussion um die Sicherheit von BAC wurde argumentiert, dass die MRZ wenig Entropie besitzt[241] (wenngleich es einen größeren Raum an Möglichkeiten gibt als für die CAN). Ausweisbehörden könnten die Bestandteile der MRZ eines gesuchten Inhabers rekonstruieren und dessen Inhaber damit Tracen (auch *ohne* im Besitz eines Terminalzertifikats zu sein).

Auch mit etwas Vorwissen ließe sich die Entropie (und damit der Aufwand für einen Brute-Force-Angriff) verringern. Für das Ablaufdatum gibt es bei einer Gültigkeit des Ausweises von zehn Jahren weniger als 3652 Möglichkeiten.[242] Ein Angreifer, dem das `EF.CardSecurity` bekannt ist, (Malware, Skimmer, Man-in-the-Middle Hardware) hätte aber die Möglichkeit den Zeitpunkt der Ausstellung (und damit des Ablaufs) anhand des öffentlichen CA-Schlüssels des nPA auf einen Zeitraum von weniger als 18 Monaten einzugrenzen, d. h. auf weniger als 547 Möglichkeiten.[243] Das Geburtsdatum stellt in dem meisten Fällen ein offenes Geheimnis dar.

Weil sich Ablauf- und Geburtsdatum stark eingrenzen lassen, steht und fällt ein Brute-Force-Angriff auf die MRZ des nPA mit der Stärke der Ausweisnummer. Sie ist beim nPA neun alphanumerische Zeichen lang, theoretisch also ein sehr großer Raum an Möglichkeiten. Werden die Ausweisnummern (pseudo-) zufällig vergeben, kann man diese kaum durch Probieren ermitteln. Genaue Mechanismen sind jedoch nicht vorgeschrieben, so dass potenziell auch leicht ermittelbare Ausweisnummern entstehen könnten.[244] Gelangt man aber anderweitig zu der MRZ, in dem man sie z. B. vom Ausweis abliest (Nearby Attacker mit Sichtkontakt zum nPA), kann man einen Ausweis auch eindeutig wiederfinden.

---

[240] BSI TR-03110 2.05, S. 54.

[241] Für eine Übersicht siehe z. B. Chothia und Smirnov 2010, S. 22; Kc und Karger 2005. Siehe auch „Password Authenticated Connection Establishment“, S. 7.

[242] Die Anzahl der möglichen Ablaufdaten in zehn Jahren beträgt $10 \cdot 365 + 2 = 3652$ abzüglich Sonn- und Feiertage.

[243] Siehe „Chipauthentisierung und Passive Authentisierung“, S. 14.

[244] In die Ausweisnummer des deutschen elektronischen Reisepasses gehen seit dem 1. Januar 2007 die vierstellige Behördenkennzahl und fünf pseudozufällige Ziffern aus einem 27 Zeichen umfassenden Alphabet ein (Bender 2010b, S. 71,a). Der Ausweisinhaber wohnt sicherlich in der Nähe der ausstellenden Behörde, die Behördenkennzahl ließe sich daher raten. (Insgesamt existieren schätzungsweise 6500 Meldestellen, denen eine Behördenkennzahl zugeordnet ist. Eine Liste aller deutschen Behördenkennziffern wurde nach dem Informationsfreiheitsgesetz am 08.03.2006 beantragt (`http://www.befreite-dokumente.de/beantragte-akten/behoerdenkennziffern`), bisher aber noch nicht veröffentlicht.) Für den pseudozufälligen Anteil der Ausweisnummer des Reisepasses gibt es insgesamt 27^5 mögliche Werte. Benötigt ein Rateversuch 0,73 Sekunden (vgl. Abbildung 2.19a), so dauert ein Brute-Force-Angriff durchschnittlich zwei Monate. Diese Zeitspanne ist sicherlich zu lang, als dass der Angriff erfolgreich wäre. Eine so konstruierte Ausweisnummer könnte auch beim nPA genutzt werden.

Tracing per Signaturanwendung

Die Speicherung des qualifizierten Zertifikats in der Signaturanwendung des nPA ist optional.[245] Nach Informationen des Autors soll es im nPA gespeichert werden damit die Verifikation einer QES durch den Ausweisinhaber auch offline möglich ist. Das qualifizierte Zertifikat wurde vom ZDA unterschrieben und enthält unter anderem den öffentlichen Signaturschlüssel und den Namen des Ausweisinhabers. Darüber ist der Ausweisinhaber bzw. nPA eindeutig verfolgbar.

Das qualifizierte Zertifikat ist nur dann im nPA gespeichert, wenn der Ausweisinhaber seinen nPA für die QES vorbereitet hat oder der Angreifer dies in seinem Namen getan hat.[246] Als weitere Bedingung benötigt der Angreifer die CAN und Kontrolle über einen Komfortleser, um sich dem nPA gegenüber als berechtigtes Signaturterminal auszuweisen (siehe „Privilege Escalation“, S. 52.

Das Tracing per Signaturanwendung ist damit durchführbar, wenn der nPA vom Ausweisinhaber für die QES vorbereitet ist, der Angreifer vollständigen Zugriff auf den nPA hat und die CAN kennt oder ermitteln kann[247] (Malware mit Basisleser, Skimmer eines PACE-fähigen Lesegeräts). Ist der Ausweis nicht für die QES vorbereitet, benötigt der Angreifer zusätzlich die eID-PIN, um eine Privilege Escalation[248] zu erreichen (Malware mit Basisleser, Skimmer eines PACE-fähigen Lesegeräts).

2.4.3. Privilege Escalation

Ein Angreifer, der in der Lage ist, die eID-Anwendung zu missbrauchen (Malware mit Basisleser, Skimmer, eingeschränkt auch Malware mit PACE-fähigem Leser und Man-in-the-Middle Hardware),[249] kann dies auch verwenden, um den nPA für die QES vorzubereiten. Dazu muss erstens eine Signatur-PIN auf dem Ausweis gesetzt werden und zweitens der Signaturschlüssel bzw. das qualifizierte Zertifikat nachgeladen werden.[250]

Der ungewollte Übergang vom eID- in den eSign-Kontext hat für den Ausweisinhaber vor allem rechtliche Konsequenzen. Soll der Ausweisinhaber für Schäden haften, muss ihm beim Missbrauch des elektronischen Identitätsnachweises eine Pflichtverletzung nachgewiesen werden.[251] Bei einer geleisteten Signatur hingegen, ist es der Ausweisinhaber, welcher den Anscheinsbeweis der QES erschüttern muss.[252] Die Privilege Escalation führt zu einer Beweislastumkehr zu ungunsten des Ausweisinhabers. [253]

Das Nachladen der Signaturschlüssel und des Zertifikats wird mittels des elektronischen Identitätsnachweises und der eID-PIN autorisiert. Ein Angreifer, der die eID-Anwendung

[245] BSI TR-03117, S. 24.
[246] Siehe „Privilege Escalation“, S. 52.
[247] Siehe „Tracing per CAN“, S. 47.
[248] Siehe „Privilege Escalation“, S. 52.
[249] Siehe „Missbrauch des elektronischen Identitätsnachweises“, S. 43.
[250] Siehe „Nutzung der Signaturanwendung“, S. 26.
[251] Siehe „Missbrauch des elektronischen Identitätsnachweises“, S. 45.
[252] Siehe „Missbrauch der Signaturfunktion“, S. 55.
[253] Eine Rechtspraxis oder ein Rechtsgutachten zur Signatur mit dem nPA liegt noch nicht vor, daher sind die hier getroffene Erläuterung lediglich informativ zu verstehen.

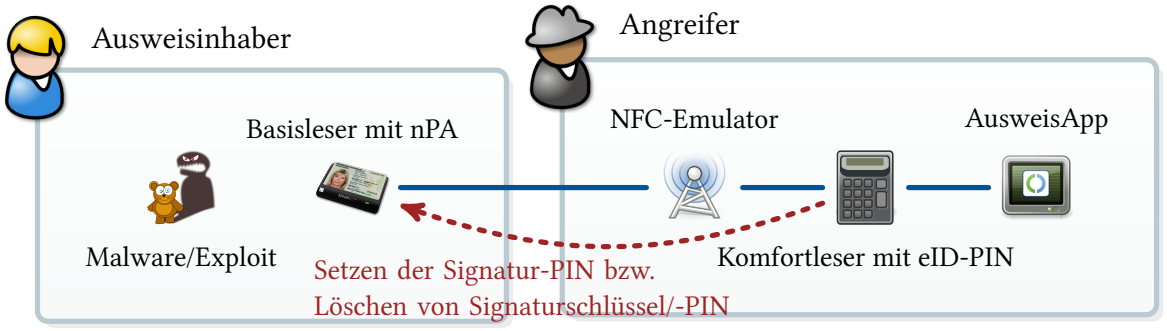


Abbildung 2.21.: Relay-Angriff zum unberechtigten Aktivieren und Terminieren des nPA als SSEE. Durch Malware oder einen lokalen Exploit werden eID-PIN und der Zugriff auf den Ausweis an den Angreifer weitergeleitet (vgl. Abbildung 2.18). Nach dem Setzen der Signatur-PIN lädt der Angreifer den Signaturschlüssel nach, indem er das Relaying zum ZDA wie in Abbildung 2.18 dargestellt durchführt.

missbrauchen kann, ist also offensichtlich in der Lage, ihn auch gegenüber eines ZDA zu verwenden.

Für die Installation der Signatur-PIN wird jedoch ein Komfortleser benötigt. Dass der nPA aber tatsächlich in einen Komfortleser eingelegt sein muss,[254] ist jedoch falsch. Der Angreifer kann den Zugriff auf den Ausweis, der sich z. B. auf einem Basisleser befindet, zu einem NFC-Emulator weiterleiten. Letzteren benutzt der Angreifer gegenüber einem Komfortleser, der unter seiner Kontrolle steht. Mit der eID-PIN kann der Angreifer am PIN-Pad seines Komfortlesers nun eine neue Signatur-PIN installieren. Sollte auf dem Ausweis bereits eine Signatur-PIN (vom Ausweisinhaber) gesetzt sein, kann der Angreifer sie ebenfalls mit der eID-PIN löschen.[255]

Der nPA ist mit diesem Relay-Angriff (siehe Abbildung 2.21) vollständig für die QES vorbereitet und der Angreifer ist im Besitz der Signatur-PIN. Ein vergleichbarer Angriff gegen den nPA wurde bereits demonstriert.[256] Dass Relay-Angriffe nicht auf die Signaturanwendung des nPA anwendbar seien,[257] wurde damit widerlegt.

Extrahiert der Angreifer einen privaten Schlüssel und das TA-Zertifikat aus einem Komfortleser, kann der Angriff automatisiert durchgeführt werden.[258] Der Angreifer muss nun nicht mehr den Komfortleser von Hand bedienen und kann die Anzahl der Angriffe in gleicher Zeit erhöhen (und damit die Anzahl potenzieller Opfer).

Tatsächlich entspricht der verteilte Aufbau im Relay-Angriff exakt der Spezifikation von

[254] bsipressemitteilungerneut
[255] Siehe „Nutzung der Signaturanwendung“, S. 26. Damit korrigiere ich meinen in Morgner und Oepen 2010 vertretenen Standpunkt.
[256] Morgner und Oepen 2010.
[257] BSI 2010a.
[258] Morgner und Oepen 2010.

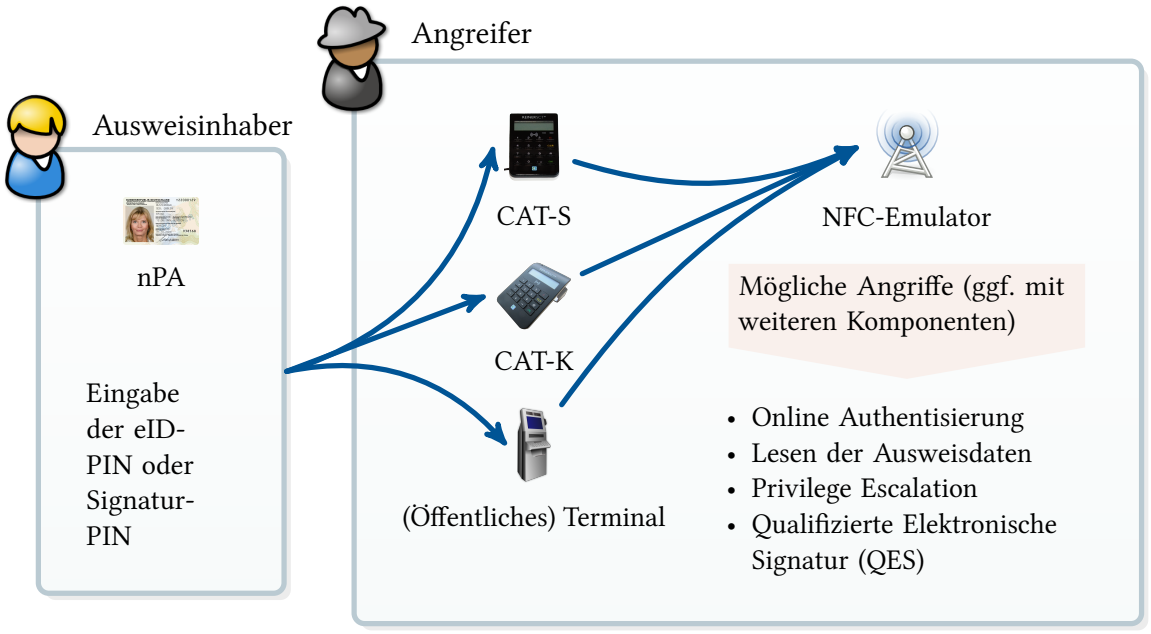


Abbildung 2.22.: Skimming eines Standardlesers (CAT-S), Komfortlesers (CAT-K) oder öffentlichen Terminals ermöglicht es einem Angreifer volle Kontrolle beider Authentisierungsfaktoren bei der Nutzung des nPA zu erlangen.

verteilten Terminals,[259] die gemeinsam ein kryptographisches Modul nutzen (siehe Abbildung 2.23), wenngleich diese böswillige Nutzung nicht beabsichtigt war.[260]

2.4.4. Missbrauch der Signaturfunktion

Mindestens zwei Arten der Umsetzung des Missbrauchs der Signaturfunktion sind denkbar. Erstens könnte ein Angreifer ohne aktives Zutun des Ausweisinhabers versuchen, den Ausweis im Signatur-Kontext zu verwenden. Zweitens ist es möglich, den Ausweisinhaber bei der legitimen Nutzung der Signaturfunktion zu täuschen, um nicht gewollte Operationen umzusetzen.

In „Privilege Escalation“, S. 52 wurde gezeigt, wie ein fremder nPA mittels eines Relaying an einem entfernten Komfortleser verwendet werden kann. Damit kann ein vom Angreifer für die QES initialisierter nPA, auch zum Signieren beliebiger Daten verwendet werden. Die digitale Unterschrift ist für andere überprüfbar und damit rechtsgültig, wenn der ZDA das qualifizierte Zertifikat öffentlich gemacht hat (SP.eSign). Allein dieser letzte Schritt der Aktivierung würde vor dem Missbrauch der Signaturanwendung schützen. Welche Maßnahmen

[259] BSI TR-03128, S. 33–34.

[260] Zugespitzt könnte man also auch einen „Signaturdienst“ anbieten, bei dem es nicht mehr notwendig ist, dass der Ausweisinhaber zum Signieren über einen Komfortleser verfügt. Er würde dabei allerdings die Kontrolle über den nPA und die Signatur-PIN abgeben, was durch SigG, §2 verboten sein dürfte. Einige Autoren halten in bestimmten Fällen die „eigentlich fehlende alleinige Kontrolle“ dennoch für zulässig (Braun u. a. 2011, S. 8).

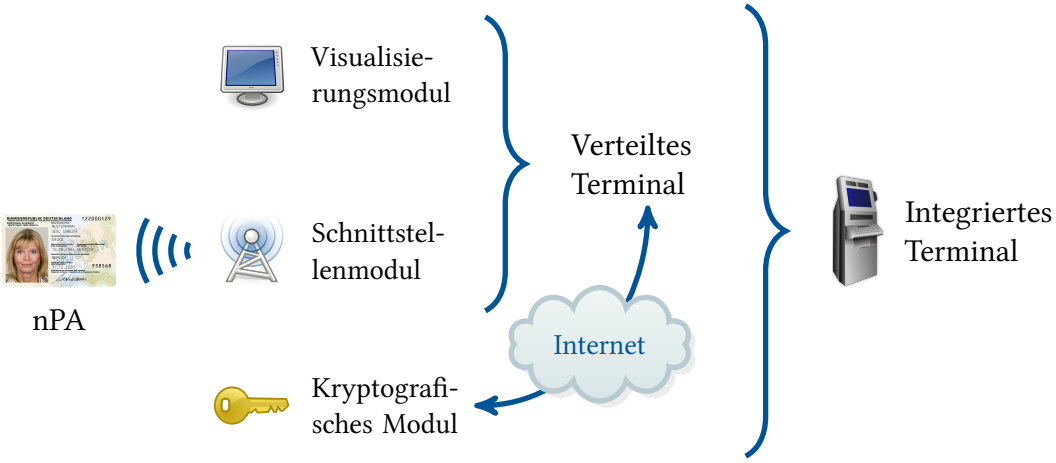


Abbildung 2.23.: Verteiltes und integriertes Terminal nach BSI TR-03128, S. 32–34

bei der Aktivierung praktisch umgesetzt werden, kann zum Zeitpunkt der Erstellung dieser Arbeit nicht geprüft werden, da bisher kein ZDA die QES mit dem nPA anbietet.

Angenommen der Ausweisinhaber will aber berechtigterweise die Signaturfunktion seines nPAs nutzen. Wie kann er dann sicher sein, dass die Aktionen, die er in Auftrag gibt, wirklich an *einen* Komfortleser gelangt? Beim Skimming eines Komfortlesers wird dem Angreifer voller Zugriff auf den nPA gewährt. Auch die Signatur-PIN wird am gefälschten Leser eingegeben. Der Angreifer kann mittels des oben dargestellten Relay-Angriffs die Operationen des Ausweisinhabers umsetzen, ohne dass dieser durch einen beschränkten Funktionsumfang den gefälschten Leser erkennen kann. Insbesondere wird jemand, der mit dem nPA signieren will, auch gegenüber dem ZDA die Aktivierung bestätigen, so dass das qualifizierte Zertifikat im Verzeichnisdienst veröffentlicht wird. Der Angreifer kann aber freilich beliebig weitere Dokumente mit dem fremden nPA signieren.

Das qualifiziert signierte digitale Dokument dient den beteiligten Parteien als Anscheinsbeweis einer willentlichen Erklärung.[261] „Der Anschein der Echtheit einer in elektronischer Form vorliegenden Erklärung, der sich auf Grund der Prüfung nach dem Signaturgesetz ergibt, kann nur durch Tatsachen erschüttert werden, die ernstliche Zweifel daran begründen, dass die Erklärung vom Signaturschlüssel-Inhaber abgegeben worden ist.“[262] Der ZDA muss die gesetzlich vorgegebenen Sicherheitsmaßnahmen nachweisen können,[263] insbesondere ist die Dokumentationspflicht zu SP.eSign zu beachten. Ist der ZDA dazu in der Lage, muss der Ausweisinhaber als Unterzeichner des Dokuments Argumente hervorbringen, die den Anscheinsbeweis erschüttern. [253]

[261] „Kartenleser für den neuen deutschen Personalausweis“, S. 765.
[262] ZPO, §371a Abs. 1.
[263] SigV, §5.

Manipulation zu signierender Daten

Angenommen der Ausweisinhaber will berechtigterweise die Signaturfunktion seines nPAs nutzen und er nutzt dabei einen echten Komfortleser. Wie kann der Ausweisinhaber sicher sein, dass die Signaturanwendungskomponente direkt mit seinem Komfortleser kommuniziert?[264] Die AusweisApp besitzt keinen sicheren Kanal zum Leser selbst und kann ihn nicht eindeutig identifizieren.[265] Damit ist es für Leser oder nPA unklar, woher die zu signierenden Daten eigentlich stammen. Wenn der Ausweisinhaber ein Dokument mit dem nPA signieren will, nutzt er den sicheren Dokumentenbetrachter der Client-Software, um das digitale Dokument zu überprüfen. Dann bildet sie einen digitalen Fingerabdruck dieses Dokuments (Hash), welcher im Anschluss zum Signieren an den Komfortleser gegeben wird. Der Leser hätte maximal die Möglichkeit den Hash als Zeichenkette anzuzeigen. Dass der Ausweisinhaber diese kryptischen Daten ignoriert und nicht mit den angezeigten Informationen der AusweisApp abgleicht (vorausgesetzt diese sind authentisch), ist abzusehen.

Ein entsprechender Man-in-the-Middle-Angriff ließe sich erstens durch manipulierte Software auf dem Computer des Ausweisinhabers realisieren (Malware). Die AusweisApp nutzt den System-eigenen PC/SC-Dienst und dessen Bibliotheken, die beim „vertrauenswürdigen Bootvorgang“ der Client-Software nicht geprüft werden kann. Schadsoftware könnte sich an dieser Stelle einnisten.[266]

Zweitens kann gleichermaßen die ungesicherte Kommunikation zum Komfortleser auf dem USB manipuliert werden (Man-in-the-Middle Hardware). Neue Kommandos können eingespeist oder legitime manipuliert werden, etwa jene zum Signieren eines (beliebigen) Hashes nach einer erfolgreicher Signatur-PIN-Eingabe. Auf dem PC selbst könnte der Angriff dann praktisch nicht mehr entdeckt werden.

Besonders elegant wäre dieser Angriff, wenn der Ausweisinhaber durch eine Prüfung der Signatur nicht fest stellen könnte, dass ein falsches Dokument signiert wurde. Die Schadsoftware müsste also sowohl das ursprüngliche als auch das ersetzte Dokument signieren lassen. In den Richtlinien[267] sind keinerlei Hinweise darauf zu finden, ob nur genau ein Dokument je Signatur-PIN-Eingabe signiert werden kann. Wäre eine solche Mehrfachsignatur mit dem nPA möglich, könnte die Schadsoftware beide Dokumente mit zwei aufeinander folgenden `PSO: COMPUTE DIGITAL SIGNATURE` Kommandos realisieren und lediglich die Signatur des ursprünglichen Dokuments an die Client-Software weitergeben. Dass noch ein weiteres Dokument signiert wurde, würde dem Ausweisinhaber gar nicht bekannt werden.

[264] Morgner und Oepen 2010.

[265] Siehe „Skimmer“, S. 37.

[266] `LD_LIBRARY_PATH` (Linux) bzw. `PATH` (Windows) enthalten die Verzeichnisse, in denen das Betriebssystem nach den Bibliotheken sucht. Die Reihenfolge der Verzeichnispfade entscheidet wo *zuerst* gesucht wird. Ein Angreifer könnte mit einfachen Benutzerrechten das Verzeichnis der Schadsoftware mit seiner manipulierten PC/SC-Bibliothek an den Anfang dieser Variablen schreiben. Die AusweisApp kommuniziert nun statt mit dem echten Komfortleser lediglich mit der Schadsoftware. Letztere leitet alle Kommandos an den Komfortleser weiter. Auch der Aufforderung zur Eingabe der Signatur-PIN wird der Ausweisinhaber nachkommen, da er ja schließlich sein Dokument signieren möchte. Wenn aber das Kommando zum Signieren des Hashes (`PSO: COMPUTE DIGITAL SIGNATURE`) die Schadsoftware passiert, werden die enthaltenen Daten durch den Hash eines anderen Dokumentes ersetzt. Als Antwort erhält die Schadsoftware die Signatur des ersetzten Dokumentes vom nPA.

[267] BSI-CC-PP-0061; BSI TR-03117; BSI TR-03119 1.2; BSI TR-03127.

Alternativ könnte der Angreifer es für den Ausweisinhaber aussehen lassen als sei ein Fehler eingetreten, obwohl EAC, Eingabe der Signatur-PIN und Signatur des (falschen) Dokuments erfolgreich waren. Der Ausweisinhaber wird sich nichts dabei denken und den Signiervorgang mit dem eigentlich zu signierenden Dokument wiederholen. Der Angreifer hätte aber in der zuvor aktiven Sitzung bereits die Signatur seines Dokuments erreicht.

Nutzung ungültiger Terminalzertifikate

Ein gesperrter nPA kann über das Sperrmerkmal erkannt werden (sofern dieser das korrekte Sperrmerkmal angibt).[268] Ein kompromittiertes Terminal kann vom nPA jedoch grundsätzlich nicht erkannt werden. Terminalzertifikate können nicht revoziert werden, sie besitzen eine kurze Laufzeit, um den Zeitraum des Missbrauchs zu begrenzen.[269] Diese Begrenzung trifft jedoch auf die drei Jahre gültigen Signaturterminalzertifikate der Komfortleser nicht zu.

Der nPA selbst besitzt keine eigenständige Energiequelle und daher keine aktuelle Uhr. Der Zustand der inneren Uhr des nPA wird entsprechend der Gültigkeit des jüngsten hoheitlichen Terminalzertifikats approximiert, das erfolgreich importiert werden konnte.[270] Die eigentliche „effective authorization" der TA durch den Nachweis des privaten Schlüssels muss das Terminal nicht erfolgreich bestehen, um die Zeit zu aktualisieren.[271] Für den erfolgreichen Import des Zertifikats genügt der Nachweis über die EAC-PKI. Ein selten genutzter Ausweis besitzt also ein entsprechend altes Referenzdatum. Kompromittierte Terminals könnten dann den Ausweis auslesen, Funktionen auslösen oder sogar Signaturen erstellen auch wenn bereits bekannt ist, dass deren Schlüssel unsicher geworden sind.

Schwerwiegend ist die Kombination mit einem Skimming-Angriff.[272] Sobald ein Angreifer den TA-Schlüssel eines Komfortlesers extrahiert hat, kann er den kompromittierten Schlüssel in beliebig vielen gefälschten Lesern verbauen. Da der Schlüssel mehrere Jahre gültig ist, eine Revozierung unmöglich ist und der AusweisApp keinerlei gesicherte Details zu einem Komfortleser vorliegen,[273] kann keine der involvierten Parteien feststellen, ob es sich um ein echtes Gerät handelt.

Zumindest für die approximierte Zeit des Ausweises ist ein „Zeitaktualisierungsdienst" angedacht.[274] Eine Umsetzung davon oder eine entsprechende Ankündigung existiert jedoch nicht. Mit einem vertrauenswürdigen Gerät könnte man diese problemlos mit einem nicht authentisierten Terminal durchführen, da hier lediglich die Zertifikatskette eines Dienstanbieters, aber nicht dessen geheimer Schlüssel benötigt wird.

[268] Siehe „Tracing", S. 46.
[269] Siehe „Terminalauthentisierung und Extended Access Control Public Key Infrastructure", S. 13.
[270] BSI TR-03127, S. 28–29; BSI TR-03110 2.05, S. 16–17; BSI-CC-PP-0061, S. 79–80,99.
[271] BSI TR-03110 2.05, S. 17–19.
[272] Siehe „Skimmer", S. 37.
[273] Siehe „Skimmer", S. 37.
[274] BSI TR-03127, S. 30.

2.5. Mitigation

Es wurden die verschiedene Prozesse und Parteien des „Systems Personalausweis“[275] beleuchtet. Neu geschaffene und etablierte Protokolle wurden genutzt, um die besonderen Anforderungen an den nPA umzusetzen. Bisherige Sicherheitsanalysen haben bisher nur je einen Teil der beteiligten Komponenten, Protokolle und Funktionen betrachtet.[276]

Auch die hier erstellte Übersicht ausgewählter Aspekte erhebt keinen Anspruch an Vollständigkeit. Es konnte dennoch gezeigt werden, dass es eine Reihe von Problemen bei der elektronischen Nutzung des nPA gibt, die bisher wenig Beachtung fanden oder gar nicht bekannt waren. Einige sind dabei in Design oder Implementierungsentscheidungen verwurzelt. Andere nutzen die Unzulänglichkeiten bei der Nutzerinteraktion und -rezeption aus.

Grundsätzlich kann beim nPA nicht zwingend sichergestellt werden, dass er sich physisch an dem Ort befindet wo er auch benutzt wird. Dies ermöglicht Relay-Angriffe in beliebigem Kontext (z. B. eID oder eSign). Die Authentisierungsgeheimnisse kann ein Angreifer durch Phishing, Exploits von Software auf dem Nutzer-PC, Malware oder per Skimming erhalten. Hierbei werden zwei der schwächsten Parteien angegriffen: Der Ausweisinhaber bzw. dessen Computer. Auch beim nPA und seiner Infrastruktur gilt:

> A security system is only as strong as its weakest link.[277]

In dem folgenden Abschnitt werden mögliche Maßnahmen zum Schutz des nPA betrachtet. Eine Reihe von Vorschlägen enthält wesentliche Änderungsvorschläge und betrifft die Gestaltung zukünftiger Komponenten im System nPA. Vorschläge zur Veränderung der Firmware von PACE-fähigen Lesegeräten oder bestimmter Verfahrensweisen der Dienstanbieter lassen sich aber ohne weiteres bereits heute umsetzen.

2.5.1. Distance Bounding

Das grundlegende Problem bei den hier dargestellten Relay-Angriffen liegt darin, dass das Geheimnis und der *Zugriff* auf den nPA genügt, um damit die zwei-Faktor-Authentisierung durchzuführen. Die gern gemachte Behauptung, dass man eine Chipkarte physisch besitzen oder unter Kontrolle haben muss, um sie verwenden zu können,[278] gilt nicht. Ein Zugriff, der die Übertragung beliebiger Daten ermöglicht, genügt. Bei kontaktlosen Chipkarten kann die Kommunikation durch ein Lesegerät mit besonders hoher Reichweite ermöglicht werden.

Das Relaying von kontaktlosen Chipkarten wird in der Literatur seit wenigen Jahren betrachtet.[279] Liegen einem Angreifer zudem noch Authentisierungsgeheimnisse vor, spricht man von „Grandmaster Chess Attack“.[280] Ein Schutz vor diesem Angriff ist schwierig, da

[275] „Kartenleser für den neuen deutschen Personalausweis“.

[276] Bender, Fischlin und Kügler 2009; „Kartenleser für den neuen deutschen Personalausweis“; Dagdelen und Fischlin 2010.

[277] Ferguson und Schneier 2003, S. 9.

[278] Z. B. von Rogall-Grothe in Rogall-Grothe 2011 oder in Welchering 2010: „Selbst wenn ein PC infiziert ist und die PIN mit einem Keylogger ausgelesen werden könnte, hilft das dem Betrüger nur wenig. Er muss ja auch den Ausweis haben.“

[279] Hancke 2005.

[280] Kc und Karger 2005; Hancke, Mayes und Markantonakis 2009, S. 616–617.

keine kryptographischen Methoden als solche gebrochen wurden. Tatsächlich behaupten einige Autoren, dass ein solcher Angriff kein Problem darstelle, sondern im Gegenteil ein viel genutztes Feature sei.[281] Sogenanntes kryptographisches Tunneln („cryptographic tunneling") erlaube es einem Nutzer seine Chipkarte auf sichere Weise an einem entfernten Server zu nutzen.

Andere Autoren schlagen „Distance Bounding" als Schutzmaßnahme gegen Relay-Angriffe vor.[282] Auf unterster Kommunikationsebene könnte beispielsweise ein Challenge-Response-Protokoll durchgeführt werden, bei dem notwendige Berechnungen im Vornherein statt finden. Dies erlaubt sehr schnelle Antwortzeiten auf eine Challenge. Das Signals kann höchstens mit der Lichtgeschwindigkeit übertragen werden. Daher kann der Chipkartenleser anhand der Antwortzeiten ableiten, in welcher Entfernung sich die Karte höchstens befindet. Für ein Distance Bounding sind die zeitlichen Anforderungen an Chipkarte und Leser allerdings sehr strikt, weshalb es derzeit allenfalls prototypische Umsetzungen gibt.[283]

Dementsprechend wurde beim nPA kein Distance-Bounding-Protokoll implementiert. Dies hat zur Folge, dass man Relay-Angriffe ausschließlich durch besonderen Schutz des Zugriffs und der Geheimnisse zum nPA verhindern kann. Die Nutzung eines Lesegeräts, welches den Zugriff auf die Chipkarte durch die zwingende Nutzung eines PIN-Pads schützt, ist unabdingbar.

2.5.2. Signatur-Nutzung mit Medienbruch

Nachdem in Morgner und Oepen 2010 auf das Problem der Privilege Escalation hingewiesen wurde, wurden erste Ideen zur medienbruchfreien Aktivierung des nPA als SSEE[284] zunächst fallen gelassen. Der technische Prozess zur Aktivierung soll nun noch um den Schritt einer schriftlichen Bestätigung ergänzt werden (im Rahmen von SP.eSign).[285] Erst nachdem der Ausweisinhaber dem ZDA seinen alleinigen Besitz des nPA über den Postweg bestätigt hat, soll das Zertifikat im Verzeichnisdienst des ZDA veröffentlicht werden. Nun kann man einwenden, dass auch dieser Bestätigungsbrief abgefangen werden kann. Dennoch ist hier der Aufwand für einen Missbrauch wesentlich höher im Vergleich zur rein elektronischen Aktivierung.

Damit wird beim nPA unabhängig vom Distance Bounding die zweite bekannte Möglichkeit zum Vorbeugen gegen Relay-Angriffe (auf die Signaturfunktion) verwendet, eine zusätzliche Verifikationsprozedur.[286] Kontaktlose Chipkarten werden gern verwendet, weil sie komfortabel und schnell zu benutzen sind. Eine zusätzliche Verifikation schränkt diese Vorteile ein.[287] So verzögert die Bestätigung des Ausweisinhabers an einen ZDA die Nutzung der Signaturfunktion um einen oder mehrere Tage. Es ist offen, welche anderen Methoden beim nPA verwendet werden können. Genügt die Bestätigung per SMS oder ist dies mit Blick auf die Angriffe gegen das mTAN-Verfahren zu unsicher? Kann die Verifikation über ein E-Mail-Postfach genutzt

[281] Kc und Karger 2005.
[282] Brands und Chaum 1994; Clulow u. a. 2006; Drimer und Murdoch 2007.
[283] Rasmussen und Čapkun 2010.
[284] Morgner und Oepen 2010; Schwarz 2011a.
[285] Schwarz 2011b.
[286] Hancke, Mayes und Markantonakis 2009, S. 622.
[287] Hancke, Mayes und Markantonakis 2009, S. 625.

werden, wenn ein Trojaner nicht nur den Zugriff auf den Ausweis weiterleiten sondern auch die Passwörter des E-Mail-Accounts ausspähen kann?

Beispielsweise wurde für den nPA eine Ad-Hoc-Signatur zum einmaligen und *sofortigen* Gebrauch ins Gespräch gebracht.[288] Dabei soll ein qualifiziertes Zertifikat mit sehr kurzem Gewährleistungszeitraum und festem Anwendungszweck auf den Ausweis geladen werden. Sofern dabei zur Bestätigung der Aktivierung als SSEE kein weiterer sicherer Kanal als die Authentisierung mittels eID zur Verfügung steht, existiert auch hier unvermindert das Problem der Privilege Escalation. Daher müssten in der Praxis sicherlich weitere flankierende Sicherheitsmaßnahmen getroffen werden. Denkbar ist die schriftliche Bestätigung der ersten Ad-Hoc-Signatur, die ein Aktivierungsgeheimnis enthält. Weitere Ad-Hoc-Signaturen könnten dieses Aktivierungsgeheimnis erfordern, um die initiale Verifikationsprozedur des Ausweisinhabers einzubinden.

2.5.3. Terminalauthentisierung zur Erkennung von Manipulationen

Zumindest ein Komfortleser besitzt ein starkes Authentisierungstoken, mit dem er seine Echtheit nachweist: Seinen privaten TA-Schlüssel. Die Prüfung dieses Schlüssels ist aber ausschließlich gegenüber dem nPA als Teil der EAC vorgesehen. Gegenüber der Chipkartenapplikation kann die Echtheit des Komfortlesers per TA nicht nachgewiesen werden. Aber man könnte sich mit einem nPA-Emulator behelfen.[289] Die virtuelle Smartcard und der NFC-Emulator aus Morgner und Oepen 2011 könnte genutzt werden, um dem Komfortleser gegenüber als nPA zu erscheinen. Per PACE beweisen sich Emulator und Komfortleser gegenseitig eine beliebige, aber auf beiden Seiten gleiche eID-PIN. In der folgenden TA weist der Komfortleser seine Echtheit nach. Zwar sollte der Emulator (im Regelfall) nicht im Besitz eines gültigen CA-Schlüssels sein, so dass die EAC insgesamt fehl schlägt. Aber das Ziel der Prozedur, die Überprüfung des Komfortlesers, konnte bereits in der TA erfüllt werden.

Revozierung für CV-Zertifikate

Wie könnte man nun auf Unregelmäßigkeiten oder Nichtbestehen der TA reagieren? Im einfachsten Fall sollte man den fragwürdigen Komfortleser nicht benutzen und das Problem dem Hersteller melden. Es ist aber kein Revozierungsmechanismus für TA-Zertifikate implementiert, so dass die drei Jahre gültigen Zertifikate der Komfortleser vom nPA stets als valide akzeptiert würden,[290] – egal ob das Terminalzertifikat bzw. der zugehörige private Schlüssel nun schon als kompromittiert erkannt wurde oder nicht.

Eine softwareseitige TA-Prüfung, könnte aber auch eine zusätzliche „Revocation List“ mit ungültigen Zertifikaten einführen. Dies könnte z. B. mit einem nPA-Emulator oder der Hersteller-Software zum Firmware-Update umgesetzt werden. Auf diese Weise können kompromittierte Geräte besser erkannt und deren Verwendung verhindert werden. Um Fälschungen eines Komfortlesers zu erkennen, müssten allerdings Chipkartenleser-individuelle Terminalzertifikate bzw. -schlüsseln – statt wie derzeit Bauart-spezifische Terminalzertifikate – eingeführt werden.

[288] Borchers 2011b; Schwarz 2011b.

[289] Vorgeschlagen wurde dies erstmals in Oepen und Morgner 2011. Siehe auch „nPA-Kartenemulator“, S. 111.

[290] Siehe „Terminalauthentisierung und Extended Access Control Public Key Infrastructure“, S. 12.

Sicherer Kanal zwischen Client-Software und Kartenleser

Auf Basis des Terminalschlüssels könnten Client-Software und Komfortleser außerdem einen sicheren Kanal etablieren, die Modifikation auf der USB-Ebene zu verhindern.[291] So könnten Leser und AusweisApp automatisch PACE mit einer Default-PIN durchführen. Der Leser könnte der AusweisApp im Anschluss bei der TA seine Echtheit nachweisen. Ein Man-in-the-Middle-Angriff würde durch die Bindung der TA an die PACE-Schlüssel von der AusweisApp erkannt werden.[292] Die AusweisApp könnte nun Operationen am Komfortleser in dem sicheren Kanal auslösen. Vertraulich und integritätsgeschützt könnten die PACE-Funktion des Lesers gestartet oder zu signierende Daten an den Leser übertragen werden.

2.5.4. Selektive Deaktivierung des nPA

Abschirmung von NFC-Signalen

Es ist möglich, den Ausweis gegen NFC-Signale abzuschirmen, so dass auch unterwegs nicht unbemerkt eine Verbindung zum Ausweis aufgebaut werden kann.[293] Denn immerhin kann man durch die elektronisch frei lesbaren Informationen des nPA zumindest die deutsche Staatsbürgerschaft ableiten. Während die Abschirmung des Ausweises ungewünschte spontane Transaktionen verhindert, ist die Manipulation von gewünschten Transaktionen weiterhin möglich.

Deaktivierung der eID-Funktion

Der Missbrauch des elektronischen Identitätsnachweis und damit die Privilege Escalation ist nicht mehr möglich, wenn die eID-Funktion auf dem nPA komplett deaktiviert ist. Dies hat offensichtlich auch den Nachteil, dass ein legitimer Identitätsnachweis erst ein aktivieren der eID-Funktion benötigt, was lediglich mit den Terminals der Ausweisbehörden möglich ist. Es bleibt eine zu untersuchende Frage, ob eine sichere Aktivierung auch im heimischen Umfeld möglich ist.

Deaktivierung der Signaturfunktion

Derzeit ist die Deaktivierung der Signaturfunktion nicht möglich. Man kann lediglich die Initialisierung als SSEE deaktivieren, indem man die eID-Anwendung abschaltet.[294] Um aber dennoch den weniger kritischen eID-Kontext aktiv nutzen zu können, aber den Missbrauch als Signaturkarte zu verhindern, sollte eine selektive Deaktivierung der Signaturfunktion geschaffen werden.

---

[291] Morgner und Oepen 2010.

[292] Siehe „Terminalauthentisierung und Extended Access Control Public Key Infrastructure“, S. 12.

[293] Die meisten Handel erhältlichen „Behältnisse“ zum Abschirmen vor NFC-Signalen sind nur bedingt einsetzbar (Paget 2012). Empfehlenswert ist die Nutzung von Störsendern.

[294] Siehe „Deaktivierung der eID-Funktion“, S. 61.

Hardware-Schalter zum Nutzungskontext

Einige SD Memory Cards besitzen einen Hardware-Schalter, mit dem Schreibvorgänge auf der Karte verhindert werden können. Auf ähnliche Weise könnte der Ausweisinhaber dem nPA vorgeben, in welchem Nutzungskontext er Transaktionen vorgeben möchte. Möglich wären die Einstellungen

- Deaktiviert
- eID
- eSign

Der Missbrauch des nPA in einem anderen als dem vorgesehenen Kontext ist nicht mehr möglich. Die Bedienung durch den Ausweisinhaber ist direkt und intiutiv. Nachteilig könnten sich jedoch Verschleißerscheinungen auswirken.

2.5.5. Schutz von CAN und MRZ

Eine simple Maßnahme gegen ungewolltes Ablesen der CAN oder MRZ, ist deren Abkleben. Um den Brute-Force-Angriffe zu verhindern, sollte der Ausweis nicht länger als nötig auf dem Kartenleser liegen. Der nPA selbst könnte aber einen Brute-Force-Angriff auch durch die Filterung einer Vielzahl von Fehlversuchen verhindern. Möglich ist z. B. eine zusätzliche Zeitverzögerung.

PACE-fähige Chipkartenleser filtern bereits die Zugriffe auf den nPA. Eine per PC/SC vorgegebene CAN sollte jedoch nur akzeptiert werden, wenn dies ausdrücklich bestätigt wird. Eine simple Rückfrage auf dem Display des Kartenlesers, z. B. „Bestätigen Sie die Nutzung der vorgegebenen CAN mit OK“, könnte nicht nur einen Brute-Force-Angriff verhindern, sondern auch andere unbemerkte Transaktionen verhindern.

Um die CAN vor unberechtigtem Auslesen durch Malware zu schützen, wäre es weiterhin sinnvoll, sie in dem PACE-fähigen Kartenleser zu speichern statt in der Client-Software.

2.5.6. Sicheres Chipkartendisplay

Gerade weil sich z. B. der Familienname oder die Anschrift des Ausweisinhabers ändern können, sind Chipkarten mit einem integrierten Display für Ausweisdokumente interessant. Das sichere Display kann auch für andere Informationen genutzt werden.

Anzeige von Verbindungsdetails

Buchstäblich alle Verbindungsdetails eines elektronischen Identitätsnachweises oder einer Signatur könnten theoretisch dargestellt werden. Das Display der Chipkarte könnte effektiv die sichere Anzeige des Kartenlesers ersetzen. Praktisch wäre die Menge an Informationen sicherlich zu groß. Zumindest den Nutzungskontext (eID oder eSign) könnte man jedoch darstellen. Auch welche Ausweisdaten *nicht* übermittelt werden sollen, ließen sich leicht darstellen, wenn sie mittels des Displays auf dem Ausweis „geschwärzt“ würden.

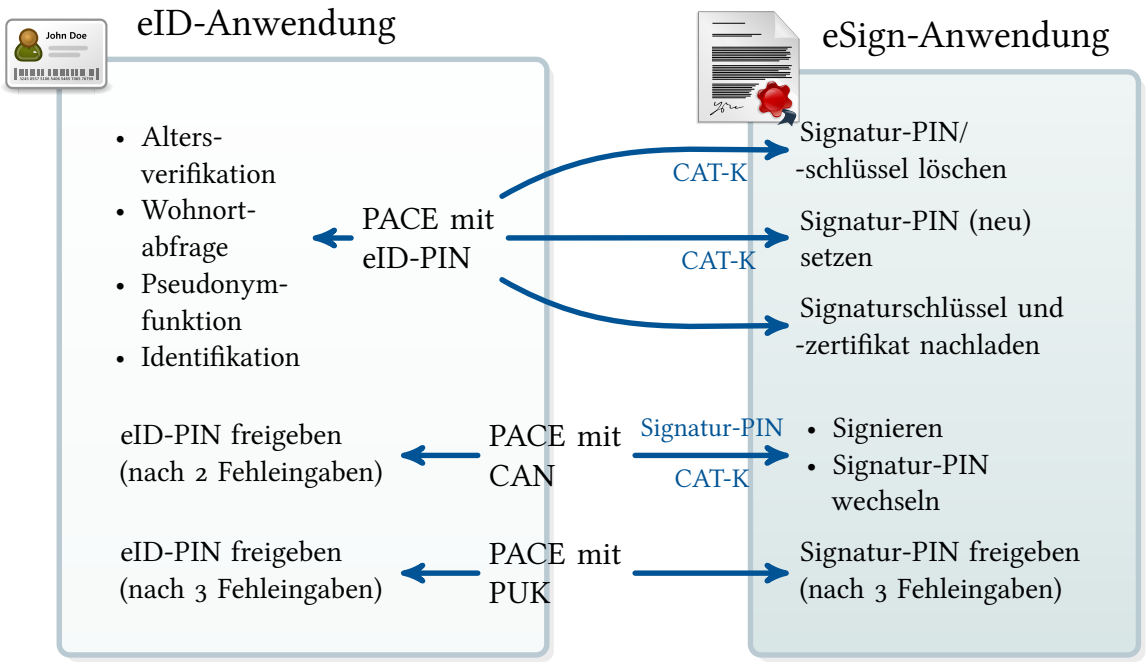


Abbildung 2.24.: Privilege Escalation. Die eID-Anwendung und ihre Geheimnisse aktivieren den nPA vollständig als SSEE.

Dynamisches Geheimnis

In Verbindung mit einem Kartendisplay kann ein dynamisches Geheimnis (CAN, eID-PIN oder Signatur-PIN) den Authentisierungsfaktor „Besitz“ stärken, da man die Informationen des Kartendisplays visuell erfassen muss. In Bastian 2011 wurde ein nPA-Emulator vorgestellt, der auf einer Java Card läuft und das Kartendisplay nutzt, um das Geheimnis mit den dargestellten Ziffern zu verknüpfen. Es entsteht dadurch ein sitzungsabhängiges Einmal-Passwort für den emulierten nPA.

2.5.7. Entflechtung von eID- und eSign-Anwendung

Die Vermischung von eID- und Signaturanwendung liegt zum einen darin, dass beide auf derselben Chipkarte vereint sind und man dieser nicht ansehen kann, in welchem Kontext sie verwendet wird. Zum anderen wird die Nutzung als SSEE mithilfe der eID-Anwendung aktiviert (und terminiert), so dass eine Privilege Escalation möglich ist (siehe Abbildung 2.24).

Eine einfache Maßnahme zur Vermeidung der Privilege Escalation wäre z. B. die Nutzung einer Aktivierungs-Signatur-PIN, die dem PIN-Brief beiliegt. Der Angreifer könnte dann nicht mehr ohne dieses Aktivierungsgeheimnis bzw. den PIN-Brief die Signatur-PIN setzen. Gleichzeitig müsste auch das Terminieren der Signaturanwendung über die eID-PIN verboten werden, damit ein Angreifer nicht unberechtigterweise die Signatur-PIN oder -schlüssel löschen kann. Stattdessen könnte man die Signatur-PIN selbst oder ein spezielles Geheimnis im PIN-Brief verwenden, um diesen Vorgang zu autorisieren.

2.5.8. „Man-in-the-Middle Defence“

Der Begriff „Man-in-the-Middle“ wird meist negativ, bezogen auf einen Angreifer, gebraucht. Tatsächlich kann die Kommunikation zwischen zwei Gesprächspartner aber auch beobachten werden, um diese zu überprüfen.[295] Insbesondere die Nutzung eines vertrauenswürdigen portablen und handlichen Geräten, könnte unterwegs an öffentlichen Terminals zum Einsatz kommen.[296]

PACE-fähiger Leser als vertrauenswürdiger Mittler

Für einen Standard- oder Komfortleser sind bereits einige Filtermaßnahmen vorgeschrieben. Diese könnten ausgebaut werden, um auch die potenziell unwirksamen Prüfungen der Client-Software zu übernehmen (API.CurrentDate, API.SecureDisplay, API.FinalAccessRights). Ein Kartenleser könnte eine noch stärkere inhaltliche Filterung und eine bessere Anzeige der Transaktionsdaten realisieren. Sensitive Nutzereingaben und -ausgaben würden ausschließlich am sicheren Kartenleser geschehen; besonderes Vertrauen in die Sicherheit des angeschlossenen Host-Computers ist nicht mehr notwendig.

Die komplexere inhaltliche Filterung kann sich nachteilig auf einen Zertifizierungsprozess auswirken oder schlicht die Kartenleser-Firmware überfordern. Außerdem fehlen den üblichen Kartenlesern genügend große Displays sowie eine unabhängige Zeitquelle.

NFC-Emulator als vertrauenswürdiger Mittler

Beim nPA legt sich das Terminal bereits frühzeitig darauf fest in welchem Kontext die Karte gebraucht wird und für welche Operationen. Insbesondere ist dies nicht durch einen sicheren Kanal geschützt, so dass diese Informationen von einem Dritten ohne weiteres gelesen werden können. Ähnlich der Filterung durch einen PACE-fähigen Leser könnte auch auf der Ebene der NFC-Kommunikation ein vertrauenswürdiger Mittler die Daten an den nPA weiterleiten, wenn sie dem Nutzer zur Anzeige gebracht wurden und dieser die Transaktion bestätigt (siehe Abbildung 2.25).

Die PIN-Eingabe wäre mit dem Emulator nur teilweise geschützt. Die AusweisApp z. B. verlangt die eID-PIN bereits bevor sie den CHAT mit den angeforderten Berechtigungen an den nPA sendet – PACE-fähige Leser müssen hingegen die PIN-Eingabe zum spätestmöglichen Zeitpunkt umsetzen. Die Kompromittierung des Geheimnisses kann mit dem NFC-Mittler nicht verhindert werden, lediglich böswillige Transaktionen können blockiert werden. Praktisch ist es sicherlich auch nicht einfach, gleich zwei unabhängige NFC-Schnittstellen in einem Gerät unterzubringen, um sowohl mit dem nPA als auch dem Terminal zu kommunizieren.

[295] Anderson und Bond 2009.
[296] Balfanz und Felten 1999.

Abbildung 2.25.: „Man-in-the-Middle Defence“ per NFC. Der vertrauenswürdige Mittler zeigt dem Ausweisinhaber den Nutzungskontext und die angeforderten Berechtigungen des CHAT an.

3. Mobiler Chipkartenleser für den neuen Personalausweis

Bereits 2007 besaßen 80% der Deutschen ein Mobiltelefon.[1] Moderne Mobiltelefone werden wegen PC-ähnlicher Funktionalität und Konnektivität auch Smartphones genannt. Typischerweise ermöglicht die Bauweise und Leistung eines Smartphones weit mehr als das bloße Telefonieren. So hat beispielsweise die mobile Nutzung des Internets stark zugenommen.[2] Offene Schnittstellen ermöglichen Benutzern oder Drittanbietern die Erstellung Smartphone-spezifischer Programme („Apps"). Hier setzt auch eine Hardware-bezogene Definition von „Smartphone" an.[3] Einfache Feature-Phones besitzen lediglich einen „Baseband Processor" (BP), der auch das GSM-Protokoll implementiert. Smartphones besitzen zusätzlich einen universellen und leistungsstärkeren „Application Processor" (AP). Auf dem AP werden Apps verarbeitet, ohne den BP oder die GSM-Operationen zu behindern.

Konnektivität kann nach außen über verschiedene Schnittstellen realisiert werden. Die Datenübertragung per GSM oder UMTS ist dabei wenig überraschend. Oft sind auch WLAN, Bluetooth, Infrarot oder USB bei den Geräten verfügbar. Wenige Smartphones besitzen aber auch eine NFC-Schnittstelle. Da besonders im Micropayment- und Ticketing-Bereich auf diese Technologie gesetzt wird, wird erwartet, dass NFC zukünftig mehr Verbreitung findet. Mit vorhandener NFC-Schnittstelle sind auch verschiedene Nutzungsszenarios des nPA möglich.[4]

Im einfachsten Fall lässt sich ein „vollmobiles Nutzungsszenario"[5] umsetzen. Im Wesentlichen kann die Client-Software auf einem PC ersetzt werden durch eine „mobile eCard-API" und damit einen Ersatz für die Kombination aus AusweisApp und Basisleser bieten. Offensichtlich wäre dies mit denselben Problemen verbunden, die oben[6] diskutiert wurden.

Müller, Morgner und Oepen schlagen zwei weitere Verwendungsmöglichkeiten vor.[7] Zunächst könnte das Smartphone als Chipkartenleser für den nPA dienen. Der „mobile Leser" könnte an einen PC angeschlossen werden und der hier laufenden AusweisApp z. B. eine sichere PIN-Eingabe ermöglichen. Ähnlich dem Standard- oder Komfortleser würde auf dem Smartphone die eID-PIN eingegeben und die Chipkartenkommunikation bereitgestellt. Eine Online-Verbindung wäre auf dem Smartphone nicht notwendig, da die Kommunikation zum Dienstanbieter durch die AusweisApp auf dem externen PC erfolgt.

Die zweite Möglichkeit der Verwendung des nPA am Smartphone ermöglicht die Realisie-

[1] Statistisches Bundesamt 2007.
[2] Statistisches Bundesamt 2011.
[3] Welte 2008, 2010.
[4] Müller, Morgner und Oepen 2011.
[5] Müller, Morgner und Oepen 2011, S. 180.
[6] Siehe „Bedrohungsszenarien", S. 40.
[7] Müller, Morgner und Oepen 2011, S. 180–181.

rung besonderer „Servicefeatures“.[8] Ein autarkes PIN-Management ermöglicht das sichere Setzen der eID-PIN und unterbindet entsprechendes Tracing. Mobile Zeitaktualisierung des nPA verhindert die Nutzung ungültiger TA-Zertifikate. Wird das Mobiltelefon als nPA-Emulator genutzt, kann die einfache Darstellung von CHAT und Berechtigungszertifikaten die Erkennung von manipulierten Terminals ermöglichen.

Die Kombination der Funktion eines Chipkartenlesers und der Möglichkeit zu umfangreichen inhaltlichen Prüfungen von eID-Anfragen als „mobiler Chipkartenleser für den neuen Personalausweis“ stellt eine vielversprechende Alternative zu Standard- oder Komfortlesern dar.[9] Sie ist nicht nur besonders preiswert.[10] Mit seiner autarken Energieversorgung, Internetverbindung und Zeitquelle könnte auch der Schutz einiger Probleme gewährleistet werden, die bisher wenig beachtet wurden und selbst von den hochklassigen Chipkartenlesern nicht gelöst werden können. Die offenen und bisher selten unter Sicherheitsaspekten betrachteten Architekturen der meisten Smartphones führen jedoch zu neuen und anderen Risiken.

Ziel dieses Kapitels ist die Konzeption des mobilen Chipkartenlesers, der eine sichere Nutzung des nPA erlaubt und gleichzeitig den Besonderheiten eines Smartphones Rechnung trägt. Zunächst werden Ergebnisse betrachtet, die zur mobilen Chipkartennutzung bereits in der Literatur erarbeitet wurden. Danach werden Elemente des mobilen Chipkartenlesers erörtert und betrachtet, mit welchen Mitteln diese umgesetzt werden könnten.

3.1. nPA-Nutzung am Mobiltelefon in anderen Arbeiten

Eine PACE-Implementierung auf einem Mobiltelefon wurde bereits 2009 für JavaME vorgelegt.[11] Zielplattform war eines der wenigen damals verfügbaren NFC-Telefone, das Nokia 6212 classic. Der Protokolldurchlauf dauerte auf dem Mobilgerät ca. 43 Sekunden, was auf die begrenzten Ressourcen des Telefons zurückgeführt wurde.[12] Die Implementierung von Horsch wurde im Rahmen mehrerer Arbeiten um das vollständige EAC-Protokoll, SM und das eCard-API-Framework erweitert.[13] In weiteren Arbeiten wurden Optimierungen vorgenommen, so dass ein einfacher PACE-Durchlauf auf 7,8 Sekunden verkürzt werden konnte.[14]

Eine andere Umsetzung der eCard-API für Mobilgeräte versuchte nicht nur eine bloße Umsetzung der Vorgaben aus BSI TR-03112. Beilke untersuchte zudem auf konzeptioneller Ebene, ob und wie die Technischen Richtlinien für eine optimale Nutzung auf einem mobilen Gerät angepasst werden könnten.[15] Das bestehende Konzept der eCard-API lässt alles in allem wenig Spielraum für konzeptionelle Optimierungen. Angeregt wird eine schlanke (nicht notwendigerweise mit der eCard-API konforme) Schnittstelle, die ausschließlich für den

[8] Müller, Morgner und Oepen 2011, S. 185–187.
[9] Morgner u. a. 2011.
[10] Zwar liegen die Preise von einigen Smartphones über denen eines Komfortlesers. Dennoch ist die nPA-Nutzung am Smartphone für jemanden, der ohnehin im Besitz eines entsprechenden Telefons ist, praktisch kostenlos.
[11] Horsch 2009.
[12] Horsch 2009, S. 64–65.
[13] Horsch 2011; Kiefer 2010.
[14] Buchmann u. a. 2010; Wiesmaier u. a. 2011.
[15] Beilke 2010.

mobilen Einsatz des nPA mit eID gedacht ist.[16] Verringerte Komplexität und der Verzicht auf verschiedene unnötige Operationen und Dienste wären möglich.

Die Umsetzung einer Client-Software auf einem Mobiltelefon mit integrierter NFC-Funktionalität bietet aber ganz offensichtlich keine höhere Sicherheit als die nPA-Nutzung an einem PC mit Basisleser.[17] Die Kommunikation zum Dienstanbieter erfordert eine Online-Verbindung, über welche das System auf gleiche Weise angreifbar ist wie ein PC.[18] Daher wurde vorgeschlagen, ausschließlich die Funktion eines Chipkartenlesers auf das Mobiltelefon auszulagern und die weniger kritischen Aufgaben weiterhin von dem angeschlossenen PC durchführen zu lassen.[19] Diese Idee wurde in Morgner u. a. 2011 in Relation zu den Lesegeräteklassen aus BSI TR-03119 1.2 gestellt und bereits um wichtige Aspekte der Sicherheit und Mobilität erweitert.

Die Umsetzung der PACE-Funktion ist dabei essenziell, um das Ausspähen der eID-PIN vom potenziell kompromittierten PC zu verhindern.[20] Eine Herausforderung stellt allerdings die Einbindung dieser Funktionalität in bestehende Standards dar.[21] Denn der Einsatz des mobilen Lesers an einem fremden PC oder Terminal ermöglicht es zuweilen nicht, entsprechende Gerätetreiber zu installieren. Die Umsetzung im Rahmen des USB-CCID-Standards wird einem drahtlosen Verfahren z. B. per Bluetooth oder GSM/UMTS vorgezogen. Die physische Verbindung zum Host-System ist für Anwender intuitiv, reduziert die Komplexität des Systems und vermeidet Probleme der logischen Zuordnung. Neben der Überprüfung und detaillierten Anzeige von CHAT und Berechtigungszertifikaten werden auch transaktionsbasierte Verbindungen vorgeschlagen, um dem Problem der Privilege Escalation[22] zu begegnen.[23]

Die Möglichkeit zur Umsetzung eines Chipkartenlesers per Mobiltelefon wird in Braun u. a. 2011 aufgegriffen. Ein spezieller PC/SC-Gerätetreiber, der die Handyansteuerung per Bluetooth übernimmt, könne dahingehend erweitert werden. Darüber hinaus könne die Authentisierung an einem Mobiltelefon statt über ein Passwort (oder eine PIN) auch über das biometrische Merkmal der Gangerkennung realisiert werden. Der Hauptgegenstand der Arbeit ist jedoch die Erarbeitung eines verteilten Komfort-Kartenleser, bei dem der nPA am Handy zur Erstellung einer QES genutzt wird. Secret-Sharing wird hierbei genutzt, um sowohl die Signatur-PIN selbst als auch den bei der EAC entstehenden Verschlüsselungsschlüssel unter den drei Parteien, Nutzer bzw. Client-Software, Trust-Center (TC) und dem universellem Authentisierungsdienst (uIdP) aufzuteilen. PACE wird lokal beim Nutzer am Handy durchgeführt, den Rest der EAC übernimmt der uIdP. Insbesondere weist der uIdP auch die Berechtigungen eines Signaturterminals nach, zu welchem sonst ein Komfortleser notwendig wäre. Die verteilte Verschlüsselung erfordert es, dass nicht nur ein Share der Signatur-PIN jeder Partei bekannt ist, sondern auch der Hash des zu signierenden Dokuments. Er wird von der Client-Software an die beiden anderen Parteien gesendet.

[16] Beilke 2010, S. 51–53, 67.
[17] Morgner u. a. 2011.
[18] Siehe „Malware“, S. 35.
[19] Müller, Morgner und Oepen 2011.
[20] Morgner u. a. 2011, S. 7–8.
[21] Morgner u. a. 2011, S. 5–6.
[22] Siehe „Privilege Escalation“, S. 52.
[23] Morgner u. a. 2011, S. 8–11.

Die Behauptung, dass „die Sicherheit der PIN garantiert [ist], solange nicht mindestens zwei Teilnehmer zusammenarbeiten“,[24] kann nach meiner Auffassung nicht bestehen. Da dem uIdP nach der EAC und dem TC zur Hashprüfung der vollständige Verschlüsselungsschlüssel zur Verfügung steht, können beide unabhängig voneinander aus der chiffrierten VERIFY-APDU die Signatur-PIN wiederherstellen. Die verschlüsselte APDU wird im Allgemeinen nicht als Geheimnis der Client-Software betrachtet. Sie wird zudem per NFC übertragen und damit abgreifbar. Außerdem kann das Verfahren nicht die Manipulation der zu signierenden Daten abwehren. Weiterhin besitzt das vorgeschlagene iPIN Verfahren zur Vermeidung von Replay-Angriffen dieselben Schwächen wie auch das iTAN-Verfahren. Durch einen simplen Trojaner könnte der Autorisierungscode einer beabsichtigten Transaktion abgegriffen und für andere Zwecke verwendet werden.[25] Die Evaluation nach Common Criteria EAL3+ ist nach Braun u. a. ein weiteres Hindernis bei der Signatur auf einem Mobiltelefon. Obwohl rechtlich die „alleinige Kontrolle des Inhabers über seine SSEE“ gefordert ist, halten sie es für „nicht ausgeschlossen“, dass ihr verteiltes System „alle Anforderungen des Signaturgesetzes und der Signaturverordnung erfüllt“.[26]

Bereits in der Konzeptphase des verteilten Komfortlesers offenbaren sich eine Reihe von Problemen. Nach meiner Einschätzung bietet ein Mobiltelefon zu viele Angriffsvektoren, als dass sich eine QES geeignet erstellen lässt. Der im Folgenden dargestellte CAT-M soll daher ausschließlich eine sichere Nutzung der eID-Anwendung gewährleisten. Die Ergebnisse der verschiedenen Autoren zeigen aber, dass durchaus die Möglichkeit gesehen wird, auf einem Smartphone sicherheitsrelevante Applikationen zu realisieren.

3.2. Erarbeitung des mobilen Chipkartenlesers für den neuen Personalausweis

Der CAT-M soll zur sicheren Nutzung der eID-Anwendung des nPA dienen. Dazu gehören die sichere PIN-Eingabe und Konsistenzprüfungen der übertragenen Daten. Gleichzeitig soll auch die übliche Smartphone-Nutzung möglich sein. Insbesondere soll sich ein Nutzer mit dem Smartphone im Internet bewegen, Anwendungen von Drittanbietern installieren und nicht zuletzt Telefonieren können. Die Nutzung der Funktionalität des Chipkartenlesers soll nicht nur im heimischen Umfeld möglich sein, sondern auch an Systemen, für die der Ausweisinhaber keine speziellen Zugriffsrechte hat. Die Funktionen des mobilen Chipkartenlesers umfassen minimal:

- Bereitstellung einer sicheren Ausführungsumgebung für die eID-Nutzung
- Zugriff auf NFC-Chipkarten, insbesondere den nPA
- Sichere PIN-Eingabe und gesicherte Kommunikation zum nPA
- Chipkartenleser-Funktionalität an einem externen Host-System
- Darstellung der Details zu einer eID-Abfrage
- Inhaltliche Prüfung der Authentisierungsdaten eines Terminals

[24] Braun u. a. 2011, S. 6.
[25] Drimer, Murdoch und Anderson 2009, S. 185.
[26] Braun u. a. 2011, S. 8.

Abbildung 3.1.: Der mobile Chipkartenleser bei der Nutzung an einem PC

- Sicheres Software/Firmware-Update

Ein Teil dieser Funktionen lässt sich ohne Weiteres mit dem Funktionsumfang eines Standard- oder Komfortlesers vergleichen. Weil aber nur die weniger sicherheitskritische eID-Nutzung für den mobilen Chipkartenleser vorgesehen ist, muss dieser nicht das aufwendige Prüfverfahren z. B. eines Komfortlesers durchlaufen. Damit lässt sich der CAT-M in den modularen Aufbau der Kartenleser nach BSI TR-03119 1.2 integrieren.

Aber eine standardisierte Host-System-Nutzung über PC/SC hinaus, Vorschriften zur Plausibilitätsprüfung (auch eines externen Geräts) und vor allem Möglichkeiten der Bereitstellung einer sicheren Arbeitsumgebung wurden bisher noch nicht beschrieben. Dies zeigt, dass der CAT-M eine eigenständige Kategorie bildet, die teilweise auch über die Möglichkeiten eines Standard- oder Komfortlesers hinaus gehen. Diese neuen Module werden Gegenstand der folgenden Untersuchung sein. Weiterhin werden zusätzliche Module vorgeschlagen, die optional eingesetzt werden können, um einige der genannten Bedrohungsszenarien zu umgehen.

3.3. Neue und angepasste Module für BSI TR-03119 1.2

3.3.1. PIN-Pad mit PACE-Unterstützung

Wie auch bei Standard- oder Komfortleser wird durch die sichere PIN-Eingabe am eigenen Gerät verhindert, dass auf dem Host-Rechner das Geheimnis abgegriffen werden kann. Die Unterstützung des PACE-Verfahrens ist zur Erfüllung dieses Moduls obligatorisch. Der CAT-M muss also in der Lage sein, selbst einen sicheren Kanal zum nPA aufbauen zu können.

In BSI TR-03119 1.2 wird vorgeschrieben, dass dazu ein zertifizierter Pseudozufallszahlengenerator genutzt werden soll. Moderne Betriebssysteme verwenden bereits verschiedene Quellen für zufällige Ereignisse genutzt, um Applikationen ein einheitliches Interface zu

Zufallszahlen bereitzustellen.[27] In diesen „Entropie-Pool“[28], der meist zur Initialisierung des Pseudozufallszahlengenerators genutzt wird, könnten auch Zufallszahlen des nPA einfließen. Dazu müsste das Smartphone einen PACE-Kanal zum nPA aufbauen und eine verschlüsselte `GET CHALLENGE`-APDU senden. Ein Angreifer kann die erhaltene Zufallszahl nicht lesen. Die Qualität des Entropie-Pools könnte so noch einmal erhöht werden.

Sichere Eingabemethode

Soll ein Chipkartenleser auch in einer unsicheren Arbeitsumgebung eingesetzt werden (wie z. B. der CAT-M), müssen Gegenmaßnahmen zum sogenannten „Shoulder Surfing“ ergriffen werden. Dass Eingaben per (On-Screen-) Tastatur oder anderer Methoden leicht analysiert werden können, ist ein seit langem bekanntes Problem.[29] Zwar kann bereits einfaches Abdecken der PIN-Eingabe das Ausspähen erschweren, dennoch sind für die sichere Eingabe am mobilen Chipkartenleser weitere Gegenmaßnahmen empfehlenswert.

Prinzipiell ließe sich die eID-PIN auch in einem sicheren Gerät speichern. Dazu müsste aber das Smartphone selber sicherstellen, dass nur der legitime Eigentümer das gespeicherte Geheimnis verwenden kann. (Eine schwache oder fehlende Nutzerauthentisierung bei einem Smartphone, welches die eID-PIN speichert, macht die eID-Anwendung jedem nutzbar, der im Besitz des Smartphones ist und Zugriff auf den nPA hat; aus den Authentisierungsfaktoren „Besitz“ (Zugriff) und „Wissen“ wird „Zugriff“ und „Besitz“.) Man könnte ein Single-Sign-On-Verfahren realisieren, das z. B. auch die Verwendung der SIM-Karte freischaltet. Allerdings sind die bisher genutzten Verfahren auf heutigen Smartphones nicht unbedingt sicher. Android bietet die Authentisierung per Gesichtserkennung, was durch Fotoaufnahmen überwunden werden kann;[30] PIN-Eingaben – vor allem mit hervorgehobenem Tastendruck – können aufgezeichnet und analysiert werden,[31] ebenso wie bei der Verwendung eines Bild-Passwortes wie es für Windows 8 möglich ist.[32]

Verschiedene Verfahren wurden in der Literatur vorgestellt, um Shoulder Surfing vorzubeugen. Eine Möglichkeit ist die Eingabe einer modifizierten PIN, wobei per Kopfhörer vorgegeben wird, wie die PIN zu modifizieren ist.[33] Die akustischen Mitteilungen stellen dabei einen quasi sicheren Kanal zum legitimen Nutzer dar. Der Angreifer sieht nur eine Zahlenfolge, die ausschließlich für die aktuelle Authentisierung des legitimen Nutzers verwendet werden kann.

Hopper und Blum schlagen die Nutzung eines Challenge-Response-Verfahrens zwischen Nutzer und Eingabegerät vor.[34] Notwendig ist hier allerdings ein vorab geteiltes Geheimnis zwischen beiden Parteien. Challenges werden entsprechend der PIN und dem geteilten Geheimnis beantwortet. Das Eingabegerät kann unter Verwendung des geteilten Geheimnisses

[27] Auf Linux-Systemen werden Zufallszahlen per `/dev/random` angeboten. Verwaltet vom Linux-Kernel gehen hier verschiedene Ereignisse von Gerätetreibern ein.

[28] Ferguson und Schneier 2003.

[29] Siehe z. B. Anderson 2001, S. 45. Seit 2002 gibt es eine entsprechende Regulierung der ISO, die Maßnahmen gegen das Shoulder Surfing für Bankautomaten und Point-of-Sales-Systeme vorschreibt (ISO 9564-1).

[30] `http://www.youtube.com/watch?v=BwfYSR7HttA`

[31] haroon@thinkst.com 2011; Maggi u. a. 2011.

[32] Build-Windows-Konferenz 2011, Anaheim

[33] Perkovi, Cagalj und Saxena 2011.

[34] Hopper und Blum 2000.

die PIN wiederherstellen. Das Ableiten der PIN ohne das geteilte Geheimnis, z. B. als passiver Beobachter, ist hingegen nicht möglich.

Ein Vorschlag, der Zero-Knowledge-Verfahren ähnelt, wurden in Roth, Richter und Freidinger 2004 gemacht. Die Eingabemöglichkeiten für eine Stelle der PIN werden in zwei zufällige aber gleich große Mengen geteilt, z. B. durch Einfärben der Ziffern. Der Nutzer gibt nun an, ob die gesuchte Ziffer sich in der einen oder anderen Menge befindet. Dieser Test wird mit zufällig ausgewählten Mengen so oft wiederholt, bis feststeht, dass der Nutzer die PIN mit sehr hoher Wahrscheinlichkeit weiß. Die geringe Unsicherheit führt dazu, dass ein Beobachter erst nach mehreren Protokolldurchläufen das vollständige Geheimnis ableiten kann. Da angenommen wird, dass die PIN-Eingabe nicht regelmäßig in derselben unsicheren Umgebung getätigt wird, ist das Geheimnis ausreichend geschützt. Das Verfahren hat den Nachteil, dass das Eingabegerät selbst die PIN kennen und entsprechend sicher verwahren muss.

In Luca, Denzel und Hussmann 2009 wird die Analyse von Augenbewegungen vorgeschlagen, um ein Geheimnis sicher einzugeben. Die Kombination aus der relativen Augenbewegung zu einzelnen Punkten und dem „Zeichnen“ spezieller Symbole mit den Augen verspricht eine hohe Erkennungsrate unter Verwendung von preiswerter Hardware. Vorteil dieser Methode ist, dass viele Smartphones bereits mit genügend guten Videokameras ausgestattet sind, um die Augenbewegung zu erkennen. Allerdings entsteht auch ein neuer Angriffsvektor. Auch ein passiver Beobachter könnte die Augenbewegung analysieren.

Filterregeln

In dieser Arbeit wurde gezeigt, dass die bisher geforderten Filterregeln nicht genügen, um einige Angriffe gegen den nPA zu unterbinden. Zwar heißt es in BSI TR-03119 1.2, S. 18, dass die Ausführung von PACE durch den Host-Rechner gänzlich verhindert werden soll. Dennoch ist PACE mit applikationsseitig vorgegebener CAN beliebig möglich. Die Prüfmethoden müssen angepasst werden, wenn Standard- oder Komfortleser auch die CAN schützen sollen.

Die Vorschläge zum Schutz vor Brute-Force-Angriffen und ungewollten Transaktionen wird hier für minimal invasive Vorschläge argumentiert.[35] Ein PACE-fähiges Lesegerät sollte zusätzlich zu den vorhandenen Anforderungen[36] die folgenden zusätzlichen Maßnahmen implementieren.

Nutzerbestätigung bei Verwendung eines vorgegebenen oder gespeicherten Geheimnisses Brute-Force-Angriffe werden durch eine simple Bestätigung des Nutzers mit „OK“ verhindert. Gleichzeitig können Geheimnisse weiterhin von der Chipkartenapplikation vorgegeben werden. Dieser Vorschlag ist kompatibel mit BSI TR-03119 1.2 und den produktiv genutzten Client-Applikationen für den nPA.

Verifizierung der TA und Prüfung der CV-Zertifikate Die zweite Ergänzung zu den Filterregeln bezieht sich auf eine Überprüfung des Dienstanbieters. Da einem PACE-fähigen

[35] Siehe „Schutz von CAN und MRZ“, S. 62.
[36] BSI TR-03119 1.2.

Chipkartenleser alle APDUs der TA im Klartext vorliegen, kann ein Chipkartenleser die Überprüfung der Zertifikatskette und der Signatur des Terminals unabhängig vom nPA durchführen. Dazu muss er lediglich mit demselben Vertrauensanker ausgestattet sein wie der nPA. Diese CVCA-Zertifikate sind recht statischer Natur und ändern sich lediglich alle drei Jahre. Sie könnten in die Firmware eines Lesers integriert werden.

Bedeutender, aber mit Hinblick auf die anderen PACE-fähigen Leser optional, ist die Prüfung der Aktualität des Terminalzertifikats. Mit der Systemzeit des CAT-M kann dies leicht sichergestellt werden.

Um die fehlende Möglichkeit zur Revozierung eines Zertifikats zu ergänzen, könnten mit der Netzwerkverbindung des CAT-M aktuelle Zertifikatsperrlisten aus vertrauenswürdiger Quelle geladen werden. Andere PACE-fähige Chipkartenleser können die Internetverbindung des Host-Computers dafür nutzen.[37]

Mit Prüfung des Terminalzertifikats auf Aktualität und mittels Zertifikatsperrlisten wird die Nutzung von ungültigen Terminalzertifikaten effektiv verhindert.

3.3.2. Speicherung der Kartenzugriffsnummer im Chipkartenleser

Die Speicherung des Geheimnisses im Lesegerät statt in der auf dem potenziell unsicheren Host-Computer sollte auf lange Sicht vorgezogen werden. Im Heimbereich werden sicherlich nur wenige Ausweise mit ein und demselben Lesegerät benutzt. Ist PACE mit CAN gewünscht, könnte der Leser nach einer Initialisierung automatisch den sicheren Kanal aufbauen. Falls dem Leser mehrere CANs zur Verfügung stehen (weil er mit verschiedenen Ausweisen initialisiert wurde), könnte er dem Benutzer anbieten, entweder zu warten bis die korrekte CAN durch Probieren ermittelt wurde[38] oder aber der Nutzer gibt die CAN selbst am Leser ein.[39]

3.3.3. Display: Vollständige und übersichtliche Darstellung

Die bisher spezifizierten Minimalanforderungen an das Display eines Chipkartenlesers genügen kaum, um den Ausweisinhaber umfassend über die Details der Nutzung seines nPA zu informieren.[40] Für die eID-Nutzung relevant sind folgende Informationen:

- Name und Anschrift des Dienstanbieters
- Auslesezweck
- Angeforderte Ausweisdaten/-funktionen

Der Nutzungskontext (eID) ist beim mobilen Chipkartenleser implizit gegeben, da der Zugriff anderer Applikationen des nPA gefiltert werden soll.[41]

Wie diese Informationen aus Usability-Sicht darzustellen sind, wird hier nicht im Detail betrachtet werden. Andere Arbeiten haben sich diesem Thema bereits genähert und sind

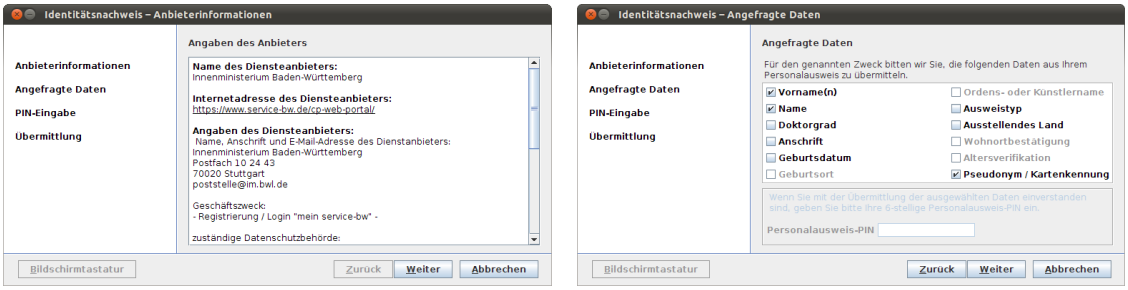
[37] Der Gerätetreiber des Chipkartenlesers müsste diese Funktion bereitstellen und die Weiterleitung der Zertifikatssperrlisten übernehmen.

[38] Die CAN besitzt keinen Fehlbedienungszähler. Das Probieren der wenigen gespeicherten CANs benötigt nur wenig Zeit (siehe Abbildung 2.19a).

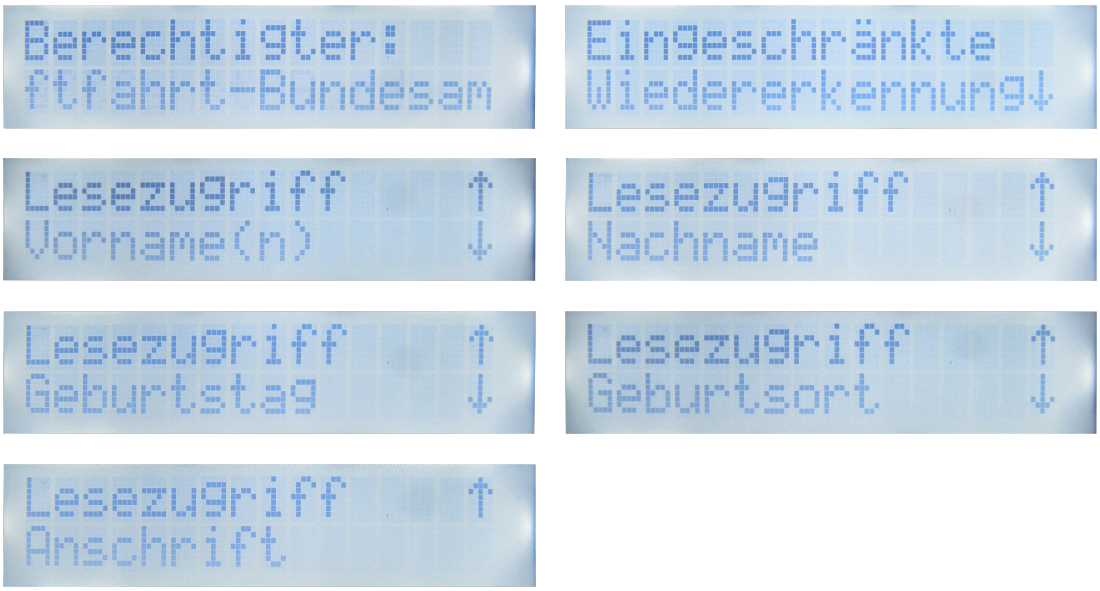
[39] Dies ließe sich auch einfach parallelisieren: Während der Nutzer nach der CAN gefragt wird, könnte der Leser bereits selbstständig die CAN durch Probieren der gespeicherten Werte ermitteln.

[40] Siehe Lesegeräteklassen nach BSI TR-03119 1.2; „Man-in-the-Middle Hardware“, S. 40

[41] Siehe „Filterregeln“, S. 73.



(a) Darstellung durch die AusweisApp



(b) Darstellung durch den Komfortleser REINER SCT cyberJack RFID komfort. Weitere Informationen, z. B. der Auslesezweck, werden nicht dargestellt.

Abbildung 3.2.: Darstellung der Verbindungsdetails einer eID-Nutzung

zu dem Schluss gekommen,[42] dass die Informationen *auf den ersten Blick* erkennbar und dennoch nicht überladen sein sollten. Die Fülle an Informationen zur eID-Nutzung führt bei der AusweisApp beispielsweise dazu, dass die Informationen in zwei aufeinander folgenden Fenstern dargestellt werden (siehe Abbildung 3.2a).

Mit einem größeren und in der Darstellung flexibleren Display kann vermieden werden, dass zur Darstellung der notwendigen Informationen mehrere Displaylängen benötigt werden, wie den aktuellen Standard- und Komfortlesern (siehe Abbildung 3.2b). Moderne Smartphones besitzen hochauflösende Displays, die für diese Zweck geeignet sind.

Auf einem Smartphone könnte die Darstellung der AusweisApp nachempfunden (siehe Abbildung 3.3) oder sogar verbessert werden. So könnten etwa die Angaben zum Aussteller des Zertifikats oder der zuständigen Datenschutzbehörde in einer erweiterten Ansicht zur Verfügung gestellt werden. Die einfache Ansicht könnte alle anderen Informationen dann bei

[42] Grote u. a. 2010; Oepen 2010.

Abbildung 3.3.: Vorschlag zur Darstellung der Verbindungsdetails einer eID-Nutzung auf dem Openmoko Neo FreeRunner (Oepen 2010)

größeren Displays auf einer Bildschirmseite darstellen.

3.3.4. Standardisierte Hardware-Treiber

In BSI TR-03119 1.1, S. 20,26,33 wurde von Basis-, Standard- und Komfortleser noch eine anwenderfreundliche Installation des Gerätetreibers auch für Laien verlangt. Auch die Nutzung von Standard-Schnittstellen, z. B. USB, wurde gefordert. Die Subsumierung unter dem Modul „Anwenderfreundlichkeit“ dieser Eigenschaften ist nach meiner Auffassung folgerichtig. Wie Standard-Schnittstellen zu benutzen sind, ist für Ungeübte nachvollziehbar. Außerdem ermöglichen sie die Nutzung von standardisierten Hardware-Treibern. Befolgen verschiedene Geräte-Modelle denselben Standard, kann der entsprechende Treiber aktuellen Betriebssystemen beigelegt werden, so dass eine explizite Installation dieser Software entfällt. Insbesondere bedeutet dies, dass kein Anwender tief in das Betriebssystem eingreifen muss, um das Gerät nutzen zu können; es sind keine Administratorenrechte notwendig.

Diese Forderungen wurden in der aktuellen Version der Richtlinie, BSI TR-03119 1.2, teilweise fallengelassen. Zwar ist Anwenderfreundlichkeit für alle Chipkartenleser eine „Grundlagenanforderung“.[43] Geprüft wird aber lediglich, ob dem Gerät eine Installationsanleitung beiliegt

[43] BSI TR-03119 1.2, S. 11.

und es nach „gängigen Installationsmethoden“[44] installierbar ist. Mit dem Verzicht auf Standard-Schnittstellen werden auch alternative Konzepte ermöglicht.[45] Zugleich muss sich ein Benutzer an diese neuen Konzepte erst gewöhnen. Dies kann nicht nur ein Problem der Anwenderfreundlichkeit, sondern auch der Sicherheit darstellen, denn ungeübten Anwendern unterlaufen leichter Fehler. Hinzu kommen die möglichen Sicherheitsprobleme von nicht erprobten Protokollen und Implementierungen. Beim drahtlosen Anschließen an den PC, z. B. per Bluetooth, WLAN oder GSM, ist nicht immer klar, zu welcher Entität der Chipkartenleser eine Verbindung aufbaut.[46] Außerdem sind drahtlose Verbindungen eher angreifbar durch Abhören oder Manipulieren als drahtgebundene Verfahren.

Damit der CAT-M zur ein- oder mehrmaligen Nutzung an verschiedenen Host-Systemen zum Einsatz kommen kann, wird hier also für standardisierte drahtgebundene Verfahren argumentiert. Seitens der Host-System-Applikationen wird auf die PACE-Funktion per PC/SC zugegriffen.[47] Wie das Host-System bzw. der Gerätetreiber aber auf den Chipkartenleser selbst zugreift, ist bisher nicht spezifiziert. Chipkartenleser werden heute üblicherweise per USB an das Host-System angeschlossen, einige wenige auch per Bluetooth. Hier bietet nur USB ein herstellerübergreifendes Profil für Chipkartenleser.

In dem USB Implementers Forum arbeiten verschiedene Hard- und Softwarehersteller gemeinsam an USB CCID. Seit 2001 gibt es diesen Industriestandard für Chipkartenleser, dem eine Vielzahl von Modellen folgt.[48] Gerätetreiber sind für Windows,[49] Linux, Mac OS X, Solaris und FreeBSD verfügbar und zumeist vorinstalliert.[50] Die CCID-Geräteklasse ist unabhängig zur Übertragungstechnik spezifiziert, so dass neben den bekannten drahtgebundenen Versionen USB 1.0/2.0/3.0 auch Wireless USB zum Einsatz kommen könnte.[51]

USB CCID definiert Funktionen zur Verwaltung des Chipkartenlesers oder der Chipkarte (im Leser), zur Datenübertragung an die Karte und zur PIN-Behandlung (siehe Tabelle 3.1). Die bereits spezifizierten Funktionen zur PIN-Verifikation/-Modifikation nach USB CCID, S. 38–45 erfolgen ausschließlich *mit Übertragung der PIN* vom Chipkartenleser an die Chipkarte. Von der Chipkartenapplikation wird dafür eine bereits formatierte APDU per PC/SC bzw. CCID an

[44] BSI TR-03119 1.2, S. 27.

[45] In KOBIL Systems GmbH 2010 wurde ein Adapter vorgestellt, der einen kontaktbasierten Klasse 2 oder Klasse 3 Chipkartenleser zu einem Standardleser für den nPA erweitert. Zur Eingabe der eID-PIN des nPA wird das PIN-Pad des kontaktbasierten Lesers genutzt, die eingegeben eID-PIN zur Durchführung von PACE an den Adapter weitergeleitet. Die verwendete Schnittstelle entspricht sicherlich keinem gängigen Konzept. Der kontaktbasierte Leser wird an den PC angeschlossen und „durch ihn hindurch“ wird der Adapter ähnlich einer kontaktbasierten Chipkarte angesprochen.

[46] Morgner u. a. 2011.

[47] BSI TR-03119 1.2; PC/SC Pt. 10 amd. 1.

[48] Eine Auswahl von Chipkartenlesern konform zu USB CCID bietet die Projektseite von Rousseau 2011 unter `http://pcsclite.alioth.debian.org/ccid/section.html`

[49] Microsoft bietet für Windows Betriebssysteme ab Windows 2000 die automatische Installation mittels Windows-Treiberupdate (`http://msdn.microsoft.com/en-us/windows/hardware/gg487509`). Auf mobilen Geräten liefert Microsoft den passenden Gerätetreiber ab Windows Embedded CE 6.0 R2 mit (`http://msdn.microsoft.com/en-us/library/ee484819%28v=winembedded.60%29.aspx`).

[50] Auf Unix-artigen Systeme wird `libccid` eingesetzt (Rousseau 2011). Es gibt jeweils fertige Installationspakete, die teilweise automatisch installiert werden oder den Betriebssystemen bereits beiliegen.

[51] Auch Wireless USB besitzt das Problem des Pairings zum Host-System. Dies müsste bei einem Chipkartenleser natürlich geeignet gelöst werden.

Bezeichnung	Beschreibung	Antwort
PC_to_RDR_IccPowerOn	Abfrage und Modifikation des Zustands einer Chipkarte in einem Slot des Kartenlesers	RDR_to_PC_DataBlock (ATR)
PC_to_RDR_IccPowerOff		RDR_to_PC_SlotStatus
PC_to_RDR_GetSlotStatus		RDR_to_PC_SlotStatus
PC_to_RDR_GetParameters	Abfrage und Modifikation der Kommunikationsparameter zur Chipkarte	RDR_to_PC_Parameters
PC_to_RDR_ResetParameters		RDR_to_PC_Parameters
PC_to_RDR_SetParameters		RDR_to_PC_Parameters
PC_to_RDR_IccClock		RDR_to_PC_SlotStatus
PC_to_RDR_T0APDU		RDR_to_PC_SlotStatus
PC_to_RDR_SetDataRateAndClockFrequency		RDR_to_PC_DataRateAndClock
PC_to_RDR_XfrBlock	Datentransfer (APDUs) zur Chipkarte	RDR_to_PC_DataBlock (R-APDU)
PC_to_RDR_Abort	Abbruch angestauter Datentransfers	RDR_to_PC_SlotStatus
PC_to_RDR_Escape	Hersteller-/Leserspezifische Funktionen	RDR_to_PC_Escape
PC_to_RDR_Mechanical	Physisches Blockieren/Auswerfen einer Karte	RDR_to_PC_SlotStatus
PC_to_RDR_Secure	Sichere PIN-Eingabe mittels PIN-Pad	RDR_to_PC_DataBlock

(a) Kommandos des PC an den Chipkartenleser

Antwort	Beschreibung
RDR_to_PC_DataBlock	Rückgabedaten der Chipkarte (APDU oder ATR, 0-65538 Bytes), möglicherweise gefiltert
RDR_to_PC_SlotStatus	
RDR_to_PC_Parameters	Verbindungsparameter: 2-wire, 3-wire oder I2C Protokoll T=0 oder T=1, Protocol Data Structure
RDR_to_PC_Escape	Rückgabedaten proprietärer Funktionen
RDR_to_PC_DataRateAndClock	Datenrate und Taktfrequenz

(b) Antworten des Chipkartenlesers an den PC

Tabelle 3.1.: Kommandos eines Chipkartenlesers nach USB CCID, S. 26–55

den Chipkartenleser gesendet. Dieser liest das Geheimnis an seinem PIN-Pad ein und druckt es geeignet formatiert in den Datenteil der APDU. Das fertige Kommando wird an die Chipkarte gesendet und das Ergebnis vom Leser an die Chipkartenapplikation zurück gegeben.

Offensichtlich sind diese Kommandos aber nicht geeignet, um die PACE-Funktion eines Chipkartenlesers aufzurufen. Denn die Verifikation der PIN soll bei PACE gerade ohne Übertragung des Geheimnisses realisiert werden. Um die PACE-Funktion des Lesers in USB CCID zu integrieren, sind folgende Möglichkeiten der Standardisierung denkbar:

1. Definition eines neuen Kommandos zur PIN-Verifikation ohne PIN-Übertragung (insbesondere für PACE), z. B. als `PC_to_RDR_PACE`
2. Erweiterung von `PC_to_RDR_Secure`. Analog zur PIN-Verifikation mit PIN-Übertragung könnte das Kommando ergänzt werden um eine Datenstruktur zur Durchführung von PACE.
3. Nutzung von `PC_to_RDR_Escape` zum Auslösen von Chipkartenleser spezifischen Funktionen
4. Datenübertragung und Auslösen von PACE per Pseudo-APDU unter Nutzung von `PC_to_RDR_XfrBlock`

Die Nutzung speziell für PACE konzipierter Kommandos (Punkt 1, Punkt 2) besitzen den Vorteil, dass keine Änderungen an der Chipkartenapplikation notwendig sind. Die PC/SC-Schnittstelle mit den Änderungen aus BSI TR-03119 1.2 bzw. PC/SC Pt. 10 amd. 1 genügen zur Realisierung des vollen Funktionsumfangs. Nachteilig ist jedoch, dass der Gerätetreiber des Chipkartenlesers das Übersetzen von der PC/SC-API nach CCID neu implementieren muss.

Gegenteilig verhält es sich mit den zwei anderen Varianten (Punkt 3, Punkt 4). Zwar löst man die CCID-Kommandos `PC_to_RDR_XfrBlock` und `PC_to_RDR_Escape` direkt mit ihren PC/SC-Gegenstücken `SCardTransmit` bzw. `SCardControl` (mit `FEATURE_CCID_ESC_COMMAND`)[52] aus. Aktuelle Gerätetreiber implementieren diese Kommandos bereits. Dennoch erfordert diese Art der Umsetzung eine entsprechende Modifikation der Chipkartenapplikation. `GetReaderPACECapabilities` und `EstablishPACEChannel` könnten dann nicht mehr nur per `SCardControl` (`FEATURE_EXECUTE_PACE`) angesprochen werden, sondern auch per `SCardTransmit`. Problematisch ist weiterhin, dass Escape-Kommandos oder Pseudo-APDUs bei einigen Geräten bereits zum Ansteuern spezieller Funktionen des Chipkartenlesers genutzt werden.[53] Will man auch das Initiieren von PACE über eine dieser Schnittstellen realisieren, muss vor allem darauf geachtet werden, dass keine Überschneidungen mit anderen Funktionen auftreten.

Punkt 3 und Punkt 4 erwecken zunächst den Eindruck als könne man sie leichter in die bestehenden Rahmenbedingungen integrieren. Neben den obligatorischen Anpassungen beim Chipkartenleser müsste lediglich die AusweisApp diese Schnittstelle implementieren. Aber mögliche negative Seiteneffekte bei der Überlagerung mit anderen Funktionen des Chipkartenlesers zwingen zu einer unabhängigen Lösung. Letztlich werden mit Escape-Kommandos oder Pseudo-APDUs nur Leser-spezifische Kommandos angesprochen, die eben gerade nicht standardisiert sind.

[52] Seit März 2010 ist in PC/SC Pt. 10, S. 3,25 die Erweiterung `FEATURE_CCID_ESC_COMMAND` spezifiziert. Damit kann über das `SCardControl`-Kommando mit entsprechenden Kontroll-Code das `PC_to_RDR_Escape` an das CCID-Gerät gesendet werden.

[53] Mit Escape-Kommandos können z. B. Firmware-Updates realisiert sein. Auch BSI TR-03119 1.2 definiert Pseudo-APDUs, um den Chipkartenleser und seine Firmwareversion zu erkennen.

Daher sollten entweder Punkt 1 oder Punkt 2 bevorzugt werden. Die Änderungen, die diese beiden Vorschläge erfordern, sind sehr ähnlich, so dass es eine Geschmacksfrage ist, zu welcher man am Ende tendiert. Logisch passt die PIN-Verifikation per PACE sehr gut zu den bestehenden Varianten des `PC_to_RDR_Secure`-Kommandos. Auch können bei dieser Möglichkeit die Meta-Informationen der sonst üblichen PIN-Verifikation/-Modifikation, wie z. B. Länge des einzulesenden Geheimnisses, Timeout der PIN-Eingabe oder Sprachauswahl des Anzeigetextes, wiederverwendet werden.[54] Hier wird also zu der Umsetzung im Rahmen des `PC_to_RDR_Secure`-Kommandos tendiert, obwohl auch ein eigenes Kommando für PACE denkbar ist.[55]

Jede der vorgeschlagenen Lösungen besitzt den Nachteil, dass sie derzeit in keiner Soft- oder Hardware integriert ist. Die eigentliche Motivation für die Integration von PACE in USB CCID, die konfigurationsfreie Nutzung an verschiedenen Host-Systemen, ist damit nur begrenzt möglich. Dennoch hat die Standardisierung das Potenzial, dem Anspruch an Komfort und zugleich Sicherheit gerecht zu werden und sich letztlich durchsetzen zu können.

3.3.5. Prüfungen als Kartenemulator

Die Sicherheitsprobleme mit dem nPA rühren teilweise daher, dass man dem Ausweis selbst nicht ansieht, in welchem Kontext er benutzt wird und welche Daten abgefragt werden sollen.[56] Die Kompatibilität moderner Smartphones zu ISO/IEC 14443 ermöglicht nicht nur die Nutzung des *Reader/Writer Mode* zur Umsetzung eines Chipkartenlesers. Möglich ist durch den *Card Emulation Mode* auch die Umsetzung eines nPA-Emulators. Ein solcher Emulator kann vor der Nutzung des echten Ausweises dem Ausweisinhaber visualisieren, ob und wie die eID- oder eSign-Funktion genutzt werden sollen.[57] Auch wenn die Nutzung des mobilen Chipkartenlesers an einem öffentlichen Terminal nicht möglich sein sollte, können auf diese Weise Skimmingversuche mit dem Emulator besser erkannt und verhindert werden. Folgende Merkmale können überprüft werden:

- Nutzungskontext und angeforderte Berechtigungen
- Verifizierung der TA und Prüfung der CV-Zertifikate
- Ermittlung und Anzeige der Zertifikatsbeschreibung

Die Nutzung des nPA-Emulators ist unkompliziert: Wird das Smartphone in den Kartenemulationsmodus versetzt, ist es nach außen hin per NFC zugreifbar wie ein echter Ausweis. Die Dateistruktur (im Wesentlichen das `EF.CardAccess`) sowie die (kryptographischen) Protokolle des nPA müssten gemäß BSI TR-03127 bzw. BSI TR-03110 2.05 nachgebildet werden. Dabei muss nicht die vollständige EAC durchgeführt werden, PACE mit anschließendem SM und TA

[54] Diese Meta-Informationen sind allerdings bei der PC/SC-Schnittstelle zu PACE nicht vorgesehen, so dass prinzipiell auch vernünftige Standard-Werte denkbar sind.

[55] Auch eine zweigleisige Strategie könnte angestrebt werden: Die gleichzeitige Umsetzung von Punkt 1/2 *und* Punkt 3/4 könnte erreichen, dass kurzfristig die Modifikation der Gerätetreiber umgangen werden kann und langfristig könnte die überschneidungsfreie Standardisierung der PACE-Funktion erreicht werden. Fraglich ist bei dieser Lösung jedoch, ob die langfristigen Ziele für die Hersteller von Chipkartenlesern und Gerätetreibern ausreichend motiviert sind und sich letztlich durchsetzen können.

[56] Siehe „Mitigation“, S. 58.

[57] Müller, Morgner und Oepen 2011, S. 181,186.

Abbildung 3.4.: Der mobile Chipkartenleser bei der Überprüfung eines externen Terminals

genügen. Insbesondere die CA, bei der ein nPA seine Echtheit nachweisen würde, muss zur Prüfung des Nutzungskontextes und der Validierung des Terminals nicht bestanden werden.

Bereits bei PACE wird mit der ersten APDU, dem `MSE:Set AT`, der effektive CHAT übertragen. Damit kündigt das Terminal seine Rolle als Inspektions-, Authentisierungs- oder Signaturterminal sowie die gewünschten (und vom Nutzer eingeschränkten) Zugriffsrechte an. Abhängig von dem Terminal sind z. B. hier die Leserechte der Iris-, Fingerabdruck- oder andere Ausweisdaten sowie die Rechte „Install qualified certificate“, „CAN allowed“ oder „Privileged Terminal“ vermerkt. Diese kritischen Informationen zur Ausweisnutzung stehen also bereits mit einem sehr minimalen nPA-Emulator zur Verfügung. PACE selbst kann der Kartenemulator mit einer zufälligen PIN durchführen, die dem Nutzer angezeigt wird, so dass er diese bei dem öffentlichen Terminal eingeben kann. In der anschließenden TA kann nun auch die Echtheit des Terminals verifiziert werden. Die Berechtigungen aus dem CHAT sind damit nachgewiesen.

Eine Überprüfung der Zertifikatsbeschreibung ist notwendig, um sicherzustellen, dass auch das richtige Terminal mit dem nPA kommuniziert. Sie gibt den Namen und die Anschrift des Dienstes sowie den Zweck des Auslesevorgangs an. Die Anzeige dieser Informationen kann jedoch nicht ohne Weiteres durch den nPA-Emulator erfolgen, weil die Beschreibung selbst nicht an die Chipkarte übertragen wird. Das Terminalzertifikat enthält aber einen Hash der Zertifikatsbeschreibung, welcher in die passende Beschreibung aufgelöst werden kann. Im Vorfeld der Prüfung muss eine Datenbank mit den möglichen Tupeln aus Hashwert und Beschreibung erstellt werden. Weil in Relation zur Länge des Hashwertes sehr wenige Dienstanbieter registriert sind, sollte es eine eins-zu-eins-Beziehung zwischen Hashwert und Zertifikatsbeschreibung geben.[58] Durch eine einfache „Lookup-Tabelle“ kann also aus dem Hash in dem Terminalzertifikat dessen Beschreibung gewonnen und angezeigt werden. Von offizieller Seite könnte ein Dienst angeboten werden, der die Zertifikatsbeschreibungen aller

[58] Die Zertifikatsbeschreibungen werden derzeit mit SHA-256 verknüpft. Nach dem gegenwärtigen Wissensstand führen selbst kleine Änderungen der Beschreibung zu einem veränderten Hashwert (Ferguson und Schneier 2003, S. 88–89). Dadurch dürften die wenigen Hashes der Zertifikatsbeschreibungen verschiedener Dienstanbieter nicht kollidieren.

Dienstanbieter veröffentlicht.

Auf einem Smartphone lässt sich weiterhin die Aktualität des Terminalzertifikats prüfen. Beginn und Ende des Gewährleistungszeitraums können mit der internen Uhr des Smartphones abgeglichen werden, so dass ungültige Terminalzertifikate nicht zum Einsatz kommen können.

Noch bevor der nPA-Emulator gezwungen ist, in der CA seine Echtheit nachzuweisen, ist er im Besitz aller wesentlichen Details zur beabsichtigten Transaktion: Anwendungskontext, angeforderte Ausweisdaten/-funktionen, Auslesezweck, Bestätigung über die Gültigkeit des Terminalzertifikats und Echtheit des Terminals. Dies gilt aber nur für die Dauer der Transaktion mit dem Emulator. Wenn der Nutzer seinen echten nPA an das öffentliche Terminal hält, könnte dieses die Details des Auslesevorgangs ändern und die Kommunikation sogar zu einem völlig anderen Dienstanbieter umleiten.

Wünschenswert wäre daher die sofortige Nutzung des nPA, bei der der Emulator auf der einen Seite mit dem Terminal kommuniziert und auf der anderen Seite mit dem Ausweis (siehe Abbildung 2.25). Wie beim PACE-fähigen Kartenleser, der eine „Firewall“ zwischen Client-Software und Chipkarte darstellt, gäbe es damit ebenfalls eine Firewall, nämlich zwischen Kartenleser bzw. Terminal und Chipkarte. Die Umsetzung wäre interessant, weil sie keinerlei Modifikation des Host-Systems erfordert. Im Moment scheint eine Implementierung praktisch aber nur sehr schwer möglich zu sein.[59]

Bei einem entsprechend gestalteten Emulator ist dieser bis zur CA elektronisch praktisch nicht von einem echten Ausweis zu unterscheiden. Denn explizite Ziele bei Design und Implementierung des nPA waren Datensparsamkeit und Anonymität des Ausweisinhabers. Es müssten weitere Methoden herangezogen werden, um einen nPA-Emulator zu erkennen. Weil die Ausführungszeit von PACE mit dem echten Ausweis relativ konstant ist,[60] könnte etwa dieser Seitenkanal zur Unterscheidung genutzt werden. Ein Angreifer könnte auch eine Kamera am Terminal anbringen, um optisch festzustellen, ob es sich um einen echten Ausweis handelt oder nicht.

Es ist zumindest fraglich, ob ein Angreifer die Unterscheidung zwischen einem echten Ausweis und einem Emulator mit einfachen Mitteln zuverlässig an einem manipulierten Terminal realisieren kann. In jedem Fall erschwert ein einfacher nPA-Emulator durch stichprobenartige Prüfung des Terminals Skimming-Angriffe und erhöht so die Sicherheit bei der elektronischen Nutzung des nPA. Weiterhin dient sie der Vertrauensbildung und Transparenz beim nPA-Einsatz an fremden Terminals.

3.3.6. Zeitaktualisierungsdienst

Die Nutzung abgelaufener Terminalzertifikate an einem Terminal, welches nicht mit dem mobilen Chipkartenleser kompatibel ist, kann auch mittels eines Zeitaktualisierungsdienst verhindert werden. Dieser kann die im Ausweis angenäherte Zeit aktualisieren, so dass alte Authentisierungszertifikate vom nPA als ungültig erkannt werden. Das BSI hat schon vor einiger Zeit die Umsetzung eines solchen Dienstes angekündigt.[61] Konkrete Pläne zur Implementierung oder gar ein Veröffentlichungstermin sind bisher jedoch nicht bekannt.

[59] Siehe „nPA-Kartenemulator“, S. 114.

[60] Siehe „Zeitverlauf beim Brute-Force-Angriff auf die CAN des nPA“, S. 48.

[61] BSI TR-03127, S. 30.

Es ist jedoch auf einfache Art möglich, das Zeitupdate beispielsweise im Rahmen des mobilen Chipkartenlesers unabhängig von dem offiziellen Dienst umzusetzen. Die Zeitaktualisierung findet bei der Validierung der Zertifikatskette vor der Prüfung der Signatur des Terminals statt.[62] Noch vor der CA kann damit einerseits die (aktuelle) Zertifikatskette ohne einen echten nPA abgegriffen werden.[63] Andererseits kann dem Ausweis eine beliebige Zertifikatskette (auch aus unsicherer Quelle) zur Validierung und Aktualisierung der Zeit gereicht werden, ohne im Besitz des privaten Terminalschlüssels zu sein. Als einzige Einschränkung muss der Anfang der Zertifikatskette des Dienstanbieters mit dem Vertrauensanker des nPA übereinstimmen, da dieser sonst keine Validierung im Rahmen der EAC-PKI durchführen kann.

In Versuchen konnte bestätigt werden, dass es bezüglich des PACE-Kanals keine Terminalspezifischen Begrenzungen gibt, wenn es um den Import der Zertifikatskette geht. Dies bedeutet insbesondere, dass durchaus auch PACE mit CAN durchgeführt werden kann, ohne dass im Anschluss ein Terminal das Recht „CAN allowed“ nachweisen muss.[64] Die Zeitaktualisierung auf dem Ausweis könnte also für den Nutzer sehr komfortabel gestaltet werden. Automatisch können mit einer gespeicherten CAN ein sicherer Kanal zum nPA aufgebaut und die aktuellen Zertifikate importiert werden.

Allerdings führen nur hoheitliche Zertifikate zu einer Aktualisierung der inneren Uhr des nPA.[65] Entsprechende Zertifikatsketten müssten von offizieller Stelle angeboten werden, da die Authentisierungsterminals der meisten Dienstanbieter nicht die entsprechenden Berechtigungen besitzen. Der Download aktueller CV-Zertifikate von einer Website, unabhängig von der eCard-API, würde genügen.

Das Zeitupdate lässt sich also leicht und ohne spezielle Rechte umsetzen. Zwar darf die Zertifikatskette durchaus durch einen unsicheren Kanal auf den Chipkartenleser gelangen. Dennoch muss darauf geachtet werden, dass der Chipkartenleser selbst nicht auf diesem Weg durch Schadprogramme kompromittiert wird.

3.3.7. Autarkes PIN-Management und temporäre PIN

Der CAT-M sollte nicht davon abhängig sein, dass das Host-System das PIN-Management anstößt. Ein öffentliches Terminal könnte beispielsweise das Entsperren einer eID-PIN schlicht nicht implementieren. Hinzu kommt, dass die eID-PIN durch den Einsatz in potenziell unsicheren Arbeitsumgebung regelmäßig gewechselt werden sollte.

Der Chipkartenleser muss selbst eine Funktion bereitstellen, um das PIN-Management entsprechend BSI TR-03119 1.2, S. 17–18 durchzuführen. Die Umsetzung ist einfach. Es wird das PACE-Protokoll durchgeführt und im entstehenden sicheren Kanal zum nPA die entsprechenden APDUs abgesetzt.

Ein Gerät, mit welchem selbstständig und ohne fremde Stromversorgung die PIN geändert

[62] Siehe „Terminalauthentisierung und Extended Access Control Public Key Infrastructure“, S. 12.

[63] Siehe „Prüfungen als Kartenemulator“, S. 80.

[64] Zwar würde z. B. das Auslesen von Ausweisdaten mit einem Authentisierungsterminal lediglich mit vorheriger eID-PIN-Eingabe möglich sein. Die eigentliche EAC mit TA und sogar CA ist aber auch nach PACE mit CAN möglich.

[65] Siehe „Nutzung ungültiger Terminalzertifikate“, S. 57.

werden kann, ermöglicht auch die Einrichtung einer temporären PIN. Angenommen, es ist nicht möglich, den mobilen Chipkartenleser an ein Terminal anzuschließen, d. h. der nPA kann ausschließlich am Terminal selbst genutzt werden. Dann wäre es mit der temporären PIN trotzdem möglich, die eID-PIN, welche sich der Ausweisinhaber merkt, zu schützen. Der CAT-M speichert dazu temporär diese Kombination und ändert die eID-PIN auf eine zufällige sechsstellige Ziffernfolge. Mit letzterer ist nun die eID-Nutzung an dem externen Terminal möglich. Nachdem diese abgeschlossen ist, ändert der CAT-M die eID-PIN wieder auf ihren ursprünglichen Wert.

Der Missbrauch des nPA durch das Terminal ist nur für den Zeitraum möglich, in dem die temporäre PIN aktiv ist. Ein Tracing über die eID-PIN des Nutzers wird verhindert. Auch dem Shoulder Surfing würde entgegengewirkt. Einige der in „Bedrohungsszenarien“, S. 40 beschriebenen Probleme treffen aber weiterhin zu. Der Nutzer sollte sich in jedem Fall davon überzeugen, dass es sich um ein echtes und nicht manipuliertes Terminal handelt, wenn er den mobilen Chipkartenleser hier nicht nutzen kann. Dazu ist die Überprüfung des CHAT und der Zertifikatsbeschreibung des Terminals durch den Kartenemulationsmodus sinnvoll (siehe „Prüfungen als Kartenemulator“, S. 80).

3.3.8. Sichere Ausführungsumgebung

Viele Angriffe auf Desktop-Systeme betreffen auch heutige Smartphones. Desktop-Computer und Smartphones unterscheiden sich jedoch entscheidend bezüglich Nutzererwartungen und Infrastruktur. Von Smartphones wird Zuverlässigkeit erwartet, während Desktop-Benutzern des öfteren mit Fehlern umgehen müssen. Zudem sind Smartphones geschlossene Plattformen, die Modifikationen und direkte Zugriffe auf Hardware oder Kern-Software verhindern. Die meisten Nutzer akzeptieren diese Einschränkung. Daher können auf dem Smartphone umfangreiche Sicherheitsmaßnahmen umgesetzt werden.

Die Module eines Basis-, Standard- und Komfortlesers werden von Prüfstellen auf funktionale Konformität zu den Modulen der Chipkartenleser geprüft.[66] Bietet der CAT-M ebenso wie ein Standard- und Komfortleser auch sicherheitsrelevante Module (z. B. PIN-Pad mit PACE-Unterstützung, Display, Firmware-Update) an, wird auch eine Common-Criteria-Zertifizierung notwendig.[67] Die Zertifizierung soll mindestens Stufe EAL3 (EAL3+, falls die QES möglich ist) betragen und umfasst folgende Mindestsicherheitsanforderungen:[68]

- Sicherer Umgang mit eingegebenen PINs
- Sichere Implementierung von PACE (z. B. Schlüsselerzeugung)
- Einhaltung der Filterregeln
- Authentische Anzeige
- Manipulationssicherheit
- Integritätssicherung und -prüfung der Firmware

Außerdem ist die Abgrenzung zu anderen Anwendungen oder Sicherheitsmodi ebenfalls Bestandteil der Evaluierung. Die bloße funktionale Umsetzung z. B. der PACE-Funktion oder

[66] BSI TR-03119 1.2, S. 23–25.
[67] BSI TR-03119 1.2, S. 23.
[68] BSI TR-03119 1.2, S. 37.

des Firmware-Updates genügt nicht für eine Zertifizierung. Zusätzlich muss gezeigt werden, dass die implementierten Verfahren sicher sind.

Aufgrund der begrenzten Einsatzmöglichkeiten eines Standard- oder Komfortlesers ist die Anzahl und Komplexität der eingesetzten Komponenten überschaubar und überprüfbar. Beim mobilen Chipkartenleser verhält es sich anders. Der Erfolg von Smartphones beruht zum Teil darauf, dass Drittanbieter oder sogar Anwender selbst Applikationen dafür entwickeln können. Um dieser Anforderung Rechnung zu tragen, muss der CAT-M Mechanismen umsetzen, um diese potenziell unsicheren Anwendungen von der nPA-Funktionalität abzugrenzen. Um festzustellen, welche Maßnahmen dabei zur Anwendung kommen sollten, wäre die reduzierte und klare Sprache eines Protection Profiles für alle Kartenleser (nicht nur dem CAT-M) sinnvoll. Für die Electronic Health Card Terminals zur Nutzung der elektronischen Gesundheitskarte wurde beispielsweise ein solches Dokument bereits erstellt.[69]

Aufgrund der kurzen Produktzyklen moderner Smartphones könnten aufwendige Zertifizierungen gescheut werden. Innerhalb eines Schutzprofils ließen sich jedoch die Anforderungen an Chipkartenleser für den nPA so fixieren, dass auch eine Prüfung relativ schnell durchzuführen ist.

Unabhängig davon, wie eine isolierte Ausführungsumgebung implementiert ist, sollten folgende Aspekte bei dem Schutz der nPA-Funktionen realisiert werden:[70]

- Schutz der Software vor externer Beeinträchtigung
- Überwachung von Daten und Berechnungen isolierter Anwendungen
- Sichere Kommunikation zwischen Programmen unterschiedlicher Ausführungsumgebungen
- Vertrauenswürdiger Kanal zwischen Ein- oder Ausgabegeräten und den isolierten Programmen

Entsprechende Umsetzungen sind so vielfältig wie komplex. Forschung fand zunächst im Bereich sicherer Hardware-Komponenten statt, um einen sicheren Boot-Prozess zu realisieren.[71] Die Execute-Only-Memory-Architektur (XOM) setzt beispielsweise auf die Manipulationssicherheit der CPU, die mit einem asymmetrischen Schlüsselpaar ausgestattet ist. Vertrauenswürdige Anwendungen (oder Betriebssysteme) werden vom Hersteller speziell für eine CPU kryptographisch aufbereitet, so dass Vertraulichkeit und Integrität der Programme und deren Daten gewährleistet wird.[72] Ein Programm oder dessen Daten werden lediglich im Cache der CPU entschlüsselt, so dass weder die Vertrauenswürdigkeit des Hauptspeichers noch des Betriebssystems vorausgesetzt wird.[73] Dazu gibt es jedoch keinen in Hardware implementierten Prototypen.[74]

Andere frühe Architekturen verwenden ein hierarchisches Modell (z. B. AEGIS).[75] Die erste Stufe besteht aus vertrauenswürdigen Hardwarekomponenten, die gegen Manipulation geschützt sind und über Mittel der Integritätsprüfung der nächsten Stufe verfügt. Im einfachsten

[69] BSI-CC-PP-0032.
[70] Gallery und Mitchell 2007, S. 298.
[71] Müller 2008, S. 92; Gallery und Mitchell 2007, S. 288.
[72] Barrett 2005, S. 92–99.
[73] Gallery 2007, S. 573–574.
[74] Barrett 2005, S. 92.
[75] Barrett 2005, S. 85; Müller 2008, S. 91–92.

Fall kann dadurch sichergestellt werden, dass ein nicht manipuliertes vertrauenswürdiges Betriebssystem geladen wird. Dieses nutzt wiederum eigene Mittel zur Integritätsprüfung und zum Schutz vertrauenswürdiger Anwendungen, der dritten Stufe in diesem Beispiel.

Der hierarchische Ansatz wurde beim *Trusted Computing* weiter verfolgt.[76] Ziel von Trusted Computing ist die Prüfung des Systemzustands (lokal und von einer entfernten Instanz) und außerdem der Schutz von Daten, so dass sie nur in definierten Systemzuständen verfügbar sind.[77] Ein Trusted Platform Module (TPM) ist ein spezieller Chip, der diese Operationen auf Hardware-Ebene bereitstellt, um einen sicheren Boot-Vorgang zu ermöglichen.[78] Es dient als eine Art Vertrauensanker, auf dem die weiteren Prozesse aufgebaut werden. Das geladene Betriebssystem, aber auch der Boot-Loader selbst müssen so konfiguriert sein, dass die Sicherungsmechanismen des TPM auch genutzt werden. Das Betriebssystem muss Prüfung und Isolierung der Anwendungen implementieren. TPMs sind in einigen Notebooks und PCs verbaut. Sie können unter Windows für die Festplattenverschlüsselung Bitlocker benutzt werden. Für Unix-artige Betriebssysteme existiert Unterstützung durch verschiedene quelloffene Software, insbesondere TrustedGRUB[79] und TrouSerS.[80] Zudem gibt es quelloffene Software-Architekturen zur vollständigen Unterstützung von Trusted Computing inklusive sicherer Beispielanwendungen.[81]

Der Stand praktischer und theoretischer Betrachtung zum Trusted Computing zeigt, dass sich mit dieser Technologie am ehesten eine Ausführungsumgebung für sicherheitskritische Anwendungen umsetzen lässt. Als Alternativen werden gelegentlich noch Intel Trusted Execution Technology (TXT) oder AMD Presidio Technology benannt.[82] Diese verwenden CPU-Speicher- und I/O-Virtualisierung in Verbindung mit einem TPM. Es handelt sich dabei also nicht um vollständig verschiedene Konzepte, sondern ist lediglich in einigen Bereichen anders ausgeprägt als beim Trusted Computing vorgesehen.

Betrachtet man mobile Geräte, gibt es ebenfalls eine Reihe von Veröffentlichungen, die sich mit Trusted Computing beschäftigen.[83] Die Trusted Computing Group hat ein Äquivalent zum TPM für den mobilen Bereich spezifiziert, das Mobile Trusted Module (MTM). Es wird erwartet, dass solche Module zukünftig in mobile Geräte integriert werden.[84] Neben einer sicheren Ausführungsumgebung ermöglicht Trusted Computing auf mobilen Geräten Anwendungen zum sicheren Software- oder Medien-Download, den Schutz von Identitätsinformationen und

[76] Gallery und Mitchell 2007, S. 286.

[77] Gallery und Mitchell 2007, S. 285.

[78] Gallery und Mitchell 2007, S. 288–290.

[79] `http://sourceforge.net/projects/trustedgrub/`

[80] `http://trousers.sourceforge.net/`

[81] Beispielsweise Open Trusted Computing (OpenTC, `http://www.openct.net`) und European Multilaterally Secure Computing Base (EMSCB, `http://www.emscb.com`) bieten als europäisches bzw. deutsches Projekt komplette Systeme zur Unterstützung von Trusted Computing. Basierend auf dem Perseus-Projekt (`http://www.perseus-os.org`) wird der Bootvorgang mit TrustedGRUB, und Prozessisolierung per L4-Mikrokern oder per Xen (`http://www.xen.org/`) umgesetzt (Gallery und Mitchell 2007, S. 287).

[82] Müller 2008, S. 65–69.

[83] Acıicmez u. a. 2008; Buchmann u. a. 2010; Ekberg und Kylänpää 2007; Gallery 2007; Gallery und Mitchell 2007.

[84] Gallery und Mitchell 2007, S. 299.

mobiles Bezahlen oder Ticketing.[85] In Hardware integrierte MTMs sind in keinem aktuellen Smartphone verbaut, auch wenn bereits 2008 ein Prototyp präsentiert wurde.[86] In der Forschung werden derzeit verschiedene Emulatoren von MTMs verwendet,[87] um hier Trusted Computing umzusetzen.

In Selhorst u. a. 2010 wurde die Sicherheitsplattform Turaya[88] auf ein mobiles Gerät portiert und Demo-Anwendungen, wie z. B. der vertrauenswürdige SMS-Versand, realisiert. Die Turaya-Architektur besteht aus Hardware-, Hypervisor- und Trusted-Software-Schicht.[89] Die Hypervisor-Schicht enthält Gerätetreiber und setzt als Virtual Machine Monitor die systemweiten Sicherheitsrichtlinien durch. Die Trusted-Software-Schicht bietet abstrakte Sicherheitsdienste wie etwa eine sichere Anwenderschnittstelle (Trusted GUI). Der Speichermanager sichert die Integrität, Vertraulichkeit, Verfügbarkeit und Aktualität von Daten und erzwingt eine strikte Isolierung einzelner „Compartments". Wegen den Beschränkungen mobiler Geräte kommt bei MoTrust ein MTM-Emulator statt eines Hardware-Moduls zum Einsatz.

Dass Virtualisierung mit mindestens zwei Gastsystemen auf einem Smartphone möglich ist, wurde auch von VMware demonstriert.[90] Im angestrebten Anwendungsszenario sollen zwei Android-Systeme parallel laufen, um einen „Unternehmensbereich" und einen „Freizeit-Bereich" zu realisieren. Im Unternehmensbereich lassen sich die Vorgaben zur Nutzung des Smartphones definieren, z. B. das Verbieten von Copy & Paste zwischen beiden Systemen oder der Zugriff auf die Kamera, GPS, Bluetooth und ähnlichem. Der Freizeit-Bereich bleibt aber für den Anwender im Rahmen dieser Vorgaben frei konfigurierbar. Insbesondere ist auch die Installation von Zusatzsoftware für den Freizeitbereich (in den vom Unternehmensbereich vorgegebenen Richtlinien) möglich.

Zur Software-Isolierung nutzen andere Autoren auch Betriebssystem-Richtlinien, z. B. SELinux.[91] Obwohl SELinux als umständlich beschrieben wird, sichert es doch auf der Ebene des Betriebssystemkerns effektiv die Zugriffsrechte von Anwendungen.[92] Die Zugriffsbeschränkung mit SELinux gilt als Ressourcen-sparend, so dass sie sich besonders für mobile Geräte eignet.[93] Zusätzlich zu diesem „Sandboxing" können auch statische oder dynamische Software-Analysen zur Überprüfung von Anwendungen genutzt werden.[94] Die heute üblichen zentralen Strukturen zum Bezug von Drittanbieter-Software würden sich gut für diese Art der Analyse eigenen.

Es lässt sich zusammenfassen, dass die Leistungsfähigkeit von modernen Smartphones genügt, um alle bisher bekannten Software-basierten Technologien zur Prozess-Isolierung[95] umzusetzen:

[85] Gallery und Mitchell 2007, S. 299–300.
[86] Acıicmez u. a. 2008.
[87] Ekberg und Kylänpää 2007; Nauman u. a. 2010; Selhorst u. a. 2010.
[88] `http://www.emscb.de/content/pages/About-Turaya-de.htm`
[89] Selhorst, Stüble und Teerkorn 2008, S. 55–57.
[90] Labs 2011.
[91] Acıicmez u. a. 2008; Muthukumaran u. a. 2008; Nauman u. a. 2010; Zhang, Aciiçmez und Seifert 2009.
[92] Barrett 2005, S. 116–119.
[93] Zhang, Aciiçmez und Seifert 2009, S. 8, 11.
[94] „An Android Application Sandbox System for Suspicious Software Detection".
[95] Gallery und Mitchell 2007, S. 291–292.

- OS-Hosted Virtual Machine Monitor (VMM)[96]
- Eigenständiger VMM[97]
- Para-Virtualisierung[98]
- Richtlinien-basierte Isolierung[99]

Zu den erprobten mobilen Plattformen gehören Android,[100] Openmoko[101] und andere Linux-basierte Systeme.[102]

Dabei kann lediglich Software-Angriffen vorgebeugt werden, weil ein manipulationssicherer Vertrauensanker, wie z. B. ein Hardware-basiertes MTM, auf mobilen Plattformen derzeit nicht verfügbar ist. Zwar bieten andere Hardware-Komponenten in mobilen Geräten, wie etwa die SIM/USIM oder das Secure Element eines NFC-Moduls, auch sicherheitsspezifische Operationen an. Aber sie können keine Aussagen über die Integrität des aufrufenden Systems machen. Damit ist Trusted Computing derzeit die einzige brauchbare Lösung zur Realisierung einer sicheren Ausführungsumgebung für den CAT-M.

Mit der fehlenden Verankerung in sicherer Hardware wird einem mobilen Chipkartenleser wohl eine dem Standardleser äquivalente Zertifizierung versagt bleiben. Die Anwendbarkeit von Software-Isolierung für Mobilgeräte zeigt jedoch, dass der CAT-M die meisten einfach durchzuführenden Angriffe abwehren kann. Der CAT-M kann damit die elektronische Nutzung des nPA im Vergleich zum Basisleser wesentlich sicherer gestalten.

Um diese Innovation zu erreichen, könnte von einer Zertifizierung eines Smartphones als CAT-M mit sicherer Ausführungsumgebung abgesehen werden. Von den umfangreichen Funktionen und Prüfungen des CAT-M würden – auch ohne Zertifizierung – alle Nutzer profitieren, deren Smartphone frei von Schadsoftware ist.

[96] Labs 2011.
[97] Selhorst u. a. 2010.
[98] Hwang u. a. 2008; Samsung Electronics Co, Ltd. 2008.
[99] Nauman u. a. 2010; Zhang, Aciiçmez und Seifert 2009.
[100] Nauman u. a. 2010.
[101] Muthukumaran u. a. 2008.
[102] Acıicmez u. a. 2008; Selhorst u. a. 2010.

4. Umsetzung eines mobilen Kartenlesers mit PACE-Funktion

Die konzeptionelle Erarbeitung des mobilen Chipkartenlesers fand im letzten Kapitel statt. Ziel dieses Kapitels ist die Realisierung wesentlicher Aspekte dieser Leserklasse. Dafür soll ein existierendes Smartphone zur Verwendung mit dem nPA erweitert werden.[1] Dabei sollen nicht nur Anforderungen nach BSI TR-03119 1.2 umgesetzt werden.[2] Außerdem werden die für den mobilen Chipkartenleser entwickelten Module berücksichtigt.[3] Zudem muss bei aller Universalität, die modernen Smartphones zugesprochen wird, beachtet werden, dass die zugreifbaren Schnittstellen und Ressourcen auf einem solchen Gerät sehr stark begrenzt sind.

Die Implementierung sicherheitskritischer Anwendungen birgt viele Stolpersteine. Auch wenn ein System theoretisch erwiesenermaßen sicher ist, kann dessen praktische Umsetzung Lücken haben.[4] Wenngleich die einzelnen Komponenten für den mobilen Chipkartenleser durchaus unter Aspekten der Systemsicherheit gestaltet werden sollen, erhebt diese Arbeit nicht den Anspruch eine zertifizierbare Lösung umzusetzen. Vielmehr soll die Realisierbarkeit eines mobilen Kartenlesers gezeigt werden, bei der die einzelnen Bestandteile separat voneinander entwickelt und durch klare und einfache Schnittstellen voneinander getrennt sind. Bei der enstehenden Software soll Transparenz durch die Verwendung einer quelloffenen Lizenz erreicht werden. Dadurch kann sie von weiteren Experten überprüft und gegebenenfalls bezüglich etwaiger Schwachstellen angepasst werden. Bei der Realisierung soll nach Möglichkeit bestehende Software verwendet oder erweitert werden, da diese i. d. R. einen Teil des Peer-Review-Prozesses bereits durchlaufen hat. Der Austausch einzelner Komponenten ist durch die angestrebte Modularität ebenfalls möglich.

Die Implementierungen einzelner Teile sollen auf verschiedenen Systemen lauffähig sein, so dass sie auch in anderen Kontexten eingesetzt werden können. Aber die Anforderungen zur Umsetzung des mobilen Kartenlesers sind recht restriktiv. Ohne Unterstützung der jeweiligen Hersteller ist sie nur auf sehr wenigen Smartphones überhaupt möglich. Im Speziellen erfordert der nPA die Möglichkeit zum Versenden von APDUs mit erweitertem Längenfeld[5] („extended length APDU“[6]). Keines der aktuellen Smartphones ist offiziell dafür ausgelegt, aber es gibt

[1] Der inverse Ansatz, die Integration eines Mobiltelefons in einen Chipkartenleser ist freilich kaum möglich. Die geschlossene und funktional reduzierte Plattform der Chipkartenleser verhindert eine solche Umsetzung.

[2] Siehe „Lesegeräteklassen nach BSI TR-03119 1.2“, S. 15.

[3] Siehe „Neue und angepasste Module für BSI TR-03119 1.2“, S. 71.

[4] Ferguson und Schneier 2003, S. 129–135. Korrekte theoretische Beweise basieren lediglich auf einem Modell der Realität und können nie die Abwesenheit von Funktionalität zeigen. Damit kann ein Beweis nie die vollständige Sicherheit zeigen.

[5] BSI TR-03110 2.05, S. 72.

[6] ISO/IEC 7816-4.

wenige, die dennoch mit diesen APDUs umgehen können.[7] Für die demonstrative Umsetzung eines mobilen Kartenlesers mit PACE-Funktion werden diese Probleme umgangen, indem Komponenten eines externen Chipkartenleser in das Smartphone integriert werden, um die eigentliche Kommunikation zum nPA zu ermöglichen.

Die Nutzung des Smartphones als USB-Gerät an einem externen Computer ist außerdem nur im USB Gadget Modus möglich. Er sorgt dafür, dass das Smartphone *als* USB-Gerät erkannt wird (im Gegensatz zum USB Host Modus, in dem USB Geräte *vom* Smartphone erkannt werden). Der Gadget Modus wird nur von wenigen Smartphones zur beliebigen Nutzung angeboten. Oft verhindern bereits die vorgegebenen Schnittstellen einen solchen Zugriff. So bieten z. B. Cocoa (iPhone OS) oder Java (Android) keine Möglichkeit, den USB-Modus umzuschalten, geschweige denn die USB-Endpunkte zu kontrollieren. Dies bleibt auf entsprechenden Systemen zumeist den Herstellern vorbehalten.

Der Neo FreeRunner von Openmoko wurde als Testplattform für die Komponenten des mobilen Chipkartenlesers ausgewählt. Der Neo FreeRunner erfüllt mit einem angeschlossenen drahtlosen Chipkartenleser alle grundlegenden Anforderungen. Das externe NFC-Lesegerät wurde per USB angeschlossen und in einer Gehäuseextension „versteckt“ (siehe Abbildung 4.1). Der Neo FreeRunner verfügt über zwei separate USB-Anschlüsse,[8] die in voneinander unabhängigen Modi betrieben werden können. Alternativ wäre es auch möglich, einen NFC-Chip samt Antenne per SPI anzuschließen.[9]

Weil der Neo FreeRunner einen herkömmlichen ARM-Prozessor besitzt, konnten neben den speziell für die Openmoko-Familie konzipierten Distributionen auch einige andere populäre Betriebssysteme auf die Plattform portiert werden.[10] Für die Testplattform wird SHR genutzt.[11] Sie verwendet den Linux Kernel sowie Openembedded[12] und bietet damit ein voll ausgestattetes Linux-Betriebssystem. Die verwendeten Programmschnittstellen sind nahezu frei wählbar. Die einzelnen Softwarekomponenten für den mobilen Chipkartenleser können daher ohne spezielle Anpassungen für eine bestimmte Zielplattform auskommen.

4.1. Kryptographische Umsetzung: OpenPACE

Eine der größten Herausforderungen ist die Umsetzung der kryptographischen Operationen der EAC. Es werden nicht nur eine Reihe grundlegender kryptographischer Primitiven benötigt (siehe Tabelle 4.1). Außerdem handelt es sich bei PACE, TA und CA um völlig neue Protokolle. Für Berechtigungszertifikate wurde ein neuer Dokumententyp, das kartenverifizierbare Zertifi-

[7] Horsch 2009, S. 89.

[8] Der Neo FreeRunner führt von den beiden USB-Anschlüssen lediglich einen als Mini-USB nach außen. Der zweite Anschluss wird intern für das Bluetooth-Modul verwendet (Openmoko Wiki 2011, Neo FreeRunner Hardware).

[9] Der Neo FreeRunner besitzt drei nutzbare SPI-Anschlüsse, von denen lediglich zwei belegt sind (Openmoko Wiki 2011, Neo FreeRunner Hardware).

[10] Zu den von Openmoko-Telefonen unterstützten Betriebssystemen gehören unter anderem Android, Debian, Gentoo, OpenWrt, Slackware (Openmoko Wiki 2011, Distributions).

[11] Openmoko Wiki 2011, Distributions.

[12] Openmoko Wiki 2011, SHR.

Abbildung 4.1.: Openmoko Neo FreeRunner mit „integriertem“ drahtlosen Kartenleser und -emulator. Aufgrund von Hardwarebeschränkungen kann der touchatag (als Emulator oder Leser) lediglich PACE durchführen (vgl. „Smartphone-Gerätetreiber“, S. 97; „nPA-Kartenemulator“, S. 115). Für einen vollständigen EAC-Durchlauf wird daher auf Komponenten eines Basislesers zurückgegriffen.

	Algorithmus	PACE	TA	CA	RI
Hashfunktionen	SHA-1	✓	○	✓	○
	SHA-224		✓		○
	SHA-256	○	✓	○	○
	SHA-384		○		○
	SHA-512		○		○
Schlüsselvereinbarung	DH	○		○	○
	ECDH	✓		✓	○
Signaturverfahren	RSA		○		
	ECDSA		✓		
MAC-Verfahren	CMAC	✓	✓	✓	
	Retail-MAC	○	○	○	
Verschlüsselungsfunkionen	AES-128	✓	✓	✓	
	AES-192	○	○	○	
	AES-256	○	○	○	
	3DES	○	○	○	

Tabelle 4.1.: Kryptographische Primitiven für den neuen Personalausweis. Unterschieden wird hier zwischen den derzeit verwendeten (✓) und den nach BSI TR-03110 2.05 weiterhin spezifizierten Verfahren (○). Der Stand der praktischen Nutzung wurde über die Nutzung verschiedener bekannter Dienste geprüft. Informationen, welche Verfahren bei der RI aktuell verwendet werden, liegen dem Autor nicht vor.

kat (CV-Zertifikat), definiert.[13] Einen Überblick über mögliche kryptographische Bibliotheken zur Umsetzung der EAC wird in Appendix A gegeben.

Mit OpenPACE entschieden sich Morgner und Oepen zur Erweiterung von OpenSSL um die kryptographischen Protokolle für den nPA.[14] Mit der Umsetzung in der bekanntesten und am häufigsten genutzten quelloffenen kryptographischen Bibliothek wird unter anderem eine breitere Nutzung von PACE erhofft. Denn als generische Passwort-basierte Schlüsseletablierung kann es immer dort eingesetzt werden, wo ein vorab geteiltes (schwaches) Geheimnis zur gegenseitigen Authentisierung genutzt werden soll – unabhängig von der Nutzung des nPA.

So startete das Projekt 2009 lediglich mit der Unterstützung für das PACE-Protokoll. 2011 wurden außerdem TA und CA hinzugefügt. In der mit dieser Arbeit erscheinenden Version 0.7 wird auch RI ein Bestandteil von OpenPACE sein, welcher im Rahmen des IT Security Workshop 2011 an der Humboldt-Universität zu Berlin in Zusammenarbeit mit Paul Wilhelm ergänzt wurde.

OpenPACE wird und wurde regelmäßig aktualisiert und gepflegt. Es wurden nicht nur neue Funktionen hinzugefügt und Fehler beseitigt. Immer wieder führten neue OpenSSL-Versionen und Modifikationen der Spezifikationen zum nPA zu Anpassungen von OpenPACE.

[13] BSI TR-03110 2.05, S. 82–91.
[14] Morgner und Oepen 2012.

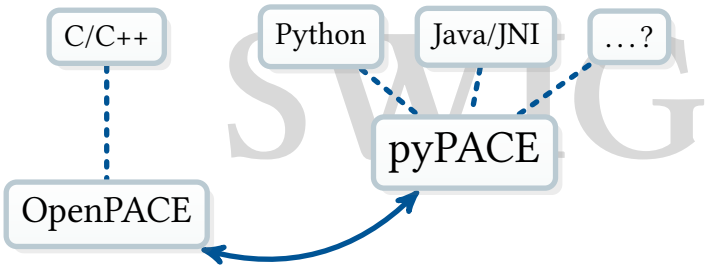


Abbildung 4.2.: pyPACE macht OpenPACE mittels SWIG in anderen Programmiersprachen verfügbar

OpenSSL unterstützt beispielsweise sehr viele, teils exotische Architekturen und Build-Systeme. Konzipiert als Patch für OpenSSL, muss OpenPACE daher Änderungen an recht flüchtigen, OpenSSL-spezifischen Dateien durchführen. Dies führte in der Vergangenheit oft dazu, dass OpenPACE für neu erscheinende OpenSSL-Versionen von Hand angepasst werden musste. Im Moment wird OpenSSL 1.0.1-beta1 unterstützt.

Die Änderungen Technischer Richtlinien führten zu inhaltlichen Änderungen von OpenPACE. Insbesondere BSI TR-03110 änderte sich von Version 2.01 zu aktuell 2.05 nicht nur in Details. In BSI TR-03110 2.02 wurde eine neue Berechnung des Authentisierungstokens vorgeschrieben.[15] In derselben Version der Technischen Richtlinie wurden ergänzende Prüfungen für die ephemeralen Domänenparameter für PACE hinzugefügt. Das „Integrated Mapping“ dieser Domänenparameter ist lediglich in BSI TR-03110 2.01 spezifiziert und wurde in den folgenden Versionen zugunsten eines Verweises auf ICAO Doc 9303 Pt. 3 Vol. 2 entfernt.[16] Mit BSI TR-03110 2.04 werden die Terminalzertifikate an die SSL/TLS-Zertifikate des Dienstanbieters gebunden. OpenPACE lässt sich bezüglich der zu verwendenden Version von BSI TR-03110 konfigurieren. Dadurch kann OpenPACE nicht nur mit den (veralteten) Testkarten, sondern auch mit echten Ausweisen genutzt werden. Die EAC wurde weiterhin mit den vom BSI vorgegebenen „Worked Examples“[17] auf Terminal- und Kartenseite validiert.[18]

Mit OpenPACE lässt sich jedes der Protokolle ganz unabhängig vom nPA oder anderen Chipkarten durchführen. Dennoch wurden mit der Zeit auch Operationen hinzugefügt, um einen einfacheren Umgang mit dem nPA zu erreichen. Die OpenPACE-Datenstrukturen können mit dem `EF.CardAccess` des nPA initialisiert werden, so dass alle relevanten Daten für EAC oder RI extrahiert werden. Auch CV-Zertifikate können gelesen und interpretiert werden. Dies ist im Besonderen zur Zertifikats- und Authentizitätsprüfung des Terminals notwendig.

Als Erweiterung von OpenSSL ist OpenPACE als C/C++ Bibliothek zu verwenden. Das

[15] Siehe „Password Authenticated Connection Establishment“, S. 10.

[16] Die Definition des Integrated Mapping in ICAO Doc 9303 Pt. 3 Vol. 2 ist im Wesentlichen jedoch identisch zu derjenigen in BSI TR-03110 2.01. Die OpenPACE-Implementierung des Integrated Mapping wurde aber nie einem ausführlichen Review bzgl. der Kompatibilität unterzogen.

[17] BSI 2011d.

[18] Bei der Überprüfung von OpenPACE sind dem Autor mehrere Fehler der „EAC Worked Examples“ aufgefallen, die zur Verbesserung von BSI 2011d dem BSI mitgeteilt wurden. Die Hinweise sollen in neuere Versionen des Dokuments eingehen.

Projekt pyPACE[19] macht OpenPACE durch den Einsatz von SWIG[20] auch in weiteren Programmiersprachen verfügbar (siehe Abbildung 4.2).[21] OpenPACE wurde sowohl unter Desktop Betriebssystemen wie Windows, Linux und Mac OS als auch mobilen Plattformen wie Openmoko und Android getestet.

4.1.1. Aufbau und Funktionsweise

OpenSSL ist in verschiedene Module gegliedert, die sich im Unterordner `crypto` befinden. Um Konflikten im gemeinsam geteilten Namensraum von C aus dem Weg zu gehen, werden Typen, Methoden oder Definitionen entsprechend dem Modul mit einem Präfix versehen. OpenPACE erweitert OpenSSL um das Module `EAC`, welches selbst untergliedert ist in `PACE`, `TA`, `CA` und `RI`. Sie dienen zur Umsetzung der einzelnen Protokollschritte der EAC und RI.

Teil dieser Module sind kontextsensitive Strukturen, die die notwendigen Daten, Operationen und Konfigurationsparameter der jeweiligen Schritte enthalten. Streng genommen handelt es sich hier aber nicht um eine aus der Objektorientierung bekannte Datenkapselung, da die lokalen Elemente nicht vor externem Zugriff geschützt sind und sogar durch die Header-Dateien nach außen propagiert werden. In C gäbe es mit untypisierten Zeigern (`void *`) zwar die Möglichkeit, den Zugriff auf untergeordnete Objekte zu erschweren. Darauf wurde jedoch bewusst verzichtet, um jedes dieser Objekte auch nach individuellen Wünschen anpassen zu können. Insbesondere ist es damit möglich, kryptographische Primitiven und Mapping-Verfahren für PACE, TA und CA zu nutzen, die nicht in BSI TR-03110 2.05 berücksichtigt sind.

Neben der Flexibilität und der Modularisierung wurden schmale und klare Schnittstellen definiert, um die Komplexität einzelner Module zu verbergen. Jedem Protokollschritt von PACE, TA und CA ist eine Methode in OpenPACE zugeordnet (ein Schlüsselaustausch erfolgt notwendigerweise in zwei Teilschritten). Entsprechend dem Protokollschritt ist durch das Präfix zum einen intuitiv klar, in welcher Reihenfolge die Funktionen verwendet werden sollen. Zum anderen erfahren sie durch die Vorgaben der BSI TR-03110 2.05 eine implementierungsunabhängige Beschreibung.

Genaues Wissen über die interne Funktionsweise einzelner Teile von OpenPACE ist für den Nutzer der Bibliothek nicht vonnöten. Nicht einmal die Elemente einzelner (Kontext-) Strukturen müssen von ihm „angefasst“ werden. Für die meisten Operationen existieren Funktionen, über deren Eingangsparameter die internen Objekte modifiziert werden. Die Komplexität von OpenPACE wird nach außen also verborgen.

Eine besondere Schwierigkeit von nicht-typsicheren Programmiersprachen wie C ist die korrekte Behandlung von Zeigern. Natürlicherweise gibt es auch in OpenSSL das Risiko zu Pufferüberläufen, Speicherzugriffsverletzungen oder Speicherlecks. In OpenSSL wie auch in OpenPACE wird dem Nutzer das eigenhändige Manipulieren von Datenobjekten erspart. Neben den oben erwähnten Methoden, die diese Aufgabe erfüllen, gehören auch das Erzeugen und Löschen entsprechender Objekte zur Minimierung der außen sichtbaren Komplexität.

[19] `http://sourceforge.net/projects/pypace/`

[20] `http://www.swig.org/`

[21] Bisher wurden Python Bindings sowie das Java Native Interface implementiert. Die Ergänzung weiterer Back-Ends ist mit SWIG einfach zu realisieren, so dass die Verwendung von OpenPACE potenziell auch in vielen weiteren Programmiersprachen möglich ist

So erzeugt z. B. `EAC_CTX_new` einen neuen `EAC`-Kontext und legt zugleich alle notwendigen Objekte an. Entsprechend gibt `EAC_CTX_clear_free` die reservierten Speicherbereiche frei und bereinigt sensitive Daten. Innerhalb von OpenPACE wird dieses Konzept auf die einzelnen Module übertragen, so dass beispielsweise aus dem EAC-Kontext nicht direkt in den PACE-Kontext eingegriffen wird. Stattdessen werden entsprechende Interfaces genutzt.

4.1.2. Besonderheiten und Schwierigkeiten

OpenPACE wurde mit Hinblick auf die Nutzung in sicherheitskritischen Kontexten konzipiert. Eingangsdaten – insbesondere Zeiger – werden intern stets überprüft. Sensitive Speicherbereiche werden, bevor sie freigegeben werden, per `OpenSSL_cleanse` überschrieben. Dennoch muss beachtet werden, dass OpenPACE letztlich immer in einem Ausführungskontext arbeitet, der kaum beeinflusst werden kann. In einem kompromittierten System können sensible Speicherbereiche ausgelesen oder funktionale Änderungen durchgeführt werden. Um Geheimnisse zu schützen und Prüfungen wirksam zu machen, ist eine sichere Ausführungsumbebung notwendig. Daher überlässt OpenSSL beispielsweise das Memory-Locking dem Betriebssystem, ohne dies selbst explizit auszulösen.

Die beschriebene Erweiterbarkeit von OpenPACE wurde durch die Nutzung von Rückruffunktion („callback functions“) und kontextbezogenen kryptographischen Objekten realisiert. Dadurch sind bei OpenPACE auch PACE, TA und CA mit Verfahren möglich, die in BSI TR-03110 2.05 derzeit nicht spezifiziert sind. Zwar können die Protokolle durch ein `EF.CardAccess` eines nPA nach den Vorgaben der Technischen Richtlinie initialisiert werden. Aber über die Manipulation der Daten des `PACE_CTX` bzw. die Initialisierung per `EAC_CTX_init_ta` und `EAC_CTX_init_ca` mit einem privaten oder öffentlichen Schlüssel[22] sind für die Protokolle alle Kombinationen aus Hashfunktionen, Verschlüsselungs- und Schlüsselaustauschverfahren von OpenSSL nutzbar. Insbesondere können auch kryptographische Hardwaremodule eingebunden werden.

Die Umsetzung von OpenPACE war außerordentlich umfangreich und zeigte an mehreren Stellen Grenzen und Schwächen von OpenSSL. So ist es beispielsweise nicht möglich, in einem OpenSSL-Modul andere elliptische Kurven zu nutzen als die vom EC-Modul vorgegebenen. Zur Integration der Brainpool-Kurven muss OpenPACE daher in dieses Kernmodul von OpenSSL eingreifen.

Der ASN.1-Parser von OpenSSL ist sehr mächtig und ermöglicht eine gute Aufspaltung der einzelnen ASN.1-Objekte in semantische Einheiten. Der hohe Funktionsumfang schlägt sich jedoch direkt in einer umständlichen Bedienung nieder. So muss etwa derzeit in OpenPACE das `SET OF SecurityInfos`[23] des `EF.CardAccess` manuell in seine Bestandteile zerlegt werden und wird erst dann an den Parser übergeben. Auch das `Discretionary Data Template`, mit dem beispielsweise die CV-Zertifikatserweiterungen[24] kodiert sind, konnte nur als spezielle `SEQUENCE`, nicht aber als `ANY DEFINED BY`, umgesetzt werden.

[22] Neben anderen Formaten unterstützt OpenSSL PKCS#8 zur Schlüsselspeicherung. Öffentliche Schlüssel werden für OpenPACE bei der TA in einem CV-Zertifikat kodiert.

[23] BSI TR-03110 2.05, S. 40.

[24] BSI TR-03110 2.05, S. 86.

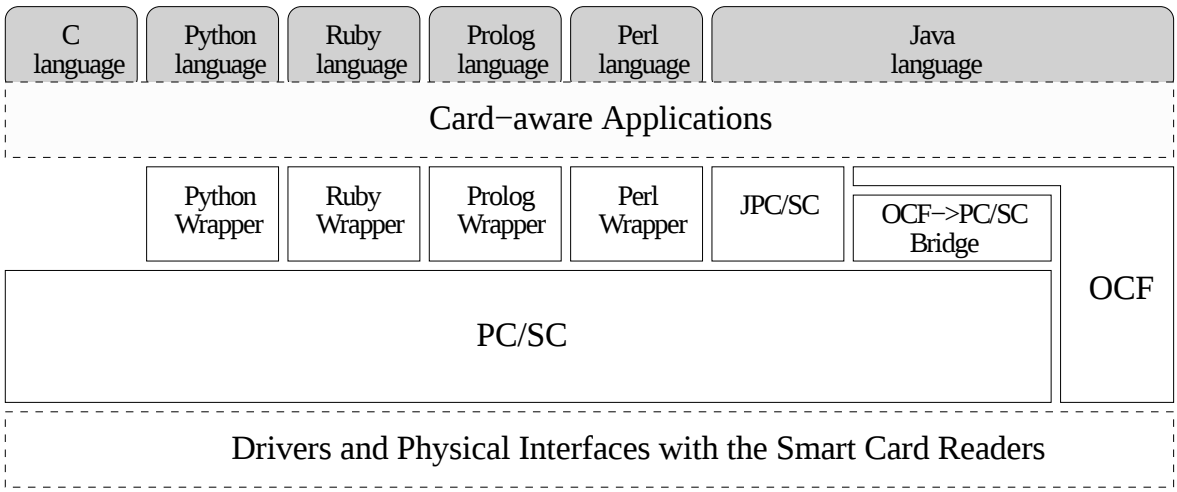


Abbildung 4.3.: Die PC/SC-Middleware und das OpenCard-Framework (OCF) zum Zugriff auf Chipkartenleser (Sauveron 2008, S. 279)

Das größte Hindernis stellte jedoch die interne Funktionsweise von OpenSSL dar, die komplex und kaum dokumentiert ist. Sie konnte meist nur durch Analyse des Quelltextes nachvollzogen werden. Trotzdem konnten die kryptographischen Anforderungen der BSI TR-03110 2.05 in OpenPACE umgesetzt werden. Neben den verschiedenen Protokollen und Konfigurationen der EAC wird auch die Verifizierung einer CV-Zertifikatskette (samt Aktualitätsprüfung) und Zertifikatsbeschreibung unterstützt. Ebenfalls werden für einen EAC-Kontext nach erfolgreichem PACE- oder CA-Durchlauf Verschlüsselungs- und Authentisierungsmethoden angeboten, die für die anschließende Kommunikation im sicheren Kanal genutzt werden können.

4.2. Schnittstelle zum nPA als Chipkarte

Chipkartenleser gibt es in vielen Variationen und Formfaktoren mit unterschiedlichen Funktionen. Sie können in ein System integriert sein oder aber per PCMCIA, USB, Bluetooth und vielen weiteren Möglichkeiten angeschlossen werden.[25] Um Anwendungen in verschiedenen Betriebssystemen eine einheitliche Schnittstelle auf Chipkartenleser bzw. Chipkarten zu geben, wurden verschiedene Standards vorgeschlagen. PC/SC, die CT-API und das OpenCard-Framework sind die bekanntesten Vertreter. Sie werden in Appendix B vorgestellt.

Alle drei dieser Plattformen lassen sich auf modernen Betriebssystemen einsetzen. Das OpenCard-Framework ist Java-basiert und benötigt lediglich eine entsprechende Laufzeitumgebung. Die CT-API wurde zwar unabhängig von einer Programmiersprache spezifiziert,[26] wird jedoch meist über eine C/C++ Bibliothek angebunden.[27] Die PC/SC-Middleware wird bei modernen Windows-Systemen mitgeliefert. Unter Unix-artigen Systemen wird sie von

[25] Sauveron 2008, S. 278.
[26] Rankl und Effing 2008, S. 261.
[27] CT-API 1.1.1, S. 9–11.

dem freien Projekt PCSC lite[28] umgesetzt. Die Programmbibliotheken zu PC/SC können unter Windows und Unix-artigen Betriebssystemen durch verschiedene Wrapper in zahlreichen Programmiersprachen verwendet werden (siehe Abbildung 4.3).

Besonders im mobilen Bereich finden sich auch verschiedene Java APIs zur NFC-Funktion neuerer Smartphones. Obwohl es mit „JSR 257: Contactless Communication API“[29] eine standardisierte Java-Schnittstelle zu NFC gibt, ergänzt jeder Hersteller seine Laufzeitumgebung mit proprietären Paketen zur NFC-Nutzung – wenn überhaupt JSR 257 unterstützt wird. Die verschiedenen teilweise inkompatiblen Schnittstellen in Android,[30] Blackberry-[31] und Nokia-Betriebssystemen[32] zeigen, dass eine portable Applikation sicherlich nur mit hohem Aufwand realisierbar wäre. So nennt etwa auch Horsch eine Reihe von Änderungen, die bei der Portierung des nPA-Projekts „MONA“ vom Nokia 6212 auf die Android-Plattform realisiert werden müssten.[33]

Zusammenfassend kann festgestellt werden, dass sich PC/SC zum Ansprechen des nPA empfiehlt. Dies ist ein Standard, der aktiv genutzt und entwickelt wird, und wenige Einschränkungen bzgl. der Ausführungs- oder Programmierumgebung auferlegt. Denkt man an die Möglichkeiten zur Weiternutzung der Ergebnisse dieser Arbeit, ist eine weitere Abstraktionsebene als diejenige zu den Kartenlesegeräten sinnvoll. Beispielsweise könnte man auch die Eigenheiten bestimmter Chipkarten in eine Middleware aufnehmen, so dass auf Applikationsebene keine weiteren Behandlungsroutinen notwendig sind.

4.2.1. Smartphone-Gerätetreiber

Die Arbeit an dem CAT-M begann mit dem touchatag, der die NFC-Kommunikation sowohl in der Rolle einer Chipkarte als auch eines Chipkartenlesers bereitstellen kann. Die Funktionen des Geräts lassen sich aber in der Ausgangskonfiguration nur recht umständlich ansteuern. An einem Host-System wird der touchatag als CCID-Leser vom PC/SC-Dienst erkannt. Über die PC/SC-API kann man zunächst direkt die im touchatag verbaute SIM-Karte ansprechen. Ein spezielles Kommunikationsprotokoll ermöglicht über diese Schnittstelle aber auch die Nutzung der NFC-Funktionen.

Verschiedene freie Projekte haben dieses Protokoll umgesetzt[34] und können direkt NFC-Chipkarten ansprechen (oder eine eben solche emulieren). Keine dieser Bibliotheken integriert jedoch die Funktion als NFC-Chipkartenleser in die PC/SC-Middleware.

Zu diesem Zweck wurde aufbauend auf `libnfc` ein Gerätetreiber für PCSC Lite realisiert.[35] Dieser „IFD-Handler“ (`ifdnfc`) fungiert als Mittler zwischen Chipkartenapplikationen und `libnfc`. Das PC/SC-Interface wird von `ifdnfc` bereitgestellt, `libnfc` übernimmt die eigentliche Ansteuerung des touchatag als Chipkartenlesers.

[28] `http://pcsclite.alioth.debian.org/pcsclite.html`

[29] `http://www.jcp.org/en/jsr/detail?id=257`

[30] `http://www.blackberry.com/developers/docs/7.0.0api/net/rim/device/api/io/nfc/package-summary.html`

[31] `http://www.blackberry.com/developers/docs/7.0.0api/net/rim/device/api/io/nfc/package-frame.html`

[32] `http://library.developer.nokia.com/topic/Java_Developers_Library/GUID-D38F02EB-1628-489B-997D-EF723C16E3CF/overview-summary.html`

[33] Horsch 2011, S. 92–95.

[34] `libnfc` (`http://www.libnfc.org/`), `cyberflex-shell` (`https://github.com/henryk/cyberflex-shell/`)

[35] Morgner 2010.

Weil der touchatag von `libnfc` selbst aber auch über die PC/SC-Schnittstelle angesprochen wird, existiert eine Race Condition. In einigen Situationen blockiert der PC/SC-Dienst, wenn er auf Antworten von `ifdnfc` wartet. Setzt dieser aber wiederum ein Kommando an `libnfc` ab, können die PC/SC-Aufrufe nicht mehr abgearbeitet werden, der PC/SC-Dienst und `libnfc` befinden sich in einem Deadlock. Entsprechende Situationen wurden bei `ifdnfc` versucht so weit wie möglich zu vermeiden. Zudem ist es möglich, den Gerätetreiber während des Betriebs des PC/SC-Dienstes zu deaktivieren oder zu aktivieren.

Weil der touchatag aber keine APDUs mit erweitertem Längenfeld („extended length APDU“[36]) unterstützt, kann er für eine vollständige EAC mit dem nPA nicht genutzt werden (die CV-Zertifikate sind schlicht zu groß). Daher muss ein anderer drahtloser Chipkartenleser bei dem elektronischen Identitätsnachweis mittels des Smartphones zum Einsatz kommen.

Weil viele Chipkartenhersteller lediglich vorkompilierte Treiber zu x86-Architekturen anbieten, empfiehlt sich ein zu USB CCID konformes Gerät. Der entsprechende Gerätetreiber, `libccid`[37], lässt sich durch den vorhandenen Quellcode auch für die ARM-Architektur des Neo FreeRunner kompilieren. Auf dem Smartphone stellt nun der Reiner SCT cyberJack RFID basis die Kommunikation zum nPA Smartphone bereit. Der Leser passt ebenso wie der touchatag in die Gehäuseextension des Neo FreeRunner.

4.2.2. OpenSC für PACE-fähige Leser

OpenSC[38] bietet eine Abstraktion von den per PC/SC übertragenen Puffern zu generischen Chipkartenkommandos. Das Projekt fokussiert in erster Linie die Umsetzung des PKCS#11- und PKCS#15-Standards, um Operationen zur Authentisierung, Verschlüsselung und Signieren mittels Chipkarten oder USB Tokens bereitzustellen. Die enthaltene Bibliothek `libopensc` bietet aber auch generische Aufrufe z. B. zum Lesen/Schreiben von Dateien auf der Karte oder zur Durchführung einer PIN-Verifikation (ggf. am Leser), die von den Eigenheiten der genutzten Geräte abstrahieren.

Einerseits können von OpenSC sowohl die CT-API als auch PC/SC als Schnittstelle zum Kartenleser genutzt werden. Andererseits wird intern auch mit Chipkarten-Treibern gearbeitet, die die üblichen Chipkartenkommandos realisieren. Da der nPA konform zu ISO 7816 ist, ist dessen Verwendung in OpenSC unproblematisch. Ein entsprechender Chipkarten-Treiber ist in `libopensc` enthalten. Um aber auch PACE zur Nutzung des mobilen Chipkartenlesers durchführen zu können, müssen die Funktionen wesentlich erweitert werden.

Im Rahmen dieser Arbeit wurde in OpenSC die Fähigkeit ergänzt, APDUs mit erweitertem Längenfeld zu verarbeiten. Dieser Patch wurde in dem Haupt-Entwicklungszweig von OpenSC übernommen.[39] Auch die Möglichkeit zur Nutzung von PACE-fähigen Lesegeräten wurde umgesetzt und ist zur Integration in OpenSC vorgeschlagen.[40]

Prinzipiell ist es wünschenswert, in OpenSC auch die Nutzung eines nPAs mit dem Basisleser

[36] ISO/IEC 7816-4.

[37] Rousseau 2011.

[38] `http://www.opensc-project.org/`

[39] `http://www.opensc-project.org/opensc/changeset/5394/opensc/trunk/src/libopensc/apdu.c`

[40] `https://github.com/OpenSC/OpenSC/commit/1141ae520201c91b9859b0cd6cbb59f4ebfbd577`; `https://github.com/frankmorgner/OpenSC`

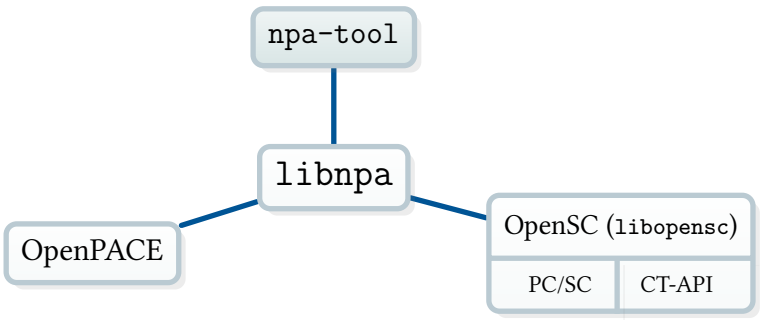


Abbildung 4.4.: Aufbau der nPA Smart Card Library (`libnpa`)

zu integrieren. Die erforderlichen Änderungen sind aber tiefgreifend, weshalb mit der nPA Smart Card Library eine eigenständige Entwicklung umgesetzt wurde (s. u.).

4.2.3. nPA Smart Card Library

Die nPA Smart Card Library (`libnpa`)[41] stellt eine Bibliothek bereit, die es Applikationen ermöglicht, PACE durchzuführen und im sicheren Kanal mit der Chipkarte zu kommunizieren. Dabei werden die von OpenPACE bereitgestellten kryptographischen Primitiven genutzt. OpenSC stellt die eigentliche Kommunikation zur Chipkarte zur Verfügung und integriert die PACE-Funktion eines Standard- oder Komfortlesers (s. o.).

Die C-Bibliothek `libnpa` bietet mit `EstablishPACEChannel` eine Möglichkeit, den SM-Kanal zu etablieren. Alle folgenden APDUs werden transparent für die Applikation mittels `sm_transmit_apdu` in dem sicheren Kanal an den nPA gesendet. Signatur und Name von `EstablishPACEChannel` sind BSI TR-03119 1.2, S. 34–36 entlehnt. Ein- und Ausgabedaten sind analog zu denen der PACE-Funktion eines Standard- oder Komfortlesers.[42]

Zur Umsetzung der nPA Smart Card Library war es nicht nur notwendig, für einen Protokolldurchlauf korrekt formatierte APDUs zusammenzusetzen. Es entstand auch eine generische Umsetzung von Secure Messaging (SM) nach ISO/IEC 7816-8. ISO/IEC 7816-8 macht lediglich Vorgaben zur Formatierung von APDUs, die in sicherheitsrelevanten Kontexten verwendet werden. Die eigentlichen kryptographischen Operationen oder die kartenspezifische Semantik bestimmter Befehle sind nicht Teil des Standards. Daher bezieht sich die SM-Implementierung von `libnpa` zunächst auch lediglich auf die Kodierung der Daten. Die kryptographischen Operationen werden über Rückruffunktionen („callback functions“) an kartenspezifische Routinen weitergegeben. Nach der Durchführung von PACE wird der SM-Kontext von `libnpa` mit Rückruffunktionen initialisiert, die die Ver- bzw. Entschlüsselung und Berechnung des MAC für den nPA übernehmen. Außerdem können in den Rückruffunktionen die Daten, die an den nPA gesendet werden sollen, inhaltlich geprüft werden. Umgesetzt wurde so etwa die passive Prüfung der TA unabhängig von dem nPA. Die vom Terminal gesendete Zertifikatskette wird

[41] Morgner und Oepen 2011.

[42] Wobei in BSI TR-03119 1.2 zur Spezifikation von `EstablishPACEChannel` die Besonderheiten von PC/SC berücksichtigt werden müssen, ist dies bei `libnpa` nicht der Fall. Die Elemente der Strukturen `establish_pace_channel_input` und `establish_pace_channel_output` können daher auf native C-Typen zurückgreifen.

Projekt/Arbeit	Kryptographische Bibliothek	Programmiersprache	Plattform	Zeit (in s)
Kampe 2010, S. 52	Bouncy Castle	Java ME	Nokia 6216 classic	41,1
Horsch 2011, S. 83	FlexiProvider mit Optimierungen	Java ME	Nokia 6216 classic	7,5
nPA Smart Card Library	OpenSSL mit OpenPACE	C	OpenMoko Freerunner*	1,4

Tabelle 4.2.: `libnpa` und OpenPACE im Vergleich zu anderen Implementierungen. Auf dem Openmoko Neo FreeRunner wird die Kommunikation zum nPA durch den Reiner SCT cyberJack RFID basis ermöglicht (*).

importiert, die Signatur geprüft und die Aktualität des Terminalzertifikats durch den Vergleich mit der Systemuhr sichergestellt.

Mit dem `npa-tool` liegt der nPA Smart Card Library bereits ein kommandozeilenbasiertes Programm bei, welches die gesamte Funktionalität von `libnpa` nutzbar macht. Einfache Operationen, wie z. B. das Ändern oder Entsperren der eID-PIN. Damit konnte bereits die temporäre eID-PIN realisiert werden.[43]

Auch das Versenden beliebiger APDUs vor oder nach PACE sind interaktiv oder durch eine vorgefertigte Datei mit dem `npa-tool` möglich. Damit ließe sich die Aktualisierung der Zeit auf dem nPA realisieren. Nachdem ein PACE-Kanal etabliert wurde, kann eine aktuelle Zertifikatskette mit hoheitlichen Zertifikaten, in mehreren `VERFIY CERTIFICATE`-Kommandos an den Ausweis übertragen werden. Dieser prüft die einzelnen CV-Zertifikate und aktualisiert seine innere Uhr (siehe Abbildung 4.6). Werden vom BSI entsprechende Zertifikatsketten bereitgestellt, kann mit dem `npa-tool` also leicht die Zeitaktualisierung umgesetzt werden.[44]

Alle Komponenten der nPA Smart Card Library wurden sowohl unter Linux als auch Windows getestet und stehen mit vorkompilierten Windows-Binaries auch unbedarften Anwendern zur Verfügung. Selbst auf einem mobilen System sind `libnpa` und OpenPACE – ohne nennenswerte Optimierungen – sehr performant (siehe Tabelle 4.2, Tabelle 2.6).

4.3. Vom Smartphone zum Chipkartenleser

Um einen Chipkartenleser am Host-System mit der PC/SC-Middleware anzusprechen, muss der Hersteller einen entsprechenden Gerätetreiber bereitstellen. Für Windows-Treiber wird die Bestätigung durch die Windows Hardware Quality Labs benötigt.[45] Um Zertifizierungsproblemen aus dem Weg zu gehen, wurde USB CCID geschaffen,[46] zu welchem Microsoft einen passenden Treiber mitliefert. Auch unter Unix-artigen Betriebssystemen wie Linux, Mac OS

[43] Siehe „Autarkes PIN-Management und temporäre PIN“, S. 83.
[44] Siehe „Zeitaktualisierungsdienst“, S. 82.
[45] Sauveron 2008, S. 286.
[46] Sauveron 2008, S. 286.

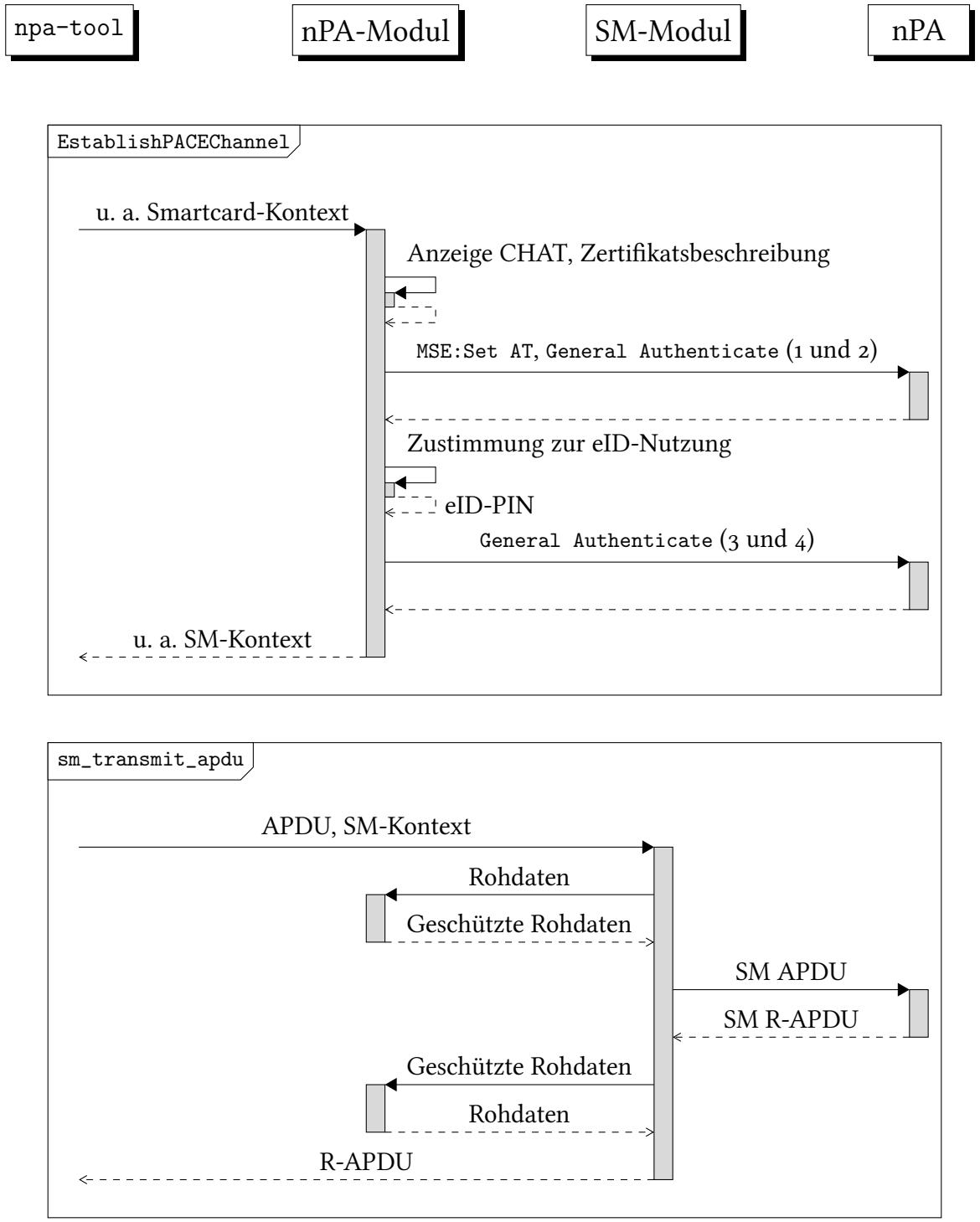


Abbildung 4.5.: Zusammenspiel von `npa-tool` und der libnpa-Module zum nPA und Secure Messaging (SM)

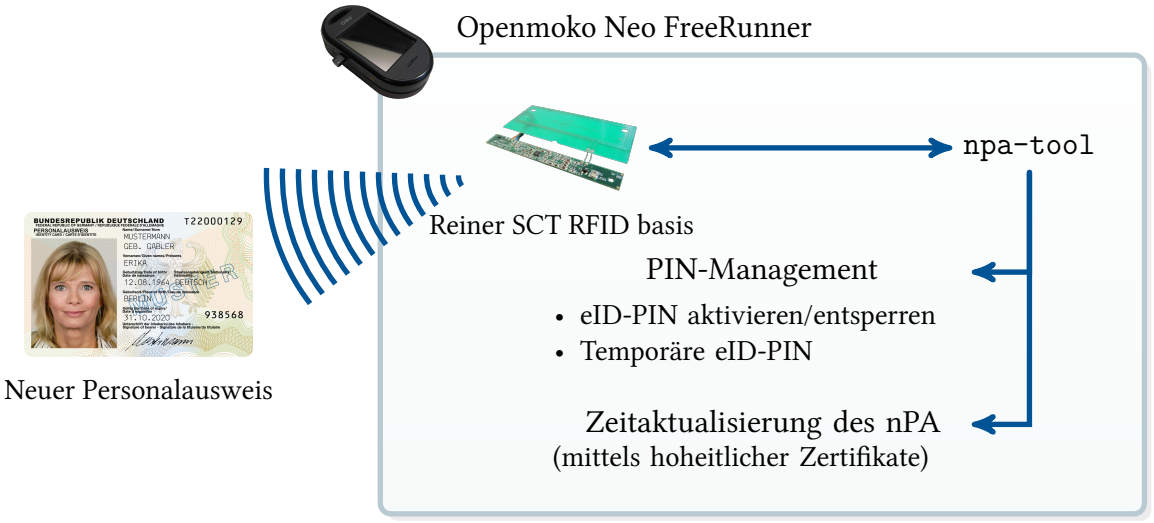


Abbildung 4.6.: Umsetzung des PIN-Management und der Zeitaktualisierung

oder Solaris ist der PC/SC-Treiber `libccid` teil der Distribution.

Neben diesen praktischen Aspekten, sprachen auch die betrachteten theoretischen Argumente für die Umsetzung von USB CCID beim mobilen Chipkartenleser.[47] Anwenderfreundlichkeit und ein erhöhtes Maß an Sicherheit durch Vermeidung von Fehlbedienung und Implementierungsfehlern wurden erläutert. Zudem gibt es schlicht kein weiteres standardisiertes Protokoll, welches ein Hardware-Interface zu einem Chipkartenleser spezifiziert. Hier soll nun gezeigt werden, wie auf dem Openmoko bzw. unter Linux im Allgemeinen eine solche Funktionalität umgesetzt werden kann.

Physikalisch bildet der USB eine Baumstruktur[48] (siehe Abbildung 4.7): Der USB-Host, bestehend aus Root-HUB und Host-Controller, bildet die Wurzel des Baumes. USB-Hubs dienen zur weiteren Verzweigung bzw. bilden neue Äste. Peripheriegeräte stellen die Blätter des Baumes dar. Logisch gesehen besteht nach der Konfigurationsphase eine sternenförmige Topologie, da vom USB-Host eine Punkt-zu-Punkt-Verbindung zu jedem Peripheriegerät existiert.[49] Es gibt stets genau einen Host. Jeglicher Datenverkehr auf dem Bus wird von diesem initiiert.

Mit USB On-the-Go (OTG) ist es nun auch möglich, dynamisch zu entscheiden, ob sich ein Gerät als Host oder Peripheriegerät verhält. Dies kann mittels des „Host Negotiation Protocol“ ausgehandelt werden, statt einmalig beim Design eines Geräts festgelegt zu werden.[50] Mit einem speziellen USB-Konnektor können damit auch Peer-to-Peer-Anwendungen zwischen Peripheriegeräten realisiert werden.

Die Erweiterungen von OTG sind vor allem gedacht für batteriebetriebene Geräte, die sowohl als ein Peripheriegerät an einem stationären Rechner genutzt werden, als auch selbst

[47] Siehe „Standardisierte Hardware-Treiber“, S. 77.
[48] Eberhardt 2003, S. 24.
[49] Eberhardt 2003, S. 25.
[50] Brownell 2004.

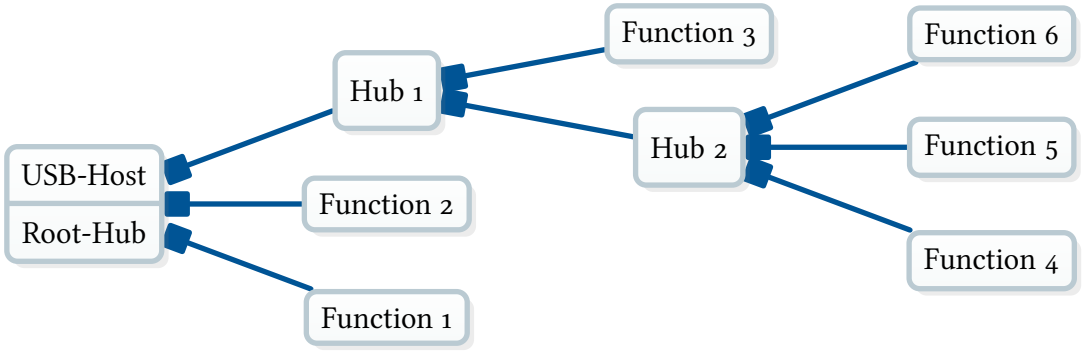


Abbildung 4.7.: Beispiel einer physikalischen Topologie des USB zwischen USB-Host und mehreren Peripheriegeräten („Functions“)

externe Geräte verwenden können.[51] Bei Smartphones kann beispielsweise der interne Speicher als Massenspeicher nach außen zugreifbar gemacht werden. Gleichzeitig könnte es etwa möglich sein, eine Tastatur an das Telefon anzuschließen.

USB-Gadgets, die auf einem Linux-System die Funktion eines Peripheriegeräts implementieren, werden als Kernel-Modul geladen. Für einen abstrakten Umgang mit den zugrundeliegenden verschiedenen Hardware-Komponenten, wurde das Linux-USB Gadget API Framework[52] konzipiert. Es zerfällt in mindestens zwei Teile:

Treiber für USB-Controller („USB Controller Driver“)[53] dienen der Abstraktion von der verwendeten USB-Hardware. Sie steuern Register-, Fifo- oder DMA-Zugriffe und stellen höheren Schichten vereinheitlichte Objekte zur Kommunikation über sogenannte USB-Endpunkte zur Verfügung. Letztere sind die Enden logischer Datenkanäle (Pipes) zwischen USB-Host und Peripheriegerät.[54]

Gadget-Treiber („Gadget Driver“)[55] implementieren hardwareunabhängige USB-Funktionen. Das Gadget muss abhängig von den verfügbaren Funktionen der Hardware-Ebene

- Control-Transfer des Endpunkts `EP0` zur Behandlung von Standard-Device-Requensts[56] interpretieren,
- Konfigurations- und Geräteeigenschaften („configuration descriptor“ bzw. „device descriptor“) propagieren,
- zustandsabhängige Operationen initiieren bei Reset- Suspend- Resume-Kommandos oder dem Abtrennen des Geräts vom Host und
- Datentransfer über weitere Endpunkte verwalten.

Höhere Schichten („Upper Layer“) sind optional und generieren oder verarbeiten die per

[51] Für batteriebetriebene Geräte ebenfalls interessant sind die im Rahmen von OTG spezifizierten Möglichkeiten zur effizienteren Energienutzung (Linux USB Project 2005).
[52] Linux USB Project 2005.
[53] Linux USB Project 2005.
[54] Eberhardt 2003, S. 57–58.
[55] Linux USB Project 2005.
[56] Eberhardt 2003, S. 60.

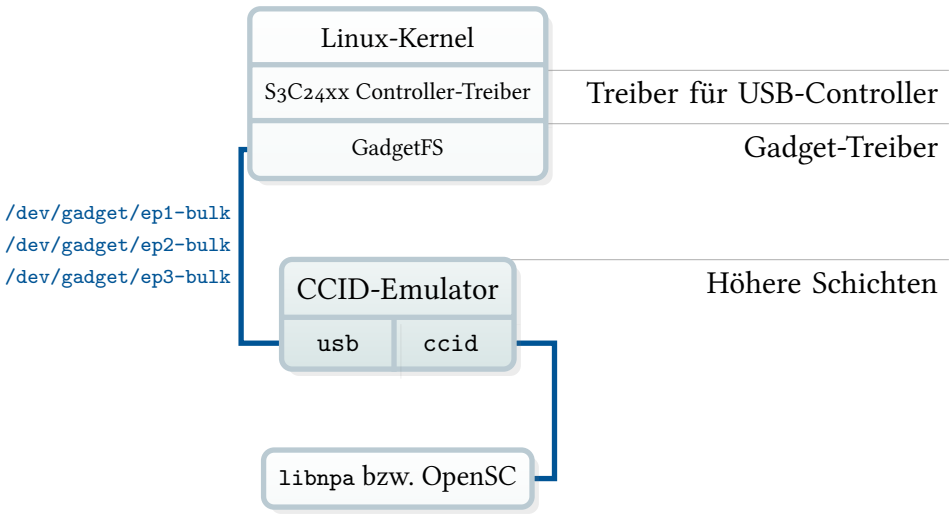


Abbildung 4.8.: Softwareschichten des CCID-Emulators auf dem Openmoko Neo FreeRunner

USB ausgetauschten Daten. Bei einem Chipkartenleser gehört dazu beispielsweise die Kommunikation zur Chipkarte.

Dem Linux-Kernel liegen mehrere USB-Gadgets bei, darunter Ethernet-, Audio-, Massenspeicher- und HID-Gadgets. Diese arbeiten vorwiegend im Kernel Space. Ein Fehler eines solchen Kernel-Moduls könnte zu einem Systemabsturz („Kernel Panic“) führen. Mit dem Kernel-Modul GadgetFS ist es jedoch möglich, wesentliche Funktionen eines USB-Gadgets in den User Space zu verlagern. Auf die verfügbaren USB-Endpunkte kann dann über ein virtuelles Dateisystem zugegriffen werden.

4.3.1. CCID-Emulator für den nPA

Der CCID-Emulator[57] nutzt GadgetFS und `libnpa`,[58] um einen PACE-fähigen Chipkartenleser bereitzustellen. Damit kann der CCID-Emulator in den Upper Layer des Linux-USB Gadget API Framework eingeordnet werden (siehe Abbildung 4.8).

Die Grundlagen des CCID-Emulators wurden im Rahmen der Studienarbeit von Morgner und Oepen erarbeitet.[59] Das Projekt wurde in der vorliegenden Arbeit um die PIN-Verifikation und -Modifikation mittels `RDR_to_PC_Secure`, einem Benachrichtigungssystem zur Übermittlung, in welchem Status sich die Chipkarte des Lesers befindet und natürlich um die für den nPA wesentliche PACE-Funktion erweitert.

Aufbau und Funktionsweise

Mit GadgetFS und `libnpa` wurden bereits zwei voneinander unabhängige Komponenten des CCID-Emulators benannt. Hinzu kommt ein Modul zur Verarbeitung von Kommandos nach

[57] Morgner und Oepen 2011.
[58] Siehe „nPA Smart Card Library“, S. 99.
[59] Morgner und Oepen 2009, S. 56–57.

```
echo 0 > /sys/class/i2c-adapter/i2c-0/0-0073/neo1973-pm-host.0/
   hostmode
echo device > /sys/devices/platform/s3c2410-ohci/usb_mode
sudo mkdir -p /dev/gadget
sudo modprobe gadgetfs default_uid=`id -u`
sudo mount -t gadgetfs gadgetfs /dev/gadget
```

(a) Das Openmoko wird in den Gadget (bzw. „device“) Modus versetzt und GadgetFS gestartet. Ein Gadget des Smartphones würde an einem externen Gerät als USB-Gerät erkannt werden.

```
sudo modprobe dummy_hcd
sudo mkdir -p /dev/gadget
sudo modprobe gadgetfs default_uid=`id -u`
sudo mount -t gadgetfs gadgetfs /dev/gadget
```

(b) Ein virtueller USB-Controller („Host Controller Device“) wird dem System hinzugefügt. Ein Gadget würde am lokalen Gerät als USB-Gerät erkannt werden.

Abbildung 4.9.: Initialisierung von USB-Controller und GadgetFS für den CCID-Emulator

USB CCID, welches für die Kommunikation zur Chipkarte letztlich `libnpa` bzw. OpenSC nutzt. Weiterhin wird in `usb.c` die Initialisierung und Verwaltung des USB-Gadgets und der im Dateisystem abgelegten USB-Endpunkte realisiert.

Um das Smartphone als CCID-Chipkartenleser an einem externen Gerät verwenden zu können, muss der USB-Controller zunächst in den Gadget Modus versetzt werden.[60] Der Openmoko Neo FreeRunner bietet hierzu einen Software-Schalter im Sys-Dateisystem.

Bevor der CCID-Emulator gestartet werden kann, muss sichergestellt werden, dass das Kernel-Modul GadgetFS geladen ist. Standardmäßig ist es für die meisten Distributionen nicht vorkompiliert und muss dann entsprechend erstellt werden.[61] Beim Laden des Kernel-Moduls kann über den Parameter `default_uid` die effektive Benutzer-ID angegeben werden, mit der auf die USB-Endpunkte im Dateisystem lesend und schreibend zugegriffen werden darf. Anschließend kann GadgetFS per `mount` im System eingebunden werden. Abbildung 4.9 zeigt die hierfür nötigen Aufrufe.

Über den Dateinamen des obligatorischen bidirektionalen Endpunkts für den Control-Transfer, `EP0`, können GadgetFS-Anwendungen den USB-Controller identifizieren. Abhängig von ihm werden USB 2.0 High-Speed oder USB 1.1 Full-Speed[62] verwendet sowie Verwendungs-

[60] Ist ein USB-Controller nach OTG verfügbar, wäre auch ein dynamisches Umschalten zwischen Host und Gadget Modus während des Betriebs möglich (vgl. „Aufbau und Funktionsweise“, S. 106). Dies ist jedoch im CCID-Emulator derzeit nicht implementiert.

[61] Hinweise zum Übersetzen des GadgetFS-Kernel-Moduls sind der Dokumentation des CCID-Emulators beigefügt. Beschrieben wird dort auch wie GadgetFS und der CCID-Emulator für das Openmoko erstellt werden können.

[62] Eberhardt 2003, S. 43.

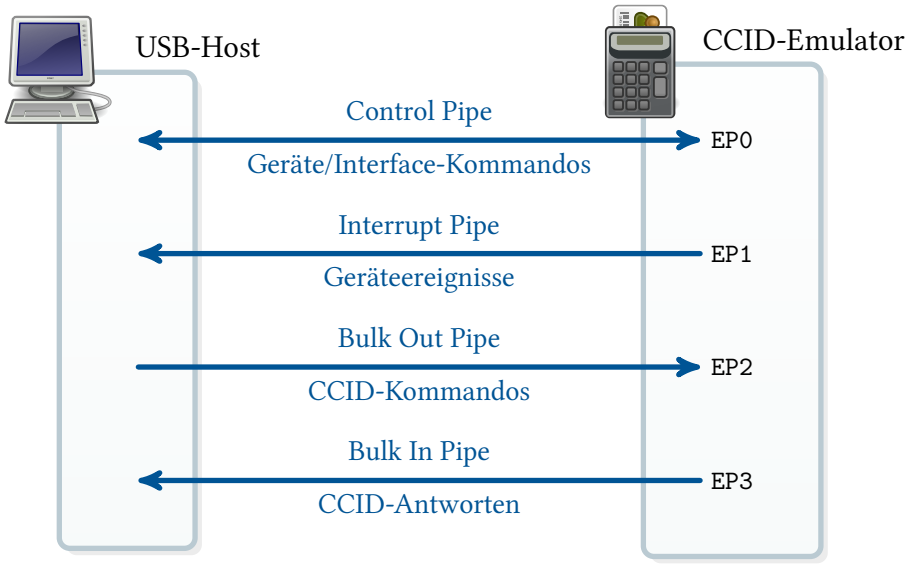


Abbildung 4.10.: Logische Organisation der Kommunikation zwischen USB-Host und CCID-Emulator

möglichkeiten und Namen weiterer Endpunkte ermittelt.[63]

Während der Konfigurationsphase wird in der `EP0` der USB Smartcard Klassendeskriptor übertragen (siehe Tabelle 4.3). Er dient zur Identifizierung des Geräts und seiner Funktionen. Um in USB CCID die noch nicht standardisierte PACE-Funktion zu erkennen, wurden dem `bPINSupport`-Parameter weitere Flags hinzugefügt (siehe Tabelle 4.4). Der Gerätetreiber am Host-System muss diese entsprechend interpretieren. Im laufenden Betrieb werden im Control-Transfer die GadgetFS-Events `NOP`, `SETUP` `CONNECT`, `DISCONNECT` und `SUSPEND` behandelt, die zu den entsprechenden USB-Ereignissen korrespondieren. Auch wenige CCID-spezifische Kommandos werden über die Control Pipe übertragen, unter anderem das `ABORT`-Kommando.[64]

Neben der bidirektionalen `EP0` dienen ein Bulk-In- und ein Bulk-Out-Endpunkt zur Datenübertragung zwischen CCID und Host. Eine Übersicht der in den letzten beiden Kanälen definierten Kommandos wird in Tabelle 4.5 gegeben. Optional nutzt der CCID-Emulator auch eine „Interrupt Pipe“ zur Benachrichtigung bei neu eingesetzten oder entfernten Chipkarten. Falls auf diese Event-basierte Signalisierung verzichtet wird, ermittelt die PC/SC-Middleware durch kontinuierliches Polling den Status des Kartenslots.

Zunächst sollen die herkömmlichen Kommandos zur PIN-Verifikation und -Modifikation betrachtet werden. Ihr Prinzip wurde bereits erwähnt:[65] Die Chipkartenapplikation übermittelt eine formatierte APDU, ohne die PIN festzulegen.[66] Weiterhin erhält der Chipkartenleser Informationen zur Formatierung des Geheimnisses. Der CCID-Emulator implementiert die

[63] Die Namenskonventionen unterscheiden sich je nach Host-Controller und erfordern daher gesonderte Behandlung.
[64] USB CCID, S. 23–24.
[65] Siehe „Standardisierte Hardware-Treiber“, S. 77.
[66] Üblicherweise werden die entsprechenden Bytes mit `0x00` oder `0xFF` initialisiert.

Feld	Beschreibung nach USB CCID	Wert des CCID-Emulators
`bLength`	Länge desc Deskriptors	Fest nach USB CCID, S. 17
`bDescriptorType`	Funktionaler Desktriptor	Fest nach USB CCID, S. 17
`bcdCCID`	Integrated Circuit(s) Cards Interface Devices (CCID)	Fest nach USB CCID, S. 17
`bMaxSlotIndex`	Höchster Kartenslot-Index	0 (genau ein Slot)
`bVoltageSupport`	Maximale Spannung	5 Volt*
`dwProtocols`	Übertragungsprotokolle	T=0 und T=1
`dwDefaultClock`	Standard-Taktfrequenz	3,58 MHz,* USB CCID, S. 17
`dwMaximumClock`	Maximale Taktfrequenz	3,58 MHz,* USB CCID, S. 17
`bNumClockSupport`	Unterstützte Taktfrequenzen	Genau eine
`dwDataRate`	Standard-Datenrate	9600 bps,* USB CCID, S. 17
`dwMaxDataRate`	Maximale Datenrate	9600 bps,* USB CCID, S. 17
`bNumDataRatesSupported`	Unterstützte Datenraten	Genau eine
`dwMaxIFSD`	Unterstützte IFSDs für T=1	Bestimmung erfolgt im CCID-Treiber
`dwSynchProtocols`	Unterstützung für 2-wire, 3-wire oder I2C Protokoll	Nicht verfügbar
`dwMechanical`	Mechanische Funktionen eines Kartenslots	Nicht verfügbar
`dwFeatures`	Spezielle Funktionen	Extended Length APDUs, Benachrichtigung bei Kartenevents, verschiedene automatische Verbindungseinstellungen
`dwMaxCCIDMessageLength`	Maximale Datengröße	65554 Bytes einer Extended Length APDU
`bClassGetResponse`	CLA bei T=0 `GET RESPONSE`	CLA Byte der vorigen APDU wird verwendet*
`bclassEnvelope`	CLA bei T=0 `ENVELOPE`	CLA Byte der vorigen APDU wird verwendet*
`wLcdLayout`	Layout des Displays	255 Zeilen mit je 255 Zeichen
`bPINSupport`	PIN-Funktionen	Siehe 4.4
`bMaxCCIDBusySlots`	Simultan verfügbare Kartenslots	Kein queuing unterstützt

Tabelle 4.3.: USB Smartcard Klassendeskriptor des CCID-Emulators. Einige Fähigkeiten des CCID-Emulator werden von dem verwendeten Back-End zum eigentlichen Chipkartenleser (*) verwaltet, d. h. von OpenSC bzw. dem PC/SC-Treiber des realen Chipkartenlesers.

	Wert	Operation
Operationen nach USB CCID, S. 20	0x01	PIN-Verifikation
	0x02	PIN-Modifikation
Ergänzungen für PACE	0x10	PACE Capabilities
	0x20	PACE Verify

Tabelle 4.4.: PIN-Funktionen des CCID-Emulators mit PACE-Erweiterung. `bPINSupport` enthält den Wert der logischen Oder-Verknüpfung der vorhandenen Fähigkeiten.

Bezeichnung	Beschreibung	
PC_to_RDR_IccPowerOn	Abfrage und Modifikation des Zustands einer Chipkarte in einem Slot des Kartenlesers	✓
PC_to_RDR_IccPowerOff		✓
PC_to_RDR_GetSlotStatus		✓
PC_to_RDR_GetParameters	Abfrage und Modifikation der Kommunikationsparameter zur Chipkarte	✓
PC_to_RDR_ResetParameters		
PC_to_RDR_SetParameters		
PC_to_RDR_IccClock		
PC_to_RDR_T0APDU		
PC_to_RDR_SetDataRateAndClockFrequency		
PC_to_RDR_XfrBlock	Datentransfer (APDUs) zur Chipkarte	✓
PC_to_RDR_Abort	Abbruch angestauter Datentransfers	
PC_to_RDR_Escape	Hersteller-/Leserspezifische Funktionen	
PC_to_RDR_Mechanical	Physisches Blockieren/Auswerfen einer Karte	
PC_to_RDR_Secure	Sichere PIN-Eingabe mittels PIN-Pad	✓

Tabelle 4.5.: Unterstützte Kommandos des CCID-Emulators (✓). Siehe auch „Kommandos eines Chipkartenlesers“, S. 78.

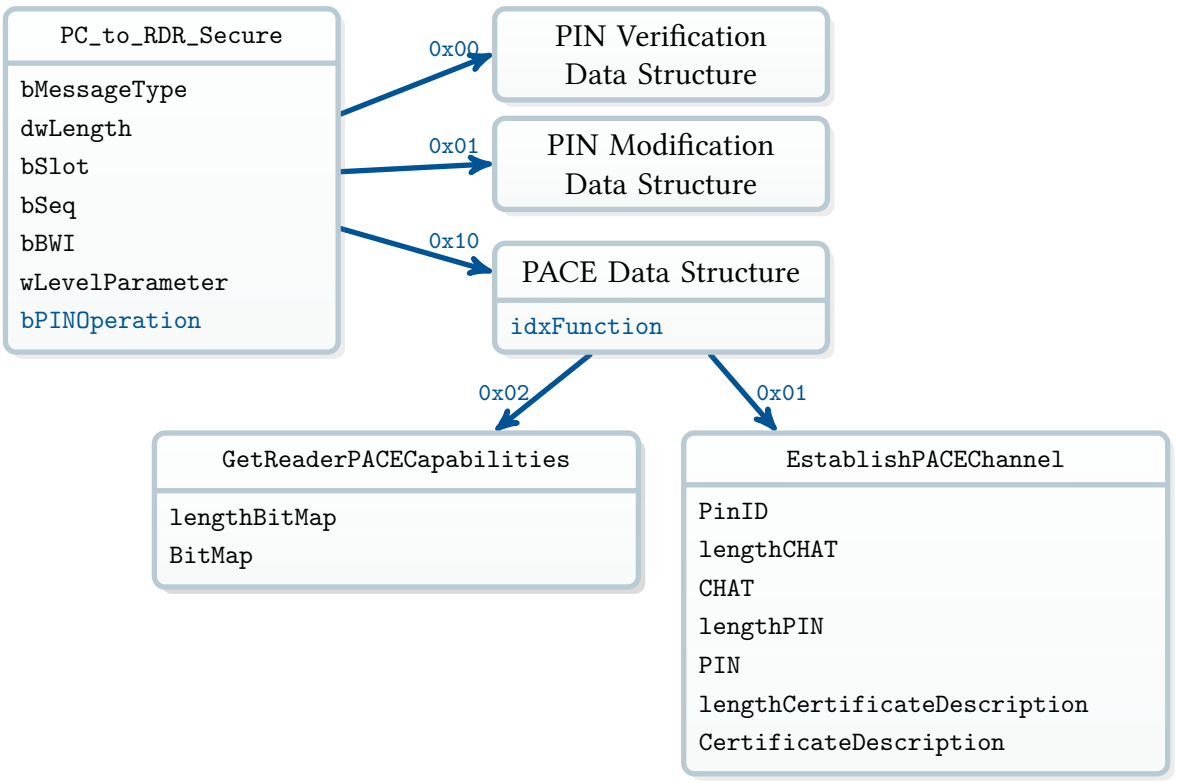


Abbildung 4.11.: Aufbau des `PC_to_RDR_Secure`-Kommandos mit Erweiterung um PACE. Die Werte (z. B. `0x10`) hervorgehobener Elemente (z. B. `bPINOperation`) verzweigen die Strukturen.

Binär-, BCD- und ASCII-Kodierung zu vollen Byte-Grenzen. Ebenso kann die Applikation den Anzeigetext auf dem Display des Lesers wählen. Nach der PIN-Eingabe am CCID-Emulator wird die APDU ausgefüllt, an die Chipkarte gesendet und deren Antwort an die Chipkartenapplikation geleitet.

Dieses Verfahren der Authentifizierung mittels PIN wird beim nPA nach erfolgreicher EAC verwendet, um die Signatur-PIN zu prüfen. Aber auch bei der eID-Anwendung wird es genutzt, um die eID-PIN zu wechseln. (Im Gegensatz zu anderen Chipkarten verwendet der nPA also nie eine unverschlüsselte Übertragung der PIN bzw. der entsprechenden APDU, weil nach PACE bzw. EAC das SM aktiv ist.) Der CCID-Emulator kann damit sowohl dem nPA als auch vielen anderen Chipkarten und -anwendungen eine sichere Möglichkeit zur PIN-Eingabe geben.

Die verschiedenen Möglichkeiten zur Integration von PACE in USB CCID wurden bereits mit ihren Vor- und Nachteilen diskutiert:[67]

- Pseudo-APDUs via `PC_to_RDR_XfrBlock`
- Proprietäres Kommando via `PC_to_RDR_Escape`
- Neuschaffung von `PC_to_RDR_PACE`

[67] Siehe „Standardisierte Hardware-Treiber“, S. 77.

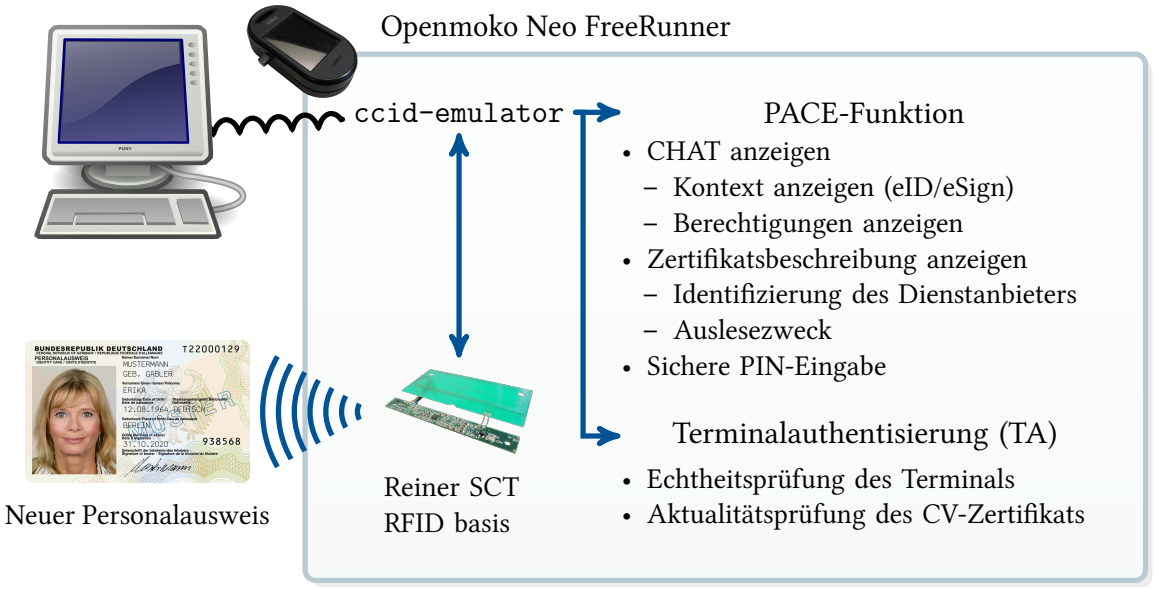


Abbildung 4.12.: Umsetzung des mobilen Chipkartenlesers

- Erweiterung von `PC_to_RDR_Secure`

Beim CCID-Emulator wurde die letzte dieser Varianten umgesetzt. Das `PC_to_RDR_Secure`-Kommando ist hierarchisch aufgebaut. Es enthält entweder die „PIN Verification Data Structure“ oder die „PIN Modification Data Structure“.[68] Genau an dieser Stelle wurde die „PACE Data Structure“ hinzugefügt (siehe Abbildung 4.11).

Die meisten CCID-Kommandos sind an die PC/SC-Spezifikation angelehnt, so dass die Umsetzung des CCID-Treibers entsprechend simplifiziert wird.[69] Analog sind Daten die Eingaben des `SCardControl`-Kommandos mit `FEATURE_EXECUTE_PACE`[70] identisch mit denen der PACE Data Structure. Lediglich die Bytereihenfolge kann sich zu dem PC/SC-Aufruf unterscheiden, da auf USB-Ebene stets mit Little-Endian kommuniziert wird. Die Rückgabedaten eines PACE-Aufrufs werden nach dem selben Schema mit `RDR_to_PC_DataBlock` an den Host gesendet.

Die eigentliche Durchführung von PACE sowie des anschließenden SM wird beim CCID-Emulator mittels `libnpa`[71] realisiert. Dadurch wird PACE nach BSI TR-03119 1.2 funktional in vollem Umfang unterstützt. Das Geheimnis wird, wie in BSI TR-03119 1.2, S. 34 gefordert, zum spätest möglichen Zeitpunkt eingegeben und frühst möglich bereinigt. Auch temporäre Sitzungsschlüssel werden via `libnpa` und OpenPACE überschrieben und freigegeben.

Wie bereits angesprochen, muss das Gegenstück zum CCID-Emulator, der CCID-Gerätetreiber, die PACE-Funktion auch geeignet auslösen. Dem CCID-Emulator liegen Patches

[68] Als Spezialfall kann das `PC_to_RDR_Secure`-Kommando auch einen früheren Aufruf einer PIN-Operation vervollständigen.

[69] Beispielsweise sind die Objekte „PIN Verification Data Structure“ (USB CCID, S. 39–40) und „PIN_VERIFY“ (PC/SC Pt. 10, S. 5) bis auf Bytereihenfolge identisch. Gleiches gilt für die „PIN Modification Data Structure“ (USB CCID, S. 43–44) und „PIN_MODIFY“ (PC/SC Pt. 10, S. 6).

[70] BSI TR-03119 1.2, S. 31–32; PC/SC Pt. 10 amd. 1, S. 4–5.

[71] Siehe „nPA Smart Card Library“, S. 99.

für `libccid` bei. Die PACE-Funktion per CCID kann damit bei allen Unix-artigen Systemen verwendet werden.

Eine Integration in den Windows-Treiber hingegen scheint unwahrscheinlich. Dieser wurde von Microsoft seit 2003 nicht verändert und enthält nicht einmal die bereits spezifizierten Verfahren zur PIN-Verifikation oder -Modifikation mit PIN-Übertragung.[72] Hier müsste der System-eigene Gerätetreiber ersetzt werden.[73]

Besonderheiten und Schwierigkeiten

Der CCID-Emulator profitiert von allen in `libnpa` umgesetzten Schutzmaßnahmen. So werden CHAT und Zertifikatsbeschreibung in allen Einzelheiten ausgegeben. Dadurch kann der Anwender kontrollieren, welche Funktion des Ausweises genutzt wird und welche Daten freigegeben werden sollen. Die Zertifikatskette und die Signatur des Terminals werden unabhängig vom nPA geprüft. Dadurch wird eine zusätzliche Sicherheit geschaffen und man muss nicht nur darauf vertrauen, dass der nPA korrekt arbeitet. Insbesondere – und dies unterscheidet den CCID-Emulator von einem Standard- oder Komfortleser – kann sichergestellt werden, dass die Terminalzertifikate aktuell sind. Ungültige Zertifikate, deren Gewährleistungszeitraum abgelaufen ist, werden von dem CCID-Emulator nicht akzeptiert.

Einige Details der Zertifikatsbeschreibung oder des CHAT werden ohne die Filterung von nicht-druckbaren Zeichen vom CCID-Emulator ausgegeben. Daher könnten ANSI-Escape-Sequenzen genutzt werden, um die Darstellung auf der Kommandozeile zu manipulieren. Die „sichere Anzeige“ wäre gefährdet. Praktisch ist dies wenig relevant, da ein solcher CHAT vom Ausweis abgelehnt würde und eine entsprechende Zertifikatsbeschreibung kaum für ein Terminalzertifikat genutzt werden dürfte. Trotzdem ist es wünschenswert, wenn sich die Ausgabe des CCID-Emulators lediglich auf druckbare Zeichen beschränken würde.

Die Nutzung von GadgetFS findet im „Upper Layer“ des Linux-USB Gadget API Framework statt. Nichtsdestotrotz müssen viele Eigenheiten des USB-Controllers in dem CCID-Emulator berücksichtigt werden. Hier wäre ein höheres Maß an Abstraktion sinnvoll. Außerdem kann der Linux Kernel stets nur ein Gadget-Modul laden. Falls also mehrere Funktionen angeboten werden sollen, z. B. ein Chipkartenleser (CCID) *und* ein Eingabegerät (HID), so muss dies derzeit in dem CCID-Emulator erfolgen. Beiden Problemen könnte mit dem Umstieg von GadgetFS auf FunctionFS begegnet werden.[74]

4.4. nPA-Kartenemulator

Der Titel dieses Abschnitts könnte beim Leser den Eindruck erwecken, es würde ein Emulator erstellt, der für einen vollständigen Durchlauf der EAC geeignet ist. Dies erscheint unwahrscheinlich, weil der Emulator sich bei der CA als echter Ausweis beweisen müsste. Zwar kann man durchaus argumentieren, dass die aktuellen zehn Jahre gültigen Ausweise ihre geheimen

[72] `http://msdn.microsoft.com/en-us/windows/hardware/gg487509`

[73] Dieses Vorgehen ist nicht unüblich, da bereits zur Unterstützung der herkömmlichen PIN-Verifikation/-Modifikation ein modifizierter Gerätetreiber vonnöten ist.

[74] Siehe „Openmoko als mobiler Chipkartenleser“, S. 122.

CA-Schlüssel nicht für die gesamte Nutzungsdauer schützen können.[75] Darum soll es an dieser Stelle aber nicht gehen. Hier wird ein Emulator des nPA vorgestellt, der vor allem die angeforderten Berechtigungen eines Terminals ermittelt und zur Anzeige bringt. Zwar wird auch die CA umgesetzt, aber mangels entsprechendem Schlüsselmaterial wird der Emulator letztlich immer als unecht erkannt.

Die theoretischen Vorbetrachtungen haben gezeigt, dass eine Prüfung als Kartenemulator denkbar einfach ist.[76] Bereits in der ersten für PACE übermittelten APDU ist der CHAT enthalten. Mit ihm legt das Terminal fest, welche Anwendung es aufrufen möchte, in welcher Rolle es den Ausweis verwenden will und welche Funktionen es aufrufen oder welche Daten es auslesen möchte. Zwar wäre es zur Authentizitätsprüfung des Terminals sinnvoll, auch die TA durchzuführen. Aber damit ein Nutzer mit dem Kartenemulator einen ungewollten Zugriff im vornherein erkennen kann, ist es noch nicht einmal notwendig PACE durchzuführen, geschweige denn die EAC.

Die eigentliche Frage ist, wie man sich gegenüber einem Terminal verhalten muss, dass man als nPA erkannt wird. Zunächst ist festzustellen, dass der nPA eine zu ISO 7816 konforme Chipkarte ist. Im Kartenidentifikationsprozess (siehe Abbildung 4.13) werden zunächst lediglich Dateien von der Chipkarte gelesen werden, um diese zu erkennen. Der AusweisApp genügt ein korrekt formatiertes `EF.CardAccess` (z. B. von einem echten Ausweis), um eine Chipkarte als nPA zu identifizieren.

Eine Emulation „in Software“ genügt aber noch nicht, um ein öffentliches Terminal zu überprüfen. Der nPA-Kartenemulator muss auch per NFC mit dem Terminal kommunizieren können. Die Virtual Smart Card Architecture[77] bietet hier eine ideale Grundlage. Sie enthält sowohl eine Softwareemulation einer Smartcard (die virtuelle Smartcard), als auch ein Programm zur Kommunikation mit einem NFC-Emulator (`pcsc-relay`).

Die virtuelle Smartcard entstand ebenfalls mit der Studienarbeit von Morgner und Oepen.[78] In der Python-Software werden alle wesentlichen Kommandos von ISO 7816 implementiert. Auch eine Umsetzung des SM ist hier verfügbar. Sie wird bei der Emulation eines deutschen elektronischen Reisepasses nach erfolgreicher BAC genutzt. Da die einfachen Datei-Kommandos in der virtuellen Smartcard bereits umgesetzt wurden, fehlt für einen nPA (nach Kartenidentifikationsprozess) lediglich noch ein entsprechendes `EF.CardAccess`. Sendet das Terminal nun nach Erkennung des nPA das `MSE:Set AT`, so kann der CHAT mit den angeforderten Berechtigungen empfangen und angezeigt werden. Diese können vom Nutzer überprüft werden. Die Anzeige-Logik für die Berechtigungen wird von pyPACE bzw. OpenPACE[79] bereitgestellt.

Zur Nutzung der virtuellen Smartcard muss am Host-System der virtuelle Smartcardleser installiert werden. Dieser ist ein spezieller Gerätetreiber für PCSC Lite[80] und kann damit nur unter Unix-artigen Systemen verwendet werden. Die virtuelle Smartcard nutzt eine (unter Umständen lokale) Netzwerkverbindung, um mit dem virtuellen Smartcardleser zu

[75] Plötz 2009.
[76] Siehe „Prüfungen als Kartenemulator“, S. 80.
[77] Morgner und Oepen 2011.
[78] Morgner und Oepen 2009.
[79] Siehe „Kryptographische Umsetzung: OpenPACE“, S. 93.
[80] `http://pcsclite.alioth.debian.org/pcsclite.html`

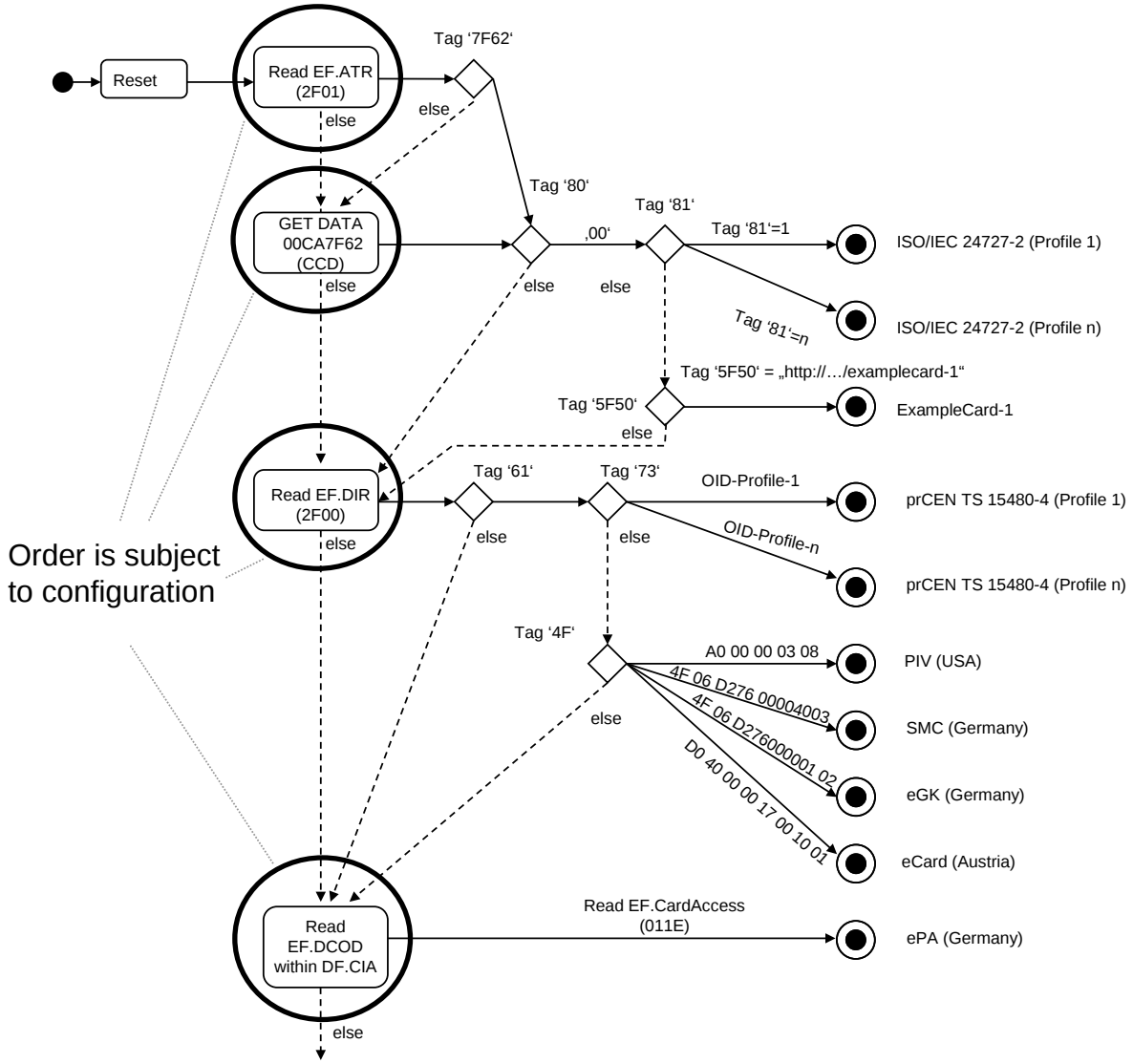


Abbildung 4.13.: Kartenidentifikationsprozess nach Pt. 4 BSI TR-03112, S. 98

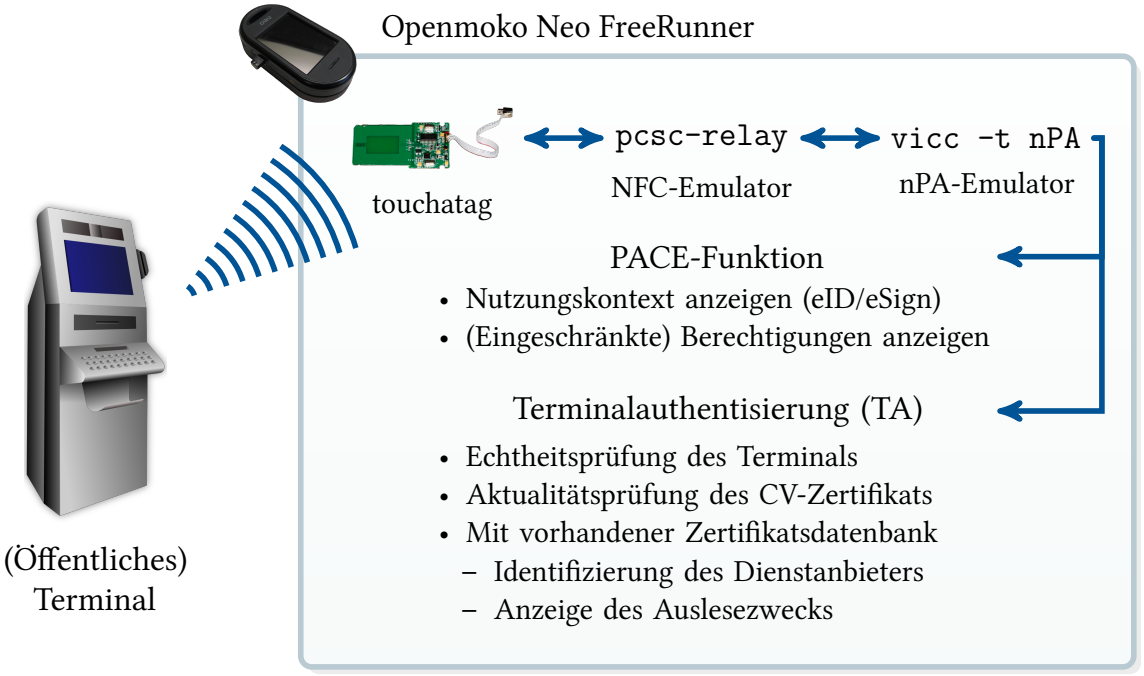


Abbildung 4.14.: Umsetzung des nPA-Emulators

kommunizieren. Der virtuelle Smartcardleser sorgt wiederum dafür, dass auf die virtuelle Smartcard per PC/SC zugegriffen werden kann.

Genau an dieser Stelle greift `pcsc-relay` die Daten der virtuellen Smartcard ab und verschickt sie über ein externes Gerät per NFC an ein Terminal. Als NFC-Back-Ends werden derzeit der OpenPICC[81] und `libnfc`[82] unterstützt. Letztere ist selbst eine Bibliothek zum abstrakten Zugriff auf NFC-Chips, welche die Kartenemulation mit den NFC-Controllern PN531, PN532 und PN533 ermöglicht. Die NFC-Emulation mit `pcsc-relay` wurde mit dem touchatag ACR122U102 (PN532) getestet und bereits auf dem 27. Chaos Communication Congress demonstriert.[83] Morgner und Oepen nutzten `pcsc-relay`, um gegenüber einem Standard- und Komfortleser (sowie der ihn nutzenden AusweisApp) erfolgreich einen nPA zu simulieren. Die Emulation gegenüber einem öffentlichen Terminal würde auf gleiche Weise erfolgen, aber mangels entsprechender Terminals konnte dies noch nicht getestet werden.

Die Überprüfung der angeforderten Berechtigungen von einem öffentlichen Terminal ist also mit dem nPA-Emulator problemlos möglich. Prinzipiell wäre mit `pcsc-relay`[84] auch die Umsetzung der „Man-in-the-Middle Defence“ auf NFC-Ebene denkbar.[85] Die EAC könnte

[81] `http://www.openpcd.org/OpenPICC_RFID_Emulator_and_Sniffer_Project`
[82] `http://www.libnfc.org/`
[83] Morgner und Oepen 2010.
[84] Die Filter-Logik könnte man in die virtuelle Smartcard verlagern. Sie bietet derzeit einen Relay-Modus, in dem sie selbst als PC/SC-Client die Daten einer weiteren Chipkarte (dem nPA) abgreift und in den virtuellen Smartcardleser speist.
[85] Siehe „Prüfungen als Kartenemulator“, S. 82.

passiv belauscht werden, um den CHAT abzufangen und anzuzeigen. Allerdings müsste man dazu die Möglichkeit haben, mit zwei NFC-Geräten zu kommunizieren, einmal als Tag (zum Terminal) und einmal als Initiator (zum nPA). Dies ist mit einem einzelnen NFC-Chip derzeit nicht gleichzeitig möglich. Die Integration zweier unabhängig voneinander arbeitenden NFC-Chips in einem Smartphone erscheint wegen Platz- und Übertragungsproblemen nicht möglich.

Die von `pcsc-relay` genutzten Hardware-Komponenten sind nicht in der Lage, APDUs mit erweitertem Längenfeld („extended length APDU“[86]) zu versenden. Daher scheiden sie für eine vollständige Emulation des nPA (insbesondere der TA zur Überprüfung des Terminals) aus. Der Proxmark III[87] hingegen ist ein preiswerter NFC-Emulator, welcher diese Anforderungen erfüllt. Er könnte als weiteres Back-End für `pcsc-relay` die vollständige NFC-Emulation eines nPA übernehmen. Ein Projekt zur Emulation einer kontaktlosen Smartcard mit dem Proxmark III wird derzeit von Michael Gehring an der Humboldt Universität zu Berlin bearbeitet.

[86] ISO/IEC 7816-4.

[87] `http://www.proxmark3.com/`

5. Diskussion und Ausblick

5.1. Nicht „die gesamte Technik ist sicher“[1]

In dieser Arbeit wurde detailliert Einblick gegeben in die Mechanismen zur elektronischen Nutzung des nPA. Die Betrachtung der verschiedenen Parteien, Protokolle und Zusammenhänge hat potenzielle Schwächen des „Systems neuer Personalausweis“ aufgedeckt. Der Fokus lag hierbei auf den Hardware- und Software-Komponenten, die beim Bürger genutzt werden. Dabei konnte auch die Forschungsfrage in der Beziehung geklärt werden, dass hochwertige Chipkartenleser nicht alle Risiken abwenden können. Verschiedene Bedrohungen wurden skizziert oder exemplarisch umgesetzt.

Es wurden zwei wesentliche prinzipielle Probleme festgestellt. Erstens sind eID- und eSign-Anwendung auf dem nPA auf ungünstige Weise miteinander verwoben. Die Chipkarte kann sowohl als Identitäts- als auch Signaturkarte verwendet werden. Der nPA wird als SSEE vollständig mittels eID-Anwendung und eID-PIN aktiviert. Dies kann von einem Angreifer genutzt werden, um von dem eID- in den kritischen eSign-Kontext zu gelangen („Privilege Escalation“). Die Prüfung des ZDA, ob der Ausweisinhaber zur Erstellung der Signaturschlüssel alleiniger Besitzer des nPA war, kann aber einen Missbrauch verhindern.

Zweitens muss man auch bei hochwertigen Lesegeräten *Vertrauen* haben in die Echtheit des Geräts selbst *und* in die Authentizität der auf dem PC angezeigten Daten. Beides ist für normale Anwender schwer zu überprüfen. Skimming kann einem Angreifer dienen, um vollen Zugriff auf den Ausweis und seine Authentisierungsgeheimnisse zu erlangen. Die Daten einer eID- oder eSign-Transaktion werden ungesichert an den Chipkartenleser übertragen. Ein Angreifer könnte mittels Malware das Ziel einer eID-Nutzung modifizieren oder zu signierende Daten ändern.

Die vorgestellten Angriffe zeigen, dass die Ziele des Systems nPA nicht in vollem Umfang erreicht werden. Die Angriffe führen zu folgenden Bedrohungen:

1. Tracing mit Wissen von eID-PIN, CAN oder MRZ
2. Unberechtigtes Ermitteln der eID-Daten
3. Nicht autorisierte oder modifizierte eID-Transaktionen (insbesondere auch Aktivierung als SSEE)
4. Nicht autorisierte oder modifizierte eSign-Transaktionen

Die praktische Bedeutung der vorgestellten Schwachstellen ist zum jetzigen Zeitpunkt schwer einzuschätzen. Ob der nPA als Ziel für einen Angriff infrage kommt, hängt maßgeblich vom Erfolg des elektronischen Identitätsnachweises und der Signatur ab. Derzeit gibt es circa 50 Dienstanbieter, die den elektronischen Identitätsnachweis nutzen (etwa die Hälfte

[1] Der Titel dieses Abschnitts wurde in Anlehnung an ein Interview mit Cornelia Rogall-Grothe gewählt (Welchering 2010). Ihre Behauptung ist konträr zu der hiesigen.

davon sind behördliche Stellen),[2] aber keinen einzigen ZDA. Welche Dienste für Angreifer ein interessantes Ziel sind, wurde hier nur überblicksartig betrachtet.[3]

Klar ist, dass in wenigen Jahren etwa 60 Millionen Deutsche einen nPA besitzen werden.[4] Folglich können selbst Angriffe mit geringer Erfolgsaussicht sehr viele Bürger in Mitleidenschaft ziehen. Wenn sich z. B. geldwerte Operationen mit dem nPA durchführen oder die eID-Daten für entsprechende Zwecke verwenden lassen, wird der nPA zu einem attraktiven Angriffsziel. Sobald mittels der eID- oder Signaturfunktion Onlineshopping betrieben, Bankkonten eröffnet oder Versicherungen abgeschlossen werden können, besteht mit den vorgestellten Angriffen definitiv die Bedrohung vom Missbrauch des nPA vor allem – aber nicht nur – für unbedarfte Anwender.

5.2. Schutzmaßnahmen des mobilen Chipkartenlesers

Standard- oder Komfortleser bieten eine sichere Ein- und Ausgabe und können inhaltliche Prüfungen der Transaktionen vornehmen. Sie sind jedoch in ihren Hardware-Komponenten sehr beschränkt und zudem kostspielig. Als separates Gerät sind sie platzraubend und nur für den stationären Einsatz geeignet. Um diese Einschränkungen aufzuheben, wurde das Konzept des mobilen Chipkartenlesers für den nPA vorgestellt. Er fügt sich als PACE-fähiger Leser nahtlos in die bestehende Infrastruktur ein. Zugleich muss kein zusätzliches Gerät zur sicheren nPA-Nutzung angeschafft werden.

Betrachtet man die Kategorisierung nach BSI TR-03119 1.2, ist der CAT-M dem Konzept nach ähnlich einem Standardleser:

A.1 Schnittstelle zum Host-Rechner

A.2 Kontaktlose Schnittstelle nach ISO/IEC 14443

A.4 PIN-Pad (sichere PIN-Eingabe) mit PACE-Unterstützung

A.5 Display

A.8 Firmware-Update

Der CAT-M erfüllt zusätzlich aber auch Module, die von den anderen Kartenlesegeräten nicht umgesetzt werden:

A.1.M.1 Sichere PIN-Eingabemethode[5]

A.1.M.2 Erweiterte Filterregeln[6]

A.5.M.3 Vollständige und übersichtliche Darstellung[7]

[2] VfB 2012.
[3] Siehe „Die elektronische Nutzung des neuen Personalausweises“, S. 5.
[4] VfB 2011.
[5] Siehe „Sichere Eingabemethode“, S. 72.
[6] Siehe „Filterregeln“, S. 73.
[7] Siehe „Display: Vollständige und übersichtliche Darstellung“, S. 74.

M.1 Sichere Ausführungsumgebung[8]

M.2 Standardisierte Hardwareschnittstelle[9]

M.3 nPA-Emulator[10]

M.4 Zeitaktualisierungsdienst[11]

M.5 Autarkes PIN-Management und temporäre PIN[12]

Der CAT-M schützt Authentisierungsgeheimnisse (A.4) und visualisiert die Transaktionsdetails (A.5). Im Gegensatz zu den anderen Chipkartenlesern kann der CAT-M den Auslesezweck darstellen und die angefragten Berechtigungen übersichtlich präsentieren (A.5.M.3). Der CAT-M kann auch in einer mobilen und potenziell unsicheren Umgebung eingesetzt werden (M.2, A1.M1). Um eine Nutzung mit „Zero Footprint" zu ermöglichen, sind jedoch Änderungen an Gerätetreibern oder Client-Software notwendig.

Als reiner Kartenleser kann der CAT-M die Authentisierung eines Dienstanbieters unabhängig vom nPA prüfen. Durch die Aktualitätsprüfung und den Abgleich mit möglichen Sperrlisten kann die Nutzung ungültiger Terminalzertifikate verhindert werden (A.1.M.2).

Als nPA-Emulator (M.3) kann der CAT-M diese Prüfungen gleichermaßen auf ein Terminal anwenden, dass nicht die Nutzung eines (externen) sicheren Lesegeräts erlaubt. Eine Zeitaktualisierung im nPA (M.4) verhindert in diesem Fall, dass abgelaufene Terminalzertifikate akzeptiert werden.

Beim Konzept des CAT-M wird davon ausgegangen, dass die einzelnen Module nach den hier gegebenen Anforderungen erfüllt werden. Ist dies der Fall, bietet der CAT-M eine in einigen Bereichen eine höhere Sicherheit als ein CAT-S. Ein Smartphone mit sicherer Ausführungsumgebung ist komplex und bietet mehr Angriffspunkte als bei dem funktional beschränkten Standardleser. Die veränderten Angriffsvektoren müssen bei der Umsetzung eines CAT-M beachtet werden.

Die IT-Sicherheitsevaluierung der Chipkartenleser mit nPA-Unterstützung[13] erfordert die Manipulationssicherheit der PIN-Eingabe und der Anzeige. Es wurde festgestellt, dass sich für eine sichere Ausführungsumgebung alle bekannten Methoden der Software-Isolierung auch auf modernen Smartphones umsetzen lassen. Ein Vertrauensanker in Form eines Hardware-Moduls scheint hier jedoch schwer zu realisieren.

Erfolgreiche Angriffe auf verschiedene Virtualisierungstechniken[14] und vertrauenswürdige Hardware-Module[15] zeigen, dass es keine absolute Sicherheit gibt. Mit dem CAT-M lässt sich zumindest *prinzipiell* eine zum CAT-S ähnliche Sicherheit gewährleistet, wobei *zusätzlich*

[8] „Sichere Ausführungsumgebung", S. 84; bei Standard- und Komfortlesern besteht kein Bedarf, sichere von unsicheren Anwendungen abzugrenzen. Sie sollten keine möglicherweise bösartigen Anwendungen anbieten.
[9] Siehe „Standardisierte Hardware-Treiber", S. 76.
[10] Siehe „Prüfungen als Kartenemulator", S. 80.
[11] Siehe „Zeitaktualisierungsdienst", S. 82.
[12] Siehe „Autarkes PIN-Management und temporäre PIN", S. 83.
[13] BSI TR-03119 1.2, S. 37.
[14] Criscione 2008, 2010; Elhage 2011.
[15] bushing und marcan 2008; bushing, marcan und sven 2010; Tarnovsky 2010; Tarnovsky und Nohl 2011.

Mögliche Bedrohungszenarien	CAT-B	CAT-S	CAT-K	CAT-M
Brute-Force-Angriff auf CAN	✗	✗	✗	
Ausspähen der eID-PIN durch Keylogger auf dem PC	✗			
Social Engineering von Authentisierungsgeheimnissen	✗	✗	✗	✗
Lesen der angezeigten Ausweisdaten durch Man-in-the-Browser	✗	✗	✗	✗
Transaktionen mit veraltetem Terminalzertifikat	✗	✗	✗	
Skimming von öffentlichen Terminals oder Chipkartenlesern	✗	✗	✗	
Manipulation von gewollten Transaktionen (z. B. MitM PC/SC oder USB)	✗	✗	○	
Unberechtigte Initialisierung mit Signaturschlüsseln/-zertifikat	✗	✗	○	
Unberechtigte Initialisierung mit Signatur-PIN	✗			
Unberechtigte operative Nutzung als SSEE mit bekannter Signatur-PIN	✗			

Tabelle 5.1.: Vergleich des CAT-M mit anderen Chipkartenlesern für den nPA. Die bei Verwendung des jeweiligen Lesers ausführbare Angriffe sind mit ✗ markiert. Einige Angriffe sind möglich, wenn der Ausweisinhaber die Transaktionsdaten auf dem Display des Chipkartenlesers nicht genau prüft (○).

die meisten der in aufgezeigten Bedrohungen verhindert werden (siehe Tabelle 5.1). Bei einer praktischen Umsetzung des CAT-M sollte ein Kompromiss aus Kosten und Nutzen angestrebt werden.

5.3. Openmoko als mobiler Chipkartenleser

Die im Rahmen dieser Arbeit entstandene und erweiterte Software erfüllt in wesentlichen Bereichen die Voraussetzungen für eine sichere Nutzung des nPA. Die zentrale Frage nach der Machbarkeit der sicheren nPA-Nutzung auf mit dem mobilen Chipkartenleser kann positiv beantwortet werden kann. Gleichzeitig sind an vielen Stellen Verbesserungen möglich.

OpenPACE bietet als kryptographische Bibliothek eine vollständige Unterstützung für die EAC nach BSI TR-03110 2.05. Es ist damit umfangreicher als andere Implementierungen. OpenPACE ist in den für EAC genutzten kryptographischen Routinen flexibel, so dass auch zukünftige Konstellationen von Hash-, Verschlüsselungs- oder Schlüsselaustauschverfahren unterstützt werden können. OpenPACE bietet dem Nutzer eine einfache Verwaltung von Objekten und schützt sensitive Speicherbereiche.

Derzeit sind die bekannten Vertrauensanker der EAC-PKI[16] in den Quelltext von OpenPACE integriert. Dies ist zunächst unproblematisch, da sie lediglich alle drei Jahre aktualisiert werden. Dennoch ist es wünschenswert, z. B. auch Vertrauensanker von Test-Systemen ohne Modifikation des Programmcodes hinzufügen zu können. Die Zertifikate vertrauenswürdiger

[16] Siehe `https://www.bsi.bund.de/DE/Themen/ElektronischeAusweise/CVCAeID/CVCAeID_node.html`

SSL CAs werden von OpenSSL in einem speziellen Verzeichnis abgelegt.[17] Es wäre sinnvoll, auf gleiche Weise auch mit den CV-Zertifikaten der CVCA zu verfahren.

Nach Erfahrung des Autors ist es schwierig, selbst etablierte Standards im Haupt-Entwicklungszweig von OpenSSL zu verankern.[18] Es ist unwahrscheinlich, dass OpenPACE übernommen wird. Weil OpenPACE als Patch stark an OpenSSL gekoppelt ist, ist die Wartbarkeit stark eingeschränkt. Die Auslagerung in eine eigenständige Bibliothek sollte angestrebt werden.

In `libnpa` ist der abstrakte Zugriff auf wesentliche Funktionen des nPA möglich, die an einem lokalen oder nicht authentisierten Terminal durchgeführt werden können. Das `npa-tool` kann für das PIN-Management und das Setzen einer temporären PIN verwendet werden. Als Back-End zum Chipkartenleser sind via OpenSC sowohl PC/SC als auch die CT-API nutzbar. Die Implementierung des SM wurde generisch gehalten, so dass sie auch für andere Chipkarten genutzt werden kann.

In `libnpa` werden derzeit bereits wichtige Prüfungen wie die Aktualität des Terminalzertifikats und die Korrektheit der Terminalsignatur durchgeführt. Die dafür verwendete Logik ist relativ simpel. Für einen nPA werden die eingehenden APDU mit denen aus BSI TR-03110 2.05, S. 67–81 abgeglichen und gegebenenfalls inhaltlich geprüft. Tatsächlich könnte man aber auch *alle* Kommandos an den nPA darauf prüfen, ob sie in der vorliegenden Form valide sind. Auch ein Zustandsautomat, der den nPA nachbildet, wäre denkbar. Unspezifizierte Kommandos oder auch inhaltliche Fehler könnten dadurch erkannt werden, noch bevor sie zum nPA gesandt werden. Diese sehr komplexen Prüfungen sollen letztlich aber auch von dem nPA selbst durchgeführt werden.

Eine Zeitverzögerung nach einer fehlgeschlagenen Authentisierung, wie oben vorgeschlagen,[19] sollte bei `libnpa` hingegen leicht zu ergänzen sein. Es kann auch im Bezug auf andere Chipkarten sinnvoll sein, nur bestimmte Transaktionen zu autorisieren. Normalerweise wird der Zugriff auf eine Chipkarte (z. B. per PC/SC) generell gewährt, ohne die Anzahl oder den Typ der Transaktionen zu begrenzen. Wird z. B. die Kommunikation zu einem Chipkartenleser manipuliert,[20] könnte eine legitime Verbindung genutzt werden, um unautorisierte Transaktionen durchzuführen. Beim nPA könnten Transaktionen oder Kommandos an dem stets unverschlüsselten APDU-Header erkannt und begrenzt werden (selbst bei aktivem SM).

Der CCID-Emulator bietet zumindest einen impliziten Schutz gegen ungewollte Transaktionen. Erst wenn das Programm gestartet ist, wird die eingelegte Chipkarte auch von außen zugreifbar. Der hier vorgestellte CCID-Emulator implementiert das standardisierte Verfahren zur sicheren PIN-Verifikation und -Modifikation nach PC/SC Pt. 10 und USB CCID. Darüber hinaus wurde die PIN-Eingabe mit anschließendem PACE umgesetzt und in den USB-Standard integriert.

Den CCID-Emulator könnte man um andere Möglichkeiten zum Auslösen der PACE-

[17] Die Vertrauensanker liegen beispielsweise unter Ubuntu und Debian in `/etc/ssl/certs/`

[18] Ein Patch für die Brainpool Kurven (RFC 5639) wurden von Dominik Oepen am 17. April 2010 eingereicht (`http://rt.openssl.org/Ticket/Display.html?id=2239&user=guest&pass=guest`) und wurde erst am 22. April 2012 eingepflegt.

[19] Siehe „Schutz von CAN und MRZ“, S. 62.

[20] Siehe „Man-in-the-Middle Hardware“, S. 40.

Funktion auf dem USB ergänzen.[21] In einigen Fällen könnte man dadurch erreichen, dass ein einfacher Gerätetreiber verwendet werden kann und lediglich die verwendete Client-Software angepasst werden müsste.

Es ist eine Eigenschaft von dem genutzten GadgetFS, dass nicht gleichzeitig ein anderes USB-Gadget aktiv sein kann. Abhilfe verspricht hier FunctionFS, welches fast auf gleiche Weise verwendet wird wie GadgetFS. Gadgets regeln jedoch nicht mehr den gesamten USB-Verkehr, sondern propagieren lediglich ihren Geräte-Deskriptor und verwalten (abstrakter als bei GadgetFS) ihre Endpunkte. Die gleichzeitige Nutzung verschiedener Funktionen bzw. Gadgets steuert der Linux-Kernel. Die Änderungen am CCID-Emulator zur Migration nach FunctionFS wären nur marginal: Die Kommunikation im virtuellen Dateisystem ist identisch, lediglich bei der Initialisierung müssten nun die Datentypen von FunctionFS genutzt werden.

Die Umsetzung des nPA-Kartenemulators kann gegenüber den bisher verfügbaren Terminals erfolgreich einen Ausweis imitieren[22] und die angefragten Berechtigungen darstellen. Der nPA-Emulator eignet sich damit zur Prüfung von Terminals, die nicht mit dem CAT-M umgehen können.

Der nPA-Emulator implementiert zwar die vollständige EAC korrekt. Bisher gibt es aber keine preisgünstige Möglichkeit, die APDUs mit erweitertem Längenfeld auch per NFC zu übermitteln; an einer entsprechenden Lösung wird jedoch gearbeitet.[23]

Die übersichtliche Darstellung des zum Auslesen Berechtigten, der angeforderten Berechtigung und des Auslesezwecks war ein Hauptanliegen dieser Arbeit. Derzeit werden sie beim nPA-Emulator und bei `libnpa` bzw. dem CCID-Emulator detailliert als formatierter Text ausgegeben. Eine graphische Nutzerschnittstelle wäre jedoch leichter verständlich. Möglich wäre die Integration von `eid_gui.py`[24], welche bereits auf dem Openmoko Neo FreeRunner eine übersichtliche Darstellung realisiert (siehe Abbildung 3.3, S. 76)

Jede einzelne der vorgestellten Komponenten ist nicht nur auf normalen Desktop-Systemen lauffähig, sondern wurde auch auf dem Openmoko Neo FreeRunner getestet.[25] Mit OpenPACE, `libnpa`, dem CCID-Emulator, nPA-Emulator und `pcsc-relay` wurde die prototypische Umsetzung wesentlicher Kernfunktionen des CAT-M auf diesem Smartphone demonstriert. Damit kann die Fragestellung dieser Arbeit auch im Bezug auf die praktische Umsetzung des CAT-M positiv beantwortet werden.

Zusätzliche Hardware wird auf dem Openmoko benötigt, um die Kommunikation zur Chipkarte bzw. zum Terminal bereitzustellen. Die gleichzeitige Nutzung sowohl des externen Kartenlesers als auch des CCID-Emulators ist wegen dieser Beschränkung im Moment nicht möglich. Geringfügige Modifikationen des Openmoko beheben dieses Problem.[26] Einzelne Funktionen wurden mit dem Smartphone sequenziell getestet, z. B. das Ändern der eID-PIN, Nutzung des CCID-Emulators an einem PC mit eingelegtem nPA-Emulator und Nutzung des nPA-Emulators gegenüber einem Standardleser.

[21] Siehe „Aufbau und Funktionsweise“, S. 109.
[22] Vgl. Kartenidentifikationsprozess, „Kartenidentifikationsprozess“, S. 113
[23] Siehe „nPA-Kartenemulator“, S. 115.
[24] Morgner und Oepen 2011.
[25] Die Websites der einzelnen Projekte listen die Plattformen auf, mit denen sie getestet wurden und auf welchen weiteren sie lauffähig sein sollten.
[26] Siehe „Umsetzung eines mobilen Kartenlesers mit PACE-Funktion“, S. 90.

Dieses Hindernis zu überwinden und außerdem eine sichere Ausführungsumgebung auf dem Smartphone bereitzustellen, konnte aufgrund des begrenzten Umfangs dieser Arbeit nicht realisiert werden. Zumindest das Problem der sicheren Ausführungsumgebung auf dem Openmoko wurde schon in anderen Arbeiten gelöst.[27]

5.4. Konklusion für den nPA und seine Infrastruktur

Mit dem nPA werden Systeme, die ausschließlich *entweder* die eID- *oder* Signaturfunktion nutzen, sicherer als es z. B. mit vielen anderen Authentisierungsverfahren oder Signaturkarten möglich ist. Der nPA bietet aber gleichzeitig neue Angriffsvektoren und führt das Risiko der Privilege Escalation ein. Ebenfalls können hochwertige Kartenleser eingesetzt werden, bei denen einige – aber längst nicht alle – bekannten Probleme gelöst wurden.

Weitet man seinen Blick und zieht auch Modifikationen des nPA, des Dienstanbieters oder anderer Komponenten in Betracht, die dem Ausweisinhaber normalerweise nicht möglich sind, könnten Angriffe auch grundsätzlicher abgewehrt werden. In „Mitigation“, S. 58 wurden verschiedene weitere Maßnahmen vorgestellt, mit denen die elektronische Nutzung des nPA sicherer gestaltet werden könnte.

Bei der Analyse ist aufgefallen, dass sichere Ein- und Ausgabegeräte unverzichtbar sind. Bei einem Standard- oder Komfortleser ist zumindest die PIN-Eingabe sicher. Die Möglichkeiten zur Anzeige sind aber sehr begrenzt. Vor allem wenn es darum geht welches Dokument mit einem Komfortleser signiert werden soll, muss der Anwender auf die Anzeige der Client-Software vertrauen (dem Chipkartenleser bzw. dem nPA wird lediglich der Hash des Dokuments gesendet).

Fügt man aber wie beim CAT-M ein detailliertes Display und komplexe Eingabemethoden hinzu, fällt auf, dass die Client-Software nunmehr lediglich als Mittler zwischen Dienstanbieter und Kartenleser fungiert. Von ihr werden kaum sicherheitsrelevante Funktionen bereitgestellt, die nicht durch einen geeigneten Kartenleser übernommen werden könnten. Die sichere und detaillierte Anzeige, die Modifikation der angefragten Berechtigungen und die PIN-Eingabe könnten auf dem Kartenleser geschehen. Was von der Client-Software bleibt, ist die eCard-API. Diese könnte direkt bei dem Dienstanbieter realisiert werden, wo der Chipkartenleser virtuell eingesteckt wird (z. B. per USB-over-IP). Beim PC des Ausweisinhabers könnte dann ein sehr schmales Programm laufen, welches diese virtuelle Verbindung realisiert. Es müsste außerdem bei der Online-Authentisierung sicherstellen, dass der SSL/TLS-Kanal mit dem aufgerufenen Webserver zu dem EAC-Kanal passt.[28]

Hätte der Ausweisinhaber lokal lediglich seinen nPA, den CAT-M und einen Browser mit einem Plug-in, dass diese wenig komplexe Funktion realisiert, gleicht die Nutzung in gewissen Teilen einem Relay-Angriff.[29] Mit den hier vorgestellten Erweiterungen zu den Kartenlesern mit PACE-Funktion wird aber eine missbräuchliche Nutzung des nPA verhindert. Hinzu käme allerdings, dass man eine Identifikation des Ausweisinhabers mittels des Chipkartenlesers verhindern muss. Der CCID-Emulator beispielsweise randomisiert hierfür seine Seriennummer.

[27] Siehe „Sichere Ausführungsumgebung“, S. 87.
[28] Siehe „Nutzung der eID-Anwendung“, S. 20.
[29] Siehe „Relay-Angriff zum unberechtigten Ermitteln der Ausweisdaten“, S. 45.

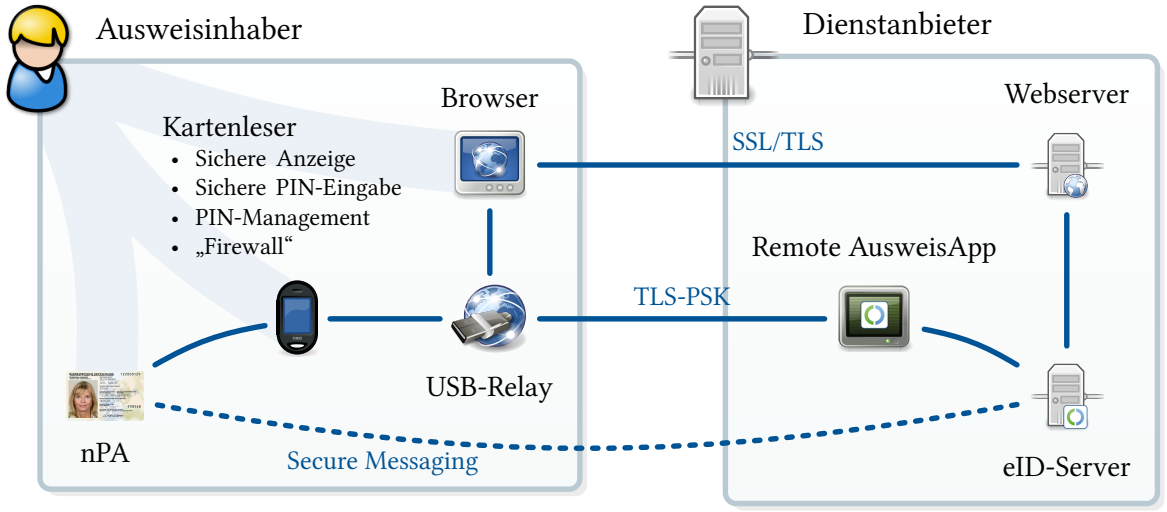


Abbildung 5.1.: Beispiel einer minimalen lokalen Client-Software. Statt der lokal installierten eCard-API wird lediglich der Zugriff auf das Lesegerät weitergeleitet. Der Gerätetreiber beim Dienstanbieter übernimmt die Ansteuerung des Lesers. Eingaben und Anzeigen geschehen beim sichern Chipkartenleser.

Ganz nebenbei würde das Problem der verschiedenen Gerätetreiber gelöst; sie müssten lediglich beim Dienstanbieter installiert sein, da der Chipkartenleser direkt auf USB-Ebene zu ihm getunnelt wird. Neben der Spezifikation des lokalen Mittlers zwischen Chipkartenleser und Client-Software beim Dienstanbieter müsste auch die Behandlung der eingeschränkten Berechtigungen modifiziert werden. Derzeit ist es dem PACE-fähigen Chipkartenleser nicht möglich, einen modifizierten CHAT an die Client-Software (bzw. den Dienstanbieter) zu melden. Entweder man fügt einen solchen Rückkanal hinzu. Oder man schickt den modifizierten CHAT schlicht nicht mehr zum Dienstanbieter. Nach erfolgreicher EAC könnte dieser dann versuchen seine Berechtigungen voll auszuschöpfen und bekäme vom nPA dann Fehlermeldungen, falls einige davon vom Ausweisinhaber am Leser eingeschränkt wurden.

A. Mögliche Kryptographische Bibliotheken zur Umsetzung der EAC

Bei der Wahl der Programmierplattform müssen aber auch nicht funktionale Anforderungen berücksichtigt werden. Bei einer sicherheitskritischen Anwendung gilt es vor allem sensitive Daten zu schützen. Bei kryptographischen Operationen muss aber z. B. auch auf die Qualität von Zufallszahlen geachtet werden. Der Anspruch dieser Implementierung an eine Plattformunabhängigkeit wurde bereits erwähnt.[1] Auch Wartbarkeit und Bedienbarkeit müssen beachtet werden. Besonders bei mobilen Geräten spielt Effizienz eine wichtige Rolle.

Bei einer Programmbibliothek stellen Bedienbarkeit, Wartbarkeit und Portabilität Erleichterungen für den Programmierer dar. Die Vernachlässigungen von Sicherheit oder Effizienz kann jedoch direkte Auswirkungen auf den Nutzer haben. Eine zu lange Ausführungszeit und hoher Ressourcenverbrauch könnte dazu führen, dass auf den Einsatz des nPA verzichtet wird. Ineffiziente Berechnungen reduzieren die sehr begrenzte Akkulaufzeit. Wird viel Zeit für die Nutzung der eID-Anwendung benötigt, könnte das unbequeme Halten des nPA am Smartphone dazu führen, dass die Chipkarte ungewollt aus dem Feld des Lesegeräts bewegt wird.

Die Effizienz verschiedener Programmierplattformen ist ein kontrovers diskutiertes Thema. Letztlich kann man meist nur Implementierungen auf verschiedenen Plattformen vergleichen. Das Ergebnis eines solchen Tests kann lediglich begrenzte Aussagen über die Effizienz der Programmierplattformen machen, da dies zum großen Teil auch vom Vermögen und Bemühen des Programmieres oder der Wahl des Compilers oder der Systemarchitektur abhängig ist. In dem Computer Language Benchmarks Game[2] werden viele verschiedene Implementierungen unterschiedlicher Probleme auf diversen Plattformen miteinander verglichen. In jedem der durchgeführten Tests erweisen sich die Implementierungen in C oder C++ als am effizientesten.

Besonders Programmierplattformen, die multiparadigmatisch, statisch und kompiliert sind, schneiden am besten ab.[3] Dies deckt sich mit der Beobachtung, dass Java-Implementierungen der Protokolle des nPA relativ lange Ausführungszeiten haben.[4] Daher wird sich hier in der Betrachtung möglicher kryptographischer Plattformen auf C oder C++ Bibliotheken beschränkt.

OpenSSL[5] ist ein Toolkit zur Unterstützung von SSL/TLS. Dabei werden auch alle für die EAC notwendigen kryptographischen Verfahren unterstützt (siehe Tabelle 4.1). CMAC

[1] Siehe „Umsetzung eines mobilen Kartenlesers mit PACE-Funktion“, S. 90.

[2] `http://shootout.alioth.debian.org/`

[3] Ellero und Pellegrini 2011, S. 13.

[4] Siehe „nPA-Nutzung am Mobiltelefon in anderen Arbeiten“, S. 68.

ist ab Version 1.0.1 verfügbar. Für ältere Versionen lässt sich die CMAC-Funktionalität durch ein Patch[6] nachrüsten.

OpenSSL unterstützt die Erstellung, Manipulation und Verwendung von kryptographischen Algorithmen durch sogenannte „Engine-Objekte". Darüber werden zur Laufzeit alternative Umsetzungen der kryptographischen Operationen geladen. Insbesondere können externe Hardware-Module wie etwa ein TPM oder eine Smartcard verwendet werden.

OpenSSL ist ein Quasi-Standard, der von vielen sicherheitsrelevanten Anwendungen verwendet wird.[7] Die große Nutzerbasis auch im professionellen Bereich hat dazu beigetragen, dass bereits viele Fehler gefunden werden konnten. Die Qualität von OpenSSL wurde in der FIPS 140-2 Validierung bestätigt.[8]

GnuTLS[9] dient ebenfalls der Umsetzung von SSL/TLS. Wie bei OpenSSL sind hier alle für die EAC nötigen kryptographischen Verfahren implementiert. Die Kryptographie mit elliptischen Kurven ist seit März 2011 verfügbar und als „experimentell" standardmäßig deaktiviert.[10]

Auch GnuTLS unterstützt externe Hardware-Module zur Durchführung kryptographischer Operationen oder zur Nutzung eines Hardware-Zufallszahlengenerators. Deren Einbindung ist über PKCS#11 URIs möglich. Obwohl sich dieser Standard noch in der Entwicklung befindet, verspricht er, den Umgang mit externen Modulen über verschiedene Anwendungen hinweg zu regeln.

Botan[11] stellt ebenfalls eine umfangreiche Sammlung von kryptographischen Primitiven bereit. Wie OpenSSL und GnuTLS auch, wird es bereits seit mehr als zehn Jahren entwickelt. Es wird versucht, die Programmierschnittstellen simpel und schlank zu halten. Sensitive Speicherbereiche werden automatisch gegen eine Auslagerung auf einen Massenspeicher geschützt. In OpenSSL oder GnuTLS hingegen wird kein Memory Locking angewandt. Als Besonderheit wird von den Entwicklern das Pipelining-Konzept bei der Bearbeitung von Nachrichten genannt – ein Aspekt der in OpenSSL durch „Bio-Objekte" realisiert werden kann. Kryptographische Hardware-Module werden ebenfalls über spezielle Engine-Objekte eingebunden. Selbst die OpenSSL-Bibliothek kann als Back-End für kryptographische Operationen genutzt werden.

Im Auftrag des BSI wurde 2007 die auf Botan aufbauende Bibliothek InSiTo entwickelt.[12] Im Speziellen sollte eine Unterstützung von CV-Zertifikaten nach BSI TR-03110 1.0.1 hinzugefügt werden. Bereits damals waren alle für den nPA nötigen kryptographischen Protokolle in Botan bzw. InSiTo integriert.

Crypto++[13] ist mit seinem ersten Erscheinen 1995 das älteste der hier genannten Projekte. Die Nutzung von Crypto++ im akademischen Umfeld hat öfter dazu geführt, dass sehr neue

[6] `http://rt.openssl.org/Ticket/Display.html?id=2162&user=guest&pass=guest`

[7] Selhorst, Stüble und Teerkorn 2008, S. 66.

[8] `http://csrc.nist.gov/groups/STM/cmvp/documents/140-1/140val-all.htm#1051`

[10] `http://lists.gnu.org/archive/html/info-gnu/2011-05/msg00015.html`

[12] `http://www.flexsecure.eu/insito/`

Algorithmen in die Bibliothek aufgenommen wurden. So wurden z. B. die beim nPA genutzten elliptischen Brainpool-Kurven bereits 2009 integriert. Trotzdem konnte die Qualität von Crypto++ in einer FIPS 140-2 Validierung bestätigt werden.[14]

Bei der Datenbearbeitung wird ebenfalls die stufenweise Verarbeitung per Pipelining genutzt. Obwohl CPU-gestützte kryptographische Operationen in Crypto++ implementiert sind, ist die Verwendung von anderen Hardware-Modulen nicht möglich.

Abschließend lässt sich sagen, dass sich die hier betrachteten kryptographischen Bibliotheken sehr ähnlich sind. Sie bieten allesamt eine ausreichende Grundlage zur Nutzung des nPA. Lediglich Crypto++ ist nicht in der Lage, kryptographische Hardware-Module einzubinden. GnuTLS setzt hierbei sogar auf ein flexibles, standardisiertes Interface. Die Nutzung von Botan ist relativ komfortabel verglichen z. B. mit OpenSSL. Aber Zertifizierungen von OpenSSL sowie Crypto++ zeigen, dass sie besonders für den Einsatz in sicherheitskritischen Umgebungen geeignet sind. Vorarbeit zur Nutzung des nPA wurde bereits in Botan bzw. InSiTo und Crypto++ geleistet.

[14] `http://csrc.nist.gov/groups/STM/cmvp/documents/140-1/1401val2007.htm#819`

B. Mögliche Middleware zum Chipkartenleser

CT-API Die CT-API definiert ein verhältnismäßig schmales Interface zu Chipkartenlesern. Mit nur drei verschiedenen Methoden[1] wird, abstrahiert von den Eigenheiten des Chipkartenlesers, eine Kommunikation zur Chipkarte ermöglicht. Im Gegensatz zu den anderen beiden Varianten der Middleware für Leser können auch Speicherkarten angesprochen werden. Die CT-API definiert keine Möglichkeit zum Auslösen spezieller Funktionen des Chipkartenlesers, z. B. der sicheren PIN-Verifikation. Die Spezifikation zu multifunktionalen Terminals (MKT) füllt diese Lücke, so dass Chipkartenleser sogar als vertrauenswürdige Anzeige genutzt werden können. In diesem Rahmen werden beispielsweise auch die komplexen Funktionen des Electronic Health Card Terminal für die elektronische Gesundheitskarte ausgelöst. Die CT-API war zwar einer der ersten Standards zum Zugriff auf Chipkarten bzw. -leser, wird jedoch vor allem im deutschen Raum eingesetzt.[2] Sie gilt technisch als überholt, auch wenn sie nach wie vor ihre Rolle als Industriestandard beibehalten wird.[3]

OpenCard-Framework Die Weiterentwicklung des OpenCard-Frameworks stoppte bereits 2000.[4] Trotzdem gibt es mit OpenSCDP[5] mindestens ein aktives Projekt, welches auf dieser Middleware aufsetzt und seit OpenSCDP 3.7 sogar Zugriff auf den nPA erlaubt.[6] Ziel des OpenCard-Frameworks war es zunächst, den Zugriff auf Chipkartenleser zu vereinheitlichen.[7] Wollten Chipkartenhersteller ihren Leser integrieren, mussten sie lediglich eine entsprechende Java-Klasse mit vorgegebenem Interface implementieren. Chipkartenapplikationen konnten Puffer an eine Karte senden oder empfangen. Mit OpenSCDP hingegen wurde das Framework wesentlich erweitert. Neben einem PC/SC-Wrapper, der bereits im OpenCard-Framework zu finden war, ist es nun auch möglich, auf Kartenleser über die Java Smart Card I/O API[8] und CT-API zuzugreifen. Außerdem

[1] CT-API 1.1.1, S. 3–7 definiert `CT_init`, `CT_data` und `CT_close`, zum Initialisieren und Terminieren von Verbindungen sowie zum Datenaustausch zur Karte.

[2] Rankl und Effing 2008, S. 260–261.

[3] Rankl und Effing 2008, S. 261.

[4] Sauveron 2008, S. 280.

[5] `http://www.openscdp.org/`

[6] OpenSCDP 3.7 erschien erst als die wesentlichen Teile der praktische Arbeit des Autors abgeschlossen waren. Daher konnten die nPA-Unterstützung in OpenSCDP für die praktische Umsetzung des mobilen Chipkartenlesers für den nPA nicht berücksichtigt werden.

[7] Sauveron 2008, S. 279.

[8] Die Java Smart Card I/O API ist derzeit im wesentlichen ein Java-Wrapper zu PC/SC, könnte aber auch mit anderen Back-Ends ergänzt werden.

werden abstrakte Methoden bereitgestellt, um Kommandos an verschiedene Kartentypen abzusetzen, ohne dass in der Applikation die konkrete Struktur der APDU vorgegeben werden muss. OpenSCDP stellt zudem eine Reihe von Skripten bereit, mit der spezielle Chipkartenfunktionen über ein Eclipse-Plug-in entwickelt und getestet werden können. So können beispielsweise die EAC mit dem nPA, BAC mit MRTDs oder Transaktionen mit EMV-Karten durchgeführt werden.

PC/SC Im Heimbereich ist PC/SC die mit Abstand am häufigsten verwendete Schnittstelle zu Chipkartenlesern. Der Standard wird stetig weiterentwickelt, was beispielsweise die jüngste Integration von PACE zeigt.[9] Er bietet generische Kommandos zur sicheren PIN-Behandlung[10] und zur Nutzung spezieller Funktionen multifunktionaler Terminals.[11] Auch Kommandos zum Zugriff auf das Display[12] oder die Tastatur des Kartenlesers[13] sind spezifiziert. Diese bergen jedoch ein potenzielles Risiko, wenn bösartige Applikationen damit z. B. eine PIN-Eingabe nachstellen. Spezielle Funktionen des Chipkartenlesers lassen sich mittels `SCardControl` auslösen, wohingegen `SCardTransmit` zum regulären Datenaustausch mit der Chipkarte gedacht ist. Letztere Funktion wird tatsächlich aber oft auch mittels Pseudo-APDUs[14] genutzt, um direkt den Kartenleser anzusprechen. Beispielsweise definiert BSI TR-03119 1.2, S. 14 vier solcher Kommandos, um die Bezeichnung und Firmwareversion abzufragen.

[9] PC/SC Pt. 10 amd. 1.

[10] Z. B. `FEATURE_VERIFY_PIN_DIRECT`, `FEATURE_MODIFY_PIN_DIRECT` (PC/SC Pt. 10, S. 23)

[11] Z. B. `FEATURE_MCT_READERDIRECT` (PC/SC Pt. 10, S. 25)

[12] Z. B. `FEATURE_WRITE_DISPLAY` (PC/SC Pt. 10, S. 31)

[13] Z. B. `FEATURE_GET_KEY` (PC/SC Pt. 10, S. 24)

[14] Pseudo-APDU haben die durch ISO/IEC 7816-4 vorgegebene Struktur. Das CLA-Byte besitzt jedoch den im ISO-Standard nicht erlaubten Wert `0xFF`. Auch die Antworten des Kartenlesers orientieren sich an dem regulärer Chipkarten; sie enthalten mindestens die beiden Statusbytes (z. B. `0x9000` falls kein Fehler aufgetreten ist).

C. EF.CardSecurity

Das folgende `EF.CardSecurity` wurde bei einem nPA erster Generation nach PACE und erfolgreicher TA vom Dienstanbieter gelesen.

```
  0:d=0  hl=4 l=1440 cons: SEQUENCE
  4:d=1  hl=2 l=   9 prim:  OBJECT                :pkcs7-
    signedData
 15:d=1  hl=4 l=1425 cons:  cont [ 0 ]
 19:d=2  hl=4 l=1421 cons:   SEQUENCE
 23:d=3  hl=2 l=   1 prim:    INTEGER            :03
 26:d=3  hl=2 l=  15 cons:    SET
 28:d=4  hl=2 l=  13 cons:     SEQUENCE
 30:d=5  hl=2 l=   9 prim:      OBJECT           :sha256
 41:d=5  hl=2 l=   0 prim:      NULL
 43:d=3  hl=4 l= 328 cons:    SEQUENCE
 47:d=4  hl=2 l=   8 prim:     OBJECT            :id-
    SecurityObject
 57:d=4  hl=4 l= 314 cons:     cont [ 0 ]
 61:d=5  hl=4 l= 310 prim:      OCTET STRING
 65:d=0  hl=4 l= 306 cons:       SET
 69:d=1  hl=2 l=  13 cons:        SEQUENCE
 71:d=2  hl=2 l=   8 prim:         OBJECT        :id-TA
 81:d=2  hl=2 l=   1 prim:         INTEGER       :02
 84:d=1  hl=2 l=  18 cons:        SEQUENCE
 86:d=2  hl=2 l=  10 prim:         OBJECT        :id-CA-ECDH-AES
    -CBC-CMAC-128
 98:d=2  hl=2 l=   1 prim:         INTEGER       :02
101:d=2  hl=2 l=   1 prim:         INTEGER       :41
104:d=1  hl=2 l=  18 cons:        SEQUENCE
106:d=2  hl=2 l=  10 prim:         OBJECT        :id-PACE-ECDH-
   GM-AES-CBC-CMAC-128
118:d=2  hl=2 l=   1 prim:         INTEGER       :02
121:d=2  hl=2 l=   1 prim:         INTEGER       :0D
124:d=1  hl=2 l=  23 cons:        SEQUENCE
126:d=2  hl=2 l=  10 prim:         OBJECT        :id-RI-ECDH-SHA
   -256
138:d=2  hl=2 l=   9 cons:         SEQUENCE
140:d=3  hl=2 l=   1 prim:          INTEGER      :01
143:d=3  hl=2 l=   1 prim:          INTEGER      :43
146:d=3  hl=2 l=   1 prim:          BOOLEAN      :255
149:d=1  hl=2 l=  23 cons:        SEQUENCE
151:d=2  hl=2 l=  10 prim:         OBJECT        :id-RI-ECDH-SHA
   -256
163:d=2  hl=2 l=   9 cons:         SEQUENCE
165:d=3  hl=2 l=   1 prim:          INTEGER      :01
```

```
 168:d=3  hl=2 l=   1 prim:          INTEGER         :44
 171:d=3  hl=2 l=   1 prim:          BOOLEAN         :0
 174:d=1  hl=2 l=  25 cons:        SEQUENCE
 176:d=2  hl=2 l=   9 prim:         OBJECT           :id-RI-ECDH
 187:d=2  hl=2 l=  12 cons:         SEQUENCE
 189:d=3  hl=2 l=   7 prim:          OBJECT          :
    standardizedDomainParameters
 198:d=3  hl=2 l=   1 prim:          INTEGER         :0D
 201:d=1  hl=2 l=  28 cons:        SEQUENCE
 203:d=2  hl=2 l=   9 prim:         OBJECT           :id-CA-ECDH
 214:d=2  hl=2 l=  12 cons:         SEQUENCE
 216:d=3  hl=2 l=   7 prim:          OBJECT          :
    standardizedDomainParameters
 225:d=3  hl=2 l=   1 prim:          INTEGER         :0D
 228:d=2  hl=2 l=   1 prim:         INTEGER          :41
 231:d=1  hl=2 l=  42 cons:        SEQUENCE
 233:d=2  hl=2 l=   8 prim:         OBJECT           :id-CI
 243:d=2  hl=2 l=  30 prim:         IA5STRING        :http://bsi.
    bund.de/cif/npa.xml
 275:d=1  hl=2 l=  98 cons:        SEQUENCE
 277:d=2  hl=2 l=   9 prim:         OBJECT           :id-PK-ECDH
 288:d=2  hl=2 l=  82 cons:         SEQUENCE
 290:d=3  hl=2 l=  12 cons:          SEQUENCE
 292:d=4  hl=2 l=   7 prim:           OBJECT         :
    standardizedDomainParameters
 301:d=4  hl=2 l=   1 prim:           INTEGER        :0D
 239:d=3  hl=2 l=  66 prim:          BIT STRING
00 04 92 5d b4 e1 7a de-58 20 9f 96 fa a0 7f 1f   ...]..z.X ......
8a 22 3f 82 3f 96 cc 5d-78 cb ef 5d 17 42 20 88   ."?.?..]x..].B .
fd d5 8e 56 bc 42 50 de-33 46 b3 c8 32 ca e4 86   ...V.BP.3F..2...
35 fb 6c 43 78 9d e8 b3-10 2f 43 93 b4 18 e2 4a   5.lCx..../C....J
13 d9                                             ..
 372:d=2  hl=2 l=   1 prim:          INTEGER          :41
 375:d=3  hl=4 l= 796 cons:    cont [ 0 ]
 379:d=4  hl=4 l= 792 cons:     SEQUENCE
 383:d=5  hl=4 l= 700 cons:      SEQUENCE
 387:d=6  hl=2 l=   3 cons:       cont [ 0 ]
 389:d=7  hl=2 l=   1 prim:        INTEGER           :02
 392:d=6  hl=2 l=   2 prim:       INTEGER           :015C
 396:d=6  hl=2 l=  12 cons:       SEQUENCE
 398:d=7  hl=2 l=   8 prim:        OBJECT           :ecdsa-with-
    SHA256
 408:d=7  hl=2 l=   0 prim:        NULL
 410:d=6  hl=2 l=  79 cons:       SEQUENCE
 412:d=7  hl=2 l=  11 cons:        SET
 414:d=8  hl=2 l=   9 cons:         SEQUENCE
 416:d=9  hl=2 l=   3 prim:          OBJECT          :countryName
 421:d=9  hl=2 l=   2 prim:          PRINTABLESTRING :DE
 425:d=7  hl=2 l=  13 cons:        SET
 427:d=8  hl=2 l=  11 cons:         SEQUENCE
 429:d=9  hl=2 l=   3 prim:          OBJECT          :
    organizationName
```

```
 434:d=9  hl=2 l=   4 prim:          UTF8STRING      :bund
 440:d=7  hl=2 l=  12 cons:        SET
 442:d=8  hl=2 l=  10 cons:         SEQUENCE
 444:d=9  hl=2 l=   3 prim:          OBJECT          :
    organizationalUnitName
 449:d=9  hl=2 l=   3 prim:          UTF8STRING      :bsi
 454:d=7  hl=2 l=  12 cons:        SET
 456:d=8  hl=2 l=  10 cons:         SEQUENCE
 458:d=9  hl=2 l=   3 prim:          OBJECT          :serialNumber
 463:d=9  hl=2 l=   3 prim:          PRINTABLESTRING :013
 468:d=7  hl=2 l=  21 cons:        SET
 470:d=8  hl=2 l=  19 cons:         SEQUENCE
 472:d=9  hl=2 l=   3 prim:          OBJECT          :commonName
 477:d=9  hl=2 l=  12 prim:          UTF8STRING      :csca-germany
 491:d=6  hl=2 l=  30 cons:       SEQUENCE
 493:d=7  hl=2 l=  13 prim:        UTCTIME           :101005110752Z
 508:d=7  hl=2 l=  13 prim:        UTCTIME           :210405083947Z
 523:d=6  hl=2 l=  71 cons:       SEQUENCE
 525:d=7  hl=2 l=  11 cons:        SET
 527:d=8  hl=2 l=   9 cons:         SEQUENCE
 529:d=9  hl=2 l=   3 prim:          OBJECT          :countryName
 534:d=9  hl=2 l=   2 prim:          PRINTABLESTRING :DE
 538:d=7  hl=2 l=  29 cons:        SET
 540:d=8  hl=2 l=  27 cons:         SEQUENCE
 542:d=9  hl=2 l=   3 prim:          OBJECT          :
    organizationName
 547:d=9  hl=2 l=  20 prim:          UTF8STRING      :
    Bundesdruckerei GmbH
 569:d=7  hl=2 l=  12 cons:        SET
 571:d=8  hl=2 l=  10 cons:         SEQUENCE
 573:d=9  hl=2 l=   3 prim:          OBJECT          :serialNumber
 578:d=9  hl=2 l=   3 prim:          PRINTABLESTRING :074
 583:d=7  hl=2 l=  11 cons:        SET
 585:d=8  hl=2 l=   9 cons:         SEQUENCE
 587:d=9  hl=2 l=   3 prim:          OBJECT          :commonName
 592:d=9  hl=2 l=   2 prim:          UTF8STRING      :DS
 596:d=6  hl=4 l= 275 cons:       SEQUENCE
 600:d=7  hl=3 l= 212 cons:        SEQUENCE
 603:d=8  hl=2 l=   7 prim:         OBJECT          :id-ecPublicKey
 612:d=8  hl=3 l= 200 cons:         SEQUENCE
 615:d=9  hl=2 l=   1 prim:          INTEGER         :01
 618:d=9  hl=2 l=  40 cons:          SEQUENCE
 620:d=10 hl=2 l=   7 prim:           OBJECT          :prime-field
 629:d=10 hl=2 l=  29 prim:           INTEGER         :
    D7C134AA264366862A18302575D1D787B09F075797DA89F57EC8C0FF
 660:d=9  hl=2 l=  60 cons:          SEQUENCE
 662:d=10 hl=2 l=  28 prim:           OCTET STRING
68 a5 e6 2c a9 ce 6c 1c-29 98 03 a6 c1 53 0b 51   h..,..l.)....S.Q
4e 18 2a d8 b0 04 2a 59-ca d2 9f 43               N.*...*Y...C
 692:d=10 hl=2 l=  28 prim:           OCTET STRING
25 80 f6 3c cf e4 41 38-87 07 13 b1 a9 23 69 e3   %..<..A8.....#i.
3e 21 35 d2 66 db b3 72-38 6c 40 0b               >!5.f..r8l@.
```

```
 722:d=9  hl=2 l=  57 prim:          OCTET STRING
04 0d 90 29 ad 2c 7e 5c-f4 34 08 23 b2 a8 7d c6   ...).,~\.4.#..}.
8c 9e 4c e3 17 4c 1e 6e-fd ee 12 c0 7d 58 aa 56   ..L..L.n....}X.V
f7 72 c0 72 6f 24 c6 b8-9e 4e cd ac 24 35 4b 9e   .r.ro$...N..$5K.
99 ca a3 f6 d3 76 14 02-cd                        .....v...
 781:d=9  hl=2 l=  29 prim:          INTEGER         :
    D7C134AA264366862A18302575D0FB98D116BC4B6DDEBCA3A5A7939F
 812:d=9  hl=2 l=   1 prim:          INTEGER         :01
 815:d=7  hl=2 l=  58 prim:        BIT STRING
00 04 3d 6a 7c 2a 6f 20-5f 83 9b 04 14 ec 58 c6   ..=j|*o _.....X.
c7 1b 75 f5 15 de c3 ae-73 3b 5f 47 88 dd c8 15   ..u.....s;_G....
f0 5b c1 f6 53 8f d9 69-54 e1 f8 40 a2 e2 18 99   .[..S..iT..@....
62 cc aa 14 90 08 24 c7-dd b9                     b.....$...
 875:d=6  hl=3 l= 209 cons:       cont [ 3 ]
 878:d=7  hl=3 l= 206 cons:        SEQUENCE
 881:d=8  hl=2 l=  14 cons:         SEQUENCE
 883:d=9  hl=2 l=   3 prim:          OBJECT          :X509v3 Key
    Usage
 888:d=9  hl=2 l=   1 prim:          BOOLEAN         :255
 891:d=9  hl=2 l=   4 prim:          OCTET STRING
03 02 07 80                                       ....
 897:d=8  hl=2 l=  31 cons:         SEQUENCE
 899:d=9  hl=2 l=   3 prim:          OBJECT          :X509v3
    Authority Key Identifier
 904:d=9  hl=2 l=  24 prim:          OCTET STRING
30 16 80 14 60 44 f2 45-f2 e0 71 d4 d5 64 f4 e5   0...`D.E..q..d..
77 d6 36 69 db eb 18 59-                          w.6i...Y
 930:d=8  hl=2 l=  65 cons:         SEQUENCE
 932:d=9  hl=2 l=   3 prim:          OBJECT          :X509v3
    Certificate Policies
 937:d=9  hl=2 l=  58 prim:          OCTET STRING
30 38 30 36 06 09 04 00-7f 00 07 03 01 01 01 30   0806...........0
29 30 27 06 08 2b 06 01-05 05 07 02 01 16 1b 68   )0'..+.........h
74 74 70 3a 2f 2f 77 77-77 2e 62 73 69 2e 62 75   ttp://www.bsi.bu
6e 64 2e 64 65 2f 63 73-63 61                     nd.de/csca
 997:d=8  hl=2 l=  43 cons:         SEQUENCE
 999:d=9  hl=2 l=   9 prim:          OBJECT          :localKeyID
1010:d=9  hl=2 l=  30 prim:          OCTET STRING
04 1c 31 2e 32 2e 32 37-36 2e 30 2e 38 30 2e 31   ..1.2.276.0.80.1
2e 31 32 2e 30 2e 32 30-2e 35 2e 31 2e 30         .12.0.20.5.1.0
1042:d=8  hl=2 l=  43 cons:         SEQUENCE
1044:d=9  hl=2 l=   3 prim:          OBJECT          :X509v3 Private
    Key Usage Period
1049:d=9  hl=2 l=  36 prim:          OCTET STRING
30 22 80 0f 32 30 31 30-31 30 30 35 31 31 30 37   0"..201010051107
35 32 5a 81 0f 32 30 31-31 30 32 30 35 30 39 33   52Z..20110205093
39 34 37 5a                                       947Z
1087:d=5  hl=2 l=  12 cons:       SEQUENCE
1089:d=6  hl=2 l=   8 prim:        OBJECT            :ecdsa-with-
   SHA256
1099:d=6  hl=2 l=   0 prim:        NULL
1101:d=5  hl=2 l=  72 prim:       BIT STRING
```

```
00 30 45 02 20 13 e9 e1-7a 9e fe 8b d7 d7 27 62   .0E. ...z.....'b
92 30 5b cc c3 2b 70 c2-b7 60 40 f4 88 30 66 62   .0[..+p..`@..0fb
26 cd 6a 4b f4 02 21 00-87 f4 71 e2 44 35 b4 c3   &.jK..!...q.D5..
4a f3 57 30 94 fb 1f 1c-2a 48 b1 3e e5 ed 67 f1   J.W0....*H.>..g.
72 6d cf 56 e3 84 e3 6f-                          rm.V...o
1175:d=3  hl=4 l= 265 cons:    SET
1179:d=4  hl=4 l= 261 cons:     SEQUENCE
1183:d=5  hl=2 l=   1 prim:      INTEGER             :01
1186:d=5  hl=2 l=  85 cons:      SEQUENCE
1188:d=6  hl=2 l=  79 cons:       SEQUENCE
1190:d=7  hl=2 l=  11 cons:        SET
1192:d=8  hl=2 l=   9 cons:         SEQUENCE
1194:d=9  hl=2 l=   3 prim:          OBJECT            :countryName
1199:d=9  hl=2 l=   2 prim:          PRINTABLESTRING   :DE
1203:d=7  hl=2 l=  13 cons:        SET
1205:d=8  hl=2 l=  11 cons:         SEQUENCE
1207:d=9  hl=2 l=   3 prim:          OBJECT            :
   organizationName
1212:d=9  hl=2 l=   4 prim:          UTF8STRING        :bund
1218:d=7  hl=2 l=  12 cons:        SET
1220:d=8  hl=2 l=  10 cons:         SEQUENCE
1222:d=9  hl=2 l=   3 prim:          OBJECT            :
   organizationalUnitName
1227:d=9  hl=2 l=   3 prim:          UTF8STRING        :bsi
1232:d=7  hl=2 l=  12 cons:        SET
1234:d=8  hl=2 l=  10 cons:         SEQUENCE
1236:d=9  hl=2 l=   3 prim:          OBJECT            :serialNumber
1241:d=9  hl=2 l=   3 prim:          PRINTABLESTRING   :013
1246:d=7  hl=2 l=  21 cons:        SET
1248:d=8  hl=2 l=  19 cons:         SEQUENCE
1250:d=9  hl=2 l=   3 prim:          OBJECT            :commonName
1255:d=9  hl=2 l=  12 prim:          UTF8STRING        :csca-germany
1269:d=6  hl=2 l=   2 prim:       INTEGER             :015C
1273:d=5  hl=2 l=  13 cons:      SEQUENCE
1275:d=6  hl=2 l=   9 prim:       OBJECT              :sha256
1286:d=6  hl=2 l=   0 prim:       NULL
1288:d=5  hl=2 l=  74 cons:      cont [ 0 ]
1290:d=6  hl=2 l=  23 cons:       SEQUENCE
1292:d=7  hl=2 l=   9 prim:        OBJECT            :contentType
1303:d=7  hl=2 l=  10 cons:        SET
1305:d=8  hl=2 l=   8 prim:         OBJECT            :id-
   SecurityObject
1315:d=6  hl=2 l=  47 cons:       SEQUENCE
1317:d=7  hl=2 l=   9 prim:        OBJECT            :messageDigest
1328:d=7  hl=2 l=  34 cons:        SET
1330:d=8  hl=2 l=  32 prim:         OCTET STRING
ef 0f da 94 2e 5a 0f 6f-c9 c5 46 ee 01 f9 10 31   .....Z.o..F....1
43 64 30 f7 5e 9d 36 54-d3 69 30 9e 8b e7 17 48   Cd0.^.6T.i0....H
1364:d=5  hl=2 l=  12 cons:      SEQUENCE
1366:d=6  hl=2 l=   8 prim:       OBJECT              :ecdsa-with-
   SHA256
1376:d=6  hl=2 l=   0 prim:       NULL
```

```
1378:d=5  hl=2 l=  64 prim:       OCTET STRING
30 3e 02 1d 00 89 75 92-5b e1 31 b7 7c 95 8c 3e   0>....u.[.1.|..>
cb 2a 5c 67 fc 5c e3 1c-bd 01 41 e3 4b c7 f0 a4   .*\g.\....A.K...
47 02 1d 00 cc 65 e6 2d-dc f2 93 96 4b 22 d7 b5   G....e.-....K"..
10 d7 81 88 07 c8 95 96-bd 34 d8 f9 bb 4c 05 27   .........4...L.'
```

Literatur

International Organization for Standardization / International Electrotechnical Commission (März 2000). *Identification cards – Contactless integrated circuit cards – Proximity cards. Transmission protocol*. ISO/IEC 14443-4.

– (Nov. 2004). *Identification cards – Integrated circuit cards. Interindustry commands for a cryptographic toolbox*. ISO/IEC 7816-8. 2.

– (Jan. 2005a). *Identification cards – Integrated circuit cards. Electronic Signals and Transmission Protocols*. ISO/IEC 7816-3. 2. Aufl.

– (Jan. 2005b). *Identification cards – Integrated circuit cards. Organization, security and commands for interchange*. ISO/IEC 7816-4. 2. Aufl.

Acıicmez, Onur u. a. (10. Jan. 2008). „A Trusted Mobile Phone Prototype“. In: *Proceedings of IEEE Consumer Communications and Networking Conference (CCNC 2008)*. Samsung Electron. R&D Center, San Jose. Las Vegas. URL: http://profsandhu.com/zhang/pub/ccnc08.pdf (besucht am 20. 09. 2011).

Adida, Ben u. a. (2009). „Phish and Chips“. In: *Security Protocols*. Hrsg. von Bruce Christianson u. a. Bd. 5087. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, S. 40–48. ISBN: 978-3-642-04903-3. DOI: 10.1007/978-3-642-04904-0_7. URL: http://dx.doi.org/10.1007/978-3-642-04904-0_7.

Adomeit, Stefanie (12. Okt. 2011). „Skimming im Supermarkt: Polizei rät zur Kontenkontrolle – Wohl über 300.000 Euro Schaden“. In: *Neue Osnabrücker Zeitung*. Hrsg. von Hermann Elstermann und Werner F. Ebke. URL: http://www.noz.de/lokales/57914905/skimming-im-supermarkt-polizei-raet-zur-kontenkontrolle--wohl-ueber-300-000-euro-schaden (besucht am 18. 01. 2012).

Anderson, Ross und Mike Bond (2009). „The Man-in-the-Middle Defence“. In: *Security Protocols*. Hrsg. von Bruce Christianson u. a. Bd. 5087. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, S. 153–156. ISBN: 978-3-642-04903-3. DOI: 10.1007/978-3-642-04904-0_20. URL: http://dx.doi.org/10.1007/978-3-642-04904-0_20.

Anderson, Ross J. (2001). *Security engineering. A Guide to Building Dependable Distributed Systems*. Wiley. ISBN: 978-0-471-38922-4.

ARIS Umfrageforschung Markt-, Media- und Sozialforschungsgesellschaft (Feb. 2010). *Studie „Neuer Personalausweis“. Aktuelle Akzeptanz in der Bevölkerung*. Studie. URL: http://www.bitkom.org/de/publikationen/38338_63570.aspx (besucht am 22. 08. 2011).

Bachfeld, Daniel (23. März 2011). „SSL-GAU zwingt Browser-Hersteller zu Updates“. In: *heise online*. Hrsg. von Christian Heise, Ansgar Heise und Christian Persson. URL: http://heise.de/-1212986 (besucht am 17. 06. 2011).

Balfanz, Dirk und Edward W. Felten (Aug. 1999). „Hand-Held Computers Can Be Bettewr Smart Cards“. In: *Proceedings of the 8th USENIX Security Symposium*. Washington, D.C, S. 15–24.

Barrett, Matthew Frederick (2005). „Towards an Open Trusted Computing Framework“. Magisterarb. Auckland, Neuseeland: The University of Auckland. URL: http://www.cs.auckland.ac.nz/~cthombor/Students/mbarrett/mbarrettThesis.pdf (besucht am 16. 09. 2011).

Bastian, Paul (7. Nov. 2011). „Display-Javakarte mit dynamischer eID-PIN für den neuen Personalausweis“. Bachelor-Thesis. Berlin: Humboldt-Universität zu Berlin. URL: http://sar.informatik.hu-berlin.de/research/publications/SAR-PR-2011-15/Bachelorarbeit_teildynamische_PIN_Zusammenfassung_.pdf (besucht am 23. 01. 2012).

Beilke, Kristian (2010). „Mobile eCard-API“. Diplomarbeit. Berlin: Humboldt-Universität zu Berlin. URL: http://sar.informatik.hu-berlin.de/research/publications/#SAR-PR-2010-12.

Bender, Jens (2. Feb. 2010a). „Standardisierung von PACE als Nachfolger von Basic Access Control: Sicherheitsbewertung und Migration“. 20. SIT-SmartCard Workshop. URL: http://www.smartcard-workshop.de/content/dam/smartcard/de/documents/WS10/Folien.zip (besucht am 23. 01. 2012).

– (2010b). „Standardization of PACE as successor of BAC: Security Evaluation and Mitigation“. In: *Tagungsband zum 20. SIT-SmartCard Workshop*. (2.–3. Feb. 2010). Hrsg. von Ulrich Waldmann. Stuttgart: Fraunhofer Verlag, S. 69–80.

Bender, Jens, Marc Fischlin und Dennis Kügler (2009). „Security Analysis of the PACE Key-Agreement Protocol“. In: *Information Security*. Hrsg. von Pierangela Samarati u. a. Bd. 5735. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, S. 33–48.

Bender, Jens u. a. „Kartenleser für den neuen deutschen Personalausweis. Vor- und Nachteile unterschiedlicher Lesertypen“. In: *DuD Datenschutz und Datensicherheit* 11/2010. Hrsg. von Benedikt Buchner u. a., S. 761–765.

Bläsing, Thomas u. a. „An Android Application Sandbox System for Suspicious Software Detection“. In: *5th International Conference on Malicious and Unwanted Software (Malware 2010) (MALWARE'2010)*. Nancy, France, France. ISBN: ISBN 978-1-4244-9353-1. URL: http://www.dai-labor.de/fileadmin/Files/Publikationen/Buchdatei/Thomas_AAS_Malware2010.pdf (besucht am 20. 09. 2011).

Borchers, Detlef (2. Mai 2011a). „Neuer Personalausweis: Eine erste Bilanz nach einem halben Jahr“. In: *heise online*. Hrsg. von Christian Heise, Ansgar Heise und Christian Persson. URL: http://heise.de/-1235748 (besucht am 20. 05. 2011).

– (2. März 2011b). „Qualifizierte AdHoc-Signatur für elektronischen Personalausweis“. In: *heise online*. Hrsg. von Christian Heise, Ansgar Heise und Christian Persson. URL: http://heise.de/-1201103 (besucht am 03. 08. 2011).

Borges, Georg (Mai 2011). „Missbrauch des elektronischen Identitätsnachweises“. In: *Tagungsband zum 12. IT-Sicherheitskongress*. Bundesamt für Sicherheit in der Informationstechnik. Gau-Algesheim: SecuMedia Verlag, S. 275–287.

Brands, Stefan und David Chaum (1994). „Distance-bounding protocols“. In: *EUROCRYPT 93*. Springer-Verlag, S. 344–359.

Braun, Johannes u. a. (Sep. 2011). „Mobile Authentisierung und Signatur“. In: *D-A-CH Security 2011*. Oldenburg. URL: http://www.cdc.informatik.tu-darmstadt.de/reports/reports/201109_DACH11_Mobile_Authentisierung_und_Signatur.pdf (besucht am 23. 08. 2011).

Brownell, David (2004). *USB Gadget API for Linux. on OMAP H2*. URL: http://www.linux-usb.org/gadget/h2-otg.html (besucht am 23. 01. 2012).

Buchmann, Johannes u. a. (2010). „Towards a mobile eCard“. In: *Tagungsband zum 13. Kryptotag*, S. 4. URL: http://www.seceng.informatik.tu-darmstadt.de/assets/Kryptotag/Tagungsband-13-Kryptotag-Darmstadt.pdf (besucht am 23. 08. 2011).

Bundesamt für Sicherheit in der Informationstechnik (2. Dez. 1999). *Application Notes and Interpretation of the Scheme (AIS)*. 1. Aufl. URL: https://www.bsi.bund.de/cae/servlet/contentblob/478152/publicationFile/30552/ais20e_pdf.pdf (besucht am 27. 07. 2011).

– (29. Nov. 2007). *Common Criteria Protection Profile. Electronic Health Card Terminal (eHCT)*. BSI-CC-PP-0032. 1.73. Bonn. URL: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Zertifizierung/ReportePP/PP0032b_pdf.pdf (besucht am 13. 09. 2011).

– (15. Dez. 2009a). *Common Criteria Protection Profile. Electronic Identity Card (ID_Card PP)*. BSI-CC-PP-0061. 1.03. Bonn. URL: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Zertifizierung/ReportePP/pp0061b_pdf.pdf (besucht am 20. 05. 2011).

– (Mai 2009b). *Technical Guideline TR-03110. Extended Access Control (EAC), Password Authenticated Connection Establishment (PACE), and Restricted Identification (RI)*. 2.01.

– (Nov. 2009c). *Technical Guideline TR-03110. Extended Access Control (EAC), Password Authenticated Connection Establishment (PACE), and Restricted Identification (RI)*. 2.02.

– (2009d). *Technical Guideline TR-03117. eCards mit kontaktloser Schnittstelle als sichere Signaturerstellungseinheit*. 1.0. URL: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/TechnischeRichtlinien/TR03117/BSI-TR-03117_pdf.pdf (besucht am 23. 07. 2011).

– (Dez. 2009e). *Technische Richtlinie TR-03119. Anforderungen an Chipkartenleser mit nPA Unterstützung*. 1.1. URL: https://www.bsi.bund.de/ContentBSI/Publikationen/TechnischeRichtlinien/tr03119/index_htm.html (besucht am 11. 06. 2011).

– (22. Sep. 2010a). *BSI weist Sicherheitsbedenken zum neuen Personalausweis erneut zurück*. Pressemitteilung. URL: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/ElekAusweise/Pressemappe_nPA/Pressemitteilung_220910_Sicherheitsbedenken_nPA.pdf (besucht am 02. 08. 2011).

– (24. Aug. 2010b). *BSI weist Sicherheitsbedenken zum neuen Personalausweis zurück*. Pressemitteilung. URL: https://www.bsi.bund.de/ContentBSI/Presse/Pressemitteilungen/Presse2010/Sicherheitsbedenken_Personalausweis_240810.html (besucht am 02. 08. 2011).

– (Okt. 2010c). *Technical Guideline TR-03110. Advanced Security Mechanisms for Machine Readable Travel Documents – Extended Access Control (EAC), Password Authenticated Connection Establishment (PACE), and Restricted Identification (RI)*. 2.05. URL: https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Publications/TechGuidelines/TR03110/TR-03110_v205_pdf.pdf (besucht am 20. 05. 2011).

– (Sep. 2010d). *Technical Guideline TR-03110. Extended Access Control (EAC), Password Authenticated Connection Establishment (PACE), and Restricted Identification (RI)*. 2.04.

– (Okt. 2010e). *Technische Richtlinie TR-03128. EAC-PKI'n für den elektronischen Personalausweis*. 1.1. URL: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/TechnischeRichtlinien/TR03128/BSI_TR-03128.pdf (besucht am 11. 06. 2011).

– (Okt. 2010f). *Technische Richtlinie TR-03130. eID-Server*. 1.4.1.

– (1. Dez. 2011). *Common Criteria Protection Profile. eID-Client based on eCard-API*. BSI-CC-PP-0066-V2. 2.05.14. Bonn. URL: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Zertifizierung/ReportePP/pp0066_V2b_pdf.pdf (besucht am 27. 03. 2012).

Bundesamt für Sicherheit in der Informationstechnik (2011). *Nach Technischen Richtlinien zertifizierte Produkte. Chipkartenleser mit nPA Unterstützung*. URL: https://www.bsi.bund.de/DE/Themen/ZertifizierungundAnerkennung/ZertifizierungnachTR/ZertifizierteProdukte/zertifizierteprodukte_node.html#doc1347416bodyText1 (besucht am 24. 07. 2011).

– (23. Mai 2011a). *Technical Guideline TR-03112. eCard-API-Framework*. 1.1.1. URL: https://www.bsi.bund.de/ContentBSI/Publikationen/TechnischeRichtlinien/tr03112/index_htm.html (besucht am 12. 06. 2011).

– (Mai 2011b). *Technische Richtlinie TR-03119. Anforderungen an Chipkartenleser mit nPA Unterstützung*. 1.2. URL: https://www.bsi.bund.de/ContentBSI/Publikationen/TechnischeRichtlinien/tr03119/index_htm.html (besucht am 24. 07. 2011).

– (Mai 2011c). *Technische Richtlinie TR-03127. Architektur elektronischer Personalausweis und elektronischer Aufenthaltstitel*. 1.14. URL: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/TechnischeRichtlinien/TR03127/BSI-TR-03127_pdf.pdf (besucht am 20. 05. 2011).

– (24. Jan. 2011d). *Worked Example for Extended Access Control (EAC). PACE, Chip Authentication and Terminal Authentication*. 1.01. URL: https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Publications/TechGuidelines/TR03110/BSI_EAC-Worked-Example.zip?__blob=publicationFile (besucht am 20. 05. 2011).

Bundeskriminalamt (10. Mai 2011). *Veröffentlichung der Zahlen zur Zahlungskartenkriminalität 2010 anlässlich der gemeinsamen Pressekonferenz des Bundeskriminalamts und der EURO Kartensysteme GmbH*. Pressemitteilung. URL: http://www.bka.de/nn_233148/DE/Presse/Pressemitteilungen/Presse2011/110510__ZahlungskartenkriminalitaetBundeslagebild.html (besucht am 04. 08. 2011).

Bundesministerium der Finanzen (2011). *Gewerblicher Rechtsschutz. Statistik für das Jahr 2010*. Pressemitteilung. URL: http://www.zoll.de/e0_downloads/f0_dont_show/zgr_jahresstatistik_2010.pdf (besucht am 04. 08. 2011).

Bundesministerium des Innern (2010a). *Der neue Personalausweis. Bildmaterial*. URL: http://www.personalausweisportal.de/DE/Presse_Service/Bildmaterial/bildmaterial_node (besucht am 24. 07. 2011).

– (2010b). *IT-Sicherheitskit für Bürgerinnen und Bürger*. URL: http://www.cio.bund.de/cln_093/DE/IT-Investitionsprogramm/IT-Sicherheit_staerken/Zuwendungsmassnahme_IT-Sicherheitskit/aktivitaeten_zuwendungsmassnahme_it_sicherheitskit_inhalt.html (besucht am 28. 07. 2011).

– (23. Nov. 2010c). *Sicherheitsrisiken und Kosten des neuen Personalausweises. Antwort der Bundesregierung auf die Kleine Anfrage der Abgeordneten Jan Korte, Petra Pau, Jens Petermann, Frank Tempel und der Fraktion DIE LINKE*. Drucksache 17/3637. URL: http://dipbt.bundestag.de/dip21/btd/17/039/1703932.pdf (besucht am 23. 01. 2012).

Bundesverwaltungsamt – Vergabestelle für Berechtigungszertifikate (Jan. 2011). *Infoblatt zur Vergabestelle für Berechtigungszertifikate*. P/ÖAnPA-VfB 01/2011. URL: http://www.bva.bund.de/cln_092/DE/Aufgaben/Abt__III/nPA/Vergabestelle/Infoblatt__Vergabestelle,templateId=raw,property=publicationFile.pdf/Infoblatt_Vergabestelle.pdf (besucht am 20. 05. 2011).

– (14. Jan. 2012). *Liste der Diensteanbieter mit Berechtigungszertifikat*. URL: http://gsb.download.bva.bund.de/VfB/npavfb.pdf (besucht am 14. 01. 2012).

bushing und marcan (29. Dez. 2008). „Console Hacking 2008: Wii Fail. Is implementation the enemy of design?“ 25th Chaos Communication Congress. URL: http://media.ccc.de/browse/congress/2008/25c3-2799-en-console_hacking_2008_wii_fail.html (besucht am 23. 01. 2012).

bushing, marcan und sven (29. Dez. 2010). „Console Hacking 2010. PS3 Epic Fail“. 27th Chaos Communication Congress. URL: http://media.ccc.de/browse/congress/2010/27c3-4087-en-console_hacking_2010.html (besucht am 23. 01. 2012).

Chothia, Tom und Vitaliy Smirnov (2010). „A Traceability Attack against e-Passports.“ In: *Financial Cryptography*. Hrsg. von Radu Sion. Bd. 6052. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, S. 20–34. ISBN: 978-3-642-14576-6.

Clulow, Jolyon u. a. (2006). „So near and yet so far. Distance-bounding attacks in wireless networks“. In: *In Security and Privacy in Ad-hoc and Sensor Networks*. Springer, S. 83–97.

Criscione, Claudio (2008). „Subverting the Xen hypervisor“. Black Hat USA 2008. URL: http://www.blackhat.com/presentations/bh-usa-08/Wojtczuk/BH_US_08_Wojtczuk_Subverting_the_Xen_Hypervisor.pdf (besucht am 29. 11. 2011).

– (2010). „Virtually Pwned. Pentesting Virtualization“. Black Hat USA 2010. URL: https://media.blackhat.com/bh-us-10/presentations/Criscione/BlackHat-USA-2010-Criscione-Virtually-Pwned-slides.pdf (besucht am 29. 11. 2011).

Cybits AG (März 2011). *Vollelektronische Personenidentifikation [verify-U] nun auch mit neuem Personalausweis möglich*. Pressemitteilung. URL: http://www.cybits.de/fileadmin/cybits-home/assets/PDFs/Pressemitteilungen/20110309_1_Pressemitteilung_nPA.pdf (besucht am 20. 05. 2011).

Dagdelen, Özgür und Marc Fischlin (Juni 2010). *Sicherheitsanalyse des EAC-Protokolls*.

Deutsche Telekom AG u. a. (7. Juni 2001). *CT-API. Anwendungsunabhängiges CardTerminal Application Programming Interface für Chipkartenanwendungen*. 1.1.1. URL: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/TechnischeRichtlinien/TR03127/BSI-TR-03127_pdf.pdf (besucht am 20. 05. 2011).

Deutsches Institut für Normung (Sep. 1983). *Informationsverarbeitung. Sinnbilder und ihre Anwendung*. DIN 66001.

Dietrich, Christian J. (Juli 2008). „Web-Authentisierung mit dem elektronischen Personalausweis (nPA)“. Magisterarb. Fachhochschule Gelsenkirchen.

Dietrich, Christian J., Christian Rossow und Norbert Pohlmann (Okt. 2010). *Restrisiken beim Einsatz der AusweisApp auf dem Bürger-PC zur Online-Authentisierung mit Penetration-Test*. URL: http://www.bmi.bund.de/SharedDocs/Downloads/DE/Themen/Sicherheit/PaesseAusweise/restrisiken.pdf?__blob=publicationFile (besucht am 06. 12. 2011).

Drimer, Saar und Steven J. Murdoch (2007). „Keep Your Enemies Close. Distance Bounding Against Smartcard Relay Attacks“. In: *SS'07: Proceedings of 16th USENIX Security Symposium*. Berkeley, CA, USA: USENIX Association, S. 1–16.

Drimer, Saar, Steven J. Murdoch und Ross J. Anderson (16. Dez. 2009). „Optimised to Fail: Card Readers for Online Banking“. In: *Financial Cryptography*. Hrsg. von Roger Dingledine und Philippe Golle. Bd. 5628. Lecture Notes in Computer Science. Springer, S. 184–200.

Eberhardt, Udo (2003). *USB 2.0. Das Praxisbuch*. Hrsg. von Hans Joachim Kelm. 2. Aufl. Franzis Verlag.

EGBGB (2011). *Einführungsgesetz zum Bürgerlichen Gesetzbuche*. Ursprüngliche Fassung vom 18. August 1896 (RGBl. S. 604), Letzte Änderung durch Art. 7 G vom 23. Juni 2011 (BGBl. I S. 1266, 1269).

Ekberg, Jan-Erik und Markku Kylänpää (14. Nov. 2007). *Mobile Trusted Module (MTM) - an introduction*. Techn. Ber. NRC-TR-2007-015. Helsinki, Finnland. URL: http://research.nokia.com/files/tr/NRC-TR-2007-015.pdf (besucht am 19. 09. 2011).

Elhage, Nelson (2011). „Virtunoid. A KVM Guest → Host privilege escalation exploit". Black Hat USA 2011. URL: https://media.blackhat.com/bh-us-11/Elhage/BH_US_11_Elhage_Virtunoid_WP.pdf (besucht am 29. 11. 2011).

Ellero, Andrea und Paola Pellegrini (2011). „Computer Language Effciency via Data Envelopment Analysis". In: *Advances in Operations Research* 2011. URL: http://downloads.hindawi.com/journals/aor/2011/154516.pdf (besucht am 29. 11. 2011).

erdgeist (21. Sep. 2010). *Praktische Demonstration erheblicher Sicherheitsprobleme bei Schweizer SuisseID und deutschem elektronischem Personalausweis*. Pressemitteilung. URL: http://ccc.de/de/updates/2010/sicherheitsprobleme-bei-suisseid-und-epa (besucht am 02. 08. 2011).

Europäische Komitee für Normung/Information Society Standardization System (2009). *Protection profiles for Secure signature creation device – Part 2: Device with key generation*. prEN 14169-1. BSI-CC-PP-0059. URL: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Zertifizierung/ReportePP/pp0059b_pdf.pdf (besucht am 20. 05. 2011).

Ferguson, Niels und Bruce Schneier (2003). *Practical Cryptography*. Indianapolis, Indiana, USA: Wiley Publishing.

Gallery, Eimear (2007). „Authorisation Issues for Mobile Code in Mobile Systems". Magisterarb. London, Großbritanien: University of London. URL: http://www.ma.rhul.ac.uk/techreports/2007/RHUL-MA-2007-3.pdf (besucht am 16. 09. 2011).

Gallery, Eimear und Chris J. Mitchell (21. Aug. 2007). „Trusted Mobile Platforms". In: *FOSAD*. Hrsg. von Alessandro Aldini und Roberto Gorrieri. Bd. 4677. Lecture Notes in Computer Science. Springer, S. 282–323. ISBN: 978-3-540-74809-0.

PAuswG (2010). *Gesetz über Personalausweise und den elektronischen Identitätsnachweis*. Letzte Neufassung vom 18. Juni 2009 (BGBl. I S. 1346), Letzte Änderung durch Art. 2 G vom 20. Juli 2007 (BGBl. I S. 1566, 1570).

SigG (2009). *Gesetz über Rahmenbedingungen für elektronische Signaturen*. Letzte Neufassung vom 16. Mai 2001 (BGBl. I S. 876), Letzte Änderung durch Art. 4 des Gesetzes vom 17. Juli 2009 (BGBl. I S. 2091, 2095).

Grote, Jasper Hugo u. a. (30. Sep. 2010). *Vom Client zur App. Ideenkatalog zur Gestaltung der Software zum Einsatz des neuen Personalausweises*.

Hancke, Gerhard P. (2005). *A Practical Relay Attack on ISO 14443 Proximity Cards*. Techn. Ber. Cambridge: University of Cambridge.

Hancke, Gerhard P., Keith Mayes und Konstantinos Markantonakis (2009). „Confidence in Smart Token Proximity: Relay Attacks Revisited". In: *Computers & Security* 28.7, S. 615–627.

haroon@thinkst.com (2011). *On Screen Keyboards Considered Harmful Today. (If the 90's are coming back, we are bringing back „Shoulder Surfing")*. Techn. Ber. URL: http://thinkst.com/stuff/ocv/osk-thinkst.pdf (besucht am 26. 08. 2011).

Hopper, Nicholas J. und Manuel Blum (Mai 2000). *A Secure Human-Computer Authentication Scheme*. Techn. Ber. Pittsburgh: Carnegie Mellon University. URL: http://www.aladdin.cs.cmu.edu/papers/pdfs/y2001/manuel_blum.pdf (besucht am 26. 08. 2011).

Horsch, Moritz (Sep. 2009). „MobilePACE. Password Authenticated Connection Establishment implementation on mobile devices". Bachelor-Thesis. Technische Universität Darmstadt. URL: http://www.cdc.informatik.tu-darmstadt.de/reports/reports/Moritz_Horsch.bachelor.pdf (besucht am 23. 08. 2011).

– (Juli 2011). „Mobile Authentisierung mit dem neuen Personalausweis (MONA)". Magisterarb. Darmstadt: Technische Universität Darmstadt. URL: http://www.cdc.informatik.tu-darmstadt.de/reports/reports/Moritz_Horsch_MONA.master.pdf (besucht am 23. 08. 2011).

Hwang, Joo-Young u. a. (10. Jan. 2008). „Xen on ARM. System Virtualization Using Xen Hypervisor for ARM-Based Secure Mobile Phones". In: *Proceedings of IEEE Consumer Communications and Networking Conference (CCNC 2008)*. Samsung Electron. R&D Center, San Jose. Las Vegas.

PC/SC Pt. 10 (Apr. 2010). *IFDs with Secure PIN Entry Capabilities*. 2.02.08. PC/SC Workgroup. URL: http://www.pcscworkgroup.com/specifications/files/pcsc10_v2.02.08.pdf.

PC/SC Pt. 10 amd. 1 (3. Juni 2011). *IFDs with Secure PIN Entry Capabilities*. AMENDMENT 1. 2.02.08. PC/SC Workgroup. URL: http://www.pcscworkgroup.com/specifications/files/pcsc10_v2.02.08_AMD1.pdf.

International Civil Aviation Organisation (2008a). *Machine Readable Travel Documents*. Doc 9303. Bd. 1.3: *MRtds with Machine Readable Data Stored in Optical Character Recognition Format*. 3. Aufl. URL: http://www2.icao.int/en/MRTD/Downloads/Doc%209303/Doc%209303%20English/Doc%209303%20Part%203%20Vol%201.pdf (besucht am 20. 05. 2011).

– (2008b). *Machine Readable Travel Documents*. Doc 9303. Bd. 2.3: *Specifications for Electronically Enabled MRtds with Biometric Identification Capability*. 3. Aufl. URL: http://www2.icao.int/en/MRTD/Downloads/Doc%209303/Doc%209303%20English/Doc%209303%20Part%203%20Vol%202.pdf (besucht am 20. 05. 2011).

– (Nov. 2010a). Supplemental Access Control for Machine Readable Travel Documents. In: *Machine Readable Travel Documents*. 1.01.

– (11. Nov. 2010b). *Supplemental Access Control for Machine Readable Travel Documents*. Techn. Ber. URL: http://www2.icao.int/en/MRTD/Downloads/Technical%20Reports/Technical%20Report.pdf (besucht am 10. 10. 2011).

ISO/IEC 14443 Eavesdropping and Activation Distance (23. Jan. 2008). *13.56 MHz proximity smart cards*. 144310. Techn. Ber.

Kampe, Ingo (1. Juli 2010). „NFC-Telefon als PACE-fähiges Lesegerät für elektronische Ausweisdokumente". Diplomarbeit. Berlin: Humboldt-Universität zu Berlin. URL: http://sar/research/publications/SAR-PR-2010-09/SAR-PR-2010-09_.pdf (besucht am 23. 01. 2012).

Kc, Gaurav S. und Paul A. Karger (11. Nov. 2005). *Preventing Attacks on Machine Readable Travel Documents (MRTDs)*. URL: http://eprint.iacr.org/2005/404.

Kiefer, Franziskus (Juli 2010). „Effiziente Implementierung des PACE- und EAC-Protokolls für mobile Geräte". Bachelor-Thesis. Darmstadt: Technische Universität Darmstadt. URL: http://www.cdc.informatik.tu-darmstadt.de/reports/reports/Moritz_Horsch_MONA.master.pdf (besucht am 23. 08. 2011).

KOBIL Systems GmbH (21. Okt. 2010). *KOBIL präsentiert neue Personalausweis Produktlinie. Sicherheitsspezialist KOBIL entwickelt Lösungen für neuen Personalausweis*. Pressemitteilung. URL: http://www.kobil.com/nc/de/presse-news/singleview2006/article/kobil-praesentiert-neue-personalausweis-produktlinie.html (besucht am 01. 09. 2011).

Kossel, Axel (22. Sep. 2010). „CCC zeigt Sicherheitsprobleme beim elektronischen Personalausweis auf [Update]". In: *heise online*. Hrsg. von Christian Heise, Ansgar Heise und Christian Persson. URL: http://heise.de/-1083649 (besucht am 17. 01. 2012).

Kraftfahrt-Bundesamt (2011). *Onlineantrag Punkteauskunft aus dem Verkehrszentralregister (VZR)*. URL: https://www.kba-online.de/vzronline (besucht am 03. 08. 2011).

Labs, Lutz (16. Feb. 2011). „VMware virtualisiert Android-Smartphones". In: *heise online*. Hrsg. von Christian Heise, Ansgar Heise und Christian Persson. URL: http://heise.de/-1191196 (besucht am 20. 09. 2011).

Linux USB Project (2005). *Linux-USB Gadget API Framework*. URL: http://www.linux-usb.org/gadget (besucht am 23. 01. 2012).

Luca, Alexander De, Martin Denzel und Heinrich Hussmann (2009). *Look into my Eyes! Can you guess my Password?* Techn. Ber. München. URL: http://cups.cs.cmu.edu/soups/2009/proceedings/a7-deluca.pdf (besucht am 26. 08. 2011).

Maggi, Federico u. a. (2011). *Don't touch a word! A practical input eavesdropping attack against mobile touchscreen devices*. Techn. Ber. URL: http://home.dei.polimi.it/fmaggi/downloads/publications/2010_maggi_volpatto_gasparini_boracchi_zanero_clearshot.pdf (besucht am 26. 08. 2011).

Morgner, Frank (2010). *IFD Handler for libnfc Devices*. URL: http://sourceforge.net/projects/ifdnfc/.

Morgner, Frank und Dominik Oepen (3. Feb. 2009). „Eine virtualisierte Smartcardarchitektur für mobile Endgeräte". Studienarbeit. Humboldt-Universität zu Berlin. URL: http://sar.informatik.hu-berlin.de/research/publications/SAR-PR-2009-10/SAR-PR-2009-10_.pdf (besucht am 23. 11. 2011).

– (Dez. 2010). „‚Die gesamte Technik ist sicher'. Besitz und Wissen: Relay-Angriffe auf den neuen Personalausweis". In: *27th Chaos Communication Congress*, S. 26–31. URL: http://media.ccc.de/browse/congress/2010/27c3-4297-de-die_gesamte_technik_ist_sicher.html.
– (2011). *Virtual Smart Card Architecture*. URL: http://vsmartcard.sourceforge.net/.
– (2012). *OpenPACE. Crypto library for the PACE protocol*. URL: http://openpace.sourceforge.net/ (besucht am 23. 01. 2012).

Morgner, Frank u. a. (Mai 2011). „Mobiler Leser für den neuen Personalausweis". In: *Tagungsband zum 12. IT-Sicherheitskongress*. Gau-Algesheim: SecuMedia Verlag, S. 227–240.

Moser, Max und Thorsten Schröder (2010). *SuisseID / Smartcard USB Takeover*. URL: http://www.vimeo.com/15155073.

Muthukumaran, Divya u. a. (2008). „Measuring Integrity on Mobile Phone Systems". In: URL: http://www.cse.psu.edu/~tjaeger/papers/sacmat08.pdf (besucht am 23. 01. 2012).

Müller, Thomas, Hrsg. (2008). *Trusted Computing Systeme. Konzepte und Anforderungen*. Zürich, Schweiz: Springer Berlin Heidelberg. ISBN: 978-3-540-76410-6. DOI: 10.1007/978-3-540-76410-6. URL: http://www.springerlink.com/content/978-3-540-76409-0/ (besucht am 16. 09. 2011).

Müller, Wolf, Frank Morgner und Dominik Oepen (2011). „Mobiles Szenario für den neuen Personalausweis“. In: *Tagungsband zum 21. SIT-SmartCard Workshop*. (2.–3. Feb. 2011). Hrsg. von Ulrich Waldmann. Stuttgart: Fraunhofer Verlag, S. 179–188.

Nauman, Mohammad u. a. (2010). „Beyond Kernel-level Integrity Measurement: Enabling Remote Attestation for the Android Platform“. In: URL: http://profsandhu.com/zhang/pub/trust10-android.pdf (besucht am 19. 09. 2011).

Oepen, Dominik (1. Sep. 2010). „Authentisierung im mobilen Web. Zur Usability eID basierter Authentisierung auf einem NFC Handy“. Diplomarbeit. Berlin: Humboldt-Universität zu Berlin. URL: http://sar.informatik.hu-berlin.de/research/publications/#SAR-PR-2010-12.

Oepen, Dominik und Frank Morgner (13. Mai 2011). „FOSS im Umfeld des neuen Personalausweis“. LinuxTag 2011. Berlin.

OpenLimit SignCubes AG (März 2010). *OpenLimit meldet Bürgerclient zur Zertifizierung an*. Pressemitteilung. URL: http://www.openlimit.com/assets/files/presse/d/OpenLimit-meldet-Buergerclient-zur-Zertifizierung-an.pdf (besucht am 21. 07. 2011).

Openmoko Wiki (12. Sep. 2011). *Openmoko – Open. Mobile. Free.* URL: http://wiki.openmoko.org/wiki/Neo_FreeRunner_Hardware (besucht am 06. 10. 2011).

Paget, Kristin (28. Jan. 2012). „Credit Card Fraud: The Contactless Generation“. ShmooCon 2012. URL: http://www.shmoocon.org/shmoocon_2012 (besucht am 15. 02. 2012).

Perkovi, Toni, Mario Cagalj und Nitesh Saxena (2011). *Shoulder-Surfing Safe Login in a Partially Observable Attacker Model*. Techn. Ber. URL: http://www.fesb.hr/~mcagalj/publications/POAM_FC10.pdf (besucht am 26. 08. 2011).

Plötz, Henryk (29. Dez. 2009). „Technik des neuen ePA“. 26th Chaos Communication Congress. URL: http://media.ccc.de/browse/congress/2009/26c3-3510-de-technik_des_neuen_epa.html (besucht am 12. 09. 2011).

R. Stumpf, N. Kohnert und (2010). *Plusminus*. WDR. Fernsehmagazin, Sendezeit: 24. August 2010 21:50–22:15.

Rankl, Wolfgang und Wolfgang Effing (2008). *Handbuch der Chipkarten. Aufbau - Funktionsweise - Einsatz von Smart Cards*. 5. Aufl. München: Hanser. ISBN: 978-3-446-40402-1.

Rasmussen, Kasper Bonne und Srdjan Čapkun (2010). „Realization of RF Distance Bounding“. In: *Proceedings of the USENIX Security Symposium*. USENIX Association, S. 389–402. URL: http://www.usenix.org/events/sec10/tech/full_papers/Rasmussen.pdf.

REINER Kartengeräte GmbH & Co. KG (2011). *Chipkartenleser-Shop REINER SCT. nPA*. URL: https://www.bsi.bund.de/DE/Themen/ZertifizierungundAnerkennung/ZertifizierungnachTR/ZertifizierteProdukte/zertifizierteprodukte_node.html#doc1347416bodyText1 (besucht am 28. 07. 2011).

PC/SC Pt. 3 (Juni 2007). *Requirements for PC-Connected Interface Devices*. 2.01.09. PC/SC Workgroup. URL: http://www.pcscworkgroup.com/specifications/files/pcsc3_v2.01.09.pdf.

Rogall-Grothe, Cornelia (10. Mai 2011). „Eröffnung des 12. IT-Sicherheitskongress“. 12. IT-Sicherheitskongress. Bonn.

Roth, Volker, Kai Richter und Rene Freidinger (2004). *A PIN-Entry Method Resilient Against Shoulder Surng*. Techn. Ber.

Rousseau, Ludovic (13. Mai 2011). *CCID free software driver*. URL: http://pcsclite.alioth.debian.org/ccid.html (besucht am 04. 09. 2011).

Samsung Electronics Co, Ltd. (10. Dez. 2008). *Secure Xen on ARM User's Guide*. Techn. Ber. URL: http://wiki.xensource.com/xenwiki/XenARM?action=AttachFile&do=get&target=Secure_Xen_on_ARM_User_Guide_v1_1.pdf (besucht am 20. 09. 2011).

Sauveron, Damien (2008). „Smart Card Reader APIS". In: *Smart Cards, Tokens, Security and Applications*. Hrsg. von Keith Mayes und Konstantinos Markantonakis. Springer Berlin / Heidelberg, S. 277–293.

Schejbal, Jan (9. Nov. 2010). *AusweisApp gehackt (Malware über Autoupdate)*. URL: http://janschejbal.wordpress.com/2010/11/09/ausweisapp-gehackt-malware-uber-autoupdate/ (besucht am 31. 07. 2011).

– (8. Aug. 2011a). *ePerso kann remote missbraucht werden*. URL: http://janschejbal.wordpress.com/2011/08/08/eperso-kann-remote-missbraucht-werden/ (besucht am 11. 08. 2011).

– (17. Jan. 2011b). *ePerso: PIN-Diebstahl ohne Malware*. URL: https://janschejbal.wordpress.com/2011/01/17/eperso-pin-diebstahl-ohne-malware/ (besucht am 31. 07. 2011).

Schwarz, Carsten (Mai 2011a). „Aktivieren der QES-Funktionalität auf dem neuen Personalausweis". In: *Tagungsband zum 12. IT-Sicherheitskongress*. Gau-Algesheim: SecuMedia Verlag, S. 199–210.

– (10. Mai 2011b). „Aktivieren der QES-Funktionalität auf dem neuen Personalausweis". 12. IT-Sicherheitskongress. Bonn.

Selhorst, Marcel, Christian Stüble und Felix Teerkorn (2. Okt. 2008). *TSS Studie. Einführung und Analyse des OSS TCG Software Stack TrouSerS und Werkzeuge in dessen Umfeld*. Techn. Ber. Bonn. URL: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/Studien/TSS_Apps/TSS-Apps_de.pdf (besucht am 20. 09. 2011).

Selhorst, Marcel u. a. (5. Juli 2010). „Towards a Trusted Mobile Desktop." In: *TRUST*. Hrsg. von Alessandro Acquisti, Sean W. Smith und Ahmad-Reza Sadeghi. Bd. 6101. Lecture Notes in Computer Science. Springer, S. 78–94. ISBN: 978-3-642-13868-3.

Soboll, Michael (18. Aug. 2011). „Betrüger plündern Konten von 140 Baumarktkunden in Hannover". In: *Hannoversche Allgemeine Zeitung*. Hrsg. von Matthias Koch und Hendrik Brandt. URL: http://www.haz.de/Hannover/Aus-der-Stadt/Uebersicht/Betrueger-pluendern-Konten-von-140-Baumarktkunden-in-Hannover (besucht am 18. 01. 2012).

Soghoian, Christopher und Sid Stamm (16. Apr. 2010). *Certified Lies: Detecting and Defeating Government Interception Attacks Against SSL*. Techn. Ber. Indiana University Bloomington.

Statistisches Bundesamt (15. Mai 2007). *80-Prozent-Marke bei der Handy-Ausstattung überschritten*. Pressemitteilung. Wiesbaden. URL: http://www.destatis.de/jetspeed/portal/cms/Sites/destatis/Internet/DE/Presse/pm/zdw/2007/PD07__019__p002 (besucht am 22. 08. 2011).

– (14. Feb. 2011). *Mobile Internetnutzung über das Handy 2010 stark gestiegen*. Pressemitteilung. Wiesbaden. URL: http://www.destatis.de/jetspeed/portal/cms/Sites/destatis/Internet/DE/Presse/pm/2011/02/PD11__060__63931 (besucht am 22. 08. 2011).

Stevens, Marc u. a. (7. Okt. 2009). „Short Chosen-Prefix Collisions for MD5 and the Creation of a Rogue CA Certificate." In: *Proceedings of the 29th Annual International Cryptology Conference on Advances in Cryptology (CRYPTO '09)*. Hrsg. von Shai Halevi. Bd. 5677.

Lecture Notes in Computer Science. Springer Berlin / Heidelberg, S. 55–69. ISBN: 978-3-642-03355-1.

Street, Jayson E. (31. Juli 2010). „Deceiving the Heavens to Cross the Sea. Using the the 36 stratagems for Social Engineering". DEF CON 18. URL: https://media.defcon.org/dc-18/video/DEF%20CON%2018%20Hacking%20Conference%20Presentation%20By%20-%20Jayson%20E%20Street%20-%20Deceiving%20the%20Heavens%20to%20Cross%20the%20Sea%20-%20Video.m4v (besucht am 29. 11. 2011).

Tarnovsky, Christopher (28. Juli 2010). „Semiconductor Security Awareness, Today and Yesterday". Black Hat USA 2010.

Tarnovsky, Christopher und Karsten Nohl (12. Aug. 2011). „Reviving smart card analysis". Chaos Communication Camp 2011. URL: http://media.ccc.de/browse/conferences/camp2011/cccamp11-4500-reviving_smart_card_analysis-en.html (besucht am 23. 01. 2012).

TÜV NORD (März 2011). *TÜV NORD prüft Sicherheitseigenschaften für ersten nPA-Komfortleser*. Pressemitteilung. URL: http://www.tuev-nord.de/cps/rde/xchg/tng_de/hs.xsl/Pressemitteilungen_575_91794_DEU_PRODUCTIVE.htm (besucht am 16. 07. 2011).

Unisys (28. Feb. 2011). *Unisys Security Index: Germany*. Studie. URL: http://www.unisyssecurityindex.com/system/reports/uploads/94/original/UnisysSecurityIndex-Germany-February2011.pdf (besucht am 22. 08. 2011).

USB CCID (Apr. 2005). *Universal Serial Bus. Device Class: Smart Card CCID*. Version 1.1. USB Implementers Forum. URL: http://www.usb.org/developers/devclass_docs/DWG_Smart-Card_CCID_Rev110.pdf (besucht am 23. 01. 2012).

VASCO Data Security International, Inc. (30. Aug. 2011). *DigiNotar reports security incident*. Pressemitteilung. URL: http://www.vasco.com/company/press_room/news_archive/2011/news_diginotar_reports_security_incident.aspx (besucht am 12. 09. 2011).

SigV (2010). *Verordnung zur elektronischen Signatur*. Letzte Änderung durch Art. 1 ÄndVO vom 15. November 2010 (BGBl. I S. 1542).

Warentest, Stiftung (2012). *Angriff aus dem Internet*. 4. Stiftung Warentest.

Welchering, Peter (1. Nov. 2010). „Interview mit Cornelia Rogall-Grothe. ‚Sicherste Technik, die es gibt'". In: *Frankfurter Allgemeine Zeitung*. URL: http://www.faz.net/-gyc-6l77o (besucht am 29. 11. 2011).

Welte, Harald (28. Dez. 2008). „Anatomy of smartphone hardware. Dissecting contemporary cellphone hardware". 25th Chaos Communication Congress. URL: http://media.ccc.de/browse/congress/2008/25c3-3008-en-anatomy_of_smartphone_hardware.html (besucht am 23. 01. 2012).

– (16. Apr. 2010). *Anatomy of contemporary GSM cellphone hardware*. URL: http://laforge.gnumonks.org/papers/gsm_phone-anatomy-latest.pdf (besucht am 23. 01. 2012).

Wiesmaier, Alexander u. a. (März 2011). „An efficient PACE Implementation for mobile Devices". In: *Proceedings of the 6th ACM Symposium*. New York, NY, USA, S. 176–185. URL: http://www.cdc.informatik.tu-darmstadt.de/mona/pubs/201103_ASIACCS11_An_efficient_PACE_Implementation_for_mobile_Devices.pdf (besucht am 23. 08. 2011).

Zhang, Xinwen, Onur Aciiçmez und Jean-Pierre Seifert (13. Okt. 2009). „Building Efficient Integrity Measurement and Attestation for Mobile Phone Platforms." In: *MobiSec*. Hrsg. von Andreas U. Schmidt und Shiguo Lian. Bd. 17. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering. Springer, S. 71–82. ISBN:

978-3-642-04433-5. URL: http://profsandhu.com/zhang/pub/mobisec09-attest.pdf (besucht am 23. 01. 2012).

ZPO (2011). *Zivilprozessordnung*. Ursprüngliche Fassung vom 30. Januar 1877 (RGBl. S. 83), Letzte Änderung durch Art. 3 G vom 22. Dezember 2011 (BGBl. I S. 3044, 3055).

Selbstständigkeitserklärung

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbstständig und nur unter Verwendung der angegebenen Quellen und Hilfsmittel angefertigt habe.

Frank Morgner

Berlin, den 10. Mai 2012