Technische Universität Darmstadt
Fachbereich Informatik

Fraunhofer SIT

Masterarbeit

Sicherheitsanalyse der Client-Server-Architektur für die Authentifizierung mit dem nPA und eine prototypische Entwicklung einer eID-kompatiblen Client-Server-basierten Software

von

Konstantin Filtschew

1. Juli 2012

Technische Universität Darmstadt
Fachbereich Informatik
Fachgebiet Sicherheit in der Informationstechnik
Mornewegstraße 30
64293 Darmstadt

Betreuer:	Dipl.-Inform. Sven Vowé
	Dipl.-Inform. Andreas Hülsing
Prüfer:	Prof. Dr. Michael Waidner

Ehrenwörtliche Erklärung

Hiermit versichern ich, die vorliegende Masterarbeit ohne Hilfe Dritter und nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die aus den Quellen entnommen wurden, sind als solche kenntlich gemacht worden. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Darmstadt, den 1. Juli 2012

Konstantin Filtschew

Danksagung

Mit der Masterarbeit beende ich das Studium und damit einen wichtigen Abschnitt meines Lebens. An diesem Abschnitt waren nicht wenige Menschen beteiligt, denen ich auf diesem Wege danken möchte.

Ich danke meiner Ehefrau Kseniya, dass sie diese lange Zeit mit mir durchgestanden hat. Während der Masterarbeit hatte ich bestimmt nicht nur zu wenig Zeit, sondern war auch öfters launisch.

Besonders möchte ich mich bei Sven Vowé und Andreas Hülsing bedanken, die sehr sehr viel Geduld mit mir und meinen „Schreibkünsten“ hatten. Nach ihrer Hilfe kann ich jetzt behaupten, dass ich ein paar vernünftige Worte auf Papier bringen kann. Auch danke ich Ulrich Waldmann, der mir hilfreiche Tipps bei der Smartcard-Kommunikation und bei der Implementierung von EAC gegeben hat.

Weiterhin möchte ich Moritz Horsch für den Ideenaustausch bei der Implementierung von PACE danken. Ich danke auch Andreas Wolf von der Uni Koblenz, der parallel zu mir an der Implementierung von Terminal Authentisierung und Chip Authentisierung gearbeitet hat und wir uns somit austauschen konnten.

Besonderen Dank geht auch an die Open Source Gemeinschaft für die vielen, sehr hilfreichen Werkzeuge, die den Fortschritt dieser Arbeit sehr stark beschleunigt haben. Ich danke den Bouncy Castle Entwicklern für die Implementierung der kryptographischen Methoden. Ein großer Dank geht an Ludovic Rousseau für die Implementierung von PCSC-Lite und der damit verbundenen Werkzeuge. Diese haben es mir ermöglicht unter Linux und damit in meiner vertrauten Umgebung zu arbeiten.

Ich danke meinen Eltern für die finanzielle und moralische Unterstützung während der schwierigen Abschnitte des Studiums. Ihr wart immer für mich da, wenn ich nicht mehr wusste, in wie viele Teile ich mich noch teilen soll.

Ich danke auch allen Freunden und Bekannten, die mich mindestens ein Mal pro Woche daran erinnert haben, dass ich meine Masterarbeit noch schreibe und sie auch bald beenden sollte.

Kurzfassung

Am 1. November 2010 wurde in Deutschland der neue Personalausweis (nPA) mit elektronischen Funktionen eingeführt. Zu den wichtigsten Neuerungen zählen der elektronische Identitätsnachweis, die qualifizierte elektronische Signatur und biometrische Daten. Über die elektronische Identität (eID) kann sich der Bürger an Diensten authentisieren. Der Haupteinsatzzweck der eID-Anwendung liegt vor allem im eBusiness und im eGovernment Bereich, da sie vom Einwohnermeldeamt verifizierte Datengruppen enthält. Die Authentifizierung erfolgt ohne augenscheinliche Überprüfung und in der Regel über das Internet, daher muss sie möglichst hohen Sicherheitsanforderungen genügen.

In der Masterarbeit wurde eine Sicherheitsanalyse des Processes und der Client-Server-Architektur für die Authentifizierung mit dem nPA durchgeführt. Für die Sicherheitsanalyse wurden vier Methoden untersucht und die Vorgehensweise nach BSI IT-Grundschutz-Standard als die am besten passende Methode ausgewählt. Als Ergebnis der Analyse wurden Sicherheitsanforderungen für den Authentifizierungsprozess erarbeitet, wobei der eID-Client als kritische Komponente identifiziert wurde. Die Sicherheitsanforderungen für den eID-Client wurden auf installierbare Anwendungen und auf Java Applets als mögliche Implementierungsvariante abgebildet, wobei sich Java Applets als die technisch bessere Lösung für den eID-Client erwiesen haben.

Im Implementierungsteil wurde eine Client-Server-Software für die Authentifizierung mit Hilfe der eID-Anwendung entwickelt. Der eID-Client kann als Anwendung und als Java Applet verwendet werden. Die Software basiert auf einer selbst entwickelten eID-Bibliothek, welche die kryptographischen Protokolle und die APDU-Kommunikation unterstützt. Die Bibliothek enthält Unittests, welche die Weiterentwicklung unterstützen sollen und zusätzlich als Referenzimplementierung für wichtige Teile von Extended Access Control dienen. Die Kommunikation zwischen Server und Client basiert auf dem standardisierten Websocket-Protokoll, welcher sehr schlank ist und fast kein Overhead besitzt. Dadurch wird die Überprüfung der Kommunikation und der Sicherheitsmechanismen deutlich vereinfacht.

Abstract

The new German electronic identity (eID) card was introduced on 1. November 2010. It has electronic features, which were missing on the old identity card. The most important features are the electronic authentication, the qualified electronic signature and biometric data. Citizens can use the electronic identity (eID) for authentication on electronic services. The primary purpose of the eID application is in eBusiness and eGovernment scope, as all data groups are verified through resident registration office. The authentication is performed without apparent review, that's why the whole process must meet high security requirements.

The first part of the master thesis is a security analysis of the process and the client-server architecture for authentication with the German eID card. The security analysis should be done on a well defined process. The German BSI IT Grundschutz Standard was identified as the most appropriate of four evaluated methods. The result of the security analysis are security requirements for the authentication process. In the analysis the eID client was identified as the most critical component in the whole process. The security requirements were applied to common software implementation methods as installable applications and as Java Applets. The Java Applets has been proven as the technically superior solution for the eID client.

The implementation part of the master thesis consists of a client-server software for authentication with eID application. The eID Client can be used as application and as Java Applet. This software is based on a self-made eID library, which supports the required cryptographic protocols and APDU communication. Almost all functions of the library are covered by unit tests. The tests should support the further development of the library and can be used as reference implementation for the components. The communication between server and client is build on the WebSocket protocol, which is very lean and has almost no overhead. It simplifies the analysis of the communication and the security mechanisms.

INHALTSVERZEICHNIS

Teil I.

Einleitung

KAPITEL 1

MOTIVATION

1.1. Einführung

Im Internet existiert eine große Anzahl von Diensten. Diese Dienste bieten häufig personalisierte Leistungen an. Um jene in Anspruch zu nehmen, muss der Benutzer eindeutig identifiziert und gegebenenfalls authentifiziert werden.

Viele der Dienste verwenden einen Benutzernamen für die Identifizierung sowie ein Passwort für die Authentifizierung des Benutzers. Diese und ähnliche Verfahren weisen entscheidende Schwächen auf, da sie auf dem Passwort als geheimes Wissensmerkmal basieren. Weiterhin existieren sehr viele Angriffe auf passwortbasierte Authentisierungsverfahren, die nur schwer identifiziert und verhindert werden können. Der Diebstahl eines Passwortes wird in der Regel erst bei einem aufgetretenen Schaden bemerkt.

Mit dem neuen Personalausweis (nPA) wurde am 1. November 2010 ein Zwei-Faktor-Authentisierungsverfahren mit einer elektronischen Identität (eID) für Dienste eingeführt, welches den nPA als Besitzermerkmal und die PIN als Wissensmerkmal für die erfolgreiche Authentisierung voraussetzt. Fehlt eines der beiden Merkmale, so kann die Authentisierung nicht durchgeführt werden. Wird dem Bürger der nPA entwendet oder er verliert ihn, so bemerkt er es in der Regel sehr schnell und kann ihn sperren lassen. Zudem muss der Angreifer nicht nur die Kontrolle über den nPA bekommen, er muss auch die PIN wissen. Dadurch entstehen entscheidende Vorteile gegenüber den Verfahren mit nur einem Merkmal.

Für die Kommunikation zwischen nPA, dem Bürger und den beteiligten Parteien wird auf dem Computer des Bürgers die Softwarekomponente eID-Client benötigt. Weiterhin benötigt der Bürger ein Kartenlesegerät, das die Kommunikation zwischen nPA und dem Computer des Bürgers übernimmt. Der eID-Client stellt den Authentisierungsprozess visuell dar und steuert die Kommunikation. Somit spielt er eine zentrale Rolle.

Der offizielle eID-Client wurde im Auftrag vom BMI[1] von der Firma OpenLimit entwickelt und trägt den Namen AusweisApp. Zum derzeitigen Zeitpunkt ist er ausschließlich für den Einsatz mit dem nPA gedacht und wird jedem Bürger kostenlos zur Verfügung gestellt.

Ziel dieser Masterarbeit ist es eine Sicherheitsanalyse der Client-Server-Architektur für die Authentifizierung mit dem nPA durchzuführen. Auf Basis der Sicherheitsanalyse soll eine prototypische Client-Server basierte Software für eID-kompatible Karten entwickelt werden.

Das erste Kapitel dient als thematische Einführung. Hier wird der nPA, die verschiedenen Passwörter, die elektronischen Funktionen und die dafür notwendige Public Key Infrastruktur beschrieben.

Das zweite Kapitel beschreibt Authentisierungsverfahren, kryptographische Grundlagen und Protokolle, die im Zusammenhang mit dem nPA verwendet werden. Diese Grundlagen werden als Basis für alle nachfolgenden Kapitel benötigt.

In der Sicherheitsanalyse soll der Authentisierungsprozess mit dem nPA untersucht werden. Dafür werden im dritten Kapitel gängige Methoden für Sicherheitsanalysen untersucht und eine Methode ausgewählt. Ziel der Sicherheitsanalyse ist es, Sicherheitsanforderungen für die Entwicklung eines eID-Clients zu erarbeiten.

Die eigentliche Sicherheitsanalyse wird im vierten Kapitel durchgeführt. Hier werden allgemeine Sicherheitsanforderungen für die Entwicklung eines eID-Clients erarbeitet. Die allgemeinen Sicherheitsanforderungen werden dann auf zwei mögliche Implementierungen als installierbare Anwendung und als Java Applet abegebildet, wobei die besonderen Sicherheitsanforderungen für einen eID-Client beachtet werden.

Im fünften Kapitel wird die Implementierung der Bibliothek beschrieben, welche die Grundlage für eine Weiterentwicklung bilden soll. Diese soll die Implementierung von EAC[2] und der Smartcard-Kommunikation für die Protokolle enthalten. Statische Tests (Unittests) sollen die Implementierung nachvollziehbar machen und gleichzeitig als Beispiel für die Verwendung der Bibliothek dienen. Basierend auf dieser Bibliothek sollen

[1] Bundesministerium des Innern
[2] Extended Access Control

Prototypen entwickelt werden, die eine Server-Client-Kommunikation beinhalten und als Anwendung sowie als Applet mit dem selbst entwickelten Server funktionieren sollen.

1.2. Der neue Personalausweis

Der neue Personalausweis (nPA) hat den Personalausweis ohne elektronische Funktionen zum 1. November 2010 ersetzt. Die Personalausweise ohne elektronische Funktionen bleiben weiterhin gültig und werden bei Neuausgabe ersetzt. Die elektronischen Funktionen des nPAs stellen die Neuerung dar und diese werden in den folgenden Abschnitten genauer erläutert.

1.2.1. Datengruppen

Im Chip des nPAs sind 21 Datengruppen (DG1 - DG21) mit persönlichen Daten des Bürgers gespeichert. Die Tabelle 1.1 enthält eine Auflistung aller Datengruppen mit ihrer Bedeutung nach BSI TR-03110 [BSI09a].

Von diesen Datengruppen werden D10-D16 beim nPA nicht verwendet. Damit enthält der nPA die Nationalität und das Geschlecht weder in aufgedruckter, noch in digitaler Form.

Bei allen 21 Datengruppen handelt es sich um persönliche Daten des Bürgers, die eine Zugriffskontrolle und damit eine Authentifizierung des Benutzers erfordern.

Für hoheitliche Zwecke muss auf die Datengruppen D17-D21 ein Schreibzugriff bereitgestellt werden. Beispielsweise müssen die Einwohnermeldeämter weiterhin in der Lage sein, bei einem Wohnortwechsel die Anschrift des Bürgers zu ändern.Die Anschrift muss auf dem Chip und auf dem Ausweis optisch sichtbar geändert werden.

Der Zugriff auf die Datengruppen erfolgt durch die eID-Funktion des nPAs, welche eine vorherige Authentifizierung aller beteiligten Parteien voraussetzt. Die Authentifizierung wird mit Hilfe von EAC durchgeführt, welches genau dafür entwickelt wurde.

1.3. Passwörter

Die elektronischen Funktionen des nPAs erfordern eine Authentifizierung des Benutzers. Dieser muss sich vorab mit einem Passwort authentisieren, um die Anwendungen auf

Datengruppe	Beschreibung
DG1	Dokumententyp
DG2	Ausgebender Staat
DG3	Ablaufdatum
DG4	Vorname(n)
DG5	Familienname
DG6	Ordensname/Künstlername
DG7	Doktorgrad
DG8	Geburtsdatum
DG9	Geburtsort als unformatierter Text
DG10	Nationalität
DG11	Geschlecht
DG12	Optionale Daten
DG13 - DG16	nicht benutzt
DG17	aktuelle Adresse in strukturierter Form
DG18	Wohnort-ID
DG19	Aufenthaltserlaubnis I
DG20	Aufenthaltserlaubnis II
DG21	Optionale Daten

Tabelle 1.1.: Alle 21 Datengruppen der eID-Anwendung nach BSI TR-03110 [BSI09a]

dem nPA freizuschalten. Das dabei verwendete Passwort unterscheidet sich abhängig von der Anwendung und den benötigten Berechtigungen, was in den folgenden Abschnitten erläutert wird.

1.3.1. CAN

Die Card Access Number (CAN) ist eine auf dem nPA sichtbar aufgedruckte Dezimalzahl. Sie hat somit keine Bindung an den Ausweisinhaber und wird für die folgenden Zwecke verwendet:

- Hoheitliche Kontrollen (z.B. Grenz- oder Personenkontrollen)
- Änderungsdienst für Adressdaten (z.B. bei Wohnortwechsel) und für die Visualisierung der auf dem Chip digital hinterlegten Daten in Ausweisbehörden
- Beantragung der Qualifizierten Elektronischen Signatur (QES)
- Freischalten des dritten Eingabeversuchs bei der eID-Funktion des nPAs

Auf den aktuellen Ausweisen ist die CAN sechs Dezimalstellen lang und besitzt keinen Fehlbedienungszähler, so dass sie beliebig oft eingegeben werden darf.

1.3.2. MRZ

Die Machine Readable Zone (MRZ) ist ein zur CAN alternatives Passwort. Sie wird bereits bei den deutschen Reisepässen eingesetzt, um ausgewählte Datengruppen in elektronischer Form abzufragen.

Eigentlich ist die MRZ kein direktes Passwort, sondern ein mit dem SHA-1 Hashalgorithmus (siehe Abschnitt 2.3.5) generierter Hashwert über die folgenden aufgedruckten und maschinenlesbaren Datengruppen:

- Dokumentennummer
- Geburtsdatum
- Ablaufdatum

Dieses Verfahren ist von der *ICAO*[3] spezifiziert und wird weltweit für Reisedokumente verwendet.

1.3.3. eID-PIN

Die Personal Identification Number (PIN) ist ein sechs Dezimalstellen langes Passwort für die eID-Funktion des nPAs. Sie sollte nur dem Bürger bekannt sein, da mit der eID-PIN sich der Bürger am nPA authentisiert und damit die eID-Funktion auf dem Chip des nPAs freischaltet.

Ein von der Bundesdruckerei gelieferter nPA hat zunächst nur eine zufällig generierte fünfstellige Transport-PIN für die eID-Funktion gesetzt. Sie wird dem Bürger als Brief an die Wohnanschrift zugestellt und lässt sich nicht für die Authentisierung einsetzen. Der Bürger muss zuerst eine sechsstellige, nur ihm bekannte eID-PIN setzen.

Die Anzahl der Fehlversuche für die eID-PIN ist auf drei begrenzt, so dass ein Angreifer nicht in der Lage sein soll, diese zu erraten. Der Benutzer des nPAs kann die PIN zweimal falsch eingeben. Danach muss er erst die richtige CAN eingeben, um den letzten Versuch freizuschalten. Wird auch beim dritten Versuch die falsche PIN eingegeben, so sperrt der nPA die eID-Funktion, bis die PIN mit dem PUK freigeschaltet wird. Damit kann ein Angreifer die eID-Funktion des nPAs nicht sperren, wenn er die CAN nicht kennt. Nach jeder erfolgreichen Authentisierung mit der eID-PIN wird der Zähler vom eID-Client wieder auf Null gesetzt. Der Fehlbedienungszähler befindet sich direkt im Chip des nPAs, so dass ein Angreifer nicht in der Lage ist, diesen von außen zu manipulieren.

1.3.4. Signatur-PIN

Die Signatur-PIN schaltet die Signaturfunktion des nPAs frei. Dadurch lassen sich qualifizierte elektronische Signaturen (siehe Abschnitt 1.5.2) erstellen.

Sie muss vor der Beantragung einer Signatur gesetzt werden. Die Authentifizierung des Bürgers erfolgt über die CAN, bevor die Signatur-PIN gesetzt werden kann.

1.3.5. PUK

Die Personal Unblocking Key (PUK) ist eine im Vergleich zur PIN lange Dezimalzahl. Sie dient lediglich dazu, die eID-PIN und die Signatur-PIN wieder zu entsperren, wenn

[3] International Civil Aviation Organization

diese durch drei Fehlversuche gesperrt wurden. Die PUK kann beliebig oft verwendet werden.

Der Bürger bekommt die PUK per Post zusammen mit der Transport-PIN für die eID-Funktion des nPAs von der Bundesdruckerei zugestellt. Die PUK sollte an einer sicheren Stelle aufbewahrt werden, da mit ihrer Hilfe die eID-PIN und die Signatur-PIN geändert werden können.

1.4. EAC-PKI für den nPA

Die Überprüfung der Terminals durch den nPA erfordert eine Public Key Infrastruktur (siehe Abschnitt 2.4). Diese soll dem nPA und dem Terminal ermöglichen, die Gültigkeit der Zertifikate zu überprüfen.

Das Zertifikat auf dem nPA wird beim Erstellen des nPAs signiert und ändert sich danach nicht mehr. Die Terminalzertifikate sind allerdings nur eine begrenzte Zeit gültig und müssen regelmäßig erneuert werden.

Die Public Key Infrastruktur (PKI) für die Terminalzertifikate besteht aus drei Stufen:

- Country Verifying CAs (CVCAs)
- Document Verifiers (DVs)
- Terminalzertifikate (Berechtigungszertifikate)

In den folgenden Abschnitten werden die einzelnen Stufen im Detail erläutert.

1.4.1. Country Verifying CA (CVCA)

Bei der PKI für elektronische Ausweise ist die oberste Instanz die Country Verifying CA, die jedes Land als Vertrauensanker stellen muss. Ausgehend von der CVCA kann die ganze Zertifikatskette überprüft werden. Die CVCA signiert die Document Verifier Zertifikate als darunter liegenden Instanzen.

1.4.2. Document Verifiers (DV)

Die Terminalzertifikate besitzen nur einen kurzen Gültigkeitszeitraum und müssen dementsprechend oft erneuert werden. Diese von der CVCA direkt zu signieren, wäre ein viel zu großer Aufwand. Deswegen werden Document Verifiers bestimmt, die Terminalzertifikate von einer Terminalgruppe signieren. In der Regel erben die Terminalzertifikate die Rechte vom DV, wobei der DV die Rechte der Terminals weiter einschränken kann.

Der Ausweis hat nur den Country Verifier seines Landes und kann dementsprechend nur ausgehend von seiner CVCA die Zertifikatskette überprüfen. Will ein Terminal eines dem Ausweis fremden Landes auf die Datengruppen zugreifen, muss der Document Verifier zusätzlich von der CA des Landes signiert werden, von dem der Ausweis stammt. Auf diesem Wege werden auch die Berechtigungen der Länder gesteuert.

Die Document Verifier haben eine für Zertifikate kurze Gültigkeit und müssen in regelmäßigen Abständen erneuert werden. Somit müssen sie nicht in einer Revokationsliste[4] gepflegt werden. Zudem macht der begrenzte Speicher der Ausweise eine Auswertung und Speicherung von Revokationslisten sehr schwierig, was bei dieser Lösung entfällt.

1.4.3. Terminalzertifikate

Bevor ein Terminal auf die Datengruppen des nPAs zugreifen darf, muss es nachweisen, dass es ein gültiges Zertifikat besitzt. Dieses Zertifikat wird auch Berechtigungszertifikat genannt, da es Zugriffsberechtigungen auf die Datengruppen des nPAs enthält. Die Terminalzertifikate haben nur einen sehr kurzen Gültigkeitszeitraum und müssen dementsprechend oft erneuert werden, was die Pflege einer Revokationsliste auf dem nPA erspart.

Sie werden vom dem Document Verifyer signiert und enthalten Berechtigungen für den Zugriff auf die Datengruppen des nPAs. Jeder Dienstanbieter muss sich ein solches Zertifikat beantragen und in seinem Terminal installieren.

1.5. Elektronische Anwendungen des nPAs

Der nPA enthält für verschiedene Einsatzzwecke elektronische Anwendungen. In den folgenden Abschnitten werden Funktionen und Einsatzzwecke dieser Anwendungen erläutert.

[4] Liste mit revozierten Zertifikaten, die bei der Überprüfung abgelehnt werden sollen

1.5.1. Biometrieanwendung

Der nPA wird innerhalb der Europäischen Union (EU) als biometrisches Reisedokument verwendet. Dafür sind die Daten der maschinenlesbaren Zone (MRZ) in digitaler Form in der ersten Datengruppe (DG1) der Anwendung gespeichert. Die zweite Datengruppe (DG2) enthält das auf dem Ausweis aufgedruckte Bild und in der dritten Datengruppe (DG3) können optional die Fingerabdrücke gespeichert werden. Sind die Fingerabdrücke nicht auf dem nPA hinterlegt, so enthält die Datengruppe zufällige Daten.

Damit entsprechen die biometrischen Daten den Anforderungen der ICAO (International Civil Aviation Organization) für biometrische Ausweise und stimmen mit den Daten des Reisepasses überein. Bei dem nPA ist im Vergleich zum Reisepass die Abgabe der Fingerabdrücke optional.

Wie auch beim Personalausweis ohne elektronische Funktionen sind die Datengruppen beim nPA außen auf dem Ausweis angebracht. Lediglich die optionalen Fingerabdrücke sind zusätzliche Daten, die nur in digitaler Form auf dem Chip gespeichert werden.

1.5.2. Signaturanwendung

Über die Signaturanwendung lassen sich elektronische Dokumente durch eine Qualifizierte Elektronische Signatur (QES) signieren. Der Bürger erzeugt auf dem nPA ein Schlüsselpaar bestehend aus einem privaten und einem öffentlichen Schlüssel. Der öffentliche Schlüssel wird von einem Anbieter für elektronische Signaturen signiert, um dessen Gültigkeit zu bestätigen. Danach können mit dem Schlüsselpaar QES erstellt werden.

Mit Hilfe des nPAs lässt sich die QES von zu Hause aus beantragen. Die Authentifizierung des Bürgers erfolgt über die eID-Funktion des nPAs. Der signierte öffentliche Schlüssel wird dadurch auch auf elektronischem Wege auf den nPA übertragen. Die Signaturfunktion des nPAs ist durch eine eigene Signatur-PIN geschützt (siehe Abschnitt 1.3.4) und muss sich von der eID-PIN unterscheiden.

Durch QES werden Verbindlichkeiten ausgelöst, was bei der eID-Funktion nicht der Fall ist. Daher lässt sie die Signaturanwendung nur mit einem Standard- oder Komfort-Chipkartenleser (siehe Abschnitt 4.1.3) verwenden. Das hat den Vorteil, dass der Bürger die Signatur-PIN nur am Kartenlesergerät eingeben muss und damit niemals über den als unsicher geltenden Computer, wo die PIN vom Angreifer abgegriffen werden könnte.

1.5.3. Master File

Die Master File stellt Systemdaten für die elektronische Anwendung des nPAs zur Verfügung, welche für Extended Access Control benötigt werden.

Aus Datenschutzgründen soll die Identifizierung einzelner Ausweise über individuelle Parameter nicht möglich sein. Daher enthalten alle nPAs einer Generation die selben öffentlichen Parameter und öffentliche Schlüssel. Somit kann ein nPA bei der Chiptauthentisierung nicht eindeutig identifiziert und auch nicht sicher verfolgt werden.

1.5.4. eID-Anwendung

Die Elektronische Identifikationsnachweis (eID) Anwendung dient zur Identifikation und Authentifikation eines Benutzers durch einen Dienstanbieter. Der Vorgang erfolgt über eine unsichere Leitung wie dem Internet. Der Einsatzzweck für die eID-Anwendung liegt im eGovernment Bereich und in eBusiness-Diensten. In den folgenden Abschnitten werden die einzelnen Funktionen erklärt.

Authentifizierung

Die Authentifizierung des Bürgers ist die Hauptfunktion der eID-Anwendung. Dabei werden vom Einwohnermeldeamt verifizierte Datengruppen vom nPA an den Dienstanbieter übertragen. Der Dienstanbieter darf nur auf Daten zugreifen, die in seinem Berechtigungszertifikat zugelassen wurden. Der Bürger kann die zu übertragenen Datengruppen vor der Authentisierung weiter einschränken.

Die Authentifizierung mit der eID-Funktion des nPAs kann unter anderem als Ersatz für das Postident Verfahren verwendet werden. Damit wird es zum Beispiel möglich, ein Bankkonto direkt vom Computer zu eröffnen (eBusiness) oder behördliche Tätigkeiten (eGovernment) durchzuführen.

Pseudonym-Funktion

Erfordert ein Dienst keine persönlichen Daten des Bürgers, so reicht dafür die Pseudonym-Funktion aus. Das bedeutet, der Bürger wird nicht authentifiziert, sondern über ein Pseudonym identifiziert. Eine Zuordnung zu einer realen Person ist darüber sehr schwer

möglich (siehe BSI TR-03110 [BSI09a]). Das Pseudonym ist nur in Verbindung mit einem Anbieter eindeutig, so dass der Bürger pro Anbieter ein eindeutiges Pseudonym besitzt.

Die Pseudonym-Funktion ist keine Anonymisierung, da der Bürger über diese Funktion bei einem Dienstanbieter identifiziert werden soll, aber seine persönlichen Daten für die Identifizierung nicht benötigt werden. Zudem kann das Pseudonym für Strafverfolgungsbehörden bis zum Bürger aufgelöst werden.

Altersverifikation

Die eID-Anwendung soll eine elektronische Altersverifikation mit Hilfe der Datengruppe DG8 ermöglichen. Dabei soll das Geburtsdatum nicht an dritte übermittelt werden, um die persönlichen Daten des Bürgers zu schützen.

Wird für einen Dienst ein bestimmtes Alter vorausgesetzt, so übermittelt der Dienst das jüngste, akzeptierte Geburtsdatum an die eID-Funktion des nPAs. Ist das Geburtsdatum nicht später als das angegebene Datum, so wird die Anfrage vom nPA positiv quittiert, ansonsten wird sie negativ quittiert. Weitere Informationen werden an den Dienstanbieter nicht übertragen.

Nach BSI TR-03127 [BSI10] ist nicht bei allen Personen das Geburtsdatum vollständig bekannt. In solchen Fällen werden die fehlenden Stellen mit Leerzeichen aufgefüllt. Für die Altersverifikation muss trotzdem sicher gestellt werden, dass die Person das Mindestalter erreicht hat. Aus diesem Grund wird das späteste mögliche Geburtsdatum der Person über die vorhanden Daten angenommen. Wenn beispielsweise nur das Geburtsjahr bekannt ist, so wird der 31.12 des Jahres als Geburtsdatum angenommen.

Wohnortverifikation

Die Wohnortverifikation (Community ID Verification) ermöglicht es, den Wohnort des Bürgers zu überprüfen, ohne seinen wirklichen Wohnort an den Dienstanbieter zu übermitteln. Damit ist es beispielsweise möglich, den Ort zu prüfen und damit sicherzustellen, dass ein Bürger Einwohner ist, ohne die vollständige Adresse abfragen zu müssen.

1.5.5. Hoheitliche Funktionen der eID-Anwendung

Die eID-Anwendung lässt sich auch für hoheitliche Funktionen verwenden. Für den Zugriff auf diese Funktionen werden hoheitliche Berechtigungszertifikate benötigt. Damit ist das Einwohnermeldeamt in der Lage, bei einem Wohnortwechsel die Datensätze mit der Adresse des Bürgers zu ändern, wie sie auch außen auf dem Ausweis durch einen Aufkleber geändert werden. Bei Grenzkontrollen können die persönlichen Daten des Bürgers elektronisch abgerufen werden. Dazu zählen auch die Fingerabdrücke und das digital gespeicherte Bild des Bürgers.

Die hoheitlichen Funktionen des nPAs sind nicht Bestandteil dieser Arbeit und werden nicht weiter behandelt.

Teil II.

Grundlagen

KAPITEL 2

GEGENSTAND DER ARBEIT

2.1. Identifizierung

Man spricht von Identifizierung, wenn ein Benutzer seine Identität durch eine eindeutige Benutzerkennung nachgewiesen hat. Die Benutzerkennung selbst reicht zwar für die Identifizierung des Benutzers aus, aber sie authentifiziert ihn nicht (siehe folgenden Abschnitt).

Eine Benutzerkennung ist bei vielen Internetdiensten ein Benutzername oder eine E-Mail Adresse. Die Benutzerkennungen müssen innerhalb des Dienstes eindeutig sein, um den Benutzer identifizieren zu können. Ein Authentizitätsnachweis für den Dienst (beispielsweise ein Passwort) ist erst für eine Authentisierung notwendig.

2.2. Authentisierung

Nach Eckert [Eck09] versteht man unter Authentizität eines Objekts bzw. Subjekts „die Echtheit und Glaubwürdigkeit des Objekts bzw. des Subjekts, die anhand einer eindeutigen Identität und charakteristischen Eigenschaften überprüfbar ist“. Subjekte stellen Benutzer des Systems dar und Objekte beschreiben Elemente im System, die durch den Benutzer aktiviert und verwendet werden. Solche Objekte stellen beispielsweise Dienste, Anwendungen und Tokens dar, die der Benutzer vorab aktiviert hat.

Die Authentifizierung durch einen Dienst erfolgt, indem sich der Benutzer am Dienst authentisiert und der Dienst den Benutzer authentifiziert. Die behauptete Authentizität des Benutzers wird mit Hilfe eines vorab ausgehandelten Identitätsnachweises erbracht. Häufig verwendete Authentizitätsnachweise sind unter anderem das Passwort oder die Persönliche Identifikationsnummer (PIN).

Gängige Authentisierungsverfahren lassen sich auf den Nachweis der Kenntnis von spezifischem Wissen oder den Nachweis eines persönlichen Besitzes aufteilen. Die Unterschiede zwischen Besitz und Wissen werden in den folgenden Abschnitten genauer erläutert.

2.2.1. Wissen

Auf Wissen basierende Authentisierungsverfahren sind weit verbreitet. Der Benutzer muss nachweisen, dass er ein vorab ausgehandeltes Geheimnis kennt. Dieses Geheimnis muss vor der ersten Authentisierung zwischen dem Benutzer und dem Dienst ausgetauscht werden. Der Nachweis des Wissens reicht für die rein wissensbasierte Authentisierung aus.

Passwortbasierte Verfahren

Die häufigsten Vertreter für wissensbasierte Authentisierung sind passwortbasierte Verfahren. Sie verwenden als Nachweis ein Passwort, welches vorab zwischen den beiden Parteien ausgetauscht werden muss.

Unter Passwortverfahren fällt auch die Authentisierung an einer Chipkarte (siehe Abschnitt 2.2.2), die über eine persönliche Identifikationsnummer (PIN) erfolgt. Sie stellt in der Regel eine kurze Folge von vier bis sechs Zahlen dar. Aus diesem Grund ist die Falscheingabe auf drei Fehlversuche begrenzt. Wurde die PIN wegen mehrmaliger Falscheingabe gesperrt, so kann sie durch den PUK (Personal Unblocking Key) wieder freigeschaltet werden.

Einmal-Passwörter

Einmal benutzbare Passwörter OTP (engl. one-time password) erfordern für jeden Authentifizierung ein eigenes Passwort. Diese müssen vorab zwischen dem Benutzer und dem Dienst ausgetauscht werden. Alternativ kann das Passwort über eine Funktion generiert werden. Dafür muss beiden Parteien die Funktion und der Initialisierungswert bekannt sein.

Bei Online-Banking werden für Transaktionen diverse Einmal-Passwort-Verfahren verwendet. Der Dienst generiert eine Liste mit Transaktionsnummern (TAN), welche dem Benutzer übergeben werden. Dieser ist verpflichtet, bei jeder Transaktion eine TAN von der Liste einzugeben, um seine Authentizität für die Transaktionen nachzuweisen.

Frage-Antwort

Ein weiteres Authentifizierungsverfahren ist die Beantwortung einer bestimmten Frage. Meistens wird die Antwort auf die Frage vorab ausgehandelt. Der Benutzer ist authentifiziert, wenn er die Frage richtig beantwortet.

Die Authentifizierung über Frage-Antwort wird häufig verwendet, wenn der Benutzer sein Passwort vergessen hat. Durch die Beantwortung der Frage erhält er die Möglichkeit, sein Passwort einzusehen oder ein neues Passwort für den Dienst zu setzen.

2.2.2. Besitz

Die Authentizität des Benutzers kann ebenfalls durch Besitz oder durch die Eigenschaft des Besitzes nachgewiesen werden. Das Verfahren wird beispielsweise bei Personenkontrollen an Grenzübergängen eingesetzt. Durch einen Personalausweis oder einen Reisepass muss sich der Ausweishalter beim Grenzbeamten authentisieren.

Digitale Schlüssel

Digitale Schlüssel werden als Besitzmerkmal in dem Authentisierungsverfahren verwendet. Sie werden mit kryptographischen Verfahren erstellt und haben dadurch bestimmte Eigenschaften. Der Nachweis des Besitzes solcher Schlüssel reicht für die Authentisierung aus. Ohne weiteren Schutzmaßnahmen können die Schlüssel kopiert und von Dritten verwendet werden, deshalb müssen sie durch zusätzliche Mechanismen abgesichert werden.

Werden die Schlüssel auf dem PC oder einem Speichergerät wie USB-Stick aufbewahrt, so besteht die Möglichkeit, sie mit einem Passwort zu verschlüsseln und dadurch vor Verwendung durch Dritte zu schützen. Werden die Schlüssel verschlüsselt, so können sie trotzdem kopiert werden. Der Angreifer kann sie zwar kopieren, aber um sie verwenden zu können, muss er erst die Verschlüsselung überwinden. Somit hängt die Sicherheit des Schlüssels vom dem Verschlüsselungsverfahren und den dafür verwendeten Passwort ab.

Eine weitere Möglichkeit digitale Schlüssel zu generieren und aufzubewahren bieten Chipkarten (siehe folgenden Abschnitt). Die Schlüssel werden direkt auf der Chip generiert. Auch die Ver- und Entschlüsselungsverfahren werden direkt auf der Chip durchgeführt. Dadurch verlässt der Schlüssel niemals die Chipkarte.

Chipkarten

Besitzbasierte Authentisierung stützt sich heute oft auf Smartcards. Dabei ist nicht jede nach Smartcard aussehende Plastikkarte auch eine Smartcard. Chipkarten in ihrer einfachsten Form stellen reine Speicherkarten dar, wie es beispielsweise Krankenversicherungskarten sind.

Die nächste Stufe bilden so genannte intelligente Speicherkarten mit zusätzlicher Sicherheitslogik. Sie schützen den geheimen Schlüssel vor der Extraktion oder dessen Funktionsweise. Der Benutzer braucht sich nicht an der Speicherkarte zu authentisieren, sondern der reine Nachweis des Besitzes des geheimen Schlüssels reicht für die Authentisierung aus.

Smartcards sind nach Eckert [Eck09] „mit einem Mikroprozessor und einem programmierbaren Speicher ausgerüstete Karten“. Sie sind in der Lage, Programme auszuführen und damit auch kryptographische Funktionen direkt auf der Karte zu implementieren. Bekannte Vertreter von Smartcards sind Geldkarten und der neue Personalausweis.

Das Format der Karten, die Kommunikationsschnittstellen und die Übertragungsprotokolle sind in dem ISO/IEC 7816 Standard beschrieben. Kontaktbehaftete Karten kommunizieren über acht (C1-C8) nach außen gelegte Kontakte. Weiterhin besteht die Möglichkeit der kontaktlosen Kommunikation, wie sie in dem ISO/IEC 14443 Standard beschrieben und beim nPA verwendet wird.

Smartcards besitzen gegenüber Speicherkarten zusätzliche physikalische Mechanismen, um sie vor diversen Angriffen zu schützen. Diese sind in dem Buch IT-Sicherheit von Eckert [Eck09] und in dem Handbuch der Chipkarten von Wolfgang Rankl [RE03] beschrieben. Die Sicherheitsarchitektur von Smartcards ist nicht Bestandteil dieser Arbeit und wird an dieser Stelle nicht weiter betrachtet.

Einmal-Passwort Generatoren

Wie bereits beschrieben, können Einmal-Passwörter durch eine Funktion generiert werden. Diese Aufgabe übernehmen elektronische Generatoren. Die gemeinsame Funktion wird von der Authentifizierungsinstanz auf dem Generator gespeichert und direkt auf

dem Chip ausgeführt. Der Benutzer sieht nur das generierte Ergebnis. Die Funktion zur Generierung des Einmal-Passwortes benötigt einen Initialisierungswert, welcher beispielsweise die aktuelle Zeit oder eine mögliche Eingabe sein kann.

Der Halter des Generators kann mit seiner Hilfe Einmal-Passwörter generieren, aber nicht die Funktion des Generators extrahieren, was durch diverse Schutzmechanismen verhindert wird.

Einmal-Passwort Generatoren können für TAN-Verfahren bei Online-Banking oder für passwortbasierte Zugangssysteme verwendet werden, wo das Einmal-Passwort als zusätzliches Sicherheitsmerkmal verwendet wird. Der Benutzer muss zu seinem festen Passwort zusätzlich das Einmal-Passwort eingeben, um sich am System zu authentisieren.

2.2.3. Biometrische Merkmale

Unter biometrischen Merkmalen versteht man nach Eckert [Eck09] „physiologische oder verhaltenstypische Eigenschaften einer Person, die diese eindeutig charakterisieren“. Biometrische Merkmale sind beispielsweise Fingerabdrücke, Gesichts-Scans und Iris-Scans. Sie werden auf elektronischen Reisedokumenten und bei Zugangskontrollen verwendet.

Biometrische Merkmal können im Vergleich zu besitz- oder wissensbasierten Merkmalen nicht verloren oder vergessen werden, da der Besitzer sie immer mit sich trägt. Bei Verlust oder Diebstahl biometrischer Merkmale gestaltet sich die Handhabung schwierig. Bei Diebstahl können die biometrischen Merkmale nicht geändert werden. Bei Verlust muss ein anderes Merkmal für die Authentisierung verwendet werden, was nicht immer möglich ist.

2.2.4. Zwei-Faktor-Authentisierung

Für die Zwei-Faktor-Authentisierung müssen zwei Bedingungen erfüllt werden, bevor die Authentizität eines Objekts oder Subjekts eindeutig bestimmt werden kann. Eine Kombination aus „Wissen und Besitz“ ist die am häufigsten verwendete Art der Zwei Faktor-Authentisierung.

Der Benutzer muss zwei Faktoren zusammen nachweisen, sonst schlägt die Authentisierung fehl. Dadurch entsteht ein stärkerer Mechanismus, da zwei Behauptungen für die erfolgreiche Authentisierung notwendig sind.

Smartcard und PIN Authentisierung

Eine häufig eingesetzte Zwei Faktor-Authentisierung sind mit PIN geschützte Smartcards. Im ersten Schritt authentisiert sich der Benutzer an der Smartcard. Erst nach der erfolgreichen Authentisierung des Benutzers an der Smartcard authentisiert sich die Smartcard am Dienst. Der Dienst authentifiziert die Smartcard und damit den Benutzer. Fehlt einer der Faktoren, so kann die Authentisierung nicht durchgeführt werden.

Einmal-Passwort-Generatoren für Zwei-Faktor-Authentisierung

Eine weitere Möglichkeit für die Zwei-Faktor-Authentisierung stellen Einmal-Passwort-Generatoren dar, die in Kombination mit festen Passwörtern verwendet werden. Der Benutzer muss zu seinem fest vergebenem Passwort zusätzlch das Einmal-Passwort eingeben. Ist das Einmal-Passwort ungültig oder das feste Passwort falsch, so schlägt die Authentisierung fehl.

2.3. Kryptographie

Im Zusammenhang mit dem nPA wird Wissen über diverse Protokolle und Teile der Kryptographie benötigt, welche in den folgenden Abschnitten kurz eingeführt werden. Weiterführende Informationen zu Kryptographie und kryptographischen Protokollen können in dem Buch *Einführung in die Kryptographie* von Johannes Buchmann [Buc10] und in IT-Sicherheit von Claudia Eckert [Eck09] nachgelesen werden.

2.3.1. Ver- und Entschlüsselung

Damit die Daten gegenüber Dritten geschützt sind, muss der Klartext durch eine Verschlüsselungsfunktion in einen Ciphertext überführt werden. Die Operation für die Umwandlung eines Klartext X in einen Ciphertext C mit Hilfe des Verschlüsselungsschlüssels K_E sieht wie folgt aus: $C = E(X, K_E)$. Um den Ciphertext C wieder in den Klartext X mit Hilfe des Entschlüsselungsschlüssels K_D umzuwandeln, wird die Entschlüsselungsfunktion $X = D(C, K_D)$ verwendet.

2.3.2. Symmetrische Verschlüsselungsverfahren

Bei symmetrischen Verfahren wird ein geheimer Schlüssel für die Verschlüsselung des Klartextes verwendet. Die Entschlüsselung des Ciphertextes erfolgt entweder mit dem selben Schlüssel oder er lässt sich leicht aus dem Verschlüsselungsschlüssel ableiten.

Für symmetrische Verfahren müssen die beiden Kommunikationspartner vorab einen gemeinsamen, geheimen Schlüssel austauschen. Dieser Schlüssel darf nur den beiden Kommunikationspartnern bekannt sein, da sonst ein Dritter den Ciphertext entschlüsseln könnte. Dadurch hängt die Sicherheit nicht nur vom verwendeten Verfahren, sondern auch vom Schlüsselaustausch und der Aufbewahrung des gemeinsamen Schlüssels ab.

Blockschiffren

Für die Verschlüsselung von Klartexten bekannter Länge verwendet man Blockschiffren. Der Klartext wird in Blöcke fester Länge unterteilt und die einzelnen Blöcke werden mit der gleichen Funktion verschlüsselt. Die Blocklänge hängt vom verwendeten Verschlüsselungsverfahren und der Schlüssellänge ab.

Entspricht der letzte Block nach der Unterteilung nicht der geforderten Blocklänge, so wird er bis zu dieser Länge aufgefüllt. Das Verfahren zum Auffüllen der Blöcke wird Padding genannt. Dieses ist nicht für alle Blockschiffren zwingend erforderlich.

Cipher Block Chaining (CBC) Mode und Electronic Code Book (ECB) sind zwei beim nPA verwendete Blockchiffre-Modi. Bei CBC geht der vorhergehende verschlüsselte Block in die Berechnung des nachfolgenden Blocks ein, wobei beim ersten Block der Initialisierungswert verwendet wird. ECB verwendet keine Verkettung, wie es bei CBC der Fall ist. Auf jeden Block wird nur der Verschlüsselungsschlüssel angewendet.

Data Encryption Standard (DES)

DES (siehe FIPS 46-3 Standard [SN99]) ist ein symmetrisches Verschlüsselungsverfahren und wurde im Jahre 1976 von der US-Regierung als offizieller Standard für symmetrische Verfahren festgelegt.

Die Ver- und Entschlüsselung basiert auf 64 Bit großen Blöcken, welche mit einem 56 Bit (7 Byte) langem symmetrischen Schlüssel verschlüsselt werden. Der Schlüssel selbst besteht aus 64 Bit (8 Byte). Die sieben höheren Bytes stellen den Schlüssel dar, während das kleinste Byte als *Paritätbyte* verwendet wird. Das Parität Byte wird von Smartcard-Betriebssystemen nicht benutzt und kann entsprechend beliebig gesetzt werden.

DES sollte nicht mehr verwendet werden, da es den heutigen Sicherheitsstandards nicht mehr entspricht (siehe BSI TR-02102 [BSI08f]). Es wird an dieser Stelle als Basis für *3DES* benötigt.

Triple DES (3DES)

3DES oder *TDES* (siehe FIPS 46-3 Standard [SN99]) basiert auf dem *DES*-Algorithmus und verwendet einen 112-bit langen Schlüssel für die Verschlüsselung der acht Byte großen Blöcke. Der Schlüssel ist gegenüber DES doppelt so lang. Dabei besteht der Schlüssel aus 16 Bytes (112-bit effektive Schlüssellänge), welche auf zwei acht Byte Blöcke (K_1 und K_2) aufgeteilt wird. Das niederwertigste Byte der beiden Blöcke ist jeweils ein Parität Byte.

Der Klartext X wird durch Verschlüsselung wie folgt in den Ciphertext C überführt:

$$C = DES(DES^{-1}(DES(X, K_1), K2), K1)$$

Der Ciphertext C wird durch Entschlüsselung in den Klartext X wie folgt überführt:

$$X = DES^{-1}(DES(DES^{-1}(C, K_1), K2), K1)$$

Der Verschlüsselungsaufwand bei 3DES verdreifacht sich gegenüber DES, wobei sich die Schlüssellänge lediglich verdoppelt. Die Schlüssellänge von 3DES entspricht nicht den heutigen Anforderungen für symmetrische Verschlüsselungsverfahren (siehe BSI TR-02102 [BSI08f]).

3DES wurde bei den ersten Testausweisen verwendet. Bei den Anwendungstests und der finalen Ausweisen wird ausschließlich AES verwendet, welches im folgenden Abschnitt beschrieben ist.

Advanced Encryption Standard (AES)

AES (von den Erfindern Rijndael genannt) wurde seit dem Jahr 2000 von der US-Regierung als offizieller Nachfolger des DES Standards für symmetrische Verschlüsselungsverfahren ernannt. Bei AES handelt es sich wie bei DES um ein blockbasiertes Verfahren. Im Gegensatz zu DES besitzt AES zusätzlich variable Block- und Schlüssellängen.

AES ist ein rundenbasiertes Blockchiffreverfahren. Der Klartextblock wird über mehrere Runden mit einer festen Abfolge von Funktionen bearbeitet und dadurch verschlüsselt. Bei AES können 128, 192 und 256 Bit Block- und Schlüssellängen verwendet werden, wobei die Block- und Schlüssellängen beliebig miteinander kombiniert werden können. Die Kombination der Block- und Schlüssellängen beeinflusst die Anzahl der Runden, in denen der Klartextblock bearbeitet wird.

AES wird von Experten als performante und sichere Chiffre eingestuft. Im Vergleich zu DES sind bei AES nach Eckert [Eck09] bis heute keine schwachen Schlüssel bekannt.

2.3.3. Asymmetrische Verschlüsselungsverfahren

Asymmetrische Verschlüsselungsverfahren basieren auf Schlüsselpaaren bestehend aus einem öffentlichen Schlüssel (Public Key) und einem privaten Schlüssel (Private Key). Jeder Kommunikationsteilnehmer benötigt ein eigenes Schlüsselpaar. Der öffentliche Schlüssel wird aus den Parametern des geheimen Schlüssels abgeleitet, wobei nur der Besitzer des geheimen Schlüssels den öffentlichen Schlüssel leicht berechnen kann. Die Berechnung des geheimen Schlüssels aus dem öffentlichen Schlüssel ist schwer.

Der öffentliche Schlüssel muss allen Kommunikationsteilnehmern bekannt sein. Um einem Kommunikationsteilnehmer eine Klartext-Nachricht zu schicken, wird dieser mit dem öffentlichen Schlüssel des Empfängers verschlüsselt. Nur der Empfänger kann die verschlüsselte Nachricht mit seinem privaten Schlüssel entschlüsseln.

Asymmetrische Verfahren werden meistens in Verbindung mit symmetrischen Verfahren eingesetzt. Die Klartext-Nachricht wird mit einem symmetrischen Verfahren verschlüsselt und der symmetrische Schlüssel wird anschließend mit dem öffentlichen Schlüssel des Empfängers verschlüsselt. Der Vorteil dieses Vorgehens liegt vor allem in der höheren Geschwindigkeit der symmetrischen Verschlüsselungsverfahren. Sollte dieselbe Nachricht an mehrere Empfänger verschickt werden, so muss der Absender den Klartext nur einmal symmetrisch verschlüsseln und dann pro Empfänger jeweils nur den symmetrischen Schlüssel mit dem öffentlichen Schlüssel des jeweiligen Empfängers verschlüsseln.

Elliptische Kurven Kryptographie

Elliptic Curve Cryptography (ECC) (siehe auch Buchmann [Buc10]) ist ein asymmetrisches Verfahren und verwendet Operationen auf elliptischen Kurven über endliche

Körper. Die Sicherheit des Systems baut auf dem mathematischen Problem des diskreten Logarithmuses auf. Wie auch bei RSA (siehe RSA PKCS#1 [RA02]) ist es schwierig, den diskreten Logarithmus in der Gruppe der elliptischen Kurven zu berechnen.

Im Vergleich zu RSA fallen bei elliptischen Kurven die Schlüssellängen für die gleichen Sicherheitsanforderungen erheblich kürzer aus (siehe Buchmann [Buc10]). Dadurch werden deutlich weniger Ressourcen für die Berechnungen benötigt, was für Smartcards einen großen Vorteil gegenüber RSA darstellt.

Vom Bundesamt für Sicherheit in der Informationstechnik wird derzeit für RSA eine Schlüssellänge von 2048 Bit gefordert, während für ECC lediglich 224 Bit für die gleichen Sicherheitsanforderungen ausreichen (siehe BSI TR-02102 [BSI08f]).

2.3.4. Anonymer Schlüsselaustausch

Zwei anonyme Parteien wollen über eine unsichere Leitung miteinander kommunizieren können. Bevor die Kommunikation beginnen kann, soll über die unsichere Leitung eine sichere Verbindung aufgebaut werden. Dafür muss ein Schlüsselaustausch durchgeführt werden, über den die Parteien ein gemeinsames Geheimnis berechnen können, welches nur Ihnen bekannt ist. Diese Anforderungen erfüllt Diffie-Hellman, welches im folgenden Abschnitt erläutert wird.

Diffie-Hellman

Das Diffie-Hellman Schlüsselaustausch-Verfahren ermöglicht es, über eine unsichere Leitung Schlüsselpaare auszutauschen, um eine sichere Verbindung zwischen den Kommunikationspartnern aufzubauen.

Vor dem Schlüsselaustausch (siehe Buchmann [Buc10]) müssen die gemeinsamen öffentlichen *Domain Parameter* (siehe Tabelle 2.1) beiden Parteien bekannt sein. Sie werden für den Schlüsselaustausch und die Generierung des gemeinsamen Geheimnisses (Shared Secret) benötigt.

Die Berechnung des *Gemeinsamen Geheimnisses (Shared Secret)* für die Absicherung der Kommunikation erfolgt in mehreren Schritten. Basierend auf den öffentlichen Domain Parametern generiert jede Partei ihren eigenen, privaten Schlüssel, welcher aus einer Zufallszahl mit bestimmten Eigenschaften besteht. Daraus berechnen Sie mit Hilfe von g und p einen eigenen, öffentlichen Schlüssel. Das Geheimnis der ersten Partei ist a und das Geheimnis der zweiten Partei ist b, aus denen die öffentlichen Schlüssel (A und B) nach der folgenden Regel abgeleitet werden:

Parameter	Beschreibung
p	Primzahl p definiert den Modulus F_p
q	Ordnung der Subgruppe
g	Generator
y	Öffentlicher Schlüssel einer Partei

Tabelle 2.1.: Diffie-Hellman (DH) Domain Parameter

$$A = g^a \mod p$$
$$B = g^b \mod p$$

Die öffentlichen Schlüssel werden zwischen den beiden Parteien ausgetauscht. Mit Hilfe des öffentlichen Schlüssel der jeweils anderen Parteien berechnen sie das gemeinsame Geheimnis (Shared Secret) K:

$$K = B^a \mod p = g^{b \cdot a} \mod p = g^{a \cdot b} \mod p = A^b \mod p$$

Allerdings ist Diffie-Hellman ohne zusätzliche Maßnahmen für Man-in-the-Middle Attacken anfällig (siehe Buchmann [Buc10]), was beachtet werden muss.

Elliptic Curve Diffie-Hellman (ECDH)

Wie bei Diffie-Hellman werden auch beim *Elliptic Curve Diffie-Hellman (ECDH)* (siehe auch Buchmann [Buc10]) gemeinsame, öffentliche Domain Parameter für die Kommunikationspartner benötigt.

Mit Hilfe der öffentlichen Domain Parameter (siehe Tabelle 2.2) können die beiden Parteien ihre privaten und öffentlichen Schlüssel berechnen und den Schlüsselaustausch durchführen.

Aus den Parametern wird die gemeinsame elliptische Kurve erstellt:

$$curve_{ecc} = ECCurve(p, a, b)$$

Parameter	Beschreibung
p	Primzahl p definiert das finite Feld F_p
a & b	definiert die Kurve: $y^2 \mod p = (x^3 + a \cdot x + b) \mod p$
G	Generator Point (x_G, y_G) ist ein Punkt auf der elliptischen Kurve gewählt durch kryptographische Operationen.
r	Ist die Ordnung der elliptischen Kurve.
h	Ist der Kofaktor.

Tabelle 2.2.: Elliptic Curve Diffie-Hellman (ECDH) Domain Parameter

Der private Schlüssel d muss im Interval $[1, r-1]$ liegen. Der öffentliche Schlüssel Q wird mit Hilfe des privaten Schlüssels d und dem Punkt G wie folgt berechnet:

$$Q = d \cdot G$$

Nach dem Schlüsselaustausch berechnet jede Partei den gemeinsamen, geheimen Schlüssel K nach der folgenden Formel:

$$K = d_A \cdot Q_B = d_A \cdot d_B \cdot G = d_B \cdot d_A \cdot G = d_B \cdot Q_A$$

Damit ist der Schlüsselaustausch abgeschlossen. Wie auch DH ist ECDH ohne weitere Maßnahmen für Man-in-the-Middle Attacken anfällig (siehe Buchmann [Buc10]).

2.3.5. Kryptographische Hashfunktionen

Kryptographische Hashfunktionen sind Einwegfunktionen, die bei einer Hashfunktion zu einer gegebenen Eingabe eine immer eindeutige Ausgabe erzeugen. Die Länge der Ausgabe wird durch die Hashfunktion definiert. Sie ist meistens deutlich kürzer als die Eingabe. Kryptographische Hashfunktionen sind Einwegfunktionen, weil durch die Ausgabe nicht auf das Urbild geschlossen werden kann. Anhand der Ausgabe können lediglich die Eingabedaten auf deren Integrität[1] überprüft werden.

Folgende Eigenschaften weisen kryptographisch sichere(starke) Hashfunktionen auf (Quelle Buchmann [Buc10]):

- Das berechnen eines Hashwertes $h(M)$ zu einer gegebene Eingabe M soll möglichst einfach sein.

[1] Unter Datenintegrität ist nach Eckert [Eck09] der Schutz vor unautorisierten und unbemerkten Modifikationen der Daten zu verstehen.

Hash Algorithmus	Länge des Hashwertes	Blocklänge	Maximale Länge der Eingabedaten
SHA-1	160 bits	512 bits	$2^{64} - 1bit$
SHA-224	224 bits	512 bits	$2^{64} - 1bit$
SHA-256	256 bits	512 bits	$2^{64} - 1bit$
SHA-384	384 bits	1024 bits	$2^{128} - 1bit$
SHA-512	512 bits	1024 bits	$2^{128} - 1bit$

Tabelle 2.3.: Eigenschaften der SHA-Hashfunktionen Familie (Quelle: FIPS 180-2 [SN02])

- Es sollte effektiv nicht durchführbar sein, für einen gegebenen Hashwert $h(M)$ einen zweite Eingabe M' zu finden, welche den selben Hashwert $h(M) = h(M')$ hat (Kollisionsresistenz).

Kryptographische Hashfunktionen werden auch für die Ableitung des gemeinsamen Geheimnisses (Shared Secrets) verwendet, in dem der ganze gemeinsame Schlüssel oder Teile des Schlüssels gehasht werden.

SHA-Hashfunktionen

SHA-Hashfunktionen gelten derzeit als kryptographisch sichere Hashfunktionen (siehe BSI TR-02102 [BSI08f]). Sie werden unter anderem für Signaturen und zur Generierung von Schlüsseln verwendet.

Die Familie der SHA-Hashfunktionen sind im NIST[2] Standard FIPS[3] 180-2 [SN02] beschrieben und werden im Zusammenhang mit dem nPA für die Schlüsselableitung eingesetzt. Die Tabelle 2.3 beschreibt die Eigenschaften der einzelnen Hashfunktionen.

SHA-1 hat eine Ausgabelänge von 160 Bit, was vom BSI für Integritätsprüfungen als zu kurz eingeschätzt wird (siehe BSI TR-02102 [BSI08f]). Beim nPA wird SHA-1 ausschließlich für die Ableitung von Schlüsseln mit einer Länge von bis zu 128 Bits verwendet.

[2]National Institute of Standards and Technology
[3]Federal Information Processing Standard

Schlüssellänge & Algorithmus	Hashfunktion
112-bit 3DES	SHA-1
128-bit AES	SHA-1
192-bit AES	SHA-256
256-bit AES	SHA-256

Tabelle 2.4.: Hashfunktionen für die Generierung der verschiedenen Schlüssellängen bei Schlüsselableitungen (Quelle: BSI TR-03111 [BSI09c])

2.3.6. Schlüsselableitungsfunktion

Mit Hilfe von kryptographischen Funktionen sollen aus einem gemeinsamen Geheimnis K_{seed} ein Schlüssel abgeleitet werden. Die Eingabe bildet das gemeinsame Geheimnis K_{seed}, eine optionale Zufallszahl r und ein Counter c.

Die in Zusammenhang mit dem nPA verwendeten Schlüsselableitungsfunktionen sind in dem Standard ANSI[4] X9.63 [ans01] beschrieben.

Die allgemeine Ableitungsfunktion sieht wie folgt aus:

$$keydata = H(K_{seed} \,||\, [r] \,||\, c)$$

Die Länge der Ausgabe hängt nicht vom Passwort, sondern von der verwendeten Hashfunktion ab. Die Ausgabe der Hashfunktion muss länger sein als die benötigte Schlüssellänge. Die jeweiligen Schlüssellängen und die im Zusammenhang mit dem nPA verwendeten Hashfunktionen sind in der Tabelle 2.4 aufgeführt. Ist die Ausgabe *keydata* länger als die benötigte Schlüssellänge, so werden nur die signifikantesten Bits für den Schlüssel verwendet.

Bei ECDH werden Punkte als x- und y-Koordinaten dargestellt. Für die Ableitungen wird lediglich der x-Wert des gemeinsamen Schlüssels K_{both} als K_{SEED} verwendet.

In der Regel wird die Zufallszahl r dazu verwendet, um die geforderte Blocklänge der Eingabe zu erreichen. Die Länge des Counters c ist fest. Wird kein Zufallszahl verwendet oder hat sie eine feste Länge, so wird der Counter c mit Nullen aufgefüllt, bis die geforderte Blocklänge erreicht ist.

[4] American National Standards Institute

2.3.7. Message Authentication Code (MAC)

Hashfunktionen helfen die Unverfälschtheit der Daten zu überprüfen, aber sie garantieren nicht den Ursprung der Daten, weil jeder den Hashwert für eine bekannte Funktion und gegebene Eingabedaten berechnen kann. Um die Authentizität der Daten überprüfen zu können, wird der Message Authentication Code (MAC) (siehe auch NIST[5] 800-38B [Dwo04]) verwendet.

Als Basis für den MAC können Hashfunktionen oder Verschlüsselungsfunktionen verwendet werden. In beiden Fällen muss vorab ein gemeinsamer, geheimer Schlüssel K_{MAC} vereinbart werden, welcher für die Generierung der MACs und für die Überprüfung der Authentizität der Eingabedaten verwendet wird.

Werden Hashfunktionen für die Erzeugung von MAC verwendet, so heißt das Verfahren Hash Message Authentication Code (HMAC). Der geheime Schlüssel wird an den Klartext angehängt und dann gehasht $mac = MAC(M \mid\mid K_{MAC})$. Die Ausgabe der Hashfunktionen mac ist der MAC. Die Länge des MACs hängt von der Länge der Hashfunktion ab.

Werden Verschlüsselungsfunktionen für die Erzeugung von MAC verwendet, so heißt das Verfahren Cipher Message Authentication Code (CMAC). Der Klartext wird mit der Verschlüsselungsfunktion verschlüsselt. Die Ausgabe des letzten Verschlüsselungsblockes ist der MAC. Die Länge des MACs hängt von der Schlüssellänge der Verschlüsselungsfunktion ab.

Für den nPA wird CMAC verwendet, um die Authentizität und die Integrität der übertragenen Daten sicherzustellen.

2.4. Public Key Infrastruktur

Um öffentliche Schlüssel verwalten zu können und deren Gültigkeit zu prüfen, muss für den nPA eine länderübergreifende Public Key Infrastruktur (PKI) geschaffen werden. Diese darf sich nicht allein auf Deutschland beschränken, da der nPA als Reisedokument innerhalb der Europäischen Union gültig ist.

Eine allgemeine Beschreibung für PKIs kann in *Understanding PKI: Concepts, Standards, and Deployment Considerations* von Adams [AL02] oder in *IT-Sicherheit* von Eckert [Eck09] nachgelesen werden. Die Grundidee hinter einer PKI sowie deren Aufbau

[5] National Institute of Standards and Technology

der PKI für den nPA wird in den folgenden Abschnitten kurz erklärt. Eine funktionierende PKI wird für diese Arbeit vorausgesetzt. Somit dient dieser Abschnitt lediglich dem notwendigen Verständnis zu dem Aufbau und Funktionsweise einer PKI. Das Wissen wird für die verwendeten Protokolle benötigt.

2.4.1. Vertrauensmodell

Für die Durchsetzung der PKI wird ein Vertrauensmodell benötigt, das auf geprüften Vertrauensstrukturen basiert. Dieses Vertrauen muss auch bei fremden Person funktionieren, wie das bei der Authentisierung mit dem nPA notwendig ist. Der Geschäftspartner oder die Behörde muss vertrauen können, dass der zugehörige öffentliche Schlüssel gültig ist. Der Bürger muss vertrauen können, dass er mit der richtigen Institution kommuniziert. Dafür ist nach Eckert [Eck09] „Vertrauen in die Echtheit und Glaubwürdigkeit, in die Gültigkeit der öffentlichen Schlüssel (und den dazu gehörigen privaten Schlüssel)“ notwendig.

Da beim nPA kein vertrauenswürdiges, zentrales Verzeichnis für öffentliche Schlüssel existiert, muss jeder Teilnehmer in der PKI die Authentizität aller Schlüssel prüfen können. Die Authentizität eines Schlüssels wird durch Signaturen überprüft. Jeder vertrauenswürdige Schlüssel muss eine Signatur besitzen, die seine Authentizität nachweist. Die prüfende Instanz kann sie über die Vertrauenskette der PKI prüfen.

2.4.2. Vertrauenskette

In jeder PKI existieren ein oder mehrere Vertrauensanker, die als Ausgangspunkt des Vertrauens innerhalb der PKI gelten. Diesen Vertrauensanker bildet eine Zertifizierungsstelle (CA[6]) als oberste Vertrauensinstanz und ihr muss explizit vertraut werden. Die CA signiert Schlüssel in der nächsten Hierarchiestufe, um deren Authentizität nachzuweisen. Das wird bis in die unterste Instanz fortgesetzt. Damit kann jeder über den CA-Schlüssel das Vertrauen in die unterste Instanz der Hierarchie prüfen. Dafür muss er die Authentizität der ganzen Vertrauenskette angefangen bei der CA prüfen. Ist die Unterschrift der zu prüfenden Instanz und die ganze Vertrauenskette bis zur Instanz gültig, so ist auch der öffentliche Schlüssel der Instanz authentisch.

[6] Certificate Authority

2.5. Basic Access Control (BAC)

BAC ist ein von der ICAO[7] spezifiziertes Verfahren für den Zugriff auf digital gespeicherte Ausweisdaten. Es soll die Eingabe von Personendaten bei Grenzkontrollen vereinfachen, indem die Daten vom Ausweis digital ausgelesen werden können. Alle auf dem Ausweis digital gespeicherten Daten sind ebenfalls auf dem Ausweis aufgedruckt und können optisch ausgelesen und verglichen werden.

Für den Zugriff auf den Ausweis über BAC muss vorab ein Schlüssel für die Verschlüsselung der Daten KB_{Enc} und ein Schlüssel für die Authentisierung der übertragenen Daten KB_{MAC} berechnet werden. Die Schlüssel werden mit Hilfe von SHA-1 (siehe Abschnitt 2.3.5) über die folgenden Daten erstellt:

- Dokumentennummer
- Geburtstag
- Ablaufdatum

Aus den signifikantesten 16 Bytes des Hashwertes werden die beiden Schlüssel KB_{Enc} und KB_{MAC} abgeleitet.

BAC soll sicherstellen, dass das Terminal für das Auslesen der Daten visuellen Zugriff auf den Ausweis hat. Somit bietet es einen schwachen Zugriffsschutz und kann auch nicht sicherstellen, dass nur zugelassene Lesegeräte die Daten des Ausweises auslesen. Für sensitive Daten wie Fingerabdrücke darf BAC nicht eingesetzt werden (siehe auch BSI TR-03110 [BSI09a]). Dafür wurde Extended Access Control (siehe Abschnitt 2.6) entwickelt.

Die vollständige Protokollbeschreibung von BAC ist im ICAO Dokument 9303 [ICA08b] erhalten. BAC ist nicht direkter Bestandteil dieser Arbeit, aber durch die Funktionsweise wird der Bedarf für Extended Access Control deutlicher. Vor allem kann mit BAC kein Zugriffsschutz mit verschiedenen Berechtigungen realisiert werden, wie er für den nPA notwendig ist.

[7] International Civil Aviation Organization

2.6. Extend Access Control (EAC)

EAC ist ein Verfahren für den kontrollierten Zugriff auf die digital vorliegenden Daten eines Ausweises, wobei der Zugriff auf sensible Anwendungen und Datengruppen (wie Fingerabdrücke) gezielt auf bestimmte Personengruppen eingeschränkt werden kann.

EAC besteht aus den folgenden drei Protokollen:

- *Password Authenticated Connection Establishment (PACE)* (siehe Abschnitt 2.7)
- *Terminalauthentisierung* (siehe Abschnitt 2.8)
- *Chipauthentisierung* (siehe Abschnitt 2.9)

Sie müssen in der genannten Reihenfolge erfolgreich durchgeführt werden, bevor ein Zugriff auf die persönlichen Daten des Bürgers möglich ist. Mit Hilfe von Berechtigungszertifikaten wird der Zugriff auf bestimmte Datengruppen und Personengruppen beschränkt.

2.6.1. EAC für hoheitliche Zwecke

Für eine erfolgreiche Durchführung von PACE wird immer ein Passwort benötigt. Weiterhin muss es möglich sein, PACE für hoheitliche Zwecke ohne das Mitwirken des Bürgers durchzuführen. Dafür ist auf dem Ausweis eine CAN (Card Access Number) aufgedruckt, welche für die hoheitlichen Zwecke verwendet wird.

2.7. PACE

Im ersten Schritt von EAC wird Password Authenticated Connection Establishment (PACE) durchgeführt. Der Benutzer authentisieirt sich am Chip des nPAs mit Hilfe eines schwachen Passwortes (siehe auch BSI TR-03110 [BSI09a]). Nach der erfolgreichen Authentisierung wird zwischen dem Chip und dem Terminal eine sichere Verbindung aufgebaut.

Durch PACE entstehen entscheidende Vorteile, wie sie in der BSI TR-03110 [BSI09a] beschrieben sind:

- Trotz eines kurzen und damit schwachen Passwortes ist es möglich, starke Sessionschlüssel zu generieren.

- Die Entropie[8] des Passwortes, welcher für die Authentifizierung des Terminals verwendet wird, kann sehr niedrig sein. Im Normalfall reichen sechs Zeichen aus.

In den folgenden Abschnitten werden die einzelnen Schritte von PACE im Detail beschrieben. Der Schwerpunkt liegt hierbei auf der Algorithmenbeschreibung, ohne auf die Chipkarten-Kommunikation einzugehen.

2.7.1. Schlüsselableitung aus dem Passwort

Für die Ableitung der Schlüssel (siehe Abschnitt 2.3.6) werden in PACE Hashfunktionen der SHA-Familie verwendet. Die Generierung erfolgt über den festen Schlüssel K_{SEED} und einen Counter c. Der Schlüssel K_{SEED} stellt bei PACE das Passwort in Form einer MRZ, CAN, PIN oder PUK dar. Die Kodierung der Passwörter ist in der Tabelle 2.5 beschrieben.

Passwort	Kodierung
MRZ	SHA-1(Seriennummer || Geburtsdatum || Gültig bis)
CAN	Oktet Zeichenfolge binär kodiert in [ISO8859-1]
PIN	Oktet Zeichenfolge binär kodiert in [ISO8859-1]
PUK	Oktet Zeichenfolge binär kodiert in [ISO8859-1]

Tabelle 2.5.: Kodierung der Passwörter für PACE (Quelle: BSI TR-03110 [BSI09a])

Im ersten Schritt von PACE muss das Passwort vom Bürger angefordert werden, welches in PACE verwendet werden soll. Danach übermittelt der Chip eine mit dem Passwort verschlüsselte Zufallszahl an das Terminal.

Die Schlüsselableitungsfunktion für die Entschlüsselung der Zufallszahl mit Hilfe einer eID-PIN oder CAN sieht wie folgt aus (Quelle: BSI TR-03110 [BSI09a]):

$$K_{SEED} = encode(password)$$
$$K_{DECR} = h(K_{SEED}||c) \quad mit\ c = 3_{10} = 00000003_{16}$$

Abhängig vom gewählten symmetrischen Algorithmus (3DES oder AES) und der Schlüssellänge (nur bei AES) wird die entsprechen Hashfunktion $h()$ eingesetzt (siehe

[8] Mittlerer Informationsgehalt

Tabelle 2.4). Für den Schlüssel K_{DECR} werden die signifikantesten Bits von der Ausgabe der Hashfunktion verwendet.

2.7.2. Zufallszahl entschlüsseln

Die verschlüsselte Zufallszahl z muss laut BSI TR-03110 [BSI09a] mit dem vom nPA vorgegebenen Algorithmus entschlüsselt werden. Bei den nPAs des Anwendungstests und bei den finalen nPAs wird ausschließlich AES mit einer Schlüssellänge von 128 Bit im CBC Modus und ohne Padding verwendet. Bis zur TR-03110 v2.01 [BSI09b] und damit auch bei den Ausweisen aus den Anwendungstests wurde ECB statt CBC verwendet. Als Entschlüsselungsschlüssel K_{DECR} dient die Ableitung aus dem verwendeten Passwort (siehe Abschnitt 2.3.6).

$$s = decrypt(K_{DECR}, z)$$

Die entschlüsselte Zufallszahl s wird beim nächsten Schritt (*Mapping Data*) benötigt.

2.7.3. Mapping Data

Bei Mapping Data berechnen beide Parteien auf Basis eines gemeinsamen Geheimnisses einen gemeinsamen Punkt. Das gemeinsame Geheimnis stellt die entschlüsselte Zufallszahl s dar:

$$\begin{aligned} X_{1_{Terminal}} &= G * x_{1_{Terminal}} \\ Y_{1_{nPA}} &= G * y_{1_{nPA}} \\ H &= Y_1 * x_{1_{Terminal}} = X_1 * y_{1_{nPA}} \\ G' &= s * G + H \end{aligned}$$

Mapping Data stellt sicher, dass der gemeinsame Punkt G' nur den beiden Parteien bekannt ist, welche die richtige Zufallszahl und damit auch das Passwort kennen. Damit werden vor allem *Man-in-the-Middle* und *Relay Attacken* verhindert, die ohne diesen Schritt bei DH und ECDH möglich wären.

2.7.4. Anonymer Schlüsselaustausch

Der Anonyme Schlüsselaustausch (Ephemeral Key Agreement) basiert auf dem *Elliptic Curve Diffie-Hellman (ECDH)* (siehe Abschnitt 2.3.4) Verfahren. Beide Parteien generieren basierend auf den Kurvenparametern einen temporären privaten und öffentlichen Schlüssel und tauschen die öffentlichen Schlüssel aus. Daraus berechnen beide Parteien einen gemeinsamen Punkt K_{both} auf der elliptischen Kurve:

$$\begin{aligned} X_{2_{Terminal}} &= G' * x_{2_{Terminal}} \\ Y_{2_{nPA}} &= G' * y_{2_{nPA}} \\ K_{both} &= Y_{2_{nPA}} * x_{2_{Terminal}} = X_{2_{Terminal}} * y_{2_{nPA}} \end{aligned}$$

Der berechnete gemeinsame Punkt K_{both} wird im nächsten Schritt für die gegenseitige Authentifizierung und für die Ableitung der Sessionschlüssel von Secure Messaging benötigt.

Damit keine Identifizierung des Bürgers über die Zertifikate möglich ist, werden an dieser Stelle nur temporäre und anonyme Zertifikate verwendet. Sie beinhalten nur den für ECDH benötigten Punkt des öffentlichen Schlüssels ($X_{2_{Terminal}}$ und $Y_{2_{nPA}}$).

2.7.5. Ableitung der Sessionschlüssel

Nach der erfolgreichen gegenseitigen Authentisierung berechnen beide Parteien aus dem gemeinsamen Schlüssel K_{both} die Sessionschlüssel für *Secure Messaging*. Diese werden durch die bereits bekannte Schlüsselableitungsfunktion (siehe Abschnitt 2.3.6) berechnet. Bei ECDH geht lediglich der x-Wert des gemeinsamen Punktes K_{both} in die Berechnung ein, welcher das gemeinsame Geheimnis K_{SEED} bildet.

Der Schlüssel K_{ENC} für die verschlüsselte Kommunikation und K_{MAC} für die Authentifizierung der Nachrichten werden durch die folgenden Funktionen generiert:

Datenobjekt	Kürzel	Tag	Typ
Object Identifier		0x06	Object Identifier
Primzahl Modulus	p	0x81	Unsigned Integer
Erster Koeffizient	a	0x82	Unsigned Integer
Zweiter Koeffizient	b	0x83	Unsigned Integer
Basis Punkt	G'	0x84	Punkt auf der elliptischen Kurve
Ordnung	r	0x85	Unsigned Integer
Öffentlicher Punkt	X_2 bzw. Y_2	0x86	Punkt auf der elliptischen Kurve
Kofaktor	h	0x87	Unsigned Integer

Tabelle 2.6.: Kodierung des öffentlichen Schlüssels PK für die gegenseite Authentisierung in PACE (Quelle: BSI TR-03110 [BSI09a])

$$
\begin{aligned}
K_{SEED} &= K_{both_X} \\
K_{ENC} &= h(K_{SEED}||c) \quad mit\ c = 1_{10} = 00000001_{16} \\
K_{MAC} &= h(K_{SEED}||c) \quad mit\ c = 2_{10} = 00000002_{16}
\end{aligned}
$$

Abhängig vom gewählten symmetrischen Algorithmus (3DES oder AES) und der Schlüssellänge (nur bei AES) wird die entsprechende Hashfunktion $h()$ eingesetzt (siehe Tabelle 2.4). Als Schlüssel werden die signifikantesten Bits aus der Ableitungsfunktion verwendet.

2.7.6. Gegenseitige Authentisierung

Wie in der TR-03110 [BSI09a] beschrieben ist, muss für die *gegenseitige Authentisierung (Mutual Authentication)* der vollständige öffentliche Schlüssel PK inklusive der Parameter mit CMac signiert und von beiden Parteien gegenseitig verifiziert werden. Das Format des öffentlichen Schlüssels ist in der Tabelle 2.6 beschrieben.

Für CMac wird ein Signaturschlüssel K_{MAC} (siehe Abschnitt 2.7.5) benötigt. Für die Blockschiffre wird AES mit 128 Bit im CBC-Modus und ohne Padding verwendet. Die ersten 64 Bit der 128 Bit langen Ausgabe $Sign_{CMAC}$ der CMAC-Funktionen dienen als Authentifizierungstoken. Diese werden für die gegenseitige Authentifizierung der Parteien ausgetauscht.

$$
\begin{aligned}
K_{SEED} &= K_{both_X} \\
K_{MAC} &= h(K_{SEED}||c) \quad mit\ c = 2_{10} = 00000002_{16} \\
Sign_{CMAC} &= CMAC(K_{MAC}, PK)
\end{aligned}
$$

Erst die gegenseitige Authentisierung zeigt, ob die davor ausgeführten Schritte richtig waren und die beiden Parteien ihren gemeinsamen Schlüssel K_{both} richtig berechnet haben.

2.7.7. Protokollablauf

Die Authentisierung erfolgt in mehreren Schritten. Die Tabelle 2.7 soll noch einmal den vollständigen Prozess beschreiben. Dieser richtet sich an der Kommunikationsart einer Chipkarte und beschreibt den Ablauf von PACE für die Authentisierung am nPA.

Die Aktionen gehen immer vom Terminal aus. Die Chipkarte ist ein passiver Kommunikationsteilnehmer und antwortet nur auf die gestellten Befehle.

Schritt	Terminal	Richtung	nPA
1	Passwort PW anfordern und daraus den Schlüssel berechnen $K_{PW} = KDF(PW)$		
2	Inhalt von EF.CardAccess anfordern	$\overset{ReadBinary()}{\longrightarrow}$	
	EF.CardAccess auswerten	$\overset{EF.CardAccess}{\longleftarrow}$	EF.CardAccess Inhalt
3	**Manage Security Environment:** Setze DST/AT (Key Identifier, Algorithm Identifier)	$\overset{MSE:Set\ AT}{\longrightarrow}$	
		$\overset{OK}{\longleftarrow}$	Überprüft und setzt die übergebenen Parameter
4	**General Authenticate:** Setze Terminaltyp und auf welche Datengruppen der eID-Funktion des nPAs nach der erfolgreichen Authentisierung zugegriffen werden soll.	$\overset{General\ Authenticate}{\longrightarrow}$	Generate nonce s
	s = decrypt(z,K_{PW})	$\overset{Encrypted\ nonce\ z}{\longleftarrow}$	z = encrypt(s, K_{PW})
5	**Mapping Data**	$\overset{X_{1_{Terminal}}}{\longrightarrow}$	Berechne G'
	Berechne G'	$\overset{Y_{1_{nPA}}}{\longleftarrow}$	(Mapping Data) OK
6	**General Authenticate**	$\overset{X_{2_{Terminal}}}{\longrightarrow}$	Ableiten des Punktes K_{both}
	Ableiten des Punktes K_{both} Sessionschlüssel ableiten	$\overset{Y_{2_{nPA}}}{\longleftarrow}$	OK Sessionschlüssel ableiten
7	**Mutual Authenticiation K_{MAC}**		
	General Authenticate	$\overset{Sign_{CMAC_{Terminal}}}{\longrightarrow}$	Externe und Interne Authentisierung
	Externe Authentisierung	$\overset{Sign_{CMAC_{nPA}}}{\longleftarrow}$	Sende (Authentisierungsdaten) OK Setze den Sicherheitsstatus für Secure Messaging
Sichergestellt, dass die PIN richtig eingegeben wurde			
Ab dieser Stelle alles über Secure Messaging mit den Sessionschlüsseln K_{ENC} und K_{MAC}			

Tabelle 2.7.: Ablauf der PACE Authentisierung

2.8. Terminalauthentisierung

Die Terminalauthentisierung (siehe BSI TR-03110 [BSI09b]) steuert den Nachweis der Zugriffsrechte eines Terminals. Das Terminal ist in der Regel ein über Netzwerk angeschlossenes Remote Terminal eines eID-Service Betreibers.

Die Aufgabe der Terminalauthentisierung (TA) ist die Übermittlung der Zugriffsrechte des Terminals an den nPA in Form eines Berechtigungszertifikates (siehe Abschnitt 1.4.3). Dieses Berechtigungszertifikat kann vom nPA auf deren Gültigkeit und die Berechtigung für den Zugriff auf die bestimmten Datengruppen überprüft werden. Der Zugriff auf die Daten kann allerdings erst nach der Chipauthentisierung erfolgen, weil die Terminalauthentisierung nicht ohne Chipauthentisierung verwendet werden darf. Ohne Chipauthentisierung kann nicht sicher gestell werden, dass es sich um einen gültigen Ausweis handelt, auf den das Terminal zugreift (siehe TR-03110 [BSI09a]).

Bei der Terminalauthentifizierung wird ein temporärer öffentlicher Schlüssel des Terminals durch das Berechtigungszertifikat authentifiziert. Dieser temporäre öffentliche Schlüssel wird bei der Chipauthentifizierung (CA) für die sichere Verbindung zwischen Terminal und nPA verwendet. Über ihn werden die Sessionschlüssel für die sichere Verbindung ausgehandelt.

Die Zugriffsrechte des authentifizierten Terminals werden an die Sessionschlüssel gebunden, so dass kein anderes Terminal auf die Datengruppen mit den gesetzten Berechtigungen zugreifen kann.

2.8.1. Protokollablauf

Am Anfang muss das Terminal sicherstellen, dass der nPA das Berechtigungszertifikat des Terminals verifizieren kann. Der Anfang der Zertifikatskette bildet die so genannte Country Verifying Certificate Authority (CVCA). Diese befindet sich auf dem nPA und bildet den Vetrauensanker der Zertifikatskette. Über die CVCA kann der nPA die ganze Zertifikatskette bis zum Berechtigungszertifikat überprüfen. Dafür übermittelt das Terminal das Document Verifier (DV) Zertifikat und anschließend das Berechtigungszertifikat des Terminals an dem nPA. Bei jedem dieser Schritte kann der nPA jedes der Zertifikate prüfen und im letzten Schritt das Berechtigungszertifikat des Terminals verifizieren.

Die TA verifiziert nur das Berechtigungszertifikat über die Zertifikatskette, sie verifiziert aber nicht den Chip des nPAs, mit welchem das Terminal kommuniziert. An dieser Stelle muss eine Verknüpfung zwischen TA und CA erfolgen. Dafür generiert das Terminal

ein temporäres Schlüsselpaar, über welches bei der CA die Sessionschlüssel für Secure Messaging abgeleitet werden.

Das temporäre Schlüsselpaar basiert auf den Domainparametern der CA, die im nPA gespeichert sind und von Terminal ausgelesen werden. Der temporäre Schlüssel wird erst später für die Ableitung der Sessionschlüssel in der CA benötigt. An dieser Stelle wird nur der X-Wert des anonymen öffentlichen Schlüssels PK_{TA_X} an den nPA übertragen.

Anschließend fragt das Terminal eine 8 Byte lange Zufallszahl vom nPA ab. Diese Zufallszahl r_{TA} geht in die externe Authentifizierung ein, welche den letzten Schritt von TA bildet.

Die externe Authentifizierung authentisiert den temporären Schlüssel von PACE und den temporären Schlüssel für die nachfolgende CA über das Berechtigungszertifikat des Terminals. Dafür wird der X-Wert des temporären öffentlichen Schlüssels von PACE PK_{PACE_X}, die Zufallszahl $r_{TA_{nPA}}$, der X-Wert des temporären öffentlichen Schlüssels des Terminals PK_{TA_X} und das optionale Auxiliary Data mit dem privaten Schlüssel des Berechtigungszertifikats SK_{Term} signiert. Die Signatur s_{Term} wird an den nPA übertragen, welcher mit Hilfe des öffentlichen Schlüssels aus dem Berechtigungszertifikat die Signatur überprüfen kann.

$$s_{Term} = sign_{K_{Term}}(PK_{PACE_X} \mid\mid r_{TA_{nPA}} \mid\mid PK_{TA_X} \mid\mid auxiliarydata)$$

Schritt	Terminal	Richtung	nPA
1	Setze das CVCA Zertifikat, um das DV Zertifikat zu verifizieren	$\overset{SetDST}{\longrightarrow}$ $\overset{OK}{\longleftarrow}$	Speichert welches Zertifikat für die Überprüfung verwendet wird.
2	Verifiziere DV Zertifikat (an den nPA übermittelt)	$\overset{DVZert.}{\longrightarrow}$ $\overset{OK}{\longleftarrow}$	Erfolgreich überprüft
3	Setze das DV Zertifikat, um das Berechtigungszertifikat des Terminals zu verifizieren.	$\overset{SetDST}{\longrightarrow}$ $\overset{OK}{\longleftarrow}$	Speichert welches Zertifikat für die Überprüfung verwendet wird.
4	Verifiziere das Berechtigungszertifikat	$\overset{Ber.-zert.}{\longrightarrow}$ $\overset{OK}{\longleftarrow}$	Erfolgreich überprüft
5	Generiere das temporäre Schlüsselpaar auf Basis der CA-Parameter aus der EF.CardAccess. Übermittele den X-Wert des anonymen temporären Schlüssel PK_{TA_X} an den nPA	$\overset{PK_{TA_X}}{\longrightarrow}$ $\overset{OK}{\longleftarrow}$	Speichert PK_{TA_X} für CA
6	**Get Challenge:** der Länge 8 Byte	$\overset{GetChallenge}{\longrightarrow}$ $\overset{r_{TA_{nPA}}}{\longleftarrow}$	Generiere Challenge $r_{TA_{nPA}}$
5	**Mapping Data** (siehe Abschnitt 2.7.3) Berechne G'	$\overset{X_{1_{Terminal}}}{\longrightarrow}$ $\overset{Y_{1_{nPA}}}{\longleftarrow}$	Berechne G' (Mapping Data) OK
6	**External Authentication:** Erstelle Signatur s_{Term} Sessionschlüssel berechnen (siehe Abschnitt 2.7.5)	$\overset{s_{Term}}{\longrightarrow}$ $\overset{OK}{\longleftarrow}$	Verify Signature Verifiziere s_{Term} Sessionschlüssel berechnen (siehe Abschnitt 2.7.5)

Tabelle 2.8.: Ablauf der Terminalauthentisierung

2.9. Chipauthentisierung

Mit BAC lässt sich die Integrität und die Authentizität der auf dem Chip gespeicherten Daten prüfen. Allerdings sind alle im BAC Protokoll verwendeten Daten öffentlich, so dass der Chip geklont werden kann (siehe Eckert [Eck09], Aktive Authentifizierung des Chips). Das verhindert die Chipauthentifizierung (CA), in dem auf dem Chip eine privater und nicht auslesbarer Schlüssel gespeichert wird. Für eine erfolgreiche CA muss der Chip nachweisen, dass er den zum öffentlichen Schlüssel passenden privaten Schlüssel besitzt.

Chipauthentisierung ist im TR-03110 [BSI09a] Standard vom BSI spezifiziert und basiert auf dem Diffie-Hellman Schlüsselaustausch Verfahren. Beim nPA für die Anwendungstests und bei den finalen Ausweisen wird ausschließlich ECDH verwendet.

Die Chipauthentisierung ist das letzte Protokoll in EAC. Mit den aus CA berechneten Sessionschlüsseln wird ein sicherer Kanal zwischen dem Remote Terminal des eID-Service Betreibers und dem Chip hergestellt. Ist der sichere Kanal aufgebaut, so kann nur das Remote Terminal auf die Datengruppen des Chips zugreifen.

2.9.1. Protokollablauf

Bei CA muss der Chip nachweisen, dass er im Besitz des privaten Schlüssels ist, der zu seinem öffentlichen Schlüssel PK_{nPA} gehört. Bevor CA durchgeführt werden kann, muss das Terminal den statischen, öffentlichen Schlüssel des Chips auslesen. Dieser ist in zusammen mit einer Signatur im $PKCS\#7$ [Kal98] Format auf dem nPA gespeichert und kann erst nach einer erfolgreichen TA ausgelesen werden. Das Terminal liest den Inhalt aus und überprüft die Gültigkeit des öffentlichen Schlüssel.

Im ersten Schritt sendet das Terminal den temporären öffentlichen Schlüssel PK_{TA_X} aus der Terminalauthentisierung an den Chip des nPAs. Der Chip kann den temporären öffentlichen Schlüssel über die Signatur aus der Terminalauthentisierung verifizieren.

Anschließend können beide das gemeinsame Geheimnis mit Hilfe von ECDH berechnen:

$$\begin{aligned} X_{Terminal} &= G * x_{Terminal} \\ Y_{nPA} &= G * y_{nPA} \\ K_{both} &= Y_{nPA} * x_{Terminal} = X_{Terminal} * y_{nPA} \end{aligned}$$

Durch ECDH weist der Chip auch nach, dass er im Besitz des privaten Schlüssels ist, der zu seinem statischen, öffentlichen Schlüssel PK_{nPA} gehört. Das Terminal kann das Zertifikat vom nPA über die Signatur auf deren Gültigkeit überprüfen. Aus Datenschutzgründen (siehe TR-03110 [BSI09a]) ist das Zertifikat nicht individuell pro Ausweis, so dass keine Nachverfolgung darüber möglich ist. Trotzdem kann das Terminal das Zertifikat auf deren Gültigkeit prüfen und somit sicherstellen, dass es sich um keine Fälschung oder eine Kopie des Ausweises handelt. Das gilt nur, so lange der Angreifer den privaten Schlüssel vom nPA nicht extrahieren oder berechnen kann.

Im letzten Schritt von CA generiert der Chip des nPAs eine Zufallszahl und leitet aus dem gemeinsamen Geheimnis K_{both} und der Zufallszahl r_{nPA} die Session-Schlüssel für Secure Messaging ab:

$$
\begin{aligned}
K_{SEED} &= K_{both_X} \\
K_{ENC} &= h(K_{SEED} \mid\mid r_{nPA} \mid\mid c) \quad mit\ c = 1_{10} = 00000001_{16} \\
K_{MAC} &= h(K_{SEED} \mid\mid r_{nPA} \mid\mid c) \quad mit\ c = 2_{10} = 00000002_{16}
\end{aligned}
$$

Mit Hilfe vom CMAC und dem Schlüssel K_{MAC} berechnet der Chip des nPAs den Authentifizierungstoken $T_{Terminal}$. Das Token wird über den temporären öffentlichen Schlüssel$PK_{Terminal}$ aus der TA berechnet. der Chip sendet ihn zusammen mit der Zufallszahl r_{nPA} an das Terminal.

Das Terminal berechnet ebenfalls die Sessionschlüssel K_{ENC} und K_{MAC}. Jetzt kann das Terminal den vom nPA gesendeten Authentifizierungstoken $T_{Terminal}$ überprüfen.

$$T_{Terminal} = CMAC(K_{MAC}, PK_{Terminal})$$

Damit ist CA abgeschlossen. Die Sessionschlüssel von PACE verlieren ihre Gültigkeit. Das Remote Terminal greift über die neu berechneten Session Schlüssel K_{ENC} und K_{MAC} auf den nPA zu, während das lokale Terminal die vom Remote Terminal geschützten APDUs unverändert an den nPA weiterleitet.

Die Berechtigung für die Anwendungen und damit auch den Zugriff auf die Datengruppen ist an diese neuen Sessionschlüssel gebunden. Durch die Kombination des privaten Schlüssels vom nPA und dem privaten Schlüssel des Remote Terminals, hat kein Dritter Zugriff auf die Datenübertragung und die Daten.

Schritt	Terminal	Richtung	nPA
1	**ReadBinary:** EF.CardSecurity PK_{nPA} extrahieren	$\overset{ReadBinary()}{\longrightarrow}$ $\overset{EF.CardSecurity}{\longleftarrow}$	
2	**MSE:Set AT**	$\overset{MSE:SetAT}{\longrightarrow}$ $\overset{OK}{\longleftarrow}$	 OK
3	**General Authenticate** Ableiten des Punktes K_{both} Sessionschlüssel K_{ENC} und K_{MAC} ableiten Überprüfe den Authentifizierungstoken $T_{Terminal}$	$\overset{PK_{Terminal}}{\longrightarrow}$ $\overset{r_{nPA},T_{Terminal}}{\longleftarrow}$	Ableiten des Punktes K_{both} Sessionschlüssel K_{ENC} und K_{MAC} ableiten
Stellt sicher, dass alle Schlüssel gültig sind			
Secure Messaging von PACE ist ab jetzt ungültig			
Secure Messaging für nachfolgenden Befehle mit den Sessionschlüsseln K_{ENC} und K_{MAC} aus CA			

Tabelle 2.9.: Ablauf der Chipauthentisierung

2.10. SAML

SAML (Security Assertion Markup Language) ist nach OASIS[9] „ein XML-basiertes Framework zur Beschreibung und zum Austausch von Sicherheitsinformationen zwischen Online-Geschäftspartnern.“ Wie aus dem Namen zu erkennen ist, handelt es sich bei SAML um eine auf XML basierende Auszeichnungssprache und beschreibt den Austausch von Authentifizierungs- und Autorisierungsinformationen über Organisationsgrenzen hinweg.

Der Haupteinsatzzweck von SAML sollen *Web Single Sign-On (SSO)* und Autorisierungsdienste werden, wobei der Haupteinsatzzweck vor allem bei den Webservices liegt. Weiterhin sollen mit SAML verteilte Transaktionen ermöglicht werden, damit Benutzer unter einer Transaktion gemeinsam arbeiten können und sich damit Sicherheitsinformationen teilen.

SAML sieht es vor, dass ein *Identity Provider (IdP)* den Benutzer authentifiziert und dem *Service Provider (SP)* die notwendigen Informationen über die Identität des Subjekts zur Verfügung stellt. In SAML wird der Identity Provider auch als *Asserting Party (AP)* bezeichnet, da er dem Service Provider die Identität des Subjekts bescheinigt. Der Service Provider übernimmt dadurch die Rolle der *Relying Party (RP)* und muss der Asserting Party vertrauen. Die Bescheinigung der Identität wird durch eine SAML Assertion übermittelt.

2.10.1. Assertions

Eine SAML Assertion (dt. Zusicherung) ist eine garantierte Zusicherung vom Identity Provider gegenüber dem Service Provider, dass das Subjekt authentifiziert wurde. In der Assertion ist ebenfalls enthalten, auf welche Art und Weise die Authentifizierung erfolgte.

Weiterhin können in die Assertion Eigenschaften über das Subjekt hinzugefügt werden, die an den Service Provider übermittelt werden sollen. Diese zusätzlichen Informationen werden Statement-Typen genannt und davon gibt es drei Stück:

- Authentication (Authentisierung): Sagt aus, dass eine Authentisierung durchgeführt wurde und enthält Informationen, wie sie stattgefunden hat.

- Attribute: Enthält eine beliebige Anzahl von Attributen mit Eigenschaften des Subjekts.

[9] Organization for the Advancment of Structured Information Standards

- Authorization Decission (Authorisierungsentscheidung): Beschreibt die Rechte des Subjekts. Sie dienen als Entscheidungsgrundlage über Zugriffsberechtigungen auf den Dienst.

2.10.2. Protocols

SAML in Version 2.0 definiert sechs Request-Response-Protokolle für die Kommunikation:

- Assertion Query and Request: Eine Relying Party fordert bei einer Asserting Party eine oder mehrere Assertions an oder eine bestimmte Assertion über eine Assertion-ID.

- Authentication Request: Ein Subjekt möchte sich bei einer Relying Party authentisieren. Dafür schickt die Relying Party einen Authentication Request an die Asserting Party. Die Bestätigung wird von der Relying Party an das Subjekt übermittelt.

- Artifact Resolution: Teile der SAML Nachricht sind nicht in der Nachricht selber enthalten, sondern nur referenziert und müssen von der Asserting Party abgerufen werden. Dafür stellt die Relying Party eine Anfrage an die Assertion Party mit den entsprechenden Referenzen (Artifacts), um die SAML-Nachrichten zu erhalten.

- Name Identifier Management: Der Benutzer oder Pseudonym eines Subjekts kann sich ändern. Dieses Protokoll erlaubt es, diese Information zwischen der Asserting Party und der Relying Party auszutauschen.

- Name Identifier Mapping: Eine Asserting Party kann für mehrere Relying Parties authentisieren. Mit diesem Protokoll wird das Mapping zwischen unterschiedlichen Relying Parties durchgeführt.

- Single Logout: Dieses Protokoll soll ein fast simultanes Abmelden eines Subjektes bei mehreren Diensten ermöglichen.

Bindings

Bindings beschreiben den Transport von SAML-Nachrichten über Standard-Protokolle wie SOAP und HTTP:

- SAML SOAP

- Reverse SOAP (PAOS)
- HTTP Redirect
- HTTP Poast
- HTTP Artifact
- SAML URI

2.10.3. Profiles

SAML in Version 2.0 definiert 13 Profile, die für bestimmte Aufgaben gedacht sind. Dabei handelt es sich um Art Standardregeln mit festgelegten Attributen.

Die SAML 2.0 Profile sind bis auf drei Stück in zwei zusätzliche Gruppen eingeteilt.

Allgemeine SAML 2.0 Profile:

- Artifact Resolultion Profile
- Assertion Query / Request Profile
- Name Identifizier Mapping Profile

Die Gruppe *Attribute Profiles* in den SAML 2.0 Profilen:

- UUILD Attribute Profile
- X.500 / LDAP Attribute Profile
- DCE PAC Attribute Profile
- XACML Attribute Profile
- Basic Attribute Profile

Die Gruppe *Single Sign-On Profiles* in den SAML 2.0 Profilen:

- Identity Provider Discovery Profile
- Enhanced client or Proxy (ECP) Profile

- Name Identifier Management Profile
- Web Browser SSO Profile
- Single Logout Profile

2.11. TLS

Transport Layer Security (TLS) ist der Nachfolger von SSL (Secure Socket Layer) und übernimmt bis auf einige wenige Änderungen die Spezifikation von SSL 3.0. Da SSL im Kontext des Personalausweises nicht verwendet wird, erfolgt an dieser Stelle die direkte Beschreibung von TLS. Die Änderungen gegenüber SSL werden an den entsprechenden Stellen erwähnt.

Die Hauptaufgabe von SSL und damit auch TLS ist nach Eckert [Eck09] die „Authentifikation der Kommunikationspartner unter Verwendung von asymmetrischen Verschlüsselungsverfahren und Zertifikaten, die vertrauliche Ende-zu-Ende-Datenübertragung unter Nutzung eines gemeinsamen Sitzungsschlüssels und schließlich auch die Sicherstellung der Integrität der transportierten Nachrichtung unter Nutzung von MAC“.

In TLS sind die verwendeten kryptographischen Protokolle nicht fest vorgegeben. Sie werden zwischen Server und Client beim Verbindungsaufbau ausgehandelt, in dem die Parteien ihre CipherSuites austauschen. Die CipherSuite enthält unterstützte Kombinationen von kryptographischen Verfahren für Verschlüsselung und Nachrichtenauthenifizierung. Für die asymmetrische Verschlüsselung und den Schlüsselaustausch stehen RSA und Diffie-Hellman zur Auswahl. SSL unterstützt zusätzlich den proprietären Fortezza Algorithmus, der in TLS ausgelassen wurde.

Die Nachrichtenauthentifizierung und Verschlüsselung von Nachrichten erfolgt durch eine Kombinationen von MAC und symmetrischen Verschlüsselungsfunktionen, wobei sie nicht frei miteinander kombinierbar sind. Für eine erfolgreiche Kommunikation müssen sich beide Parteien auf ein gemeinsames Verfahren einigen.

Die gemeinsamen geheimen Schlüssel werden nicht über ein Hybridverfahren aus asymmetrischen und symmetrischen Verfahren ausgehandelt, da TLS möglichst wenig geheime Informationen über die unsichere Leitung übertragen soll. Aus diesem Grund wird nur eine Basisinformation zwischen dem Client und dem Server vereinbart, aus welcher der gemeinsame Verschlüsselungsschlüssel und die MAC-Schlüssel berechnet werden.

2.11.1. TLS-Schichten

TLS setzt auf der Transportschicht des OSI-Schichtenmodells auf und benutzt TCP/IP als Grundlage für die Kommunikation. TLS besteht selber aus der *Recordschicht* und der *Handshake-Schicht*, auf denen Anwendungsprotokolle aufsetzen.

Recordschicht

Die Recordschicht übernimmt die Fragmentierung der Daten von der Anwendungsschicht, in dem es die Daten in TLS-Records unterteilt. Sie komprimiert die Daten vor der Verschlüsselung und berechnet den MAC für die TLS-Records. Die Berechnung des gemeinsamen Sitzungsschlüssels und die Verschlüsselung der TLS-Records ist ebenfalls Aufgabe der Recordschicht. Danach werden die Daten über TCP/IP an den Empfänger übermittelt.

Da das Protokoll auf TCP/IP aufsetzt, ist der Schutz der TCP- und IP-Header nicht Bestandteil von TLS. Lediglich die übermittelten Daten können authentifiziert und entschlüsselt werden.

Handshake-Schicht

Die Handshake-Schicht übernimmt die Authentifikation der Kommunikationspartner, sowie das Aushandeln der zu verwendeten kryptographischen Verfahren und den Austausch der notwendigen Geheimnisse.

2.11.2. Reserverierte Ports

TLS wird vor allem zur Authentifizierung und Verschlüsselung bereits bestehender und auf der Anwendungsschicht aufsetzender Protokolle verwendet. Damit das Anwendungsprotokoll für TLS nicht modifiziert werden muss, sind für die Protokolle Netzwerkports ausschließlich für die TLS-Kommunikation reserviert.

2.11.3. Alert Protokoll

TLS besitzt mit dem Alert Protokoll ein weiteres Protokoll, welches für die Übermittlung TLS-spezifischer Warnungen dient. Die Nachrichten bestehen aus zwei Teilen, welche jeweils ein Byte lang sind.

Das erste Byte zeigt, wie schwerwiegend die Warnung ist. Eine Eins bedeutet, dass es sich um eine Warnung handelt und eine Zwei bedeutet, dass es sich um einen fatalen Fehler handelt. Bei allen fatalen Fehlern wird die Kommunikation abgebrochen. Das zweite Byte enthält den Grund der Warnung.

In TLS sind zusätzliche fatale Fehler enthalten, die in SSL nicht vorhanden waren.

2.11.4. ChangeCipherSpec

Die ChangeCipherSpec setzt auf dem TLS-Record Protokoll auf und umfasst lediglich eine Nachricht. Bei Übermittlung dieser Nachricht übernimmt der Kommunikationspartner die ausgehandelten Sitzungsinformationen als aktuell zu verwendendes Verfahren.

2.11.5. Handshake-Protokoll

TLS etabliert eine Sitzung zwischen dem Server und den Client, was es zu einem zustandsbehafteten Protokoll macht. Der Client nutzt diesen Zustand, um mehrere Sitzungen zu einem Server aufzubauen. Die Handshake-Schicht verwaltet die Sitzungen und erlaubt auch parallele Sitzungen zu anderen Server zu betreiben.

ClientHello

Nach dem Aufbau der TCP-Verbindung schickt der Client zuerst eine ClientHello-Nachricht an den Server bestehend aus der Datenstruktur R_c. Diese Datenstruktur wird aus einem 32-Bit Zeitstempel, einer 28-Bit Zufallszahl, einem Sitzungsidentifikator und einer CipherSuite zusammengesetzt. Durch die Zufallszahl sollen Wiedereinspielungsattacken verhindert werden.

ServerHello

Der Server antwortet auf das ClientHello mit ServerHello und übermittelt, wie auch der Client eine Datenstruktur R_s, bestehend aus einem 32-Bit Zeitstempel, einer 28-Bit Zufallszahl und einer an den Client angepassten CipherSuite. Die CipherSuite enthält nur Verfahren, die der Server und der Client gemeinsam unterstützen. Die Verfahren werden vom Server über Ihre Sicherheitseigenschaften nach Priorität sortiert. Der Server sucht sich aus dieser Liste ein Verfahren mit möglichst hoher Priorität raus.

Um Sitzungen fortsetzen zu können, hält sich der Server einen Sitzungs-Cache mit Sitzungsidentifikatoren der Clients vor. Über den Sitzungsidentifikator kann ein Client auch mehrere Verbindungen zum Server aufbauen.

Soll der Server authentifiziert werden, so übermittelt er dem Client mit dem ServerHello sein Zertifikat. Ist die Serverauthentifizierung nicht erforderlich, so übermittelt er mit dem ServerKeyExchange einen temporären öffentlichen Schlüssel.

Die CertificateRequest-Nachricht fordert den Client auf, sich über ein Zertifikat zu verifizieren.

Mit ServerHelloDone wird die Nachricht abgeschlossen.

Client-Antwort auf ServerHello

Soll der Client ebenfalls authentifiziert werden, so übermittelt er dem Server sein eigenes Zertifikat und eine CertificateVerify-Nachricht. Wird keine Authentifizierung benötigt, so übermittelt der Client mit der ClientKeyExchange-Nachricht einen temporären öffentlichen Schlüssel.

Damit der Empfänger das Zertifikat des Clients verifizieren kann, sendet der Client dem Server eine CertificateVerify-Nachricht. Diese besteht aus einem MAC über alle bislang im Protokoll ausgetauschten Nachrichten.

Mit der ChangeCipherSpec-Nachricht teilt der Client dem Server mit, dass die im Handshake ausgehandelten Verfahren jetzt angewendet werden können. Mit der Finished-Nachricht wird das Handshake-Protokoll auf der Client-Seite beendet.

Antwort des Servers

Der Server bestätigt dem Client mit der ChangeCipherSpec-Nachricht die ausgehandelten Verfahren und beendet mit der Finished-Nachricht das Handshake-Protokoll. Ab jetzt können die Daten über die gesicherte Verbindung übertragen werden.

2.11.6. Schlüsselaushandlung

Bei SSL wird das Geheimnis nicht direkt über die Leitung übertragen, sondern mit Hilfe des Pre-Master Secrets berechnet. Aus dem Pre-Master Secret wird das Master Secret berechnet.

Pre-Master Secret

Das Pre-Master Secret wird mit der ClientExchange-Naricht als geheime Basisinformation (48 Byte lang) an den Server übermittelt. Bei RSA wird das Geheimnis mit dem öffentlichen Schlüssel des Server verschlüsselt und an den Server geschickt.

Wird das DH-Verfahren verwendet, so wird an dieser Stelle nur der öffentliche Schlüssel des Clients übermittelt, da DH nur zur dezentralen Berechnung des Pre-Master Secrets dient.

Master Secret

Aus dem Pre-Master Secret und den Zufallszahlen aus R_s imd R_c wird das 48-Byte Master Secret berechnet. Aus dem Master Secret werden die geheimen Schlüssel für Verschlüsselung und Authentifizierung der Nachrichten abgeleitet.

Teil III.

Sicherheitsanalyse

KAPITEL 3

METHODEN DER SICHERHEITSANALYSE

In diesem Kapitel sollen Methoden der Sicherheitsanalyse evaluiert werden. Der Schwerpunkt der Untersuchung liegt auf dem Authentisierungsprozess an einem Dienstanbieter mit dem nPA des Bürgers. An dem Prozess sind mehrere Parteien beteiligt, die über Netzwerke kommunizieren und diverse Softwarekomponenten verwenden. Der eID-Client spielt hier eine wichtige Rolle, da er als zentrale Komponenten zwischen dem nPA, eID-Service und dem Dienst agiert. Seine weitere Aufgabe ist die Visualisierung des vollständigen Prozesseses für den Bürger.

Aus der Analyse sollen Sicherheitsanforderungen für den Prozess und die beeinflussbaren Softwarekomponenten erarbeitet werden. Dafür werden in diesem Kapitel vier Normen und Standards evaluiert: ISO 27000-Normenfamilien, NIST Handbook, SQUARE und BSI IT-Grundschutz-Standards.

Als Ergebnis der Evaluierung soll aus den Normen und Standards eine Methode für die Sicherheitsanalyse ausgewählt werden, die der Aufgabenstellung möglichst nahe kommt.

3.1. Einheitliche Begriffe für Beschreibungen

In den verschiedenen Normen und Standards werden unterschiedliche Begriffe für gleiche Sachverhalte verwendet. Besonders die älteren Normen verwenden nicht mehr aktuelle Begriffe. Durch eine einheitliche Benennung soll der Vergleich zwischen den Normen und Standards verbessert werden. Die spezifischen Begriffe der Normen und Standards werden an den jeweiligen Stellen explizit erwähnt.

Die aktuellsten und die in Deutschland bekanntesten sind die ISO 27000-Normenfamilien und die IT-Grundschutz-Standards vom BSI. Beide sind auf Deutsch verfügbar, wobei das BSI viele Begriffe aus der ISO 27001 Norm [ISO05a] übernommen hat. Aus diesem Grund werden überwiegend Begriffe aus der ISO 27000-Normenfamilie verwendet.

Leitungsebene: Nach BSI hat jede Organisation ein Management, welches mit *Leitungsebene* bezeichnet wird, „wenn die verantwortlichen Führungskräfte gemeint sind und Verwechslungsgefahr zum Management als *Leitungsprozess* (Leiten, Lenken und Planen) besteht (Quelle: BSI 100-1 [BSI08a])“.

Management: Unter Management versteht man nach BSI den „*Leitungsprozess*“. Damit soll eine Abgrenzung vom englischen Begriff Management erfolgen, der für die *Leitungsebene* einer Organisation steht.

Informationssicherheit: Informationssicherheit nach ISO 27001 ist „Aufrechterhaltung der Vertraulichkeit, Integrität und Verfügbarkeit von Informationen; andere Eigenschaften wie Authentizität, Zurechenbarkeit, Nicht-Abstreitbarkeit und Verlässlichkeit können ebenfalls berücksichtigt werden“ (Quelle: ISO/IEC 27001 [ISO05a]). Dabei bezieht sich die Informationssicherheit auf alle Informationen „unabhängig von der gewählten Form, dem Medium der Weitergabe oder Speicherung“ (Quelle: ISO/IEC 27001 [ISO05a]).

In älteren Dokumenten findet der Begriff *Computersicherheit* oft Verwendung, da früher Computersysteme mit wichtigen Informationen im Fokus der Analysen standen. Dieser Begriff wurde später zur Informationssicherheit verallgemeinert.

ISMS: Ein Informationssicherheit-Managementsystem (ISMS) ist nach BSI „der Teil des Managementsystems, der sich mit Informationssicherheit beschäftigt“ (Quelle: BSI 100-1 [BSI08a]). Die ISO 27000-Normenfamilie hat eine speziellere Definition und bezieht sich direkt auf Geschäftsrisiken einer Organisation, was nicht bei allen Normen der Fall ist.

Richtlinien: Nach der ISO 27001 Norm ist eine *Richtlinie* eine „Beschreibung, die klarstellt, was wie getan werden soll, um die in der Leitlinie formulierten Ziele zu erreichen“ (Quelle: ISO/IEC 27001 [ISO05a]). Oft findet auch der englische Begriff *Policies* Verwendung.

Prozess: Nach ISO 27002 [ISO05b] sind Prozesse „alle Aktivitäten mit genutzten Ressourcen, die eine Eingabe in eine Ausgabe überführen“. Damit werden auch Tätigkeiten und Vorgänge innerhalb einer Organisation beschrieben.

3.2. Methoden der Sicherheitsanalyse

Dieser Abschnitt stellt Methoden für Sicherheitsanalyse vor. Der Fokus liegt vor allem auf dem Einsatzzweck und dem Vorgehen der Methode. Auf Basis dieser Information soll eine Evaluierung im Bezug auf die gestellte Aufgabe ermöglicht werden.

3.2.1. ISO 27000-Normenfamilie

Die *ISO 27000-Normenfamilie* behandelt ein ISMS. Sie teilt sich auf mehrere Normen auf. An dieser Stelle wird die *ISO 27001* Norm als Einführung in die ISO 27000-Normenfamilie und die *ISO 27002* als Leitfaden für die Einführung eines ISMS in einer Organisation betrachtet.

ISO 27001

Die ISO 27001 beschreibt ein Modell für die Einführung (establishment), Implementierung (implementing), Betrieb (operating), Überwachung (monitoring), Überprüfung (reviewing), Aufrechterhaltung (maintaning) und Verbesserung (improving) eines ISMS.

Nach ISO 27001 muss die Einführung eines ISMS eine strategische Entscheidung der Organisation sein. Der Entwurf und die Implementierung eines ISMS sollte nach Bedarf, Zielvereinbarung, Sicherheitsanforderungen, Prozessen, Größe und Struktur des Unternehmens gerichtet werden.

Die Einführung des ISMS ist kein einmaliger Prozess, da sich die darauf aufbauenden Systeme mit der Zeit ändern werden. Aus diesem Grund sollte das ISMS so ausgelegt werden, dass es mit der Organisationsgröße skaliert und sich an die Bedürfnisse der Organisation anpasst.

Der Fokus dieser Norm liegt auf Aktivitäten und darauf aufbauenden Prozessen. Eine Organisation besitzt viele Aktivitäten, die identifiziert und gesteuert werden müssen, damit sie effizient funktionieren. Nach ISO 27001 werden „alle Aktivitäten mit genutzten Ressourcen, die eine Eingabe in eine Ausgabe überführen, werden als Prozesse bezeichnet“ (Quelle: ISO/IEC 27001 [ISO05a]).

Die ISO 27000-Normenfamilie verfolgt damit den so genannten Prozessansatz, welcher aus einem System von einzelnen Prozessen besteht. Für das Informationssicherheit-Management ist beim Prozessansatz wichtig, dass das Verständnis für die Anforderungen der Informationssicherheit existieren. Dafür ist die Einführung von Richtlinien und Zielen für die Informationssicherheit innerhalb von Prozessen von großer Bedeutung.

Die Ausrichtung der Informationssicherheit sowie deren Implementierung und Kontrollen sollten sich an die Geschäftsrisiken der Organisation ausrichten. Weiterhin soll eine ständige Überwachung und Überprüfung der Leistung und der Effektivität des ISMS erfolgen. Basierend auf objektiven Messungen des ISMS soll eine kontinuierliche Verbesserung der Informationssicherheit erreicht werden.

Dabei spielt das „Plan-Do-Check-Act“ PDCA-Modell von Dr. Demming (siehe Abbildung 3.1) eine wichtige Rolle, da alle ISMS Prozesse in der ISO 27000-Normenfamilie danach strukturiert sind.

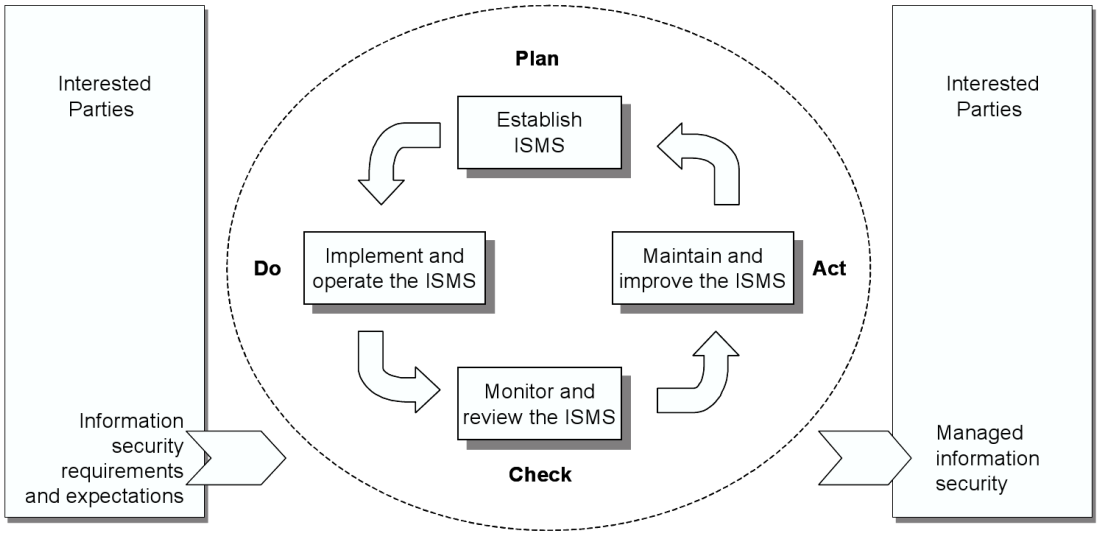

Abbildung 3.1.: PDCA Modell (Quelle: ISO 27001 [ISO05a])

Die Norm beschreibt das PDCA-Modell als Zyklus für die Einführung und Betrieb eines ISMS. Als Eingabe für den Zyklus gelten Anforderungen und Erwartungen an die Informationssicherheit, die von der Leitungsebene vorgegeben werden. Diese Anforderungen und Erwartungen werden in der ersten Phase geplant (plan) und für die Organisation als verpflichtend eingeführt (establish). Die zweite Phase der Umsetzung (do) setzt die Planung um und führt den operativen Betrieb des ISMS ein. Um die Qualität der Umsetzung zu überprüfen (check), wird in der dritten Phase das ISMS überwacht (monitor). Die Ergebnisse der Überwachung ergeben Korrekturen und Verbesserungsvorschläge für das ISMS. Diese werden in der vierten Phase umgesetzt, was als Wartung (maintain) und Verbesserung (improve) bezeichnet wird. Die Informationen aus dem Zyklus fließen der interessierten Partei zu, welche neue Anforderungen und Erwartungen für den nächsten Zyklus definiert.

ISO 27002

Die ISO 27002 Norm ist ein Leitfaden für die Implementierung eines ISMS. Sie baut auf den Grundlagen der ISO 27001 Norm auf. Die Ausrichtung der beiden Normen gelten für Informationen in allen Formen, unabhängig von Medien oder Art der Speicherung. Es macht kein Unterschied, ob es sich beim Medium um Papier, digitale Dokumente, Gespräche oder Videoaufnahmen handelt, da die Informationen in allen Formen geschützt werden sollen.

Bevor das ISMS umgesetzt wird, muss vorher eine Risikoabschätzung für die Prozesse durchgeführt werden. Erst danach folgt die Umsetzung für die als geschäftskritisch eingestuften Prozesse mit Hilfe des PDCA Modells.

Die Norm ist generisch aufgebaut und bezieht sich auf keine direkte Technik oder gar Implementierung. Sie behandelt generelle Themen der Informationssicherheit. Das ist auch an den behandelten Themen zu sehen:

- Sicherheitsleitlinie
- Organisation der Informationssicherheit
- Management von organisationseigenen Werten
- Sicherheit im Personalbereich
- physische und umgebungsbezogene Sicherheit
- Management der Kommunikation und des Betriebs
- Zugangskontrolle
- Beschaffung, Entwicklung und Wartung von Informationssystemen
- Management von Informationssicherheitsvorfällen
- Aufrechterhaltung des Geschäftsbetriebs
- Einhaltung von Verpflichtungen

Jedes dieser Themen beinhalten durchzuführende organisatorische *Maßnahmen* und als Ergänzung *weitere Informationen*. Die Maßnahmen beinhalten eine *Anleitung zur Umsetzung*. Die *weiteren Informationen* bestehen aus Verweisen auf weitere Dokumente wie Normen, die das Thema betreffen.

Dadurch entsteht ein Maßnahmenkatalog mit generischen Beschreibungen, ausgerichtet auf die Leitungsebene einer Organisation. Er beinhaltet keine Verweise auf eine bestimmte Technik oder gar Implementierung.

3.2.2. NIST Handbook

Das *NIST Handbook* ist vom *National Institute of Standards and Technology (NIST)* im Auftrag des *U.S. Department of Commerce* entwickelt worden. Es wurde vor allem als Unterstützung für die Absicherung informationsbasierter (computerbasierter) Ressourcen entwickelt. Die aktuelle Version ist aus dem Jahre 1995 und enthält viele nicht mehr aktuelle Begriffe. Diese sind jeweils in Klammern hinter den aktuellen Begriffen angegeben.

Der Fokus des NIST Handbooks liegt auf Ressourcen wie Hardware, Software und Informationen. Diese werden im Bezug auf deren Absicherung, Kosten und Sicherheitskontrollen betrachtet. Durch die Vorstellung wichtiger Techniken und Ansätze sollen Vorteile von Sicherheitskontrollen gezeigt werden. Die Sicherheitskontrollen beschreiben damit ein ISMS, in dem die Umsetzung der Richtlinien kontrolliert wird.

Das NIST Handbook wurde für Personen mit Verantwortung in Informationssicherheit entwickelt, um Ihnen Grundkonzepte und Technik der Informationssicherheit zu vermitteln. Es richtet sich an Organisationen in den Vereinigten Staaten, die sensible Informationssysteme (Computersysteme) im Bezug auf Sicherheit betreiben.

Es soll vor allem als eine Übersicht über die Informationssicherheit (Computersicherheit) für Verantwortliche dienen und den Anwender bei der Auswahl der richtigen Kontrolltechniken unterstützen. In der Einführung des NIST Handbooks wird darauf hingewiesen, dass es keine Methoden für Vorgehen enthält. Auch beschreibt es nicht, wie ein Sicherheitssystem in der Organisation implementiert werden soll. Es ist mehr als Verständnishilfe für Verantwortliche gedacht. Am Ende der entsprechenden Kapitel folgen Hinweise und Referenzen zu speziellen Techniken, die angewendet werden können. Im NIST Handbook wird ganz klar darauf hingewiesen, dass es nicht für Sicherheitsanalysen geeignet ist.

Das NIST Handbook ist auf drei große Bereiche aufgeteilt, die im Folgenden erläutert werden.

Managementkontrollen

Zuerst werden Kontrollen aus Sicht des Leitungsebene der Organisation betrachtet. Daraus soll ein Informationssicherheitsprogramm (Computersicherheitsprogramm) entwickelt werden, welches durch die Leitungsebene eingeführt und deren Durchsetzung kontrolliert werden soll.

Operationelle Kontrollen

Die operationelle Kontrolle baut auf dem Informationssicherheitsprogramm der Leitungsebene auf. Auf dieser Ebene werden Sicherheitskontrollen durch Menschen implementiert und ausgeführt, was zur Verbesserung der Informationssicherheit innerhalb der Organisation führen soll. Für diese Aufgabe wird technisches und IT-Sicherheit spezifisches Wissen gefordert. Zusätzlich müssen Management- und Wartungsaktivitäten sowie technische Kontrollen für die Maßnahmen eingeführt werden.

Technische Kontrollen

Technische Kontrollen sind durch Menschen implementierte Kontrollen, die automatisch ausgeführt werden. Darunter fallen beispielsweise Zugangskontrollen zu Informationssystemen. Die Kontrollen hängen direkt von der korrekten Funktionalität und Effektivität der Implementierung ab.

Die Einführung technischer Kontrollen muss sich an das Informationssicherheitsprogramm der Leitungsebene anlehnen und erfordert operationelle Eingriffe in Informationssysteme. Die Eingriffe und Änderungen müssen mit der Leitungsebene der Organisation abgestimmt werden.

3.2.3. SQUARE

Security Quality Requirements Engineering (SQUARE) [NRM05] wurde vom *Carnegie Mellon Software Engineering Institute* entwickelt und beschreibt eine Methode für eine systematische Analyse von Softwarekomponenten. Der Fokus liegt auf Sicherheit und Qualität beim Anforderungsmanagement (Requirements Engineering) für die Softwareentwicklung. SQUARE sollte so früh wie möglich in den Entwicklungsprozess der Softwarekomponenten integriert werden, um die besten Ergebnisse zu liefern. Nach SQUARE stellen Sicherheitsanforderungen nicht-funktionale Anforderungen dar und richten sich explizit an die Qualität des Produktes.

Durch die Einstufung der Sicherheitsanforderungen als nicht-funktionale Anforderungen müssen die Projektverantwortlichen (Stakeholder) in den Prozess eingebunden werden. Die Sicherheitsanforderungen werden nach SQUARE vom Anforderungsmanagement erarbeitet und mit den Projektverantwortlichen abgestimmt. Dadurch sollen aus Sicht der Projektverantwortlichen nur notwendige Sicherheitsanforderungen als Anforderung aufgenommen werden.

Das Vorgehen nach SQUARE beschreibt in neun Stufen ein systematisches Vorgehen für die Erhebung, Kategorisierung und Priorisierung der gestellten Sicherheitsanforderungen an die Softwarekomponenten. Es soll bereits beim Anforderungsmanagement für die Softwarekomponenten zum Einsatz kommen und gezielt Verbesserungen für das zukünftige System während der Softwareentwicklung bieten.

Vorgehen

Das Vorgehen nach SQUARE ist in neun Stufen aufgeteilt. Jede dieser Stufen decken erforderliche Eingaben, eingesetzte Technik, Beteiligte und die daraus resultierenden Ausgaben ab. Im Folgenden werden die einzelnen Stufen nach SQUARE beschrieben.

In der ersten Stufe sollen passende Definitionen und Begriffe für alle beteiligten Parteien erarbeitet werden. Gängige Definitionen können von der IEEE[1] oder aus anderen Standards entnommen werden. Diese sollten durch Interviews in der Gruppe der Projektverantwortlichen zusammen mit den Anforderungsingenieuren erarbeitet werden. Darin sollen auch Begriffe aus dem Geschäftsbereich nicht fehlen. Die vordefinierten Begriffe bilden eine gemeinsame Sprache, die das Verständnis zwischen den Parteien fördert.

In der zweiten Stufe werden Sicherheitsziele für das Projekt identifiziert und priorisiert. Als Eingabe für diesen Schritt dienen die vorab definierten Begriffe, mögliche Sicherheitsziele, Geschäftsziele, Richtlinien und Prozeduren für Vorgehen in einer Institution. Zusätzlich können passende Beispiele eingebracht werden. Die Sicherheitsziele sollen in Arbeitssessions, Anwenderbefragungen und Interviews erarbeitet werden. Im Anschluss werden sie durch die Geschäftsleitung priorisiert, wobei hier die Projektverantwortlichen sowie das Anforderungsmanagement beteiligt sein müssen.

In der dritten Stufe sollen Artefakte gesammelt und entwickelt werden, die das System beschreiben. Dazu zählt die Beschreibung der Systemarchitektur, Use Cases, Szenarien für falsche Verwendung und Angriffsbäume. Diese Aufgaben können vom Anforderungsmanagement durchgeführt werden. Sind diese Informationen nicht vorhanden, so sollten sie nach Möglichkeit erstellt werden, da der Erfolg des Vorgehens maßgeblich davon abhängt. Die Szenarien sollten möglichst vollständig sein, da darauf alle nachfolgenden Stufen aufbauen.

[1] Institute of Electrical and Electronics Engineers

In der vierten Stufe soll eine Risikoanalyse über die Fälle für falsche Benutzung und Szenarien durchgeführt werden, indem sie im Bezug auf die vorab festgelegten Sicherheitsziele betrachtet werden. Dazu können diverse Methoden der Risikoanalyse und Bedrohungsanalysen verwendet werden. Über die verwendete Methode sollen für die Sicherheitsrisiken Wahrscheinlichkeiten bestimmt werden. Diese Aufgaben können durch das Anforderungsmanagement oder Experten für Sicherheitsanalysen durchgeführt werden. Die Ergebnisse sollen den Projektverantwortlichen vorgelegt werden.

In der fünften Stufe muss eine Erhebungstechnik für das Erarbeiten der Sicherheitsanforderungen ausgewählt werden. Diese hängt maßgeblich von Organisationsstil, Unternehmenskultur, Stufe der geforderten Sicherheit und den Kosten ab. Für das Vorgehen wird Expertenwissen der Institution benötigt, um aus den Sicherheitszielen geschäftsspezifische Sicherheitsanforderungen zu erarbeiten.

In der sechsten Stufe sollen mit Hilfe der im vorhergehenden Schritt festgelegten Methode Sicherheitsanforderungen erhoben werden. Die Sicherheitsanforderungen müssen mit den vorab ausgewählten Experten mit geschäftsspezifischen Wissen erarbeitet und anschließend mit der Geschäftsleitung abgestimmt werden. Als Ergebnis dieses Schrittes sollen initiale Sicherheitsanforderungen erstellt werden.

In der siebten Stufe müssen die Sicherheitsanforderungen nach Stufen kategorisiert werden. Minimale Kategorien für solches Vorgehen wären Systemebene, Softwareebene oder Beschränkungen durch die Architektur und das Festlegen, ob diese Sicherheitsanforderung wichtig ist. Zusätzlich sollte festgestellt werden, ob die Anforderungen noch weitere Abhängigkeiten besitzen. Diese Aufgabe kann vom Anforderungsmanagement durchgeführt werden.

In der achten Stufe soll mit Hilfe der kategorisierten Anforderungen und den Ergebnissen der Risikoanalyse eine Priorisierung der Anforderungen durchgeführt werden. Die Priorisierung bestimmt die Reihenfolge für der Implementierung der Sicherheitsanforderungen, da meistens nicht genug Ressourcen für die Implementierung aller Sicherheitsanforderungen vorhanden sind. Das muss mit der Geschäftsleitung und den Projektverantwortlichen abgestimmt werden.

In der neunten Stufe sollen die priorisierten Sicherheitsanforderungen durch Dritte begutachtet werden. Diese kann durch Mitarbeiter der Organisation oder Personen mit projektspezifischen Wissen durchgeführt werden. Das Ziel dieser Begutachtung ist es Fehler, Unklarheiten, Inkonsistenzen oder falsche Annahmen zu finden und damit die Qualität der Arbeit zu prüfen und bei Bedarf zu korrigieren.

Wie an den Stufen zu erkennen ist, gibt SQUARE nicht für jede der Stufen ein genaues Vorgehen oder eine konkrete Methode vor. Bei der Anwendung muss für einige Stufen erst eine Methode ausgewählt werden. SQUARE gibt Unterstützung durch eine Beschreibung der einzelnen Stufen und schlägt konkrete Methoden für die Durchführung vor.

3.2.4. BSI IT-Grundschutz-Standards

Die *IT-Grundschutz-Standards* wurden vom deutschen *Bundesamt für Sicherheit in der Informationstechnik (BSI)* entwickelt. Darin sind „Methoden, Prozesse und Verfahren sowie Vorgehensweisen und Maßnahmen mit Bezug zur Informationssicherheit“ (Quelle: BSI 100-1 [BSI08a]) enthalten. Die BSI IT-Grundschutz Standards sind auf vier Dokumente unterteilt. Sie werden im Einzelnen erläutert und deren Einsatzzweck beschrieben.

Managementsysteme für Informationssicherheit (ISMS)

Das erste Dokument *Managementsysteme für Informationssicherheit (ISMS)* [BSI08a] beschreibt ein Managementsystem mit Anlehnung an die *ISO 27001* Norm und soll laut BSI kompatibel zur *ISO 27001* Norm sein.

Informationssicherheit spielt für Organisationen (Behörden und Unternehmen) eine immer wichtigere Rolle und bezieht sich auf Informationen in allen Formen, unabhängig vom Medium der Weitergabe oder Speicherung. Mit Hilfe von *IT-Grundschutz* soll durch systematisches Vorgehen ein Sicherheitsniveau für Informationen erreicht werden. Der Fokus von IT-Grundschutz liegt auf der Vorgehensweise zur Erarbeitung von Richtlinien und der systematischen Einführung der Informationssicherheit innerhalb einer Organisation. Erst zum Schluss sollen technische Maßnahmen für die Durchsetzung der organisatorischen Richtlinien angewendet werden.

Die Leitungsebene der Organisation trägt die Verantwortung für die Einführung und den Betrieb des ISMS, da Geschäftsprozesse von den Maßnahmen direkt beeinflusst werden können. Der wirtschaftliche Einsatz von Ressourcen muss auch für die Informationssicherheit garantiert werden.

Dieser Standard soll zeigen, was ein erfolgreiches Informationssicherheitsmanagement ausmacht und welche Aufgaben auf die Leitungsebene in Organisationen dabei zukommen. Die Ausrichtung des ISMS nach IT-Grundschutz ist für Verantwortliche der Informationssicherheit, Sicherheitsbeauftragte, Sicherheitsexperten und Sicherheitsberater gedacht.

Beim ISMS nach IT-Grundschutz handelt es sich um eine generische Beschreibung für ein ISMS. Die Einführung und das Vorgehen sind in der technischen Richtlinie *IT-Grundschutz Vorgehensweise* BSI 100-2 [BSI08b] genauer beschrieben.

IT-Grundschutz Vorgehensweise

Das zweite Dokument *IT-Grundschutz Vorgehensweise* [BSI08b] beschreibt eine Methode für das Management der Informationssicherheit. Sie kann nach Bedarf auf die Organisation angepasst werden.

Die IT-Grundschutz Vorgehensweise baut auf dem BSI-Standard 100-1 Managementsysteme für Informationssicherheit (ISMS) auf und stellt eine Vorgehensweise für die Einführung der Informationssicherheit in einer Institution dar. Das Ziel der Einführung ist es, ein angemessenes Sicherheitsniveau für Informationen zu erzielen und dieses Sicherheitsniveau aufrecht zu erhalten.

Während im BSI-Standard 100-1 die Sensibilisierung für die Einführung ein ISMS erfolgt, bietet die IT-Grundschutz Vorgehensweise Hilfestellungen für die Einführung eines ISMS. Eine Organisation kann es auch als Gerüst für die individuelle Umsetzung des ISMS ansehen, die durch Hinweise und Metriken die praktische Umsetzung unterstützt.

Zuerst folgt eine Übersicht über die wichtigsten Schritte für die Einführung eines ISMS, sowie die Vorgehensweise für die Erstellung eines Sicherheitskonzeptes.

Darauf aufbauend folgt eine Erklärung, wie die Phase der Initiierung des Informationssicherheitsprozesses aussehen kann. Diese Phase bedarf unter anderem Eingriffe in die Organisationsstrukturen der Organisation, um die Einführung des ISMS zu ermöglichen. Die IT-Grundschutz Vorgehensweise soll dabei helfen, in dem ein systematischer Weg für ein funktionierendes Sicherheitsmanagement aufgezeigt wird. Zusätzlich wird beschrieben, wie die Weiterentwicklung im laufenden Betrieb aussehen soll.

Nach der Initiierung folgt die eigentliche Durchführung der IT-Grundschutz Vorgehensweise (siehe auch Abbildung 3.2), die in einem Sicherheitskonzept enden soll. Dafür werden zuerst Grundinformationen über Komponenten mit Informationsbezug (nach BSI Informationsverbund) erhoben und anschließend in Gruppen eingeteilt. Die Einteilung soll zusammengehörige Komponenten gruppieren und dadurch die Komplexität reduzieren. Dann wird aus Sicht der Geschäftsprozesse der Schutzbedarf für Anwendungen, IT-Systeme, Kommunikationsverbindungen und Räume in Gebäuden erstellt. Für Gruppen mit hohem oder sehr hohem Schutzbedarf soll im Anschluss eine ergänzende Sicherheitsanalyse und eine Risikoanalyse nach BSI-Standard 100-3 [BSI08c] durchgeführt werden. Für alle anderen erfolgt ein Abbildung der Bausteine und Maßnahmen aus den IT-Grundschutz Katalogen auf die jeweiligen Komponenten des Informationsverbundes.

Wenn alle Sicherheitsanalysen durchgeführt sind, werden die Maßnahmen konsolidiert und in einen Umsetzungsplan überführt. Erst danach soll die eigentliche Umsetzung erfolgen.

Abschließend folgt noch eine Beschreibung für die Aufrechterhaltung des Sicherheitsniveaus durch das ISMS. Dieses ist für die gestellte Aufgabe nicht relevant und wird nicht weiter betrachtet.

Risikoanalyse auf der Basis von IT-Grundschutz

Das dritte Dokument *Risikoanalyse auf der Basis von IT-Grundschutz* [BSI08c] beschreibt eine Risikoanalyse mit Anlehnung an die beiden vorhergehenden Dokumente und soll nach erfolgreicher Umsetzung der *IT-Grundschutz Vorgehensweise* durchgeführt werden.

Die Risikoanalyse zielt nach BSI ausschließlich auf Gruppen des Informationsverbundes mit hohem oder sehr hohem Schutzbedarf und soll nach einer ergänzenden Sicherheitsanalyse angewendet werden.

Die ergänzende Sicherheitsanalyse soll Gefährdungen aufdecken, die nicht direkt in den Grundschutzkatalogen aufgeführt sind. Bei der Risikoanalyse kommen unter Umständen zusätzliche Gefährdungen dazu. Diese werden bewertet und Maßnahmen für die Behandlung der Risiken erstellt. Sie sollen in den Umsetzungsplan nach BSI-Standard 100-2 einfließen und am Ende umgesetzt werden (siehe auch Abbildung 3.2).

Notfallmanagement

Das vierte Dokument *Notfallmanagement* [BSI08d] behandelt Notfälle, die nicht durch direkte Maßnahmen abgedeckt werden können. Es soll nach BSI „Schäden durch Notfälle oder Krisen minimieren und die Existenz der Behörde oder des Unternehmens auch bei einem größeren Schadensereignis zu sichern."

Das Notfallmanagement liegt außerhalb des Kontextes dieser Arbeit und soll nicht weiter betrachtet werden.

3.3. Untersuchung der Aufgabe

Aus dem vorangegangenen Abschnitt haben sich bereits einige wichtige Anhaltspunkte herausgebildet, die zuerst der Aufgabenstellung gegenübergestellt werden müssen. Diese Punkte sollen später bei der Entscheidung für eine der Methoden helfen.

Der eID-Client soll vorwiegend im zivilen Bereich eingesetzt werden. Das bedeutet, vom Anwender wird kein Verständnis in IT-Sicherheit erwartet. Die Absicherung von IT-Systemen oder Softwarekomponenten bedarf zusätzliches spezialisiertes Wissen in IT-Sicherheit, welches in der Regel nicht gegeben ist.

Zudem sind die Benutzer keiner Organisation oder Leitungsebene unterstellt. Das bedeutet, ein Sicherheitsmanagement vergleichbar mit Organisationen kann nicht eingeführt werden, da keine Kontrollen für die Umsetzung der Maßnahmen möglich sind.

Eine Sicherheitsanalyse auf der Leitungsebene würde zwar organisatorische Richtlinien definieren, aber die Durchsetzung der Richtlinien kann nicht erfolgen, weil keine Kontrollen beim Endanwender möglich sind. Allerdings kann aus den Richtlinien ein Handbuch für den Betrieb des eID-Clients abgeleitet werden. Die Anwendung der Richtlinien aus dem Handbuch liegt in der Verantwortung des Benutzers und ist nicht verpflichtend durchsetzbar.

Die Richtlinien stellen die Grundlage für technische Maßnahmen und die Durchsetzung der Vorgaben dar. Aus den Richtlinien können Sicherheitsanforderungen abgeleitet werden. Diese Sicherheitsanforderungen können in die Softwareentwicklung einfließen, um die Sicherheit der Software sicherzustellen.

3.4. Auswahl einer Methode

Die ISO 27000-Normenfamilie (siehe Abschnitt 3.2.1) behandelt die Einführung und den Betrieb eines ISMS, wobei der Fokus auf dem Erarbeiten von organisatorischen Richtlinien liegt. Durch die fehlenden organisatorischen Strukturen im zivilen Bereich eignen sich die aus der ISO 27000-Normenfamilie erarbeiteten Richtlinien lediglich als Grundlage für das weitere Vorgehen. Es fehlt eine konkrete Methode, um technische Maßnahmen oder Sicherheitsanforderungen zu erarbeiten. Dadurch ist es für die gestellte Aufgabe nur zum Teil geeignet.

Das NIST Handbook (siehe Abschnitt 3.2.2) ist mehr als ein Handbuch der Informationssicherheit zu sehen, welches strategische, taktische und operationelle Kontrollen für die Informationssicherheit innerhalb einer Organisation beschreibt. Es ist für Sicherheitsanalysen ungeeignet, da es keine direkte Methode für ein Vorgehen vorgibt, sondern lediglich Verweise auf mögliche Methoden für die einzelnen Kontrollen enthält. Die Verweise erfordern weiteres Einarbeiten und ermöglichen kein einheitliches Vorgehen. Dadurch ist das NIST Handbook für eine Sicherheitsanalyse ungeeignet.

Mit SQUARE (siehe Abschnitt 3.2.3) sollen Sicherheitsanforderungen für die Softwareentwicklung erarbeitet werden. Es beschreibt ein generische Methoden und bietet mehrere Vorschläge für ein genaues Vorgehen in den einzelnen Schritten, was die Einarbeitung in weitere Methoden erfordert. Die vorgeschlagenen Methoden sind nicht für SQUARE geschrieben und decken nur Teile von SQUARE-Stufen ab oder greifen in andere Stufen rein. Dadurch müssen die ausgewählten Methoden angepasst werden, bevor sie für die Aufgabe verwendet werden können.

Die BSI IT-Grundschutz-Standards (siehe Abschnitt 3.2.4) stellen ein umfassendes Konzept für die Informationssicherheit dar, wobei vom ISMS bis zur technischen Implementierung alle Teile der Informationssicherheit abgedeckt werden. Dadurch entsteht eine in sich geschlossene und einheitliche Methode, die auch für Analysen von Softwarekomponenten geeignet ist. Das Ergebnis der Methode sind Sicherheitsanforderungen, die in die Softwareentwicklung aufgenommen werden können.

Besonders das einheitliche und systematische Vorgehen prädestiniert die *Vorgehensweise nach IT-Grundschutz* für diese Aufgabe. Lediglich die IT-Grundschutz-Kataloge müssen bei der Sicherheitsanalyse ausgelassen werden, da sie keine der zu untersuchenden Komponenten abdecken.

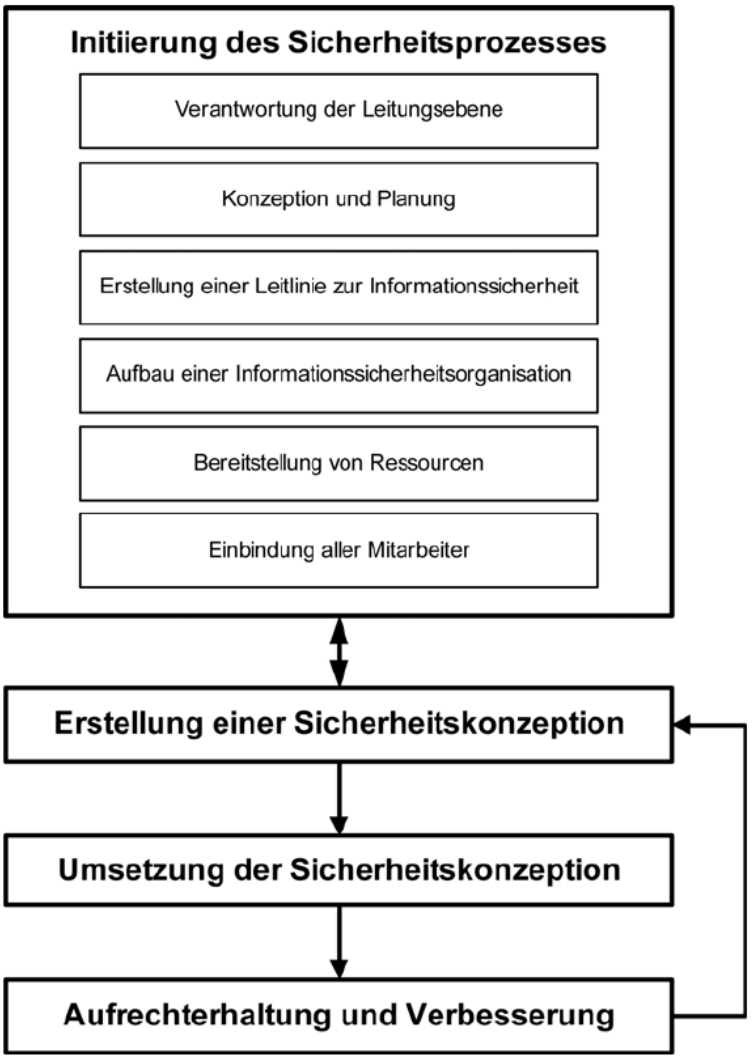

Abbildung 3.2.: Phasen des Sicherheitsprozesses (Quelle: BSI 100-2 [BSI08b])

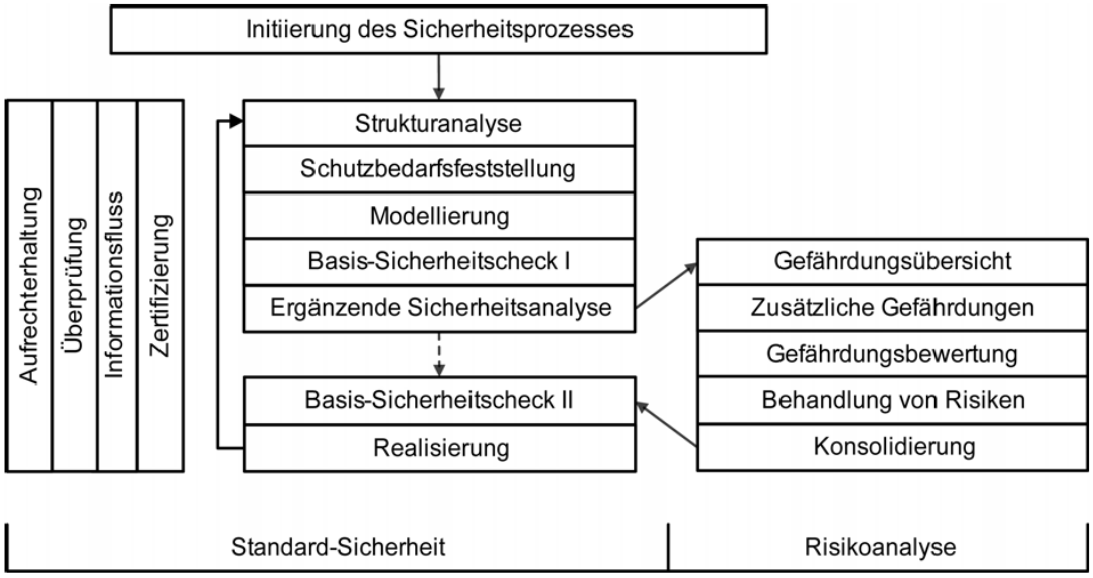

Abbildung 3.3.: Integration der Risikoanalyse in den Sicherheitsprozess (Quelle: BSI 100-3 [BSI08c])

KAPITEL 4

DURCHFÜHRUNG DER SICHERHEITSANALYSE

Die Sicherheitsanalyse des Authentisierungsprozesses mit dem nPA wird nach dem BSI IT-Grundschutz-Standard durchgeführt, welche im vorhergehenden Kapitel für diese Aufgabe ausgewählt wurde. Ziel der Sicherheitsanalyse ist es, Sicherheitsanforderungen für den Authentisierungsprozess und die Softwarekomponente eID-Client zu erarbeiten. Die Sicherheitsanforderungen sollen dann auf zwei konkrete Implementierungsvarianten für den eID-Client abgebildet werden.

4.1. Strukturanalyse

In der Strukturanalyse sollen die zu untersuchenden Prozesse, Informationen und Kommunikationswege für die Sicherheitsanalyse detailliert aufgeschlüsselt werden. Zuerst wird der Geltungsbereich definiert, indem die Strukturanalyse statt findet. Anschließend werden die sich im Geltungsbereich befindenden Informationen, Prozesse und Kommunikationswege im Detail untersucht.

4.1.1. Definition des Geltungsbereiches

Am Prozess sind drei Parteien beteiligt: Bürger, Dienstanbieter und eID-Service. Die Kommunikation zwischen den Parteien findet über das Internet statt. Der Bürger authentisiert sich ohne augenscheinliche Überprüfung mit Hilfe des nPAs beim Dienstanbieter. Der eID-Client ist die Softwarekomponente auf dem Computer des Bürgers, welche

die Kommunikation zwischen allen Parteien steuert. Die Analyse erfolgt aus Sicht des Bürgers.

Für die Analyse wird vorausgesetzt, dass die Komponenten beim legitimen Dienstanbieter und beim legitimen eID-Service Betreiber korrekt funktionieren. Auf die Sicherheit des Computers des Bürgers kann kein Einfluss genommen werden, da dieser keinen Sicherheitsrichtlinien unterliegt oder von einer Person mit Wissen in IT-Sicherheit betreut wird (siehe auch Abschnitt 3.3).

Die Strukturanalyse bezieht sich auf den eID-Client und die damit verbundenen Informationen, Prozesse und Kommunikationswege. Sie werden in den folgenden Abschnitten genauer beschrieben.

4.1.2. Datengruppen auf dem nPA

Auf dem nPA sind persönliche Daten in Form von Datengruppen (siehe Abschnitt 1.2.1) digital gespeichert. Nach der erfolgreichen Authentifizierung kann der eID-Service über die eID-Funktion des nPAs auf die Datengruppen zugreifen. Diese Datengruppen werden anschließend an den Dienstanbieter übermittelt.

4.1.3. Prozessbeschreibung: Authentisierung

Die Authentifizierung des Bürgers durch die eID-Anwendung (siehe Abschnitt 1.5.4) ist der Hauptprozess und soll im Folgenden genauer untersucht werden.

Nach BSI IT-Grundschutz-Vorgehensweise (siehe auch BSI-Standard 100-2 [BSI08b]) kann für Prozesse die Top-Down Vorgehensweise verwendet werden, indem im ersten Schritt die Prozessbeteiligten identifiziert werden und im Anschluss der Prozess untersucht wird. Aus dem Prozess lassen sich die Kommunikationswege ableiten.

In den folgenden Abschnitten wird die Top-Down Vorgehensweise auf den Authentisierungsprozess angewendet.

Prozessbeteiligte

Am Authentisierungsprozess sind drei Parteien sowie diverse Informationstechnik beteiligt:

- Der **nPA** ist der neue Personalausweis (siehe Abschnitt 1.2) des Bürgers, welcher als Besitz-Merkmal (siehe Abschnitt 2.2.2) in den Prozess eingeht.

- Das **lokale Terminal** (siehe auch BSI TR-03128 [BSI09d]) interagiert mit dem Bürger und kommuniziert mit dem *Remote Terminal* des eID-Service-Betreibers. Es besteht aus mehreren Komponenten, die im folgenden beschrieben werden:

  - Der **Computer** des Bürgers bietet die notwendige Software- und Hardware-Basis für die Durchführung der Authentifizierung.

  - Der **Chipkartenleser** stellt die Verbindung zwischen dem nPA und dem Remote Terminal her. Nach derzeitigen Stand existieren drei verschiedene Typen von Chipkartenlesern (siehe auch BSI TR-03119 [BSI09e]), welche für den Einsatz mit dem nPA zertifiziert sind:

    * Der **Basis-Chipkartenleser (CAT-B)** ist ein kontaktloser Chipkartenleser. Er besitzt kein eingebautes Pinpad, so dass die PIN-Eingabe über den Computer erfolgen muss.

    * Der **Standard-Chipkartenleser (CAT-S)** hat einen Pinpad und erlaubt die sichere PIN-Eingabe direkt am Chipkartenleser. Er unterstützt das PACE Protokoll direkt auf dem Chipkartenleser, so dass die PIN nicht zum Computer übertragen werden muss.

    * Der **Komfort-Chipkartenleser (CA-K)** hat ein Pinpad für die sichere PIN-Eingabe und ein Display, welches 2x16 alphanumerische Zeichen darstellen kann. Dadurch können im Vergleich zu dem Standard-Chipkartenleser, die Berechtigungen auf der authentischen Anzeige des Chipkartenlesers angezeigt werden.

- Der **Bürger** will sich mit dem nPA an einem personalisierten Dienst authentisieren. Dafür benötigt er den nPA und das lokale Terminal.

- Der **Dienstanbieter** stellt eine personalisierte Dienstleistung zur Verfügung, für die sich der Bürger authentisieren muss.

  - Der **Dienst** wird vom Dienstanbieter betrieben und stellt personalisierte Leistungen zur Verfügung. Der Bürger muss sich am Dienst authentisieren, um ihn benutzen zu können.

- Das **Remote Terminal** (siehe BSI TR-03128 [BSI09d]) stellt dem lokalen Terminal die Berechtigungszertifikate zur Verfügung und kommuniziert über das lokale

Terminal mit dem nPA. Das Remote Terminal ist nach der erfolgreichen Authentifizierung in der Lage, auf die eID-Funktion des nPAs zuzugreifen und Daten auszulesen.

- Der **eID-Service** führt die Authentifizierung des Bürgers mit Hilfe des nPAs über das Remote Terminal durch. Anschließend liest das Remote Terminal die Datengruppen des nPAs aus und übermittelt sie über den eID-Service an den Dienstanbieter.

Der eID-Service kann vom Dienstanbieter selbst betrieben oder von einem externen Betreiber verwendet werden. Das spielt für die Authentisierung aus Sicht des Bürgers keine Rolle.

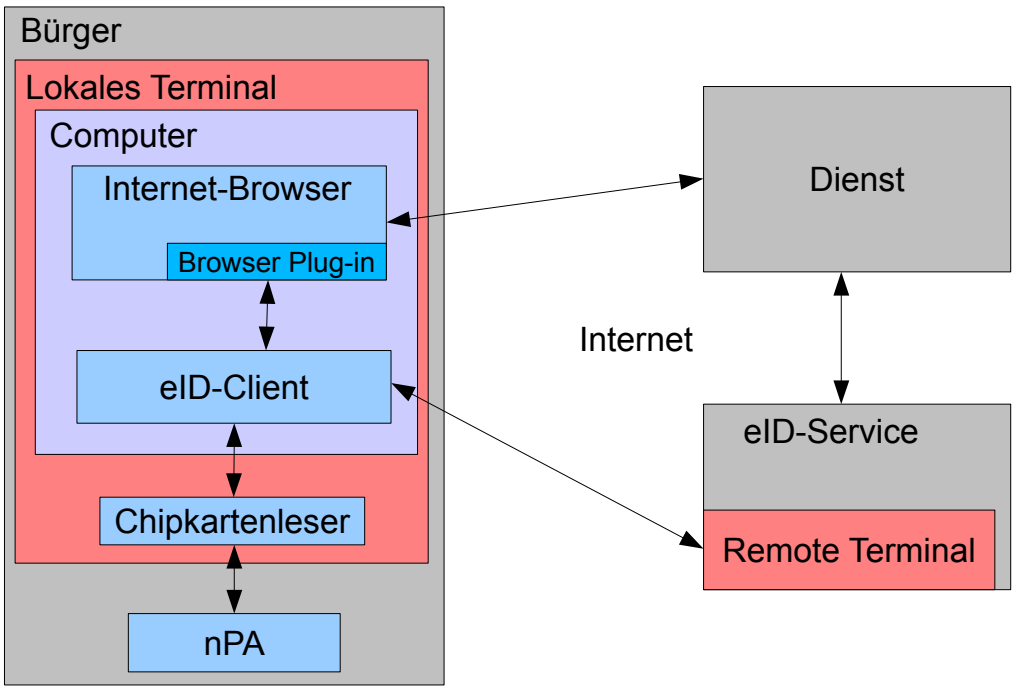

Abbildung 4.1.: Beteiligte Komponenten und Kommunikationswege bei der Authentisierung über die eID-Funktion des nPAs

Systemarchitektur

Bei dem System handelt es sich um ein Client-Server-Architektur. Die Kommunikation zwischen den beteiligten Parteien findet über ein unsicheres Netzwerk (in der Regel das

Internet) statt. Der eID-Client bildet die Schnittstelle zwischen Bürger, nPA, personalisiertem Dienst und dem eID-Service.

Prozessablauf im Detail

Im Folgenden wird der Authentisierungsprozess in einzelnen Schritten beschrieben. Er richtet sich nach der aktuell verfügbaren Version des eID-Clients vom BMI (Stand 20.01.2012):

1. Der Bürger möchte sich an einem personalisierten Dienst im Internet authentisieren. Er wählt die Authentisierung mit dem nPA aus, welche über die eID-Anwendung und EAC durchgeführt wird.

2. Für die Zuordnung der übertragenen Daten zwischen dem Dienstanbieter und dem eID-Service, muss der Dienstanbieter dem eID-Service eine eindeutige Session-ID übergeben. Die Session-ID und die Adresse des eID-Services muss über den Browser an den eID-Client übergeben werden. Ein im Browser integriertes Plugin erkennt, dass die Authentifizierung mit dem nPA durchgeführt werden soll und übergibt dem eID-Client die Adresse des eID-Services und die Session-ID des Dienstes.

3. Der eID-Client prüft anschließend, ob ein nPA mit dem Chipkartenleser verbunden ist und startet den Prozess mit EAC.

4. Bevor die Authentifizierung durchgeführt werden kann, muss der Bürger die Allgemeinen Geschäftsbedingungen des Dienstanbieters bestätigen.

5. Dem Bürger werden die Namen der Datengruppen angezeigt, auf die der Dienstanbieter zugreifen darf. Diese sind im Berechtigungszertifikat des Dienstanbieters definiert. Der Bürger muss bestätigen, auf welche der geforderten Datengruppen er den Zugriff zulässt. Diese Informationen werden an den nPA übertragen und gelten für den gesamten Authentisierungsprozess.

6. Der Bürger gibt anschließend die eID-PIN in das lokale Terminal ein. Über das PACE-Protokoll und die eID-PIN authentisiert sich der Bürger am nPA. Durch PACE wird eine sichere Verbindung zwischen dem lokalen Terminal und dem Chip des nPAs aufgebaut.

7. Nach PACE muss sich das Remote Terminal des eID-Services beim nPA authentisieren. Dafür wird die Terminalauthentisierung (TA) durchgeführt, indem das Remote Terminal des eID-Services das Berechtigungszertifikat an den nPA

übermittelt. Dieses Zertifikat enthält die notwendigen Daten für die Authentifizierung des Remote Terminals und die Berechtigungen für den Zugriff auf die Datengruppen.

8. Im letzten Schritt von EAC muss sich der Chip des nPAs beim Remote Terminal authentisieren. Das erfolgt mit der Chipauthentisierung (CA). Der Chip des nPAs authentisiert sich am Remote Terminal. Dadurch wird eine sichere Verbindung auch zwischen dem Chip und dem Remote Terminal aufgebaut.

9. Nachdem EAC vollständig durchgeführt wurde, darf der eID-Service auf die vom Bürger zugelassenen Datengruppen des nPAs zugreifen. Die Daten werden anschließend vom eID-Service an den Dienst übermittelt.

10. Im letzten Schritt wird der Bürger über den Browser zurück zum Dienst umgeleitet und ist dort authentifiziert.

Im Ablauf sind die verwendeten kryptographischen Protokolle PACE (siehe Abschnitt 2.7), Terminalauthentisierung (siehe Abschnitt 2.8), Chipauthentisierung (siehe Abschnitt 2.9) nur grob beschrieben. Die vollständigen Protokollbeschreibungen können in den jeweiligen Abschnitten nachgelesen werden.

4.1.4. Lokale Kommunikation

An der lokalen Kommunikation ist der Bürger, das lokale Terminal sowie der nPA beteiligt. Sie enthält keine Kommunikation über lokale Netzwerke (LAN) oder dem Internet.

Bevor die Authentisierung gestartet werden kann, muss sich der Bürger gegenüber dem nPA mit dem PACE-Protokoll (siehe Abschnitt 2.7) authentisieren und damit den gesamten Prozess auf dem nPA freischalten.

4.1.5. Externe Kommunikation

Unter externer Kommunikation ist an dieser Stelle die Kommunikation über ein Netzwerk wie dem Internet zu verstehen, welche vom Bürger nicht beeinflusst werden kann.

Für die externe Kommunikation zwischen dem Dienst, dem eID-Client und dem eID-Service bestehen mehrere Integrationsmöglichkeiten. In der Analyse soll der eID-Service sowie der Dienst als getrennte Systeme betrachtet werden. Um die Kombinationsmöglichkeiten zu reduzieren, wird der Dienst und der eID-Service von verschiedenen Anbietern betrieben. Die Kommunikation zwischen allen Parteien findet über ein ungeschütztes Netzwerk wie dem Internet statt.

Für die Kommunikation zwischen den drei Parteien bestehen derzeit zwei Abläufe, die in den folgenden Abschnitten einzeln erläutert werden.

Direkte Kommunikation über Webservices

Bei der direkten Kommunikation über Webservices muss die Authentisierung zwischen den drei Parteien synchronisiert werden. Der personalisierte Dienst muss dem eID-Service vorab mitteilen, dass er eine Authentifizierung des Bürgers benötigt. Dafür übergibt er dem eID-Service vor der Authentifizierung ein eindeutiges Erkennungsmerkmal (Session-ID) des zu authentifizierenden Bürgers.

Diese Session-ID inklusive der Adresse (URL) des eID-Services muss er dem Client mitteilen, damit sich der Client beim eID-Service authentisieren kann. Nach der erfolgreichen Authentifizierung des Bürgers teilt der eID-Service dem eID-Client mit, dass er für den Dienst authentifiziert ist. Der authentifizierte Bürger wird zur Webanwendung umgeleitet. Der Dienst muss die Bestätigung der Authentifizierung direkt beim eID-Service abholen, da kein Vertrauensverhältnis zwischen dem Dienst und dem Bürger besteht.

Indirekte Kommunikation über SAML

Bei der direkten Kommunikation über Webservices erfordert die Authentisierung eine zusätzliche Interaktionen zwischen dem eID-Service und dem Dienstanbieter, um die Bestätigung der erfolgreichen oder erfolglosen Authentisierung zu erhalten. Dieser Schritt soll durch SAML (siehe Abschnitt 2.10) gespart werden, indem die Authentisierung mit Hilfe eines *Security Tokens* durchgeführt wird.

Wie auch bei der direkten Kommunikation muss der Dienst der eID-Client ein eindeutiges Erkennungsmerkmal für den Bürger mitteilen. Der Bürger erhält auch die Adresse des eID-Services, bei dem er sich authentisieren soll. Nach der Authentisierung übergibt der eID-Service dem Bürger ein digital signiertes und verschlüsseltes Security Token (SAML), welches ihn beim Dienst authentisiert. Durch das Security Token kann der Dienstanbieter überprüfen, ob die Authentifizierung erfolgreich durchgeführt wurde, ohne den eID-Service erneut kontaktieren zu müssen.

4.2. Sicherheitsanalyse

Für die Sicherheitsanalyse müssen zuerst IT-Sicherheit-Schutzziele bestimmt werden. Mit Hilfe der in BSI IT-Grundschutz-Vorgehensweise definierten Schutzbedarfskategorien soll eine Schutzbedarfsfeststellung für Informationen, Prozesse und Kommunikationswege durchgeführt werden. Dabei werden die Schutzziele der IT-Sicherheit mit Schutzbedarfskategorien belegt.

Nach BSI IT-Grundschutz-Vorgehensweise sollen zuerst die IT-Grundschutz-Kataloge verwendet werden, um einen Grundschutz zu erreichen. Allerdings enthalten die aktuellen IT-Grundschutz-Kataloge keine Modellierung des eID-Clients. Auch das Einsatzszenario im zivilen Bereich liegt außerhalb des Rahmens der IT-Grundschutz-Vorgehensweise (siehe BSI 100-2 [BSI08b]). Dadurch kann die Schutzbedarfsfeststellung, Auswahl und Anpassung von Maßnahmen, sowie der Basis-Sicherheitscheck ausgelassen werden. An dieser Stelle wird sofort die ergänzende Sicherheitsanalyse angewendet, wie sie in der *IT-Grundschutz-Vorgehensweise* für solche Fälle empfohlen wird. Das Ziel der ergänzenden Sicherheitsanalyse ist es, möglichst alle kritischen Stellen des eID-Clients im Bezug auf die IT-Sicherheit-Schutzziele zu analysieren.

4.2.1. IT-Sicherheit-Schutzziele

Die folgenden IT-Sicherheit-Schutzziele werden in der Sicherheitsanalyse betrachtet:

- *Authentizität* stellt nach Eckert [Eck09] den Nachweis der Echtheit und Glaubwürdigkeit der Identität eines Objekts bzw. Subjekts.

- Unter *Datenintegrität* ist nach Eckert [Eck09] der Schutz vor unautorisierten und unbemerkten Modifikationen der Daten zu verstehen.

- Für die *(Informations-)Vertraulichkeit* muss nach Eckert [Eck09] garantiert werden, dass unautorisierte Informationsgewinnung unterbunden wird.

- Bei der *Verfügbarkeit* zählt nach Eckert [Eck09] der Schutz vor Beeinträchtigung der Funktionalität der Dienste.

- Unter *Verbindlichkeit* versteht man nach Eckert [Eck09] die Unabstreitbarkeit von durchgeführten Handlungen.

- Für *Privatheit* muss nach Eckert [Eck09] der Schutz der personenbezogenen Daten, Schutz der Privatsphäre und Gewährleistung des informationellen Selbstbestimmungsrechts garantiert werden.

- Unter *Anonymisierung* versteht man nach Eckert [Eck09] das Verändern der personenbezogenen Daten der Art, dass die Einzelangaben über persönliche oder sachliche Verhältnisse nicht mehr oder nur mit einem unverhältnismäßig großen Aufwand an Zeit, Kosten und Arbeitskraft einer bestimmten oder bestimmbaren natürlichen Person zugeordnet werden können.

- *Pseudonymisierung* ist nach Eckert [Eck09] eine schwächere Art der Anonymisierung. Die personenbezogenen Daten müssen durch eine Zuordnungsvorschrift so verändert werden, dass die Einzelangaben über persönliche oder sachliche Verhältnisse ohne Kenntnis oder Nutzung der Zuordnungsvorschrift nicht mehr einer natürlichen Person zugeordnet werden können.

- *Nichtverknüpfbarkeit* ist nach Pfitzmann [PH10] dann gegeben, wenn zwei oder mehrere für den Angreifer interessante Gegenstände (wie beispielsweise Personen, Nachrichten, Aktionen, ...) in einem System nicht verknüpft werden können. Das bedeutet, der Angreifer kann nicht unterscheiden, ob diese Gegenstände in irgend einem Zusammenhang stehen.

Des Weiteren wird die Anonymisierung nicht als Sicherheitsziel betrachtet, da dieses Ziel durch die eID-Anwendung nicht bereitgestellt wird. Es wurde lediglich als Grundlage für die Pseudonymisierung benötigt.

4.2.2. Schutzbedarfskategorien

Der Schutzbedarf nach BSI IT-Grundschutz-Vorgehensweise teilt sich auf drei Kategorien auf:

- **normal:** Die Schadensauswirkungen sind begrenzt und überschaubar.

- **hoch:** Die Schadensauswirkungen können beträchtlich sein.

- **sehr hoch:** Die Schadensauswirkungen können ein existentiell bedrohliches, katastrophales Ausmaß erreichen.

Die Tabellen 4.1, 4.2 und 4.3 enthalten die Voraussetzungen für die Schutzbedarfskategorie „normal“, „hoch“ und „sehr hoch“.

Schutzbedarfskategorie „normal“	
1. Verstoß gegen Gesetze/Vorschriften/-Verträge	• Verstöße gegen Vorschriften und Gesetze mit geringfügigen Konsequenzen • Geringfügige Vertragsverletzungen mit maximal geringen Konventionalstrafen
2. Beeinträchtigung des informationellen Selbstbestimmungsrechts	• Es handelt sich um personenbezogene Daten, durch deren Verarbeitung der Betroffene in seiner gesellschaftlichen Stellung oder in seinen wirtschaftlichen Verhältnissen beeinträchtigt werden kann.
3. Beeinträchtigung der persönlichen Unversehrtheit	• Eine Beeinträchtigung erscheint nicht möglich.
4. Beeinträchtigung der Aufgabenerfüllung	• Die Beeinträchtigung würde von den Betroffenen als tolerabel eingeschätzt werden. • Die maximal tolerierbare Ausfallzeit ist größer als 24 Stunden.
5. Negative Innen- oder Außenwirkung	• Eine geringe bzw. nur interne Ansehens- oder Vertrauensbeeinträchtigung ist zu erwarten.
6. Finanzielle Auswirkungen	• Der finanzielle Schaden bleibt für die Institution tolerabel.

Tabelle 4.1.: Schutzbedarfskategorie „normal“ (Quelle: BSI 100-2 [BSI08b])

Schutzbedarfskategorie „hoch“	
1. Verstoß gegen Gesetze/Vorschriften/-Verträge	• Verstöße gegen Vorschriften und Gesetze mit erheblichen Konsequenzen • Vertragsverletzungen mit hohen Konventionalstrafen
2. Beeinträchtigung des informationellen Selbstbestimmungsrechts	• Es handelt sich um personenbezogene Daten, bei deren Verarbeitung der Betroffene in seiner gesellschaftlichen Stellung oder in seinen wirtschaftlichen Verhältnissen erheblich beeinträchtigt werden kann.
3. Beeinträchtigung der persönlichen Unversehrtheit	• Eine Beeinträchtigung der persönlichen Unversehrtheit kann nicht absolut ausgeschlossen werden.
4. Beeinträchtigung der Aufgabenerfüllung	• Die Beeinträchtigung würde von einzelnen Betroffenen als nicht tolerabel eingeschätzt. • Die maximal tolerierbare Ausfallzeit liegt zwischen einer und 24 Stunden.
5. Negative Innen- oder Außenwirkung	• Eine breite Ansehens- oder Vertrauensbeeinträchtigung ist zu erwarten.
6. Finanzielle Auswirkungen	• Der Schaden bewirkt beachtliche finanzielle Verluste, ist jedoch nicht existenzbedrohend.

Tabelle 4.2.: Schutzbedarfskategorie „hoch“ (Quelle: BSI 100-2 [BSI08b])

Schutzbedarfskategorie „sehr hoch“	
1. Verstoß gegen Gesetze/Vorschriften/-Verträge	• Fundamentaler Verstoß gegen Vorschriften und Gesetze • Vertragsverletzungen, deren Haftungsschäden ruinös sind
2. Beeinträchtigung des informationellen Selbstbestimmungsrechts	• Es handelt sich um personenbezogene Daten, bei deren Verarbeitung eine Gefahr für Leib und Leben oder die persönliche Freiheit des Betroffenen gegeben ist.
3. Beeinträchtigung der persönlichen Unversehrtheit	• Gravierende Beeinträchtigungen der persönlichen Unversehrtheit sind möglich. • Gefahr für Leib und Leben
4. Beeinträchtigung der Aufgabenerfüllung	• Die Beeinträchtigung würde von allen Betroffenen als nicht tolerabel eingeschätzt werden. • Die maximal tolerierbare Ausfallzeit ist kleiner als eine Stunde.
5. Negative Innen- oder Außenwirkung	• Eine landesweite Ansehens- oder Vertrauensbeeinträchtigung, eventuell sogar existenzgefährdender Art, ist denkbar.
6. Finanzielle Auswirkungen	• Der finanzielle Schaden ist für die Institution existenzbedrohend.

Tabelle 4.3.: Schutzbedarfskategorie „sehr hoch“ (Quelle: BSI 100-2 [BSI08b])

4.2.3. Schutzziele

Nicht alle Schutzziele gelten für Informationen, Prozesse und Kommunikationswege. Aus diesem Grund werden in den Tabellen 4.4, 4.5 und 4.6 die Schutzziele einzeln auf ihren Einsatzzweck untersucht.

Für Informationen gültige IT-Sicherheitsziele		
Sicherheitsziel	**Geeignet**	**Begründung**
Authentizität	nein	Informationen sind nur im Zusammenhang mit einem Prozess authentisch.
Integrität	ja	Die Integrität von Informationen kann ohne einen bestimmten Prozess überprüft werden.
Vertraulichkeit	ja	Informationen alleine können vertraulich sein.
Verfügbarkeit	nein	Informationen alleine besitzen keine Verfügbarkeit, da sie ohne einen Prozess nicht genutzt werden können.
Verbindlichkeit	nein	Informationen sind erst im Zusammenhang mit einem Prozess verbindlich.
Privatheit	ja	Informationen können privat sein.
Nichtverknüpfbarkeit	ja	Informationen können über ihren Inhalt mit anderen Subjekten oder Objekten verknüpft werden.
Pseudonymisierung	nein	Eine Pseudonym kann nur in einem Prozess verwendet werden.

Tabelle 4.4.: Für Informationen gültige IT-Sicherheitsziele

Für Prozesse gültige IT-Sicherheitsziele		
Sicherheitsziel	**Geeignet**	**Begründung**
Authentizität	ja	In einem Prozess kann es erforderlich sein, dass man die Authentizität von einem Subjekt oder Objekt sicherstellen muss.
Integrität	ja	Ein Prozess kann integer ablaufen, indem keine Manipulationen an ihm statt finden.
Vertraulichkeit	ja	Ein Prozess kann vertraulich sein.
Verfügbarkeit	ja	Ein Prozess beschreibt einen definierten Vorgang und kann Verfügbarkeitsansprüche haben.
Verbindlichkeit	ja	In einem Prozess können Verbindlichkeiten ausgelöst werden, die erfüllt werden müssen.
Privatheit	ja	Ein Prozess kann privat sein oder private Vorgänge enthalten.
Nichtverknüpfbarkeit	ja	Bei der Durchführung eines Prozesses kann die Nichtverknüpfbarkeit aus Sicht von Dritten gefordert werden.
Pseudonymisierung	ja	Ein Pseudonym in einem Prozess kann einem Subjekt oder Objekt helfen sich zu Authentisieren, ohne sich mit seinen richtigen Daten zu identifizieren.

Tabelle 4.5.: Für Prozesse gültige IT-Sicherheitsziele

Für Kommunikationswege gültige IT-Sicherheitsziele		
Sicherheitsziel	**Geeignet**	**Begründung**
Authentizität	nein	Kommunikationswege haben alleine keine Authentizität, da sie lediglich Informationen in Form von Daten transportieren.
Integrität	ja	Die Kommunikation zwischen zwei Parteien kann über unsichere Verbindungen durchgeführt werden. Eine Integritätsprüfung der übertragenen Daten stellt sicher, dass sie bei der Übertragung nicht manipuliert wurden.
Vertraulichkeit	ja	Kommunikationswege können vertraulich sein.
Verfügbarkeit	ja	Kommunikationswege besitzen eine Verfügbarkeit, da sie ausfallen oder durch Dritte manipuliert werden können.
Verbindlichkeit	nein	Kommunikationswege alleine stellen keine Verbindlichkeiten dar.
Privatheit	ja	Kommunikationswege können privat sein.
Nichtverknüpfbarkeit	ja	Durch Kommunikationswege kann eine Verknüpfung zwischen den Kommunizierenden erfolgen.
Pseudonymisierung	nein	Kommunikationswege haben ohne einen Prozess keinen Einfluss auf die Pseudonymisierung der Kommunizierenden.

Tabelle 4.6.: Für Kommunikationswege gültige IT-Sicherheitsziele

4.2.4. Schutzbedarfsfeststellung

In der Schutzbedarfsfeststellung werden Schutzkategorien für Informationen, Prozesse und Kommunikationswege ermittelt. Dafür werden für die einzelnen IT-Sicherheitsziele Schutzkategorien bestimmt.

Datengruppen auf dem nPA

Der Schutzbedarf für die Datengruppen (siehe Tabelle 4.7) bezieht sich ausschließlich auf die Datengruppen selbst, ohne den Authentisierungsprozess, in dem sie verwendet werden.

Authentisierung über die eID-Anwendung

Die Hauptaufgabe der eID-Anwendung ist die Authentifizierung des Bürgers über den nPA und dieser stellt somit den Hauptprozess für das Szenario dar. Die dabei verwendeten Kommunikationswege werden separat betrachtet. Die Tabelle 4.8 enthält die Schutzbedarfsfeststellung für die Authentisierung über die eID-Anwendung.

Kommunikationsverbindungen

An dieser Stelle werden die beiden Varianten der externen Kommunikation (siehe Abschnitt 4.1.5) zusammengefasst betrachtet. Die Unterschiede spielen für die Schutzbedarfsfeststellung keine direkte Rolle, da in beiden Fällen die Kommunikation zwischen den drei Parteien statt findet und damit den gleichen Schutzbedarf fordert. Die Tabelle 4.11 enthält die Schutzbedarfsfeststellung für die lokale Kommunikation. Die Tabelle 4.12 enthält die Schutzbedarfsfeststellung für die externe Kommunikation.

Schutzbedarfsfeststellung: Datengruppen		
Sicherheitsziel	**Schutzbedarfskategorie**	**Begründung**
Integrität	„sehr hoch“	Die Manipulation der Datengruppen verletzt deren Integrität. Da es sich um hoheitlich beglaubigte Daten handelt, ist die Manipulation ein Verstoß gegen das Gesetz. Zudem würde der Bürger durch die Manipulation das informationelle Selbstbestimmungsrecht über seine persönlichen Daten verlieren.
Vertraulichkeit	„sehr hoch“	Können die Datengruppen von Dritten ausgelesen werden, so verliert der Bürger sein informationelles Selbstbestimmungsrecht. Werden die Daten von Dritten missbraucht, so kann ein finanzieller Schaden entstehen. Weiterhin kann es zu einem Image- oder Ansehensverlust führen.
Privatheit	„sehr hoch“	Die Datengruppen auf dem nPA enthalten persönliche und damit auch private Daten. Ein Diebstahl der Daten führt zum Verlust des informationellen Selbstbestimmungsrechts. Weiterhin kann es zu einem Image- oder Ansehensverlust führen.
Nichtverknüpfbarkeit	„sehr hoch“	Die Datengruppen können mit dem Besitzer verknüpft werden. Werden die Datengruppen durch Dritte entwendet, so verliert der Bürger das informationelle Selbstbestimmungsrecht über seine persönlichen Daten. Werden die Daten von Dritten missbraucht, so kann ein finanzieller Schaden entstehen. Weiterhin kann es zu einem Image- oder Ansehensverlust führen.

Tabelle 4.7.: Schutzbedarfsfeststellung für Datengruppen auf dem nPA

Schutzbedarfsfeststellung: Authentifizierung über die eID-Anwendung (Teil 1)		
Sicherheitsziel	**Schutzbedarfskategorie**	**Begründung**
Authentizität	„hoch“	Die Authentifizierung über der eID-Anwendung findet über räumliche Entfernungen statt und muss aus diesem Grund ohne augenscheinliche Überprüfung funktionieren. Sie baut auf den kryptographischen Verfahren vom BSI auf. In dieser Arbeit werden keine Analysen der kryptographischen Verfahren durchgeführt, so dass sie als sicher angenommen werden. Der eID-Service Betreiber und die Dienstanbieter müssen sich auf die vom BSI spezifizierten Verfahren verlassen. Die Verletzung der Authentizität kann zu finanziellen Schäden für beide Parteien führen. Weiterhin kann ein Ansehens- oder Imageverlust für den Dienstanbieter sowie den eID-Service Betreiber entstehen. Der Bürger muss sich bei der Authentisierung auf die Verfahren vom BSI verlassen, da seine übertragenen Datengruppen damit geschützt werden. Eine Verletzung der Authentizität kann zu hohen finanziellen Schäden führen.
Integrität	„hoch“	Der nPA wird bei der eID-Anwendung als Identitätsdokument mit hoheitlich geprüften Datengruppen verwendet. Werden die Daten im Prozess manipuliert, so ist das ein Verstoß gegen das Gesetz. Eine Manipulation der Daten während der Authentifizierung muss erkannt werden, da sonst eine falsche Identität erzeugt wird. Das kann zum Ansehensverlust und zu finanziellen Schäden für den Dienstanbieter und den Bürger führen.

Tabelle 4.8.: Schutzbedarfsfeststellung: Authentifizierung über die eID-Anwendung (Teil 1)

Schutzbedarfsfeststellung: Authentifizierung über die eID-Anwendung (Teil 2)		
Sicherheitsziel	**Schutzbedarfskategorie**	**Begründung**
Vertraulichkeit	„hoch“	Durch den Identitätsnachweis erhält der Bürger Zugang zu personalisierten Dienstleistungen. Die Datengruppen werden für den Dienstanbieter benötigt, um die Leistung erfüllen zu können. Aus Sicht des Bürgers ist ein Schaden gering, da der Dienstanbieter in Verantwortung steht, die Daten vertraulich zu behandeln. Der Bürger muss lediglich die eID-PIN geheim halten und den nPA vor unberechtigten Zugriff schützen. Der Dienstanbieter benötigt die Datengruppen, um die Dienstleistung erfüllen zu können. Damit trägt er die Verantwortung für die Vertraulichkeit der Daten und muss entsprechenden hohen Schutz für die Daten bieten. Für den Bürger und den Dienstanbieter kann die Verletzung der Vertraulichkeit zu einem Ansehensverlust führen. Weiterhin kann abhängig vom Dienst ein hoher finanzieller Schaden entstehen.
Verfügbarkeit	„normal“	Die Verfügbarkeit hängt maßgeblich von der spezifischen Anwendung ab. Eine allgemeine Abschätzung ist deswegen schwer möglich. Der Bürger muss vor allem bei der Authentisierung eine Wahl zwischen eID-Verfahren und mindestens einem anderen Verfahren haben, bei dem kein nPA benötigt wird. Daher ist die Verfügbarkeit nicht primär von der eID-Anwendung abhängig.
Verbindlichkeit	„hoch“	Im Unterschied zu den hoheitlichen Anwendungen sind die Datengruppen der eID-Anwendung nicht hoheitlich beglaubigt. Sie werden aber trotzdem als höchst wahrscheinlich richtig betrachtet, wenn sie gestohlen werden. Dritte können mit den gestohlenen Datengruppen Handlungen im Namen des Bürgers durchführen, die zu einem Ansehensverlust und zu erheblichen finanziellen Schäden führen können.

Tabelle 4.9.: Schutzbedarfsfeststellung: Authentifizierung über die eID-Anwendung (Teil 2)

Schutzbedarfsfeststellung: Authentifizierung über die eID-Anwendung (Teil 3)		
Sicherheitsziel	**Schutzbedarfskategorie**	**Begründung**
Privatheit	„hoch“	Die Datengruppen des nPAs sind persönlich. Das bedeutet, sie müssen geschützt werden, um die Privatsphäre und das informationelle Selbstbestimmungsrecht des Bürgers zu garantieren. Wird die Privatheit verletzt, so kann das für den Bürger zum Ansehensverlust führen und unter Umständen finanzielle Schäden verursachen.
Nichtverknüpfbarkeit	„hoch“	Die Verknüpfbarkeit bei der Verwendung des Pseudonyms bei der eID-Anwendung lässt sich berechnen. Sie hängt maßgeblich von gesetzeswidrigen Absprachen zwischen Anbietern ab, die bei genügenden Anzahl von Informationen das Pseudonym einer realen Person zuordnen können.
Pseudonymisierung	„hoch“	Die eID-Anwendung ermöglicht es eine Authentisierung über einen Pseudonym durchzuführen. Wird das Pseudonym aufgedeckt und mit persönlichen Daten des Bürgers verknüpft, so kann das zum Ansehensverlust führen und unter Umständen finanzielle Schäden verursachen.

Tabelle 4.10.: Schutzbedarfsfeststellung: Authentifizierung über die eID-Anwendung (Teil 3)

Schutzbedarfsfeststellung: Lokale Kommunikation		
Sicherheitsziel	**Schutzbedarfskategorie**	**Begründung**
Integrität	keine	Die Kommunikation zwischen dem lokalen Terminal und dem nPA findet über eine Funkverbindung statt. Diese erlaubt keinen physikalischen Schutz. Die Integrität der übertragenen Daten muss durch kryptographische Verfahren sicher gestellt werden.
Vertraulichkeit	keine	Wie bereits bei Integrität beschrieben, ist kein physikalischer Schutz der Verbindung möglich. Auch die Vertraulichkeit der übertragenen Daten muss durch kryptographische Verfahren sicher gestellt werden.
Verfügbarkeit	normal	Die Verfügbarkeit der lokalen Verbindung kann vom Bürger überprüft werden. Daher ist hier der normale Schutzbedarf ausreichend.

Tabelle 4.11.: Schutzbedarfsfeststellung für die lokale Kommunikation

Schutzbedarfsfeststellung: Externe Kommunikation		
Sicherheitsziel	**Schutzbedarfskategorie**	**Begründung**
Integrität	keine	Die Kommunikation findet über eine unsichere Verbindung statt. Die Integrität der Daten muss durch kryptographische Verfahren sicher gestellt werden. Diese wurden vom BSI spezifiziert, deren Betrachtung nicht Bestandteil dieser Arbeit ist. Sie werden an dieser Stelle als sicher angenommen.
Vertraulichkeit	keine	Die als unsicher geltende Verbindung kann keine Vertraulichkeit garantieren. Dies muss durch kryptographische Verfahren sicher gestellt werden, die nicht Teil der Kommunikationsverbindung sind.
Verfügbarkeit	normal	Die Kommunikation zwischen allen Parteien findet über unsichere Netzwerke (in der Regel das Internet) statt. Aus Sicht des Bürgers kann keine hohe Verfügbarkeit für die Internetverbindung verlangt werden, da in der Regel keine Service Level Agreements (SLA) zwischen Internet Provider und Bürger bestehen. Zudem schränkt eine gestörte Kommunikationsverbindung in den meisten Fällen die komplette Kommunikation mit dem Internet ein. Die Kommunikationsverbindung des Dienstanbieters und des eID-Service-Betreibers wird als funktionierend vorausgesetzt, da der Bürger keinen Einfluss darauf hat.

Tabelle 4.12.: Schutzbedarfsfeststellung für die externe Kommunikation zwischen dem eID-Client, dem eID-Service und dem Dienst.

4.3. Bedrohungsanalyse

In der Bedrohungsanalyse sollen für die Schutzziele (siehe Abschnitt 4.2.1) der einzelnen Komponenten Bedrohungen identifiziert werden. Die Analyse wird mit Hilfe von Angriffsbäumen durchgeführt.

4.3.1. Angriffsbäume

Angriffsbäume (attack trees) bzw. Bedrohungsbäume sollen potenzielle Bedrohungen für die einzelnen Schutzziele grafisch erfassen. Sie sind sehr stark an Fehlerbäume angelehnt, besitzen aber keine einheitliche Beschreibung. Für die Analyse werden die Angriffsbäume nach dem Aufbau von Eckert [Eck09] verwendet, wobei sie für diese Aufgabe vereinfacht wurden.

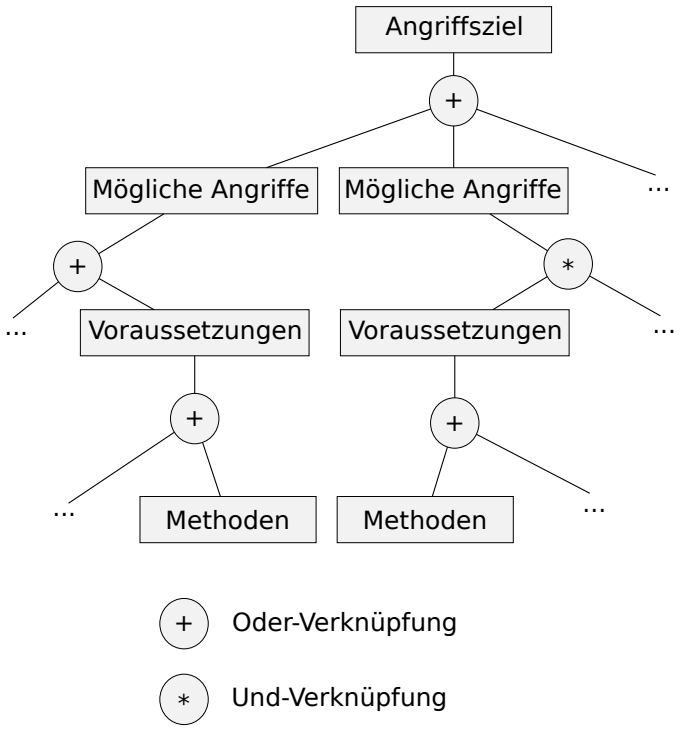

Abbildung 4.2.: Erklärung der Angriffsbäume und verwendete Symbole

Der Grundaufbau der Angriffsbäume ist in Abbildung 4.2 dargestellt. Die Knoten des Baums werden durch UND- (Multiplikation-Zeichen) bzw. ODER-Verknüpfungen (Plus-Zeichen) verbunden. Die Wurzel des Baums ist das Angriffsziel. Die zweite Ebene des Angriffsbaums beschreibt mögliche Angriffe, die für das gegebene Angriffsziel möglich sind. Für diese Angriffe sind Voraussetzungen notwendig, die für einen erfolgreichen

Angriff erfüllt sein müssen. Sie sind auf der dritten Ebene dargestellt. Um diese Voraussetzungen zu erfüllen, werden in der vierten Stufe mögliche Methoden dafür genannt.

4.3.2. Szenario

Das folgende Szenario soll als Grundlage für die Bedrohungsanalyse dienen und gleichzeitig die Anzahl der möglichen Szenarien auf eins reduzieren. Durch die Abgrenzung wird ein Authentisierungsprozess mit allen beteiligten Komponenten beschrieben. Bei der Kommunikation und den Datengruppen werden Abstraktionen durchgeführt, da die einzelnen Komponenten keinen Einfluss auf die Sicherheit des Prozesses aufweisen.

Es wird davon ausgegangen, dass der Chipkartenleser ordnungsgemäß funktioniert und sich wie erwartet verhält. Bei der Untersuchung werden drei verschiedene Typen von Chipkartenlesern betrachtet: Basis-Chipkartenleser, Standard-Chipkartenleser und Komfort-Chipkartenleser.

Es wird davon ausgegangen, dass der personalisierte Dienst und der eID-Service sich korrekt verhalten. Die Bedrohungen beziehen sich auf die Softwarekomponente eID-Client und die Kommunikationsverbindungen, an denen der eID-Client beteiligt ist.

Der Zugriff auf die Datengruppen ist durch EAC (siehe Abschnitt 2.6) geschützt, weil eine sichere Ende-zu-Ende Verbindung zwischen dem Chips des nPAs und dem eID-Service aufgebaut wird. Alle Angriffe auf Datengruppen beziehen sich lediglich auf die kryptographischen Protokolle und sind nicht Bestandteil dieser Arbeit.

4.3.3. Vertraulichkeit

Die PIN und die CAN (siehe Abschnitt 1.3) des nPAs stellen für den Bürger vertrauliche Informationen dar und müssen entsprechend geschützt werden. Diese werden in den folgenden Abschnitten getrennt betrachtet, da für die Angriffe verschiedene Voraussetzungen erfüllt sein müssen.

Angriffe auf die eID-PIN

Die eID-PIN stellt eine vertrauliche Information dar. Um sich zu authentisieren, muss der Bürger die eID-PIN eingeben und damit die eID-Funktion auf dem nPA freischalten.

Die Wurzel des Angriffsbaums in Abbildung 4.3 zeigt mögliche Bedrohungen für die eID-PIN. Für die erfolgreiche Authentisierung im Namen des Bürgers, muss der Angreifer

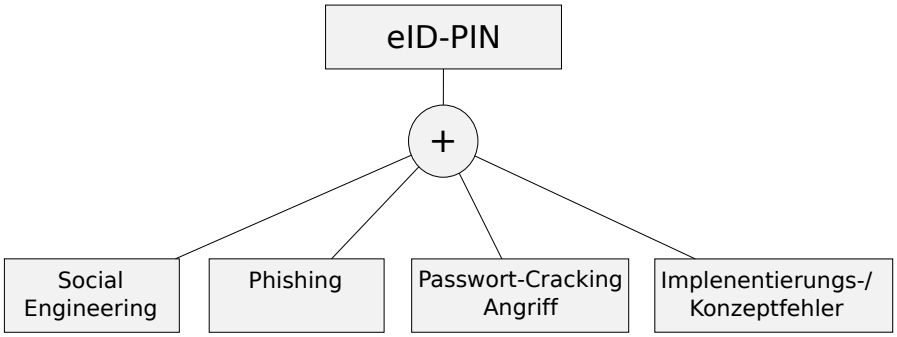

Abbildung 4.3.: Wurzel des Angriffsbaums auf die PIN

noch zusätzlich auf den nPA zugreifen können. An dieser Stelle soll lediglich die eID-PIN als vertrauliche Information betrachtet werden.

Um die eID-PIN zu erhalten, sind folgende Angriffsmöglichkeiten gegeben:

- **Social Engineering:** Social Engineering oder Social Hacking ist ein Versuch des Angreifers, das Opfer dazu zu bringen, dass es unabsichtlich oder absichtlich im guten Glauben sensitive Informationen an den Angreifer preisgibt (Quelle: Eckert [Eck09]). In diesem Fall wäre die eID-PIN die sensitive bzw. vertrauliche Information, die das Ziel des Angreifers ist.

- **Phishing**: Phishing Angriffe sind eine spezielle Ausprägung von Identitätsdiebstählen (Quelle: Eckert [Eck09]). Dabei versucht der Angreifer durch Täuschungsversuche die PIN des Bürgers zu stehlen. Die notwendigen Voraussetzungen und Methoden sind in der Abbildung 4.4 dargestellt und separat erläutert.

- **Passwort-Cracking:** Passwort-Cracking ist eine Möglichkeit durch eine Methode das Passwort zu erraten bzw. auf das Passwort zu schließen. Für die PIN gelten spezielle Voraussetzungen, die in der Abbildung 4.5 detailliert dargestellt sind.

- Weiterhin können **Implementierungs- und Konzeptfehler** dazu führen, dass der Angreifer Zugriff auf die PIN erhält. Die dafür notwendigen Voraussetzungen und Methoden für Angriffe sind in der Abbildung 4.6 dargestellt und separat erläutert.

Der Teilbaum in Abbildung 4.4 zeigt mögliche Phishing-Angriffe, um die eID-PIN beim Einsatz der eID-Anwendung abzufangen. Die einzelnen Wege sind von Links nach Rechts erläutert:

- Der Angreifer kann über den Computer des Bürgers die PIN abgreifen. Dafür muss er sicherstellen, dass der Bürger die PIN über den Computer eingibt oder

diese zum Computer übertragen wird. Verwendet der Bürger einen Standard- oder Komfort-Chipkartenleser, so kann ihn der Angreifer durch Social Engineering Techniken überzeugen, die eID-PIN über den Computer einzugeben. Wird ein Basis-Chipkartenleser verwendet, so wird die PIN über den Computer eingegeben, was zu folgenden drei möglichen Angriffen führt:

- Durch einen untergeschobenen und manipulierten eID-Client kann der Angreifer die eID-PIN-Eingabe abfangen. Der Angreifer kann auch die Benutzeroberfläche des eID-Clients nachahmen und sie dem Bürger unterschieben (Phishing). Bei der eID-Anwendung wird die Authentifizierung über den Browser angestoßen. Das Browser-Plugin übernimmt die Informationen und leitet Sie an den eID-Client weiter (siehe auch Abschnitt 4.1.3). Dadurch ergeben sich drei Komponenten, die angegriffen werden können:

  * Der Angreifer kann den eID-Client manipulieren indem er eine Schwachstelle im eID-Client ausnutzt und dadurch ausführbaren Code einschleust. Er kann aber auch dem Bürger eine manipulierte eID-Client-Installation unterschieben und auf diesem Weg die PIN abgreifen.

  * Der Browser ist eine kritische Komponente, weil über ihn die Authentifizierung angestoßen wird. Mit Hilfe einer Schwachstelle im Browser kann der Angreifer ausführbaren Code einschleusen, welcher den vorhandenen eID-Client manipuliert oder einen manipulierten eID-Client dem Bürger unterschiebt. Auch durch Cross Site Scripting (XSS) (siehe Eckert [Eck09]) kann der Angreifer den Browser manipulieren und dem Bürger einen manipulierten eID-Client unterschieben (siehe auch ePerso „PIN-Diebstahl ohne Malware“ von Jan Schejbal [Schb]).

  * Ein manipuliertes Browser-Plugin kann einen vom Angreifer manipulierten eID-Client öffnen. Dafür kann der Angreifer eine Schwachstelle im Browser-Plugin ausnutzen und ausführbaren Code einschleusen.

- Ein Dritter kann unberechtigten Systemzugriff erhalten, indem er auf den Computer des Bürgers einen Trojaner oder ein Rootkit einschleust. Dadurch kann er das System beliebig steuern.

- Um die eID-PIN abzufangen, besteht die Möglichkeit, die Eingabegeräte am Computer zu belauschen. Auf der Software-Seite können Betriebssystemkomponenten oder Treiber belauscht werden, wenn die Informationen ungeschützt übertragen werden. Alternativ kann die Eingabe direkt an der Tastatur abgehört werden, indem die Übertragung am Kabel abgehört oder bei Funktastaturen die Funkübertragung belauscht wird. Für die Funkübertragung muss sich der Angreifer in Kommunikationsreichweite befinden und zudem

die übertragenen Daten entschlüsseln können, da sie in der Regel verschlüsselt sind.

- Durch einen Späher-Angriff besteht die Möglichkeit, die PIN optisch abzulesen. Dafür benötigt der Angreifer optischen Zugang auf die PIN-Eingabe. Für diesen Angriff wären alle Chipkartenleser anfällig.

- Die Eingabe der eID-PIN kann auch direkt am Chipkartenleser abgefangen werden, wenn ein Standard- oder Komfort-Chipkartenleser zum Einsatz kommen. Der Angreifer kann die Tastatur des Chipkartenlesers ersetzen oder dessen System-Bus abhören, um die PIN abzufangen.

Für die PIN existieren eingeschränkte Passwort-Cracking Angriffe, die in der Abbildung 4.5 zu sehen sind und im Folgenden erläutert werden:

- Um Brute-Force Angriffe durchführen zu können, muss der Angreifer mit dem nPA kommunizieren können. Das ist gegeben, wenn sich der nPA in Funkreichweite befindet oder der Angreifer unberechtigten Zugriff auf den Computer besitzt, mit dem der nPA verbunden ist. Den unberechtigten Zugriff kann der Angreifer durch Einschleusen eines Trojaners oder Rootkits erreichen. Wie bei allen Chipkarten darf die PIN beim nPA maximal dreimal falsch eingegeben werden, bevor die Chipkarte sich selber sperrt. Beim nPA wird der dritte Fehlversuch erst nach der Eingabe der richtigen CAN freigeschaltet. Damit der Angriff nicht auffällt, darf der Angreifer maximal einen Fehlversuch nach einer erfolgreichen Authentisierung durchführen, da sonst der Bürger die Fehlversuche auf jeden Fall bemerkt. Damit der Angriff funktioniert, muss der nPA vom Bürger sehr häufig verwendet werden. Dadurch wird der Angriff deutlich erschwert. Zudem kann der Bürger die eID-PIN jederzeit ändern.

- Eine weitere Möglichkeit besteht darin, einen Man-in-the-Middle Angriff durchzuführen. Der Angreifer muss in der Lage sein, die Kommunikation zwischen dem nPA und dem eID-Client zu kontrollieren. Dafür muss er zusätzlich unberechtigten Zugriff auf den Computer bekommen oder dem Bürger ein von ihm kontrolliertes Terminal unterschieben. Auch muss der Bürger die eID-PIN über den Computer eingeben, indem er ein Basis-Chipkartenleser verwendet oder der Angreifer den Bürger durch Social Engineering überzeugt, die eID-PIN über den Computer einzugeben. Sind diese Bedingungen erfüllt, so kann der Angreifer sich als Terminal ausgeben und die Zufallszahl mit einer vorgegebenen eID-PIN verschlüsseln. Wird PACE erfolgreich durchgeführt, so weiß er, dass er die richtige eID-PIN erraten hat. Für einen erfolgreichen Angriff muss der Bürger allerdings viele Vorsichtsmaßnahmen missachten und die eID-PIN sehr oft ohne Verdacht eingeben.

Implementierungs- und Konzeptfehler können zusätzliche Sicherheitslücken öffnen (siehe auch Abbildung 4.6). An dieser Stelle werden nur mögliche Fehler in dem eID-Client

untersucht. Für die Ausnutzung solcher Fehler benötigt der Angreifer auf jeden Fall unberechtigten Zugriff auf den Computer, da der eID-Client auf dem Computer des Bürgers ausgeführt wird. Damit die eID-PIN über den eID-Client eingegeben wird, muss ein Basis-Chipkartenleser verwendet werden oder der Angreifer muss den Bürger durch Social Engineering (beispielsweise Phishing-Angriffe) überzeugen, die eID-PIN über die Tastatur des Computer einzugeben. Durch Implementierungs- oder Konzeptfehler kann der Angreifer Zugriff auf die Eingabe der eID-PIN erhalten, indem er einer der folgenden Sicherheitslücken ausnutzt:

- Werden Felder, Variablen oder get-Methoden für das Auslesen der eID-PIN als öffentlich deklariert, so kann ein Angreifer diese auslesen.
- Die eID-PIN muss eine gewisse Zeit im Arbeitsspeicher vorgehalten werden, um PACE mit dem Basis-Chipkartenleser durchführen zu können. Der Angreifer kann zu diesem Zeitpunkt ein Abbild des Arbeitsspeichers erstellen oder die eID-PIN aus dem Arbeitsspeicher direkt auslesen.
- Der eID-Client könnte die eID-PIN auf dem Computer abspeichern. Der Angreifer mit lokalen Zugriff kann sie auslesen.
- Für die Fehlersuche im eID-Client kann die eID-PIN in das Protokoll geschrieben werden, wenn beispielsweise der DEBUG-Modus der Anwendung aktiv ist. Ein Angreifer könnte diese Funktionalität ausnutzen, indem er die Anwendung im DEBUG-Modus ausführt und die eID-PIN anschließend aus dem Protokoll ausliest.

Angriff auf die CAN

Die CAN ist eine auf der Vorderseite des nPAs aufgedruckte sechsstellige Dezimalzahl. Sie wird für das Entsperren des dritten Eingabeversuchs und für hoheitliche Zwecke (siehe auch Abschnitt 1.3) benötigt. Weiterhin wird mit Hilfe der CAN die Qualifizierte Elektronische Signatur (QES) beantragt.

Die CAN ist vor allem für hoheitliche Verwendung bestimmt. Der Angreifer kann zwar mit der CAN erfolgreich PACE durchführen, aber er ist nicht in der Lage, mit einem Berechtigungszertifikat von einem eID-Service Betreiber auf die Datengruppen des nPAs zuzugreifen. Dazu ist ein hoheitliches Berechtigungszertifikat notwendig, welches nur in hoheitlichen Inspektionsterminals für Behörden und Grenzkontrollen verwendet wird.

Beim Bestellen der digitalen Signaturen (QES) ist keine Signatur-PIN für die Signaturanwendung gesetzt. Daher muss sich der Halter des Ausweises zuerst mit der CAN authentisieren. Kennt ein Angreifer die CAN und hat er Zugang zum nPA, so ist er in der Lage, eine digitale Signatur im Namen des Bürgers zu beantragen. Allerdings kann

er mit der Signatur-PIN nichts signieren, da für die Signatur-Anwendung ein Standard- oder Komfort-Chipkartenleser notwendig ist und dieser die Signatur-PIN nur über das Pinpad annehmen darf.

Um die CAN zu erraten, muss der Angreifer über längeren Zeitraum Zugang zum nPA haben oder sich in Funkreichweite befinden. Morgen und Oepen [DO10] haben gezeigt, dass sie mit ihrer PACE-Implementierung für den *Bruteforce-Angriff* auf die CAN in Durchschnitt 4,5 Tage benötigen. Die Geschwindigkeit beim Raten hängt vor allem von der Geschwindigkeit eines PACE-Durchlaufs auf dem nPA ab, da die Anzahl möglicher Versuche bei einer sechsstelligen CAN auf 999.999 begrenzt ist. Vor allem kann die CAN nicht geändert werden, ohne einen neuen nPA zu beantragen.

Ist dem Angreifer die CAN bekannt, so kann er mit ihrer Hilfe eine Person verfolgen. Dafür führt er in Funkreichweite des nPAs PACE mit der CAN aus. Wird PACE erfolgreich durchgeführt, so kann er sich ziemlich sicher sein, dass die verfolgte Person noch immer die selbe Person ist.

Ein Angreifer kann auch mit der CAN einen finanziellen Schaden anrichten, so dass sie ebenfalls geheim gehalten werden sollte. Sollte ihm gelingen, eine QES im Namen des Bürgers zu beantragen, so kann das zum Ansehensverlust und unter Umständen zu einem finanziellen Schaden für den Bürger führen.

4.3.4. Authentizität

Die Authentizität des Bürgers wird durch die eID-Service überprüft. Dafür wird der Authentisierungsprozess zwischen nPA und dem eID-Service mit Hilfe von EAC (siehe Abschnitt 2.6) durchgeführt. Nach CA beseht eine sichere Ende-zu-Ende Verbindung zwischen nPA und eID-Service. Aus diesem Grund liegen die Angriffsmöglichkeiten nur vor oder nach der erfolgreichen Durchführung von EAC. Die Abbildung 4.7 zeigt den Angriffsbaum auf den Authentisierungsprozess.

Die folgenden drei Angriffe sind vor der eigentlichen Authentisierung möglich. Das Ziel des Angriffs ist es, den Bürger auf einen vom Angreifer kontrollierten eID-Service umzuleiten:

- Der Angreifer kann die DNS-Einträge vom eID-Service manipulieren und den Bürger damit auf einen von ihm kontrollierten eID-Service umleiten. Das kann durch DNS-Spoofing[1] erfolgen.

[1] Manipulation der Zuordnung zwischen Domainnamen und der dazugehörigen IP-Adresse, so dass das Opfer über die IP-Adresse auf einen vom Angreifer kontrollierten Server umgeleitet wird.

- Ein manipuliertes Browser-Plugin oder ein manipulierter eID-Client kann den Browser des Bürgers auf einen anderen eID-Service umleiten. Die Manipulation des Browser-Plugins oder des eID-Clients kann durch eine Sicherheitslücke erfolgen, indem ausführbarer Code in die Anwendung eingeschleust wird.

- Die Manipulation des Browsers ermöglicht es, die Umleitung zum eID-Service zu verändern, bevor das Browser-Plugin eingreift. Das ist durch Cross Site Scripting oder durch Einschleusen von ausführbaren Code in den Browser möglich.

Weiterhin besteht die Möglichkeit, durch unautorisierten Systemzugriff den Authentisierungsvorgang auf dem Computer zu manipulieren, indem ein Trojaner oder Rootkit auf das System eingeschleust wird. Die Manipulation kann vor oder nach der Authentisierung erfolgen, wobei der Angreifer keinen Zugriff auf die Datengruppen des nPAs erhält, wenn er nicht im Besitz eines gültigen Berechtigungszertifikats ist.

Nach der erfolgreichen Authentifizierung wird der Bürger über den Browser zum Dienst umgeleitet. Dafür sendet der eID-Service gesichert über SSL eine HTTP-Umleitung an den eID-Client, damit der Bürger zu dem personalisierten Dienst umgeleitet wird. Nach der Umleitung glaubt der Bürger den richtigen Dienst zu benutzen. Der Angreifer kann die Umleitung zum richtigen Dienst manipulieren. Dadurch werden Phishing-Angriffe möglich. Die Manipulation kann auf die folgenden Art und Weise erfolgen:

- Ein manipulierter Browser kann die Umleitung zum Dienst verändern. Das wird beispielsweise durch Cross-Site Scripting möglich. Weiterhin kann der Angreifer eine Sicherheitslücke im Browser ausnutzen und dadurch ausführbaren Code in den Browser einschleusen.

- Ein manipuliertes Browser-Plugin oder ein manipulierter eID-Client kann den Bürger auf eine vom Angreifer kontrollierte Seite umleiten. Die Manipulation kann durch Einschleusen von ausführbaren Code erfolgen.

- Eine weitere Möglichkeit für den Angreifer ist es, die DNS-Einträge vom Dienst zu manipulieren und den Bürger damit auf eine von ihm kontrollierte Seite umzuleiten. Das kann durch DNS-Spoofing erfolgen.

Um eine Authentifizierung im Namen des Bürgers durchzuführen (siehe Abbildung 4.8), muss der Angreifer die eID-PIN des nPAs (siehe Abschnitt 4.3.3) kennen und auf den nPA zugreifen können. Der Zugriff auf den nPA kann auf mehreren Wegen erreicht werden:

- Durch Social Engineering (siehe Eckert [Eck09]) kann der Angreifer den Bürger überzeugen, den nPA an ihn zu übergeben oder den Bürger überreden den nPA auf ein von ihm kontrollierten Chipkartenleser zu legen.

- Befindet sich der nPA in Funkreichweite und kann der Angreifer auf den nPA zugreifen, so ist er in der Lage sich mit Hilfe der PIN an einem Dienst zu authentisieren.

- Auch das gewaltsame Entwenden ist eine Möglichkeit, in Besitz des nPAs zu kommen.

- Erhält der Angreifer durch einen Trojaner oder Rootkit unautorisierten Zugriff auf den Computer des Bürgers, so muss er nur noch den Bürger überzeugen den nPA auf den Chipkartenleser zu legen. Der Angreifer kann warten, bis der Bürger den nPA verwenden will oder er kann den Bürger durch Social Engineering überzeugen, den nPA mit dem Chipkartenleser zu verbinden.

4.3.5. Integrität

Die Integrität der Daten muss überprüfbar sein, damit Manipulationen erkannt werden. Das vom BSI spezifizierte EAC stellt bei der Übertragung die Integrität der Daten sicher. Alle Angriffe auf die Integrität gehen auf kryptographische Protokolle und Verfahren zurück, deren Untersuchung nicht Bestandteil dieser Arbeit ist.

Der Bürger muss vor der Übertragung entscheiden, welche Datengruppen er an den Dienstanbieter übertragen will. Diese Auswahl kann von einem Angreifer manipuliert werden, wenn dieser die Kontrolle über den Computer des Bürgers hat. Der Bürger kann eine solche Manipulation nur beim Komfort-Chipkartenleser erkennen, da alle anderen Chipkartenleser kein Display für die Darstellung der Informationen besitzen. Weiterhin kann der Angreifer durch eine solche Manipulation einen schwer zu erkennenden Denial of Service Angriff durchführen, indem er auf dem nPA Berechtigungen setzt, die nicht im Berechtigungszertifikat enthalten sind. Eine solche Manipulation ist nur sehr schwer zu erkennen.

Sollte ein Angreifer auch die Kontrolle über einen eID-Service mit einem gültigen Berechtigungszertifikat besitzen, so wäre er in der Lage, die Datengruppen vom nPA auszulesen.

4.3.6. Verbindlichkeit

Durch die Authentifizierung über die eID-Funktion des nPAs werden für den Bürger keine direkten Verbindlichkeiten ausgelöst, da die eID-Anwendung keine hoheitliche Anwendung ist, wie es bei der Qualifizierten Elektronischen Signatur (QES) der Fall ist. Das ist allerdings rechtlich unklar, weil zu dieser Anwendung noch keine Urteile existieren.

4.3.7. Verfügbarkeit

Die Verfügbarkeit des Authentisierungsprozesses hängt maßgeblich von der Verfügbarkeit der Kommunikationsverbindungen und der beteiligten Softwarekomponenten auf dem Computer des Bürgers ab.

In der Abbildung 4.9 ist der Angriffsbaum auf die Verfügbarkeit des Authentisierungsprozesses zu sehen, welcher im Folgenden beschrieben ist:

- Eine Manipulation des Browsers durch Cross Site Scripting oder durch Einschleusen von Code kann die Ansteuerung des Browser-Plugins stören. Solche Angriffe können auch die Umleitung vom personalisierten Dienst zum eID-Service stören.

- Eine Manipulation des Browser-Plugins oder des eID-Clients kann die Authentifizierung verhindern. Das wird möglich, indem man eine Schwachstelle in einer der beiden Komponenten ausnutzt und dadurch ausführbaren Code einschleust.

- Weiterhin kann die lokale Kommunikation (siehe Abschnitt 4.1.4) zwischen dem nPA und dem eID-Client gestört werden. Dafür benötigt der Angreifer unberechtigten Zugriff auf den Computer, da diese Kommunikation nur lokal statt findet. Dadurch ist er in der Lage durch mehrmaliges Eingeben der falschen PIN den nPA zu sperren (Denial of Service Angriff). Alternativ kann er auch jegliche Kommunikation auf dem Computer durch einen Trojaner oder Rootkit stören.

- Die externe Kommunikation (siehe Abschnitt 4.1.5) kann durch einen Denial of Service Angriff gestört werden. Ziel des Angriffs können alle Prozessbeteiligten sein. Weiterhin kann der Angreifer die Verbindung zwischen Computer des Bürgers und dem eID-Service, bzw. dem personalisierten Dienst durch Spoofing-Angriffe stören. Durch DNS Spoofing ist der Angreifer in der Lage, die IP-Adresse des eID-Services oder des personalisierten Dienstes zu manipulieren und damit eine erfolgreiche Authentifizierung zu verhindern.

- Generell ermöglicht ein unberechtigter Zugriff auf den Computer sehr viele Angriffsmöglichkeiten, welcher durch das Einschleusen von Trojanern oder Rootkits erreicht werden.

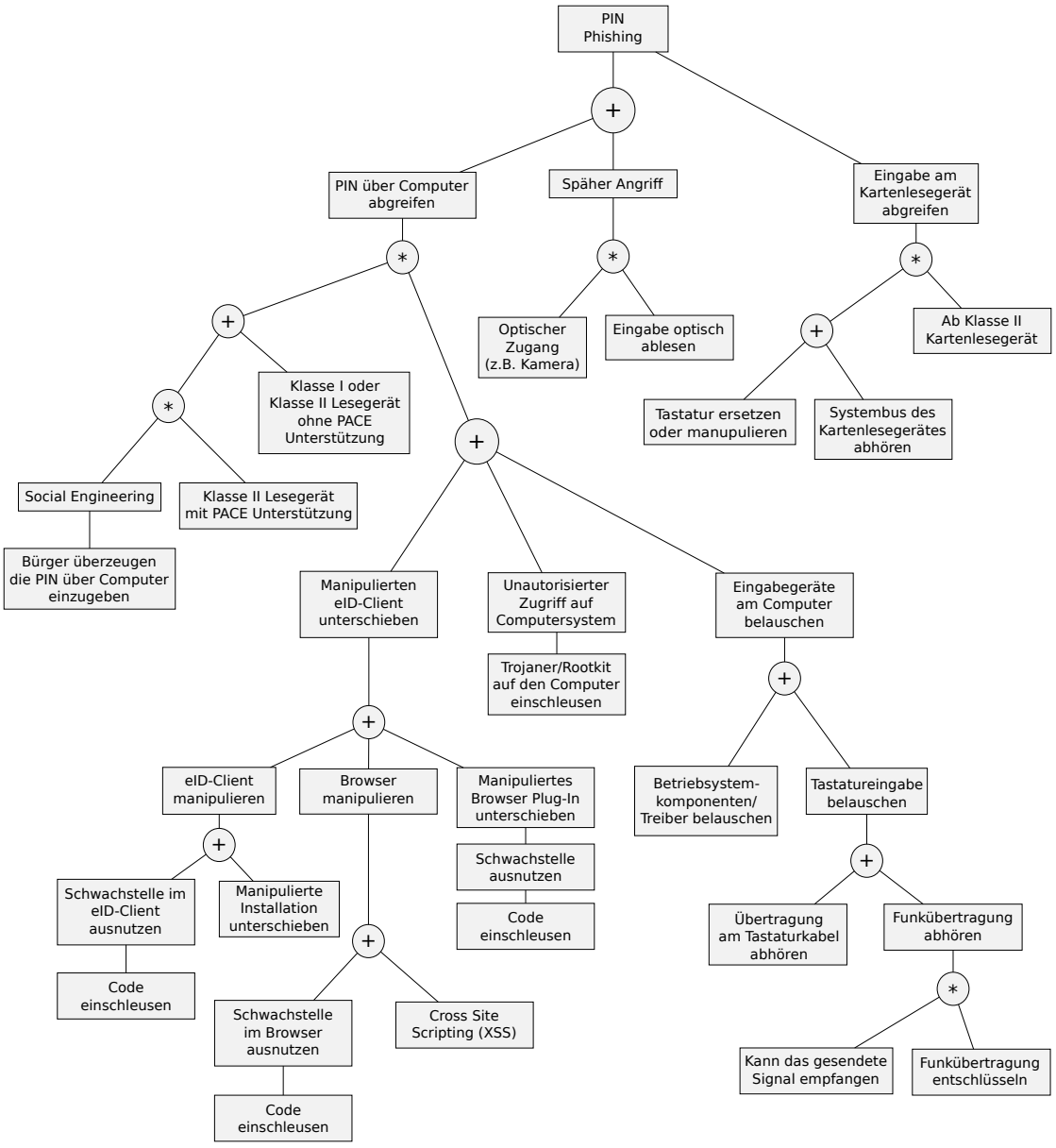

Abbildung 4.4.: Phishing Angriffe auf PIN

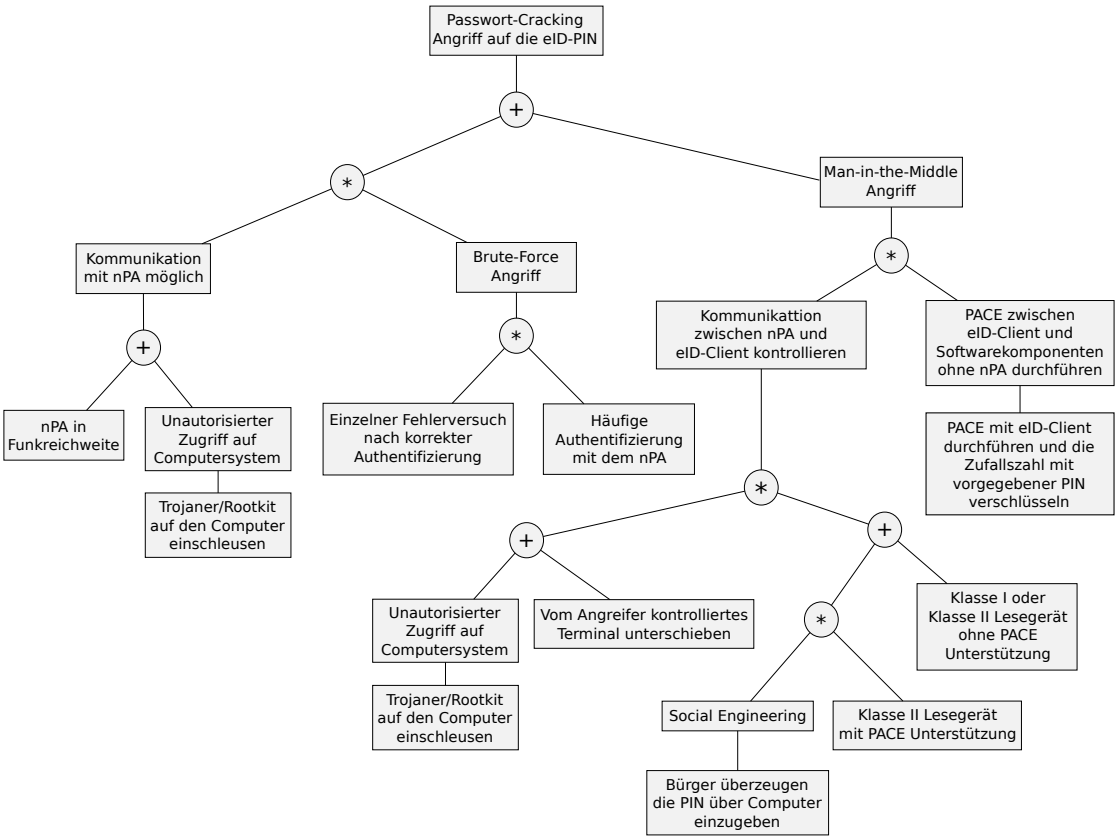

Abbildung 4.5.: Passwort-Cracking Angriffe auf die PIN

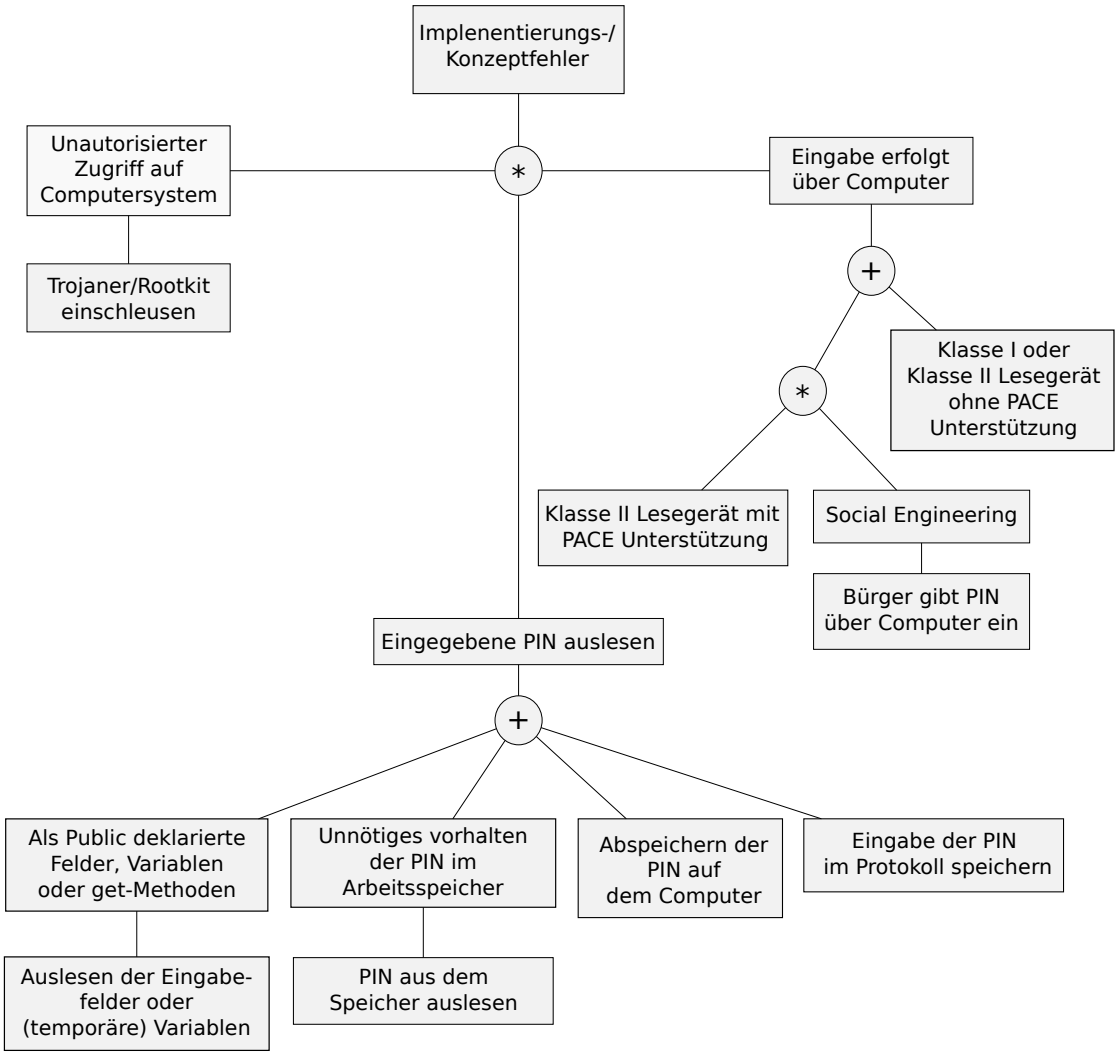

Abbildung 4.6.: Bedrohungen für eID-PIN/CAN von Implementierungs- oder Konzeptfehlern

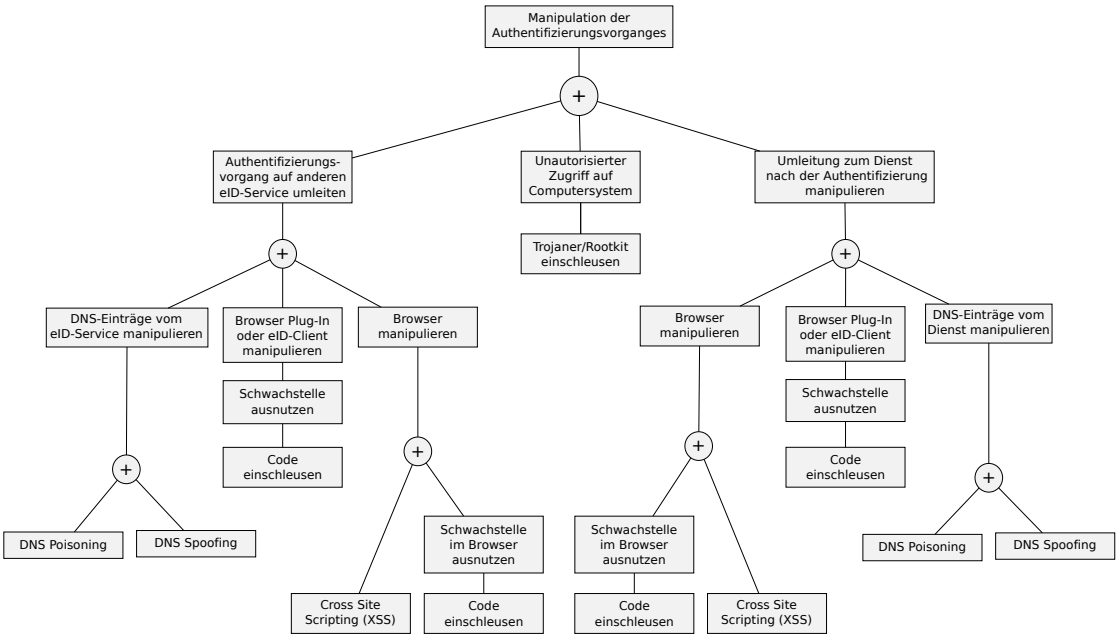

Abbildung 4.7.: Angriffsbaum für die Authentisierung mit dem nPA

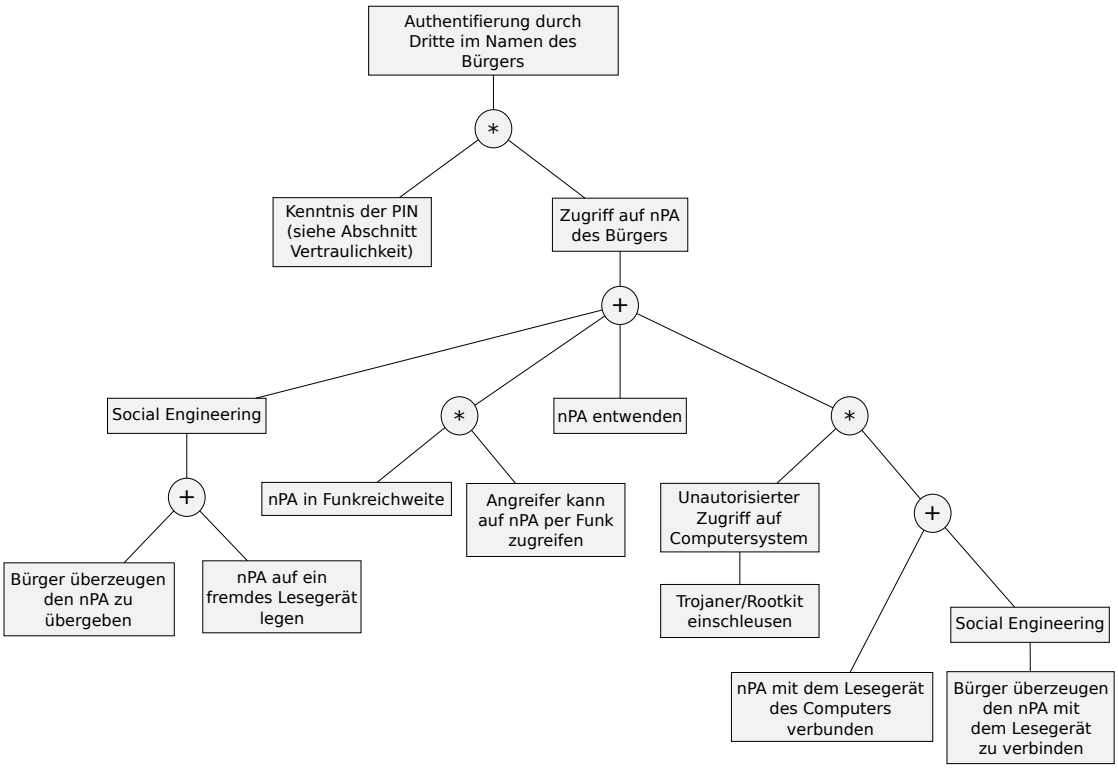

Abbildung 4.8.: Unberechtigte Authentifizierung durch Dritte im Namen des Bürgers

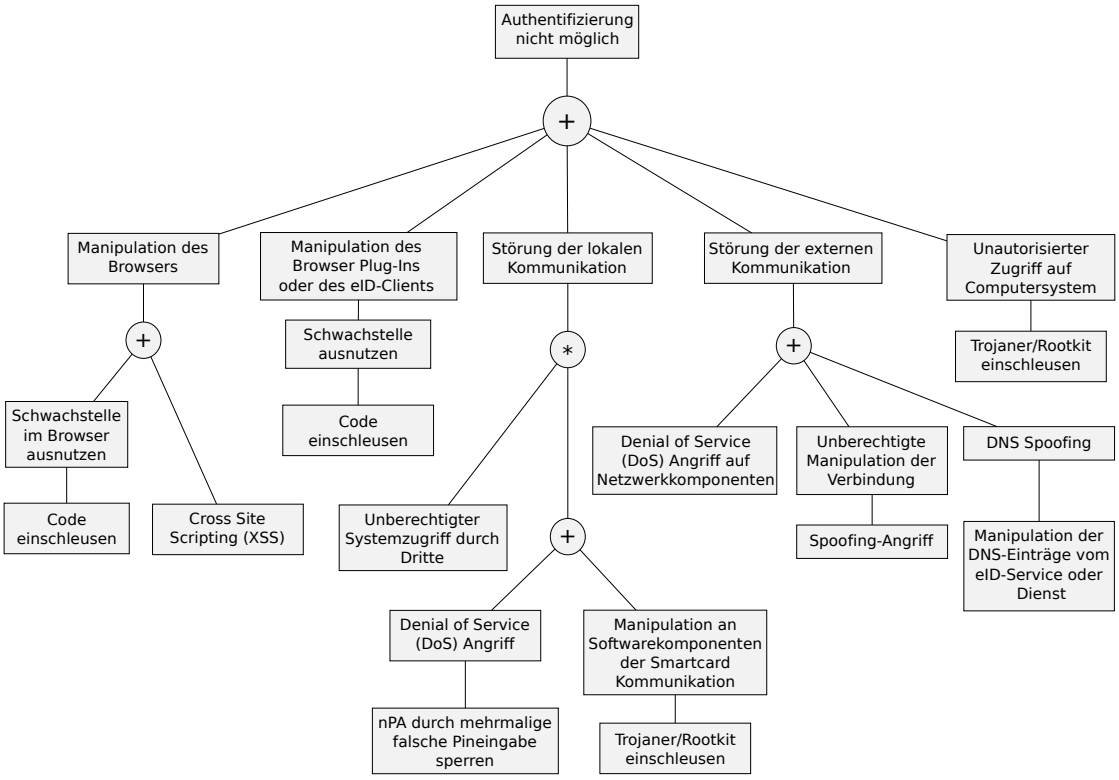

Abbildung 4.9.: Angriffsbaum für die Verfügbarkeit der Authentifizierung

Sicherheitsanforderungen: Datengruppen		
Sicherheitsziel	**Einstufung**	**Begründung**
Integrität	„normal“	Integrität (Abschnitt 4.2.4 & Abschnitt 4.3.5)
Vertraulichkeit	„normal“	Vertraulichkeit (Abschnitt 4.2.4 & Abschnitt 4.3.3)
Privatheit	„normal“	Privatheit (Abschnitt 4.2.4 & Abschnitt 4.3.3).
Nichtverknüpfbarkeit	„normal“	Nichtverknüpfbarkeit (Abschnitt 4.2.4).

Tabelle 4.13.: Sicherheitsanforderungen für Datengruppen des nPAs

4.4. Sicherheitsanforderungen

In den folgenden Abschnitten werden aus der Sicherheitsanalyse und der Bedrohungsanalyse Sicherheitsanforderungen abgeleitet.

4.4.1. Ableitungsregeln für Sicherheitsanforderungen

Die Sicherheitsanforderungen werden aus dem Schutzbedarf für das Schutzziel und den Bedrohungen für das Schutzziel abgeleitet. Dafür wird für jedes Schutzziel der Schutzbedarf der vorliegenden Bedrohung gegenüber gestellt. Daraus ergibt sich eine Sicherheitsanforderung mit einer Einstufung bestehend aus Schutzbedarf und Bedrohung. Diese richtet sich an den Stufen des Schutzbedarfskategorien (siehe Abschnitt 4.2.2).

4.4.2. Sicherheitsanforderungen ableiten

Die Sicherheitsanforderungen werden für die jeweiligen Schutzziele einzeln betrachtet. Jede Sicherheitsanforderung erhält eine Einstufung und eine Begründung.

Datengruppen auf dem nPA

Mit Hilfe von EAC (siehe Abschnitt 2.6) wird eine Ende-zu-Ende Sicherheit zwischen nPA und eID-Service aufgebaut. Danach hat außer dem eID-Service niemand mehr Zugriff auf die Datenübertragung. Das Berechtigungszertifikat beschränkt den Zugriff auf die Datengruppen des nPAs, in dem es vorschreibt, auf welche Datengruppen der eID-Service zugreifen darf. Aus diesen Gründen sind keine weiteren zusätzlichen Sicherheitsanforderungen für den Schutz der Datengruppen notwendig (siehe auch Tabelle 4.13).

Die Sicherheit der Datengruppen hängt vor allem von der Sicherheit der spezifizierten kryptographischen Protokolle und der Teilnehmer ab. Der eID-Service und der Dienst müssen sicherstellen, dass die Daten nicht von Dritten gestohlen werden. Der Bürger schützt die digital vorliegenden Datengruppen auf dem nPA durch die Geheimhaltung der eID-PIN. Daher sind keine zusätzlichen Sicherheitsanforderungen für den Schutz der Datengruppen notwendig.

eID-PIN und CAN

Die eID-PIN hat einen hohen Schutzbedarf und ist bei der Bedrohungsanalyse eine oft notwendige Voraussetzung für den erfolgreichen Angriff. Die daraus resultierenden Sicherheitsanforderungen sind in der Tabelle 4.14 aufgeführt.

Die CAN muss ebenfalls geschützt werden, weil sie die Verfolgung einer Person ermöglicht und zum Setzen der Signatur-PIN beim Beantragen einer QES benötigt wird. Durch das Beantragen einer QES entstehen Kosten für den Halter des Ausweises.

Authentisierung über die eID-Anwendung

Die Authentisierung über die eID-Anwendung wird zwischen eID-Service und nPA durchgeführt. Der eID-Client stellt lediglich Schnittstellen für die Kommunikation zwischen den Parteien zur Verfügung. Er hat keinen Zugriff auf die Datengruppen. Durch EAC (siehe Abschnitt 2.6) sind alle notwendigen Sicherheitsanforderungen aus Sicht der Software für den Authentisierungsprozess abgedeckt. Daher sind keine zusätzlichen Sicherheitsanforderungen notwendig (siehe auch Tabelle 4.15).

Lokale Kommunikation

Die lokale Kommunikation zwischen dem nPA und dem eID-Client wird über den als unsicher eingestuften Computer durchgeführt. Der eID-Client kann die Sicherheit der übertragenen Daten nicht signifikant beeinflussen. Der bereits vorhandene Schutz durch EAC ist ausreichend. Die Sicherheit der lokalen Kommunikation ist aus diesen Gründen als unkritisch zu bewerten, so lange EAC als sicher gilt (siehe auch Tabelle 4.16).

Sicherheitsanforderungen: eID-PIN und CAN		
Sicherheitsziel	**Einstufung**	**Begründung**
Vertraulichkeit	„hoch“	Vertraulichkeit (Abschnitt 4.2.4 und Abschnitt 4.3.3) Der Schutz der eID-PIN und der CAN hängt maßgeblich vom Bürger ab, da er für die Geheimhaltung der Beiden verantwortlich ist. Für sie gelten die gleichen Sicherheitsrichtlinien wie für die PIN der EC-Karte. Zusätzlich sollte die eID-PIN des nPAs in regelmäßigen Abständen geändert werden. Wie bereits in der Bedrohungsanalyse geschrieben, kann die Sicherheit des Computers nicht garantiert werden. Lediglich die konsequente Verwendung eines Standard- oder Komfort-Chipkartenlesers schützt die eID-PIN und die CAN vor Diebstahl. Manipulationen des Chipkartenlesers lassen sich nur schwer durchführen und rechtfertigen oft nicht den Aufwand für den Angriff. Mit Beachtung der angegeben Sicherheitsrichtlinien lässt sich der hohe Schutzbedarf der eID-PIN und der CAN bezüglich Vertraulichkeit erreichen. Ein softwareseitiger Schutz auf dem Computer des Bürgers wird die Sicherheit nicht signifikant steigern können.
Privatheit	„hoch“	Bei der eID-PIN und der CAN handelt es sich um private Informationen, die vor Dritten geschützt werden müssen. Deswegen gelten hier die gleichen Anforderungen, wie sie für die Vertraulichkeit notwendig sind.

Tabelle 4.14.: Sicherheitsanforderungen für die eID-PIN und die CAN

Sicherheitsanforderungen: Authentifizierung mittels eID-Anwendung		
Sicherheitsziel	**Einstufung**	**Begründung**
Authentizität	„normal“	Authentifizierung (Abschnitt 4.2.4 & Abschnitt 4.3.4)
Integrität	„normal“	Integrität (Abschnitt 4.2.4 & Abschnitt 4.3.5)
Vertraulichkeit	„normal“	Vertraulichkeit (Abschnitt 4.2.4 & Abschnitt 4.3.3)
Verfügbarkeit	„normal“	Verfügbarkeit (Abschnitt 4.2.4 & Abschnitt 4.3.7)
Verbindlichkeit	„hoch“	Verbindlichkeit (Abschnitt 4.2.4 & Abschnitt 4.3.6)
Privatheit	„hoch“	Privatheit (Abschnitt 4.2.4 & Abschnitt 4.3.6)
Nichtverknüpfbarkeit	„hoch“	Nichtverknüpfbarkeit (Abschnitt 4.2.4)
Pseudonymisierung	„hoch“	Pseudonymisierung (Abschnitt 4.2.4)

Tabelle 4.15.: Sicherheitsanforderungen für die Authentisierung

Sicherheitsanforderungen: Lokale Kommunikation		
Sicherheitsziel	**Einstufung**	**Begründung**
Integrität	„normal“	Integrität (Abschnitt 4.2.4 & Abschnitt 4.3.5)
Vertraulichkeit	„normal“	Vertraulichkeit (Abschnitt 4.2.4 & Abschnitt 4.3.3)
Verfügbarkeit	„normal“	Verfügbarkeit (Abschnitt 4.2.4 & Abschnitt 4.3.7)
Privatheit	„normal“	siehe Abschnitt 4.2.4 Privatheit
Nichtverknüpfbarkeit	„normal“	siehe Abschnitt 4.2.4 Nichtverknüpfbarkeit

Tabelle 4.16.: Sicherheitsanforderungen für die lokale Kommunikation

Sicherheitsanforderungen: Externe Kommunikation		
Sicherheitsziel	**Einstufung**	**Begründung**
Integrität	„normal“	siehe Integrität in Abschnitt 4.2.4 und Abschnitt 4.3.5 Die Integrität der Kommunikationsverbindungen mit einem eID-Service kann erhöht werden, indem beispielsweise DNS-Sec (siehe Ateniese [AM01]) verwendet wird.
Vertraulichkeit	„normal“	siehe Vertraulichkeit in Abschnitt 4.2.4 und Abschnitt 4.3.3
Verfügbarkeit	„normal“	siehe Verfügbarkeit in Abschnitt 4.2.4 und Abschnitt 4.3.7
Privatheit	„normal“	siehe Abschnitt 4.2.4 Privatheit
Nichtverknüpfbarkeit	„normal“	siehe Abschnitt 4.2.4 Nichtverknüpfbarkeit

Tabelle 4.17.: Sicherheitsanforderungen für die externe Kommunikation

Externe Kommunikation

Wie in der Bedrohungsanalyse (siehe Abschnitt 4.3) zu sehen ist, kann kein physikalischer Schutz für die externen Kommunikationsverbindungen aufgebaut werden. Dieser muss durch TA und CA von EAC (siehe Abschnitt 2.6) sicher gestellt werden. Aus diesem Grund sind keine zusätzlichen Sicherheitsmaßnahmen für die externe Kommunikation notwendig (siehe auch Tabelle 4.17). Auch der eID-Client kann keinen signifikanten Einfluss auf die Sicherheit der externen Verbindung nehmen.

4.5. Das Java Sicherheitsmodell

Die Programmiersprache Java wird auf der Java Virtual Machine (JVM) ausgeführt. Die JVM wird mit der Java Runtime Environment (JRE) ausgeliefert, welche weitere, für die Laufzeitumgebung notwendigen Komponenten enthält. Für die Entwicklung von Java Software wird das Java Development Kit (JDK) benötigt. Dieses enthält einen Compiler, Werkzeuge und Klassen, um Anwendungen für die JVM zu erstellen. Heute spielt die JVM nicht nur im Zusammenhang mit Java eine wichtige Rolle, weil viele weitere Programmiersprachen sie als Basis benutzen.

Im Folgenden wurden die JRE, JVM und den JDK unter der JVM zusammengefasst, da der compilierte Bytecode auf der JVM ausgeführt wird und sie die Laufzeitumgebung für Anwendungen bereit stellt. Der ByteCode wird im Folgenden nur als Code bezeichnet, da die JVM nur für sie compilierten ByteCode ausführen kann.

Die Java Sicherheitsarchitektur (siehe Java Security Architecture von Li Gong [Gon98a]) hat sich mit der Entwicklung der JVM und des JDK stark verändert. Das ist vor allem auf die gestiegenen Anforderungen und die starke Weiterentwicklung der Java Plattform zurückzuführen.

Die Laufzeitumgebung der JVM ermöglicht es auf Netzwerke, lokale Ressourcen wie das Dateisystem und auf Geräte zuzugreifen. Diese Ressourcen stellen zu schützende Werte dar. Daher werden Schutzmechanismen benötigt, die den Zugriff auf die verschiedenen Ressourcen steuern.

Im Folgenden werden die Schutzmechanismen über die Entwicklungsversionen der JVM betrachtet, um Entscheidungen für deren Implementierung verständlicher zu machen.

4.5.1. JVM 1.0

Mit der Veröffentlichung der JVM Version 1.0 im Jahre 1996 wurde ein einfaches Sicherheitsmodell für lokalen und extern geladenen Code (auch Remote Code genannt) eingeführt. Lokaler Code wird ohne weitere Schutzmechanismen direkt auf der JVM ausgeführt und kann auf alle Funktionen der JVM und damit auch auf alle verfügbaren Ressourcen zugreifen. Lokale Anwendungen galten zu der Zeit als vertrauenswürdig und erforderten somit keine speziellen Schutzmechanismen.

Aus dem Quellcode wird mit Hilfe des Compilers JVM-kompatibler Bytecode erstellt. Der Bytecode Verifier stellt vor jedem Start sicher, dass nur legitimer Java Bytecode ausgeführt wird. Zusammen garantieren sie auch, dass zur Laufzeit die Sprachsicherheit

gewährleistet ist, wie es in „Java Security Architecture“ von Li Gong [Gon98a] beschrieben ist.

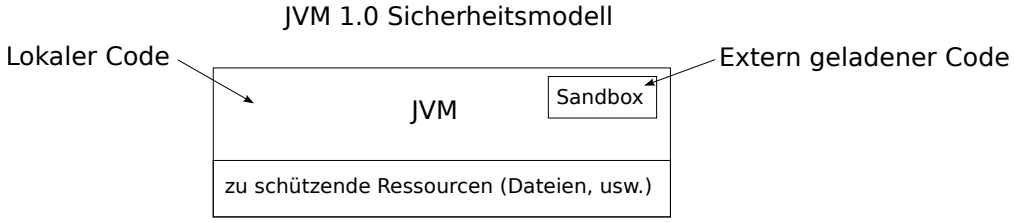

Abbildung 4.10.: JVM 1.0 Sicherheitsmodell (Quelle: Java Security Architecture von Li Gong [Gon98a])

Externer Code (auch Remote Code genannt) wird in einer Sandbox[2] ausgeführt, welche den Zugriff auf Funktionen der JVM einschränkt und den Zugriff auf lokale Ressourcen wie das Dateisystem vollständig blockiert. Damit kann der externe Code ohne Warnungen oder Abfragen direkt ausgeführt werden. Der Zugriff auf Netzwerke inklusive dem Internet ist gesperrt. Die Anwendung darf lediglich mit der Quelle kommunizieren, von welcher der externe Code geladen wurde (Point of Origin). Damit kann die Anwendung in der Sandbox nicht mit fremden Systemen kommunizieren, da sie in einer Sandbox ausgeführt wird. Unter externen Code fallen vor allem Applets, welche über den Browser von einer externen Quelle geladen und anschließend auf dem lokalen Computer ausgeführt werden.

Dieses einfache Sicherheitsmodell (siehe auch Abbildung 4.10) erlaubte es zwar, Anwendungen ohne Warnungen in der Sandbox auszuführen, aber einfache Aufgaben wie den Ausdruck von Dokumenten ist bei extern geladenem Code in der JVM Version 1.0 nicht möglich.

4.5.2. JVM 1.1

Die Einschränkung extern geladenen Code nur in der Sandbox auszuführen, wurde im Jahre 1997 mit Version 1.1 der JVM aufgehoben. Extern geladener Code ohne eine Signatur wird weiterhin nur in der Sandbox ausgeführt und entspricht den Einschränkungen der JVM 1.0. Signierter extern geladener Code wird wie lokaler Code direkt auf der JVM ohne weitere Schutzmechanismen ausgeführt, wenn der Benutzer das Vertrauen gegenüber dem Autor bestätigt (siehe auch Abbildung 4.11).

Der Grundgedanke hinter signiertem Code ist die Möglichkeit, die Herkunft des Codes nachzuweisen und damit auch einen Verantwortlichen für eventuell entstandene Schäden

[2]Eine Sandbox ist nach Eckert [Eck09] eine Technik die gewährleistet, dass ein Code-Modul nur auf Adressen eines festgelegten Bereichs (Sandbox) zugreift.

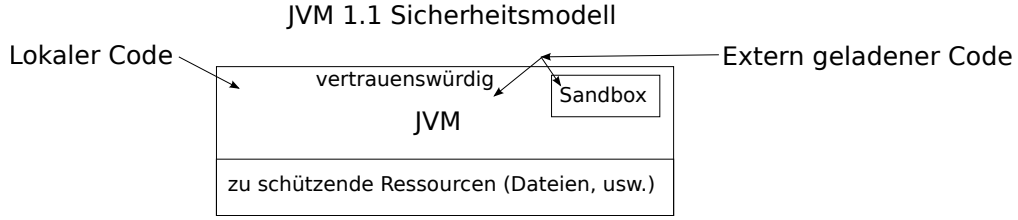

Abbildung 4.11.: JVM 1.1 Sicherheitsmodell (Quelle: Java Security Architecture von Li Gong [Gon98a])

zu bestimmen. Zudem muss der Benutzer vor der ersten Ausführung der Anwendung explizit das Vertrauen gegenüber dem Autor des externen Codes bestätigen, sonst kann der Code nur in der Sandbox ausgeführt werden.

4.5.3. JVM 1.2

Die Trennung zwischen lokal und extern geladenem Code wurde im Jahr 1998 in Java Version 1.2 aufgehoben. Das implizit definierte Vertrauen gegenüber lokalen Code gilt seitdem nicht mehr. Das neu eingeführte erweiterte Sicherheitsmodell stellt eine zentrale Prüfung für jeglichen Code zur Verfügung. Der Code muss diverse Überprüfungen durchlaufen, bevor er ausgeführt wird. Die Sicherheit der Java Plattform baut auf dem Class Loader, den Security Policies und dem Security Manager auf (siehe auch Abbildung 4.12), die im Folgenden im Detail erläutert werden.

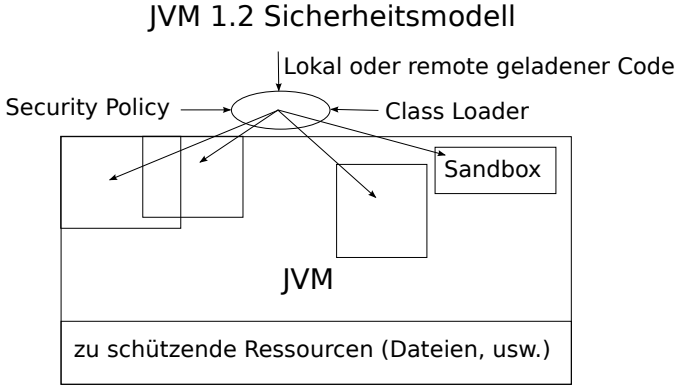

Abbildung 4.12.: JVM 1.2 Sicherheitsmodell (Quelle: Java Security Architecture von Li Gong [Gon98a])

Security Policies

Security Policies beschreiben Zugriffsregeln auf Klassen und deren Funktionen. Sie können sehr fein granuliert werden. Die Zugriffsregeln werden in Security Policy Dateien beschrieben und in der JVM beim Starten der Anwendung geladen.

Bei jeder Java Runtime Environment (JRE) existiert eine vordefinierte *java.policy* Datei, welche in *$JAVA_HOME/lib/security/* zu finden ist. Sie beschreibt grundsätzliche Sicherheitsregeln, die für alle Benutzer des Computers gelten. Jeder Benutzer kann eigene Regeln definieren, die in seinem Home-Verzeichnis unter */.java.policy* gespeichert werden. Weiterhin besteht die Möglichkeit, bei jedem Aufruf der Anwendung weitere Security Policies durch den Parameter *„-Djava.security.policy=...“* an die JVM zu übergeben. Die Klassen können in Sicherheitsdomains (Security Domains) zusammengefasst werden, um die Handhabung zu vereinfachen (siehe Abbildung 4.13).

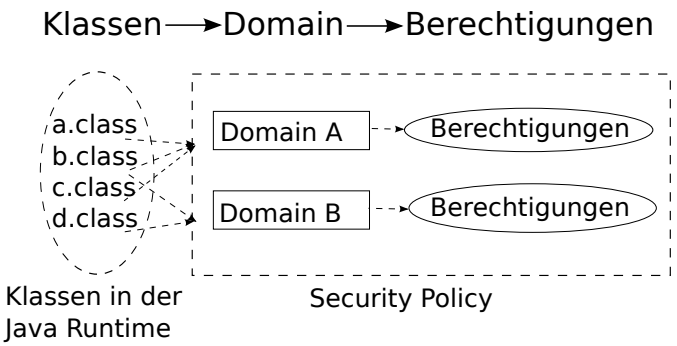

Abbildung 4.13.: Java Security Domains (Quelle: Java Security Architecture von Li Gong [Gon98a])

Ein Beispiel für eine solche Regel sieht wie folgt aus:

```
1 grant {
2   permission java.io.FilePermission "/opt/someFile.txt",
3                                                   "read";
4  };
```

Die Security Policies legen Zugriffsrechte für die Anwendung fest, bevor diese ausgeführt wird. Die Durchsetzung der Zugriffsregeln erfolgt durch den Class Loader und den Security Manager, welche die Einhaltung der Regeln garantieren. Bei einer Zugriffsverletzung wird eine *Security Exception* geworfen.

Class Loader

Der Class Loader lädt Klassen, die in einer Anwendung verwendet werden. Dieser ist nach dem Verifier (siehe Abschnitt 4.5.1) das zweite Element in der Sicherheitskette der JVM. Er sorgt dafür, dass nur in den Security Policies zugelassene Klassen in den aktuellen Kontext der Anwendung geladen werden (siehe auch Java Security Architecture von Li Gong [Gon98b]). Dadurch wird einem Applet in der Sandbox der Zugriff auf das Dateisystem versperrt, weil es nicht die dafür notwendigen Klassen laden kann. Auch kann das Applet nicht über eigene Klassen auf lokale oder externe Ressourcen zugreifen, weil das nur über die Klassen des JDK erlaubt ist (siehe Abbildung 4.14).

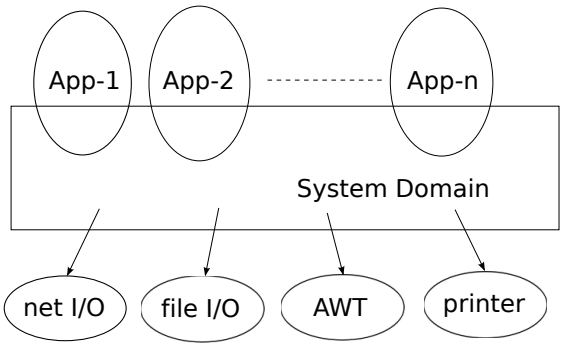

Abbildung 4.14.: Java Ressourcen Sicherheitsmodell (Quelle: Java Security Architecture von Li Gong [Gon98a])

Mit dem JDK kann der Entwickler eigene Class Loader für seine Zwecke erstellen, um beispielsweise Programmiersprachen wie Groovy oder Ruby zu unterstützen. Damit wäre es auch bei Applets möglich, eigene Klassen nachzuladen, um Sicherheitsmaßnahmen der JVM zu umgehen. Aus diesem Grund sind eigene Class Loader in der Sandbox nicht erlaubt.

Damit Anwendungen in der Sandbox nicht auf andere laufende Anwendungen zugreifen oder mit ihnen kommunizieren können, stellt der Classloader für jede Anwendung ein lokales Namespace zur Verfügung. Dieser gilt ausschließlich für diese eine Anwendung und versperrt somit die lokale Kommunikation, wie sie beispielsweise durch statische Klassen innerhalb der JVM möglich wäre.

Unabhängig von der Herkunft wird nicht signierter Code weiterhin nur in der Sandbox ausgeführt. Damit wird sichergestellt, dass über andere Quellen geladener Code aus fremder Quelle nur in der schützenden Sandbox ausgeführt wird. Diese Einschränkung kann allerdings durch Security Policies vom Benutzer aufgehoben werden.

Weitere Informationen zu dem Java Sicherheitsmodell können in „Java Security Architecture (JDK1.2)“ von Li Gong [Gon98a] nachgelesen werden.

4.6. Sicherheitsvergleich von Anwendungen und Java Applets

Im folgenden Abschnitt werden installierbare Anwendungen und Java Applets gegenüber gestellt. Zuerst werden die Unterschiede zwischen den beiden Anwendungsarten gezeigt und die dahinter stehenden Sicherheitsarchitekturen verglichen. Dieser Vergleich soll eine Basis für verschiedene Implementierungsarten des eID-Clients dienen. Am Ende soll eine Bewertung des Vergleichs zwischen einer Java Applet Implementierung und einer installierbaren Anwendung erstellt werden.

4.6.1. Allgemeiner Vergleich zwischen Anwendungen und Java Applets

In den beiden folgenden Abschnitten werden die Unterschiede sowie Gemeinsamkeiten zwischen einer installierbaren Anwendung und einem Java Applet beschrieben.

Anwendungen

Anwendungen werden meistens von einer externen Quelle (Internet) oder von einem Datenträger (CD oder DVD) auf den Computer übertragen. Über eine mitgelieferte Installationsroutine wird die Anwendung auf dem Computer installiert. Für die Installation einer solchen Anwendung benötigt der Benutzer des Computers in der Regel Administratorrechte, sonst kann er die Installation nicht durchführen.

Nach der Installation steht die Anwendung allen Benutzern des Computers zur Verfügung. Alle Sicherheitswarnungen und das Bestätigen des Vertrauens gegenüber dem Autor der Anwendung erfolgen beim Start der Installation (bei Windows, Linux, ...) oder beim ersten Ausführen (bei MacOS) der Anwendung. Dieses Vorgehen gilt für die meisten gängigen Betriebssysteme.

Die Installationsroutine der Anwendung muss alle notwendigen Komponenten für deren Betrieb mitbringen. Nach der Installation kann die Anwendung ohne weitere Unterbrechungen verwendet werden.

Installierbare Anwendungen werden im Weiteren nur als Anwendungen bezeichnet.

Java Applet

Java Applets sind Java Anwendungen, welche direkt aus dem Browser auf dem Computer ausgeführt werden. Der so genannte Externe Code (auch Remote Code genannt) wird von einer externen Quelle geladen und auf dem Computer lokal ausgeführt.

Um eine Java Applet ausführen zu können, muss auf dem System die Java Runtime Environment (JRE) installiert sein. Laut den genannten Quellen ist die Java JRE bei etwa 80% (QUELLE: [Gro], [Sta]) der heutigen Computer bereits installiert. Diese enthält das für viele Browser notwendige Plugin, um Java Applets direkt aus dem Browser auszuführen. Viele Browser auf heutigen Desktopsystemen haben das Java Browser Plugin installiert und können dementsprechend mit Applets umgehen.

Die grundsätzliche Unterscheidung zwischen signiertem und nicht signiertem Code (siehe Abschnitt 4.5) wird durch das Java Browser Plugin gesteuert. Benötigt das Applet keinen Zugriff auf lokale Ressourcen, so muss es nicht signiert werden. Damit kann es ohne Warnungen direkt im Browser ausgeführt werden.

Muss das Applet auf lokale Ressourcen wie das Dateisystem oder den Drucker zugreifen, so muss der Autor das Applet signieren. Mit Hilfe der Signatur kann der Anwender den Autor des Applets bestimmen. Bestätigt der Benutzer das Vertrauen gegenüber dem Autor, so wird die Anwendung außerhalb der Sandbox ausgeführt und kann dementsprechend auf lokale Ressourcen zugreifen. Das Vertrauen gegenüber dem Autor für dieses Applet bleibt in der Regel bis zur nächsten Aktualisierung der Anwendung erhalten.

4.6.2. Vor- und Nachteile installierbarer Anwendungen

Die folgenden Auflistungen beschreiben Vor- und Nachteile einer zu installierenden Anwendung.

Vorteile

- Mit der Installation werden alle notwendigen Voraussetzungen wie Bibliotheken oder Laufzeitumgebungen (zum Beispiel JRE) installiert.
- Die Anwendung muss nur ein einziges Mal auf den Computer übertragen und installiert werden. Danach steht sie allen Benutzern des Computers zur Verfügung.

- Das Look & Feel der Plattform kann durch eine plattformspezifische Anwendung vollständig realisiert werden, was bei einem Applet nicht immer möglichst ist. Durch die Anpassung an die Plattform werden auch die gewohnten Bedienkonzepte übernommen. Mit diesen kommt der Benutzer in der Regel besser zurecht.

- Die Anwendung wird immer vom lokalen System ausgeführt und weist somit auf dem lokalen Computer des Bürgers ein höheres Vertrauen auf, als extern geladener Code.

Nachteile

- Die Anwendung muss für jedes Betriebssystem eine eigene Aktualisierungstechnik implementieren. Die dadurch entstehenden Zusatzrisiken müssen dementsprechend beachtet werden (siehe Update Problem mit ZIP-Dateien beim BürgerClient [Scha]).

- Die Installation sowie das Aktualisieren der Anwendung erfordert in der Regel Administratorrechte auf dem Computer, welche nicht jeder Benutzer auf seinem Computer hat.

- Die Anwendungen können durch Dritte manipuliert werden, da diese nicht zwingend signiert sein müssen und nach der ersten Zustimmung in der Regel nicht mehr auf Manipulationen geprüft wird.

4.6.3. Vor- und Nachteile von Java Applets

Die folgenden Punkte zeigen Vor- und Nachteile von Java Applets. Hier wird vor allem auf Java Applets eingegangen, wobei viele dieser Punkte auch für andere Techniken mit extern geladenem Code gelten.

Vorteile

- Das Applet muss nicht auf dem Computer installiert werden.

- Für Applets werden keine Administratorrechte benötigt. Ein Benutzer mit eingeschränkten Rechten kann in der Regel ein Applet ausführen.

- Applets aktualisieren sich automatisch ohne Benutzereingriff. Damit werden wichtige Änderungen wie Sicherheitsaktualisierungen automatisch eingespielt.

- Für das Aktualisieren des Applets sind keine Administratorrechte notwendig. Ein Benutzer mit eingeschränkten Rechten kann die neue Version sofort verwenden.
- Um die Sicherheit des Aktualisierungsprozesses kümmert sich der Browser-Plugin Hersteller oder der Browserhersteller.
- Applets können aus vielen Browsern heraus ausgeführt werden.
- Das Browser Plugin muss vom Entwickler nicht installiert, gepflegt und für neue Browserversionen angepasst werden. Diese Aufgabe wird durch den JVM-Anbieter oder den Browserhersteller durchgeführt.
- Sicherheitsaktualisierungen für das Browser Plugin und die JRE wird vom JVM-Anbieter bereitgestellt.
- Anwendungen ohne benötigten Zugriff auf lokale Ressourcen können direkt in der Sandbox ausgeführt werden (siehe Abschnitt 4.5.1).
- Signierte Applets können nicht vor der eigentlichen Ausführung manipuliert werden, da die JVM die Signatur vorab überprüft.
- Der Benutzer muss keine Entscheidungen treffen, wie sie bei einer Installationsroutine (beispielsweise Speicherort, mitinstallierte Komponenten) notwendig sind.
- Applets werden nur ausgeführt, wenn sie benötigt werden. Sie stellen in der Regel keine Dienste zur Verfügung und werden auch nicht im Hintergrund ausgeführt.

Nachteile

- Für das Ausführen von Java Applets muss die JRE und das notwendige Browser Plugin auf dem Computer bereits installiert sein.
- Die JVM wird nicht von jedem Betriebssystem unterstützt.
- Java Applets werden nicht von jedem Browser unterstützt, da nicht für jeden Browser ein entsprechendes Plugin existiert.
- Java Applets werden von vielen Smartphones nicht unterstützt (Android, IPhone, Windows Phone 7 OS, Palm WebOS)
- Das Java Applet muss vor der ersten Ausführung heruntergeladen werden. Wird der Cache der JVM gelöscht, so muss dieser Schritt wiederholt werden.

- Java Applets sind in Unternehmen nicht immer erwünscht, weil sie schlecht kontrolliert werden können (siehe auch [JSF96]).

- Applets werden beim Start automatisch aktualisiert. Das bedeutet, der Benutzer muss beim Start des Applets auf das Ende der Aktualisierung warten. Diese kann nicht auf später verschoben oder abgebrochen werden, um mit dem Programm arbeiten zu können.

- Das Applet muss pro Benutzer des Computers runtergeladen werden und jeder Benutzer muss für sich das Vertrauen gegenüber dem Autor bestätigen.

4.7. eID-Client als Anwendung und Java Applet

Für die Authentisierung am Dienstanbieter muss der eID-Client auf das System übertragen und gegebenenfalls installiert werden. Das ist für die zu untersuchenden Anwendungsarten grundsätzlich unterschiedlich und bedarf einer näheren Betrachtung.

4.7.1. eID-Client als Anwendung

Der Authentisierungsprozess wird über den Browser gestartet. Das bedeutet der verwendete Browser benötigt ein entsprechendes Plugin, um mit dem lokal installierten eID-Client kommunizieren zu können.

Der Ablauf der Authentisierung über eine vorinstallierte Anwendung entspricht der allgemeinen Prozessbeschreibung (siehe Abschnitt 4.1.3).

4.7.2. eID-Client als Java Applet

Die Authentisierung über den eID-Client als Java Applet verläuft fast identisch zu einer Lösung als installierbare Anwendung. Im Unterschied zu der Anwendung muss das Applet vor der ersten Verwendung oder bei einer Aktualisierung von der Seite des Dienstanbieters oder des eID-Service Betreibers heruntergeladen werden.

Der eID-Client muss über den Computer und den Chipkartenleser auf den nPA und das Netzwerk zugreifen können. Aus diesem Grund kann die Anwendung nicht in der Sandbox ausgeführt werden. Das bedeutet, das Applet muss von einer vertrauenswürdigen Instanz signiert und vom Benutzer bei der ersten Ausführung als vertrauenswürdig

bestätigt werden. Beim erneuten Aufruf ist keine erneute Bestätigung notwendig, solange keine neuere Version des Applets vorliegt.

Nachdem das Vertrauen gegenüber dem Autor der Anwendung bestätigt wurde, kann die Authentisierung nach der allgemeinen Prozessbeschreibung vom Abschnitt 4.1.3 erfolgen. Damit existieren beim Applet nur geringe Abweichungen gegenüber einer Anwendung.

4.7.3. Vor- und Nachteile eines eID-Clients als Anwendung

Vorteile

- Die aktuelle Version des eID-Clients vom BMI (AusweisApp) kann beim Einwohnermeldeamt auf CD abgeholt werden. Diese Quelle wird als vertrauenswürdig eingestuft.
- Für die Authentisierung wird immer der gleiche eID-Client mit dem gleichen Aussehen verwendet. Weitere Warnhinweise wie sie beim Applet erscheinen können, bleiben dem Bürger erspart.

Nachteile

- Die Anwendung muss alle verwendeten Kommunikationsprotokolle und deren Versionen unterstützen, da Sie mit jedem Service funktionieren muss.
- Für jeden zu unterstützenden Browser muss ein eigenes Plugin entwickelt und gepflegt werden.
- Für die Installation und Aktualisierung der Anwendung werden Administratorrechte benötigt.

4.7.4. Vor- und Nachteile eines eID-Clients als Java Applet

Vorteile

- Für den Browser muss kein Plugin entwickelt, installiert und aktualisiert werden. Das wird durch den JVM-Anbieter durchgeführt.

- Der eID-Client bedarf an keiner Stelle Administratorrechte, da keine Installation stattfinden muss.

- Der eID-Client als Applet funktioniert mit fast jedem Gerät, welches Java Applets ausführt und mit einem passenden Chipkartenleser kommunizieren kann.

- Der eID-Client wird von der Seite des Dienstanbieters oder des eID-Service Betreibers bereit gestellt und garantiert damit die beste Kompatibilität mit dem jeweiligen Service.

- Der eID-Client kann auch von einer dritten vertrauenswürdigen Instanz wie dem BSI signiert und zur Verfügung gestellt werden.

- Erweiterungen für weitere Chipkarten (z.B. österreichische Bürgerkarte) können in das Applet integriert werden, ohne sich von einem Softwareprodukt oder Anbieter abhängig zu machen.

- Der Dienstanbieter oder eID-Service-Betreiber kann das Applet für seine Bedürfnisse (beispielsweise Look & Feel) anpassen.

- Es ist keine zwingende Unterstützung der eCard-API (siehe BSI TR-03112 [BSI08e]) notwendig. Das Applet muss nur mit ausgewählten Diensten funktionieren, wenn es nicht für alle Dienste bestimmt ist.

- Es ist keine Versions- und Kompatibilitätspflege notwendig, da das Applet nur mit der aktuellen Version der Serversoftware funktionieren muss.

Nachteile

- Der Bürger muss bei der ersten Ausführung des Applets das Vertrauen gegenüber dem Autor bestätigen. Das wird bei vielen Diensten der Fall sein, so dass die Bürger ohne Aufmerksamkeit die Warnungen bestätigen werden, weil sie an der Stelle eine Warnung erwarten und diese immer wieder bestätigen müssen. Zudem kann ein Bürger ohne tiefergehendes Wissen in IT-Sicherheit nur schlecht entscheiden, ob er dem Autor der Software und der Signatur vertrauen kann.

- Bei Applets kann kein einheitliches Aussehen und kein einheitliches Bedienkonzepte garantiert werden, da eine Anpassung des eID-Clients durch den eID-Service Betreiber oder Dienstanbieter möglich ist.

4.7.5. Benutzerfreundlichkeit

Die Anforderungen an die Benutzerfreundlichkeit der Software beinhalten auch sicherheitskritische Aspekte. An dieser Stelle werden Anforderungen bezüglich Benutzerfreundlichkeit mit dem Schwerpunkt auf Sicherheit betrachtet. Dabei werden die besonderen Anforderungen eines eID-Clients berücksichtigt.

Die Authentisierung ist aus Sicht des Bürgers der kritischste Teil. Der eID-Client ist das visuelle Werkzeug für den Bürger. Er stellt den Authentisierungsprozess visuell dar und hilft dem Bürger Entscheidungen zu treffen. Der Vorgang sollte möglichst einheitlich dargestellt werden und sich in gewohnten Abläufen widerspiegeln. Ein einheitliches Aussehen und Bedienkonzept über alle Betriebssysteme hinweg würde die Anwendung für den Bürger vertrauter machen, aber die Vorteile der jeweiligen Plattform würden dabei verloren gehen.

Die plattformspezifische Anpassung kann aus Sicht des Prozesses (siehe Abschnitt 4.1.3) vernachlässigt werden, da die einzelnen Schritte für alle Systeme allgemein genug dargestellt werden können.

Anwendung

Der eID-Client vom BMI ist eine Anwendung auf Basis von Java. Er besitzt ein für alle Betriebssysteme einheitliches Aussehen und Bedienkonzept. Dadurch kann ihn jeder Bürger unabhängig vom Betriebssystem bedienen.

Das einheitliche Aussehen und die einheitlichen Bedienkonzepte bauen Vertrauen beim Bürger auf, wenn auch ein ihm fremdes Betriebssystem verwendet wird.

Java Applet

Durch Java Applets kann jeder eID-Service Betreiber und Dienstanbieter einen eigenen eID-Client ausliefern. Daher kann dieser visuell und vom Bedienkonzept an die Webanwendung angepasst werden. Allerdings wird dem Bürger dadurch nicht sofort ersichtlich, dass es sich um die legitime und damit eine ihm vertrauenswürdige Anwendung handelt.

Aus diesen Gründen sollte die Anpassung an das Aussehen des Dienstanbieters oder eID-Service Betreibers nicht durchgeführt werden, um kein unnötiges Misstrauen beim Bürger auszulösen.

4.7.6. Verletzung der Privatheit durch Profilbildung

Die Privatheit des Bürgers muss auch durch den eID-Client gewahrt bleiben. Diese kann zwar nicht auf die Datengruppen des nPAs zugreifen, aber er kann Informationen über verwendete Dienste und eID-Services sammeln. Damit wäre er in der Lage, ein Profil des Bürgers zu erstellen und an Dritte zu übermitteln.

Wird aber für jeden Dienst eine anderer eID-Client (in Form von einem Java Applet) verwendet, so wäre er nur in der Lage, die Anzahl der Authentisierungsvorgänge zu protokollieren, könnte aber nicht ermitteln, welche anderen Diensten der Bürger verwendet.

4.7.7. Distribution

Die verschiedenen Implementierungen des eID-Clients erfordern verschiedene Wege der Distribution und technisch unterschiedliche Aktualisierungsprozesse. Sie werden in den folgenden Abschnitten aus Sicht der beiden Implementierungsmöglichkeiten betrachtet.

Anwendung

Der eID-Client als Anwendung bedarf einen höheren Entwicklungsaufwand, da er für jeden zu unterstützenden Browser ein eigenes Plugin benötigt. Dieses Plugin muss über Versionen der Browser hinweg gepflegt werden. Durch die schnellen und vielen Änderungen bei Browsern und mit der steigenden Anzahl an Browsern wird der Pflegeaufwand sehr hoch sein.

Weiterhin bestehen Probleme mit Kompatibilität der JVM-Versionen. Aus diesem Grund liefert der eID-Client vom BMI eine eigene JVM mit sich. Damit kann er bei der Ausführung davon ausgehen, dass die richtige JVM Version verwendet wird. Das hat allerdings den großen Nachteil, dass bei einer sicherheitskritischen Aktualisierung der JVM auch der eID-Client aktualisiert werden muss, um den Computer des Benutzers nicht zusätzlich zu gefährden.

Abhängig vom Betriebssystem und deren Version muss der eID-Client ein eigene Aktualisierungstechnik mit sich mitbringen. Dieses stellt ein weiteres zusätzliches Risiko dar, welches nicht unterschätzt werden darf (siehe Update Problem mit ZIP-Dateien beim BürgerClient [Scha]).

Der eID-Client vom BMI fungiert als ein eigener Webservice, der immer auf dem Computer des Bürgers läuft. Damit haben alle anderen Anwendungen mit lokalen Zugriff auf den Computer die Möglichkeit die Dienste des eID-Clients zu benutzen. Sollte der eID-Client eine Sicherheitslücke im Webservice enthalten, so wäre diese ständig geöffnet und damit durch viele Anwendungen angreifbar.

Java Applet

Der eID-Client als Java Applet benötigt bestimmte Voraussetzungen, bevor es ausgeführt werden kann. Dazu zählt unter anderem ein vorinstalliertes Browser Plugin für die JVM und die richtige Version der JVM. Das Applet kann nicht selbstständig entscheiden, unter welcher vorinstallierten JVM-Version es ausgeführt wird. Die dadurch entstehenden Kompatibilitätsprobleme sind dem Benutzer nur sehr schwer zu vermitteln und können von einem Leihen nur schwer gelöst werden, besonders wenn andere Anwendungen oder Applets eine andere JVM-Version erwarten. Allerdings betrifft dieses Problem alle Java Anwendungen und Applets. Eine Lösung wie bei dem eID-Client vom BMI mit einer eigenen JVM ist für ein Java Applet nicht möglich.

Der große Vorteil des eID-Clients als Applet ist die Distribution und die Aktualisierung über vordefinierte Mechanismen. Die Aktualisierung der JVM wird durch den JVM-Hersteller sichergestellt. Weiterhin wird die Pflege des Browser Plugins durch den JVM-Hersteller oder den Browserhersteller durchgeführt. Damit entfallen viele sicherheitskritische Aufgaben für die Entwickler des eID-Clients. Die große Verbreitung der JVM bietet die Integration in fast jeden gängigen Browser. Die JVM stellt sicher, dass die Aktualisierungen sicher durchgeführt werden und nimmt diese Fehlerquelle dem Applet-Autor ab.

4.8. Bewertung des Sicherheitsvergleichs

Aus der Strukturanalyse (siehe Abschnitt 4.1) hat sich ergeben, dass am Prozess mehrere Parteien beteiligt sind. Diese tragen verschiedene Verantwortungen. In diesem Fall wurde der Authentisierungsprozess aus Sicht des Bürgers untersucht. Dieser beinhaltet externe Parteien und lokale Komponenten wie den Computer und den Chipkartenleser. Zusätzlich muss die externe und die lokale Kommunikation in die Sicherheitsanalyse einbezogen werden.

Die Sicherheitsanalyse (siehe Abschnitt 4.2) des Authentisierungsprozesses hat mehrere Schutzziele identifiziert, die in der anschließenden Bedrohungsanalyse (siehe Abschnitt 4.3) untersucht wurden. Daraus wurden Sicherheitsanforderungen (siehe Abschnitt 4.4) für den Authentisierungsprozess abgeleitet. Die Analyse hat ergeben, dass die komplette

Kommunikation zwischen den Parteien durch EAC ausreichend geschützt ist. Der eID-Client als zentrale Komponenten zwischen allen Parteien hat nur geringen Einfluss auf die Sicherheit des ganzen Prozesses. Sie stellt allerdings den ganzen Vorgang visuell dar und hilft dem Bürger, bei den einzelnen Schritten Entscheidungen zu treffen.

Bei der Analyse haben sich die Passwörter (PIN und CAN) als Hauptangriffspunkt heraus gestellt, welche nur durch Vorsichtsmaßnahmen und den konsequenten Einsatz eines Standard- oder Komfort-Chipkartenlesers geschützt werden können. Kennt ein Angreifer die PIN oder die CAN, so kann er nicht PACE über den Computer durchführen, wenn ein Standard- oder Komfort-Chipkartenleser vom Bürger verwendet wird. Der eID-Client hat keinen entscheidenden Einfluss auf die Sicherheit der Passwörter, wenn ein Basis-Chipkartenleser verwendet wird.

Anschließend wurde ein Vergleich zwischen einer Anwendung und einem Java Applet durchgeführt. Dabei wurden Vor- und Nachteile der verschiedenen Techniken erarbeitet. Diese wurden im Anschluss auf die Implementierung eines eID-Clients abgebildet. Diese Bewertung wird auf Basis dieses Vergleiches durchgeführt.

Der eID-Client als Anwendung bietet vor allem nicht technische Vorteile gegenüber einer Implementierung als Java Applet (siehe Abschnitt 4.7.7). Darunter fällt das einheitliche Aussehen und Bedienkonzept, weil für den Authentisierungsprozess immer derselbe eID-Client verwendet wird. Zudem kann der Bürger zu der installierten Anwendung ein höheres Vertrauen aufbauen, da sie von einer vertrauenswürdigen Quelle einmalig übertragen wurde und bis zur nächsten Aktualisierung als recht sicher angenommen werden kann. Diese Vorteile bringen allerdings einen nicht zu unterschätzenden Mehraufwand mit sich. Darunter fällt die Plugin-Entwicklung für verschiedene Browser und ein eigener Aktualisierungsprozess, der für einige Betriebssysteme notwendig ist. Vor allem der Aktualisierungsprozess ist sicherheitskritisch, wie die Sicherheitslücke im Update-Mechanismus des eID-Clients vom BMI gezeigt hat (siehe Update Problem mit ZIP-Dateien beim BürgerClient [Scha]).

Das Applet bietet in diesem Fall sehr viele technische Vorteile. Die Integration in den Browser und der Aktualisierungsprozess für den eID-Client wird vom JVM-Hersteller zur Verfügung gestellt, da es ein Teil der JVM-Technik ist. Das einheitliche Aussehen der Applets (siehe Abschnitt 4.7.5) stellt in diesem Fall sogar ein Vorteil dar, weil der Prozess unabhängig vom Betriebssystem gleich dargestellt werden kann. Der verminderte Entwicklungsaufwand, die nicht unbedingt notwendige Versionspflege (siehe Abschnitt 4.7.4) und die Unabhängigkeit von der komplexen eCard-API sind weitere Vorteile. Zudem kann durch die Applet-Lösung das Risiko für eine Profilbildung (siehe Abschnitt 4.7.6) reduziert werden.

Zusammengefasst bietet das Applet gegenüber einer Anwendung einige technische Vorteile und bringt zudem einen reduzierten Entwicklungsaufwand mit sich. Allerdings liegt der entscheidende Nachteil in der Bestätigung von extern geladenem Code. Die meisten

Bürger würden sich daran gewöhnen, extern geladen Code zu bestätigen und diesem zu vertrauen. Dieser Nachteil kann gravierende Auswirkungen auf die Sicherheit aller Computersysteme haben, so dass eine Signatur des eID-Clients durch den eID-Service Betreiber oder den Dienstanbieter nicht umgesetzt werden sollte.

Eine Implementierung als Java Applet ist aus den genannten technischen Gründen zu empfehlen. Allerdings sollte der eID-Client von einer vertrauenswürdigen Instanz wie dem BSI signiert werden, um dem Bürger eine Vertrauensquelle für den eID-Client in Form einer ihm bekannten Signatur zu geben. Allerdings müsste auch hier die vertrauenswürdige Signatur dem Bürger richtig vermittelt werden, um keine zusätzlichen Risiken zu schaffen.

Teil IV.

Implementierung

KAPITEL 5

IMPLEMENTIERUNG

5.1. Kodierungsschemas

Für die eID-Anwendung, EAC und die Kommunikation müssen Daten zwischen den beteiligten Parteien ausgetauscht werden. Diese liegen in verschiedenen Formaten vor, die in den folgenden Abschnitten erläutert werden.

5.1.1. ASN.1

Abstract Syntax Notation One (ASN.1) wurde von der ITU-T[1] und der ISO/IEC[2] erstellt und beschreibt nach ITU-T X.680 [IT02a] „eine standardisierte Notation für Datentypen, Werte und Nebenbedingungen für Datentypen.“ Er wurde gemeinsam von der ITU-T entwickelt.

Der Standard (Quelle ITU-T X.680 [IT02a]):

- definiert eine gewisse Anzahl von primitiven Datentypen, mit vordefinierten Tags und spezifiziert eine Annotation für die Referenzierung dieser Typen und deren Wertebereiche;

[1] International Telecommunication Union – Telecommunication Standardization Sector
[2] International Organization for Standardization/International Electrotechnical Commission

- definiert Mechanismen für die Konstruktion von neuen Datentypen auf Basis der primitiven Datentypen, spezifiziert eine Notation für die Definition solcher Typen, so wie man ihnen Tags und Wertebereiche definiert;

- definiert Zeichensätze (Referenzen zu anderen Empfehlungen und/oder Internationalen Standards) für die Verwendung im ASN.1.

Ein Beispiel für die Kodierung ist im folgenden Beispiel zu finden:

```
1 MyProtocol DEFINITIONS ::= BEGIN
2
3     MyPersonalNumber ::= SEQUENCE {
4         myNumber          INTEGER,
5         myName            IA5String
6     }
7 END
```

Daraus entstanden abgeleitete Kodierungsverfahren die *Basic Encoding Rules (BER)*, die *Distinguished Encoding Rules (DER)* und die *Canonical Encoding Rules (CER)*, welche in dem ITU-T Standard X.690 [IT02b] publiziert wurden.

Die Definition der primitiven Datentypen nach ASN.1 wie Integer und String werden nicht im vollen Umfang benötigt und können im Standard nachgeschlagen werden.

5.1.2. Basic Encoding Rules

Basic Encoding Rules (BER) wurden im ITU-T Standard X.690[3] [IT02b] publiziert. Sie basieren auf den ASN.1 Notation und beschreiben Regeln für die Kodierung und Dekodierung von Werten.

Die kodierten Daten bestehen aus vier Komponenten (siehe auch ITU-T X.690 [IT02b]):

- **Bezeichner (Tag) als Oktett**: Enthält einen Tag in der ASN.1 Notation, welcher den Typ des Inhalts definiert.

- **Länge des Oktetts**: Beschreibt die Länge des darauf folgenden Oktett-Inhalts.

- **Inhalt als Oktett**: Der eigentliche Inhalt in der definierten Kodierung.

- **Ende des Inhalts**: Wird nur definiert, wenn es sich um einen unbegrenzten Inhalt handelt.

[3] Auch als ISO/IEC 8825-1 Standard publiziert

Beim nPA werden ausschließlich Elemente mit definierter Längenangabe verwendet. Sie werden als TLV Objekte (Tag Length Value) oder BER-TLV Objekte bezeichnet.

Für eine logische Aufteilung und Gruppierung von Elementen existieren Sequenzen und Mengen, die im folgenden erläutert werden. Sie können auch ineinander verschachtelt werden.

- **SEQUENCE:** Eine Sequenz definiert eine feste und geordnete Liste von Typen, wobei einige Typen als optional deklariert werden können. Die Elemente bilden eine geordnete Liste mit definierten ASN.1 Elementtypen.

- **SET:** Ein Menge bildet eine definierte, aber im Vergleich zu der Sequenz nicht sortierte List von Typen. Die Elemente müssen einen bestimmten Typ haben, wobei einige der Typen als optional deklariert werden können.

5.1.3. Distinguished Encoding Rules

Distinguished Encoding Rules (DER) definieren Regeln für die Kodierung von Dokumenten, die von einer anderen Instanz eindeutig dekodiert werden können. Diese Regeln sind im ITU-T X.690 [IT02b] Standard beschrieben und wurden auch als ISO/IEC 8825-1 Standard veröffentlicht.

DER ist als Anforderung an den *BER* Standard entstanden, welche es nicht erfüllen konnte. Die fehlende Eindeutigkeit von Kodierungsregeln ist ein Nachteil des *BER* Verfahrens. Dadurch ist es beispielsweise möglich ein Boolean-Wert True (Wahr) in 255 Zuständen zu kodieren, während False (Falsch) lediglich mit 0 kodiert wird. Die fehlende Eindeutigkeit bei der Kodierung macht es zum Beispiel unmöglich, signierte Dokumente zu prüfen, da die Interpretation der Werte bei den BER nicht eindeutig ist.

Die meisten Informationen auf dem nPA und in Verbindung mit dem nPA werden DER kodiert, damit sie eindeutig überprüft werden können.

5.1.4. Notation für Object Identifier Typen

Die Object Identifier (OID) Typen beschreiben bestimmte Typen von Objekten in einer ASN.1 Struktur. Sie werden nach einem Baumschema gebildet, welches im dem ITU-T Standard X.660 [IT04] definiert ist.

Bevor die OID in BER oder DER kodierten Dokumenten gespeichert werden, müssen Sie nach dem ITU-T Standard X.690 [IT02b] kodiert werden. Dabei wird der erste Subidentifier durch die ersten zwei Werte aus dem Object Identifier nach der folgenden Formel generiert:

$(X \cdot 40) + Y$

X steht für den ersten Identifier und Y für den zweiten Identifier. Alle weiteren Subidentifier werden direkt in Hexadezimalwerten kodiert.

Als Beispiel wäre für den SHA1 Object Identifier '1.3.14.3.2.26' die erste Stelle $(1 * 40) + 43_{10} = 2B_{16}$. Der ganze Identifier ergibt den folgenden Hexadezimalwert: $2B0D03021A_{16}$.

5.1.5. Card Verifiable Certificate

Card Verifiable Certificate (CVC) sind digitale Zertifikate, die im Vergleich zu anderen Zertifikatsformaten wie beispielsweise *X.509* in einer besonders kompakter Form gespeichert werden, um den Anforderungen der Smartcards gerecht zu werden. Dadurch soll mit der begrenzten Leistung und dem begrenzten Speicher möglich sein, Zertifikate zu speichern, zu verarbeiten und zu validieren.

Zertifikatsformat

Die CV Zertifikate werden im kompakteren Format gespeichert und enthalten nicht alle Informationen, die X.509 Zertifikate enthalten.

Von den CV Zertifikaten existieren zwei Typen, die sich von der Struktur her unterscheiden. Selbstbeschreibende Zertifikate und nicht selbstbeschreibende Zertifikate. Die Unterschiede werden in den folgenden zwei Abschnitten genauer erläutert.

Selbstbeschreibende Zertifikate

Selbstbeschreibende Zertifikate werden *ASN.1* kodiert und enthalten damit Tags und die Längen der einzelnen Felder. In CV Zertifikaten sind nicht alle möglichen *ASN.1* Felder erlaubt. Eine Auflistung erlaubter Felder kann in der Norm ISO/IEC 7816-6 [ISO04c] nachgeschlagen werden.

Schicht	Name	Normen
7	Anwendungsschicht	ISO-7816-4,7,8,9
2	Leitungsschicht	ISO-7816-3, ISO-14443-4
1	Physikalische Schicht	ISO-7816-2, ISO-14443-2

Tabelle 5.1.: ISO-Schichten der Smartcard Kommunikation nach ISO/IEC 7498-1 [ISO94]

Nicht selbstbeschreibende Zertifikate

Nicht selbstbeschreibende Zertifikate sind nur die Zertifikatsdaten ohne Struktur. Das bedeutet, für die Dekodierung der Daten muss eine vordefinierte Struktur vorhanden sein. Der Vorteil dieser Speicherstruktur ist eine weitere Minimierung der Zertifikatsgröße. Sie hat aber den Nachteil, dass die Struktur vorab festgelegt und allen Parteien bekannt sein muss.

5.2. Ansteuerung von Smartcards

Die Kommunikation zwischen Smartcards und der Anwendung erfolgt über Kartenlesegeräte, welche die kartenspezifischen Protokolle unterstützen. Der Computer schickt dabei standardisierte APDUs an die Smartcard und erhält von der Smartcard Antworten auf die Anfragen. In den folgenden Abschnitten werden die Kommunikationsschichten und der grobe Aufbau der APDUs beschrieben.

5.2.1. ISO-Schichten der Kommunikation

Die Kommunikationsschichten der Smartcards sind nach den OSI-Schichtenmodell (siehe ISO/IEC 7498-1 [ISO94]) aufgebaut. Die Tabelle 5.1 enthält alle für die Smartcardkommunikation relevanten Schichten sowie die einzelnen ISO/IEC-Normen, welche auf den jeweiligen Schichten arbeiten. Wie aus ihr zu erkennen ist, werden nur die Schichten eins, zwei und sieben des OSI-Schichtenmodells umgesetzt.

Die Normen-Familie ISO-7816 beschreibt die kontaktbehaftete Kommunikation, während die Normen-Familie ISO-14443 die kontaktlose Kommunikation behandelt.

5.2.2. Kommunikationsprotokolle

Für die Kommunikation mit Smartcards existieren mehrere Protokolle. Sie alle arbeiten auf der zweiten Schicht (Leitungsschicht) und haben Einfluss auf die darüber liegenden Schichten. Das Protokoll muss am Anfang der Kommunikation ausgewählt werden.

Die Kommunikation wird abhängig von der Übertragungstechnik auf verschiedenen Protokollen abgewickelt. An dieser Stelle erfolgt eine kurze Auflistung der Protokolle abhängig vom Standard:

- ISO/IEC 7816-3 [ISO04a]: $T = 0$, $T = 1$
- ISO/IEC 10536-4 [ISO95]: $T = 2$

Für diese Arbeit sind nur die Protokolle $T = 0$ und $T = 1$ relevant. Aus diesem Grund werden alle anderen genannten Protokolle nicht weiter behandelt.

Das Protokoll $T = 0$ ist das ältere Protokoll und unterstützt APDUs mit Längen von bis zu 256 Bytes. Diese Einschränkung wurde beim Nachfolgeprotokoll $T = 1$ aufgehoben, um der Anforderung an längere APDUs gerecht zu werden. Durch die Verwandtschaft des Protokolls $T = 1$ zu $T = 0$ wirken sich allerdings auch einige Einschränkungen auf $T = 1$ aus, welche für diese Arbeit nicht relevant sind und im ISO/IEC 7816-3 [ISO04a] Standard nachgeschlagen werden können.

5.2.3. APDU

APDU (Application Protocol Data Unit) ist nach ISO-Norm ISO/IEC 7816-4 [ISO04b] eine Kommunikationseinheit zwischen Smartcard und Anwendung. Bei APDUs wird zwischen Command APDUs und Response APDUs unterschieden. Die Command APDUs werden an die Smartcard geschickt. Response APDUs sind Antworten von der Smartcard auf geschickte Kommandos. Die Smartcard ist ein passiver Kommunikationsteilnehmer, der nur auf Kommandos reagiert und selber nur Antworten auf Kommandos verschickt.

APDUs sind ähnlich IP-Paketen in Netzwerken aufgebaut. Sie besitzen einen Header und einen optionalen Body. Bei APDUs erhält der Header die zusätzliche Bedeutung der Befehlsbildung. Im Vergleich zu IP-Paketen ist die vollständige Struktur der APDUs strikt vorgeschrieben. Zusätzlich kommt auf eine Command APDU immer eine Antwort in Form einer Response APDU, um den Status des Befehls zu erhalten, was bei IP-Paketen nicht zwingend der Fall sein muss.

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

Tabelle 5.2.: Genereller Aufbau eines Command ADPUs nach ISO/IEC 7816-4 [ISO04b]

Während in IP-Netzwerken die Pakete primär dem Datenaustausch dienen und Befehle in applikationsspezifischer Form in Body enthalten sein können, sind Command-APDUs immer Befehle an die Smartcard, welche eine Interaktion durchführen und einen Zustand in Form einer Response ADPU zurückliefern.

Command APDU

Command APDUs bestehen aus einem Kopf (Header) und einem optionalen Körper (Body). Die Tabelle 5.2 beschreibt den generellen Aufbau einer Command APDU.

Die Tabelle 5.3 beschreibt die Bedeutung der einzelnen Felder der Command APDU und die möglichen Längen der Felder.

Werden keine Antwortdaten erwartet, so wird das L_e-Byte weggelassen. Dadurch erfolgt die Antwort nur in Form des Statuswortes ohne Antwortdaten.

Das L_c-Byte (Länge des Datenfeldes) inklusive Daten des Befehls (Data) sind optional und von L_c unabhängig. Dadurch entstehen genau vier Kombinationen für mögliche Strukturen des Command APDUs, welche in der Tabelle 5.4 dargestellt werden.

Response APDU

Reponse APDUs werden als Antworten auf die Command APDUs verschickt. Sie bestehen aus einem optionalen Body (Körper) und einem Trailer. Der Trailer besteht aus einem Statuswort (SW) von 2 Bytes, welche Auskunft über die Abarbeitung des Kommando APDUs zurückgibt. Die Struktur der Response APDU ist in der Tabelle 5.5 beschrieben.

Ist die Länge der Antwort $L_e = 0$ in dem Command APDU mit Null definiert, so entfällt der Data-Teil und die Antwort enthält nur das Statuswort.

Bezeichner	Name	Länge	Bedeutung
CLA	Class	1 Byte	Beschreibt die Klasse des Befehls (Commands). Zusätzlich wird damit signalisiert, ob Secure Messaging bei der Kommunikation verwendet werden soll.
INS	Instruction	1 Byte	Ist das eigentliche Kommando, welches an die Smartcard geschickt wird. Das byteorientierte Protokoll T=0 beschränkt dabei die Befehle auf geradzahlige Instruction-Bytes.
P1	Parameter 1	1 Byte	Der erste Parameter zu dem Befehl (Command)
P2	Parameter 2	1 Byte	Der zweite Parameter zu dem Befehl (Command)
Lc	Length Command	0-3 Bytes	Länge der Kommandodaten
Data	Data	Definiert in Lc	Kommandodaten
Le	Length Expected	0-3 Bytes	Länge der erwarteten Antwortdaten

Tabelle 5.3.: Aufbau von Command APDUs und die Größen der Felder nach ISO/IEC 7816-4 [ISO04b]

CLA	INS	P1	P2			
CLA	INS	P1	P2	L_e		
CLA	INS	P1	P2	L_c	Data	
CLA	INS	P1	P2	L_c	Data	L_e

Tabelle 5.4.: Vier mögliche Strukturen für Command APDUs

Data	SW1	SW2
Data	Trailer	

Tabelle 5.5.: Aufbau des Response ADPUs nach ISO/IEC 7816-4 [ISO04b]

Statuswörter

Statuswörter dienen als Antwort-Codes auf ausgeführte Command APDUs und können weitere Informationen im Datenfeld enthalten. Die Antworten 9000 und $61xx$ sind Antworten auf fehlerfrei ausgeführte Befehle. Das letzte Byte (hier beschrieben durch xx) von $61xx$ sagt aus, wie viele Bytes noch abgeholt werden sollen, falls die Daten in eine Antwort nicht rein passen.

Die Statuswörter sind in der Norm ISO/IEC 7816-4 [ISO04b] definiert, wobei viele der negativen Codes im Betriebssystem der Smartcard definiert sind. Weiterhin hat jede Anwendung ihre spezifischen Fehlercodes. Dadurch ist für die Ansteuerung der Smartcard eine Dokumentation der jeweiligen Codes notwendig.

Kodierung der Längenfelder

Die Längenfelder L_c und L_e sind ein Byte lang. Damit lassen sich Werte von 0x01 bis 0xFF (1-255) definieren. Für L_e existiert der Sonderfall 0x00, welcher 256 als erwartete Länge der Antwort festlegt. Dadurch lassen sich für L_c 255 Bytes und für L_e 256 Bytes als maximal Länge definieren.

Im Zuge der Anforderungen müssen moderne Smartcards auch größere Datenmengen pro APDU übertragen können. Aus diesem Grund wurden für die *Extended APDUs* die beiden Längenfelder auf bis zu drei Byte erweitert. Über ATR (Answer to Reset) durch das Historical Byte kann auf der Smartcard festgelegt werden, ob sie Extended APDUS unterstützt.

Mit 3 Bytes lässt sich die Länge im Bereich 1 bis 65536 definieren. Die Kodierung erfolgt in 3 Bytes. Dabei hat das Byte B1 den Wert 00, während Byte B2 und B3 einen beliebigen Wert annehmen könne. Sind B2 und B3 beide gleich “00„, so definiert das Längenfeld den maximalen Wert von 65.536.

5.2.4. Strukturen für Anwendungen und Daten

Die Anwendungen und Daten werden in einer Struktur ähnlich dem Unix-Dateisystem organisiert, wobei sie bei Smartcards aus zwei verschiedenen Typen bestehen. Das Format und die Stelle, an der die Daten gespeichert werden, sind nicht Bestandteil der ISO/IEC 7816 Normenfamilie. Sie werden intern durch das Smartcardbetriebssystem verwaltet.

Zwei Kategorien von Strukturen werden unterstützt: Dedicated File (DF) und Elementary File (EF).

- DFs enthalten Applikationen, Gruppen von Dateien (ähnlich Ordnern auf Computerdateisystemen) und/oder speichern Datenobjekte. Dabei enthält eine Applikation-DF eine Applikation. Eine DF kann auch eine Eltern-Datei von weiteren Dateien sein. Diese Dateien sind direkt dahinter zu finden.

- EFs speichern Daten (ähnlich einer Datei auf Computerdateisystemen). Ein EF kann nicht eine Eltern-Datei von anderen Dateien sein. Generell existieren zwei Kategorien von EF-Dateien:

  - Ein internes EF speichert Daten, welche direkt von der Karte interpretiert werden. Sie werden meistens von der Karte selbst für Management und für Kontrollzwecke genutzt.

  - Eine Arbeits-EF wird nicht von der Karte interpretiert und ist ausschließlich für die Benutzung außerhalb der Karte bestimmt.

Als Anfang des Dateisystems (ähnlich dem Root-Slash „/“ bei Unix und Linux) fungiert die sogenannte Master File (MF), welche den Ausgangspunkt für die Zugriffe bildet.

Strukturauswahl

Durch Auswahl einer Struktur wird der Zugriff auf deren Daten ermöglicht. Ist die Struktur eine DF-Struktur, so wird der Zugriff auf die darunter liegenden Daten möglich. Strukturen werden implizit beispielsweise nach dem Reset ausgewählt oder explizit über die folgenden vier Methoden:

- **Durch den DF-Namen:** Ein DF-Name kann ein beliebiges DF referenzieren. Er besteht aus einem String von bis zu 16 Bytes. Jeder *Application Identifier (AID)* kann als DF-Name genutzt werden. Damit der Zugriff per Name möglich wird, muss jeder DF-Name nur einmal auf der Karte vorhanden sein.

- **Durch den File Identifier:** Der *File Identifier* kann beliebige File (DF oder EF) referenzieren, wobei der Identifier aus zwei Bytes besteht. Einige Werte sind für bestimmte Zwecke reserviert (siehe ISO/IEC 7816-4 [ISO04b]). Damit die Auswahl der Dateien über *File Identifier* möglich ist, muss jede Datei einen eigenen, einzigartigen Identifier haben.

- **Durch den Pfad:** Ein Pfad kann eine beliebige Datei Referenzieren. Er besteht aus Konkatenation von *File Identifiern*. Der Pfad beginnt mit einem DF für relative

Pfade oder mit einem MF für absolute Pfade und endet mit dem Identifier der referenzierten Datei. Die Reihenfolge des Pfades ist immer vom Eltern-Element zum Kind.

- **Durch den kurzen EF Identifier:** Ein kurzer *EF Identifier* kann eine beliebige EFs referenzieren. Er besteht aus 5 Bits, welchen nicht alle gleich sein dürfen. Der Zahlenbereich von 1 bis 30 kann für die Referenzierung genutzt werden, wobei kurze EF Identifier können nicht in einem Pfad verwendet werden.

5.2.5. Secure Messaging

Secure Messaging (SM) nach ISO/IEC 7816-4 [ISO04b] sichert Teile oder vollständige Command-Response Paare durch kryptographische Methoden ab. Das wird durch Datenauthentifizierung (data authentication) und Datengeheimnis (data confidentiality) sichergestellt. Durch die Anwendung einer oder mehrerer Verfahren wird sicher gestellt, dass die Daten gegen Geheimhaltung (Confidentiality), Integrität (Integrity) und Authentizität (Authenticity) geschützt sind. Für die Sicherheitsmechanismen müssen entsprechende Verfahren, Schlüssel und wenn nötig Initialisierungsinformationen zur Verfügung gestellt werden.

Um die Sicherheitsziele zu erreichen, können zwei Mechanismen eingesetzt werden. Die Verschlüsselung stellt die Geheimhaltung der Informationen sicher, wobei sie mit Hilfe von symmetrischen Schlüsseln bewerkstelligt wird.

Die Authentizität und Integrität der Daten wird durch Message Authentication Codes (MAC) sichergestellt. Die Strukturen von MAC basieren auf symmetrischen kryptographischen Blockchiffren.

5.2.6. Datenobjekte

Datenobjekte (Data objects) sind spezielle Objekte, welche TLV (Tag, Length, Value) kodiert in Datenfeld bestimmter APDUs gespeichert werden. Sie stellen in der Regel eine Sequenz von Datenobjekten im Datenfeld der APDU dar. In der Norm ISO/IEC 7816-4 [ISO04b] sind *SIMPLE-TLV* und *BER-TLV* als Datenobjekte definiert. In den folgenden Abschnitten erfolgt eine genau Beschreibung der beiden Objekttypen sowie deren Verwendung.

SIMPLE-TLV Datenobjekte

Jedes *SIMPLE-TLV* Datenobjekt besteht aus zwei oder drei aufeinanderfolgenden Feldern. Als erstes Feld steht das Tag Feld des Objekts, danach folgt die Länge des Value-Feldes (Werte selber) und anschließend der Wert (Value). Ein Record (Datensatz innerhalb einer EF) kann ein einfaches *SIMPLE-TLV* Datenobjekt darstellen.

Die folgende Aufzählung erläutert die Felder im Detail (Quelle: ISO/IEC 7816-4 [ISO04b]):

- **tag field**: Das Tag Feld besteht aus einem einzigen Byte und stellt Werte von 1 bis 254 dar. Ist ein Record ein *SIMPLE-TLV* Objekt, so kann der Tag als Identifier für den Record verwendet werden.

- **length field**: Das Längenfeld besteht aus einem oder drei Bytes und muss die folgenden Bedingungen erfüllen:

  - Ist das erste Byte des Längenfeldes ungleich FF_{16}, so handelt es sich um ein ein Byte langes Längenfeld und stellt Zahlen zwischen 1 und 254 dar.

  - Ist das erste Byte des Längenfeldes gleich FF_{16}, so können mit den zwei folgenden Bytes Werte von 0 bis 65535 kodiert werden.

- **value field**: Beschreibt das Längenfeld den Wert Null, so ist dieses Feld leer. Ansonsten besteht das Feld aus einer Folge von Bytes der Länge definiert im Längenfeld.

BER-TLV Datenobjekte

Jedes *BER-TLV* Datenobjekt besteht aus zwei oder drei aufeinanderfolgenden Feldern. Sie sind nach den Kodierungsregeln von ASN.1 (siehe Abschnitt 5.1.1) kodiert. Als erstes Feld steht das Tag Feld des Objekts, danach folgt die Länge des Value-Feldes (Werte selber) und anschließend der Wert (Value).

Die folgende Aufzählung erläutert die einzelnen Felder im Detail (Quelle: ISO/IEC 7816-4 [ISO04b]):

- **tag field**: Das Tagfeld besteht aus einem oder mehreren Bytes. Es beschreibt eine Klasse, Kodierung der Daten und kodiert eine Tagnummer.

CAN	43 28 66

Tabelle 5.6.: CAN as password

- **length field**: Das Längenfeld besteht aus einem oder mehreren Bytes und kodiert die Länge des Wertes.
- **value field**: Ist in dem Längenfeld die Länge mit Null kodiert, so ist dieses Feld leer. Ansonsten besteht der Wert aus aufeinander folgenden Bytes der definierten Länge.

Die Kodierung und Bedeutung der Felder im BER-TLV Format ist in der ISO-Norm [ISO04b] spezifiziert und kann dort nachgeschlagen werden.

5.3. Testfälle

Standards und Spezifikationen lassen Interpretationen für die Implementierung offen. Diese Interpretationen erschweren Vergleiche zwischen verschiedenen Implementierungen durchzuführen, ohne dabei den Quellcode direkt zu vergleichen. Der Vergleich von Quellcode oder Ausgaben macht bei diesen Algorithmen auch wenig Sinn, da die Eingaben von der Smartcard Zufallszahlen enthalten.

Mit diesen definierten Testfällen soll das Verhalten verschiedener Implementierungen verglichen werden, ohne dabei den Quellcode oder das Programm selber zu benötigen. Die Entwickler der verschiedenen Implementierungen müssen lediglich das Setzen von statische Parameter an allen notwendigen Stellen der Implementierung ermöglichen.

5.3.1. Vorbereitungen der Testfälle

Für alle nachfolgenden Testfälle gelten die gleichen statischen Parameter. Diese Parameter sind in den folgenden Unterabschnitten beschrieben.

PACE mit Passwort (CAN)

Bei allen Testfällen soll PACE mit einer CAN als Passwort durchgeführt werden. Dadurch werden die Testfälle vereinfacht, da beliebig oft getestet werden kann, ohne den Counter für mögliche Versuche im Code abzufragen.

Parameter	Wert
P (Int)	76 88 49 56 39 70 45 34 42 20 80 97 46 62 90 01 64 90 93 03 79 50 20 09 43 05 52 03 73 56 01 44 50 31 51 61 97 75 1
A (Hex)	a9 fb 57 db a1 ee a9 bc 3e 66 0a 90 9d 83 8d 72 6e 3b f6 23 d5 26 20 28 20 13 48 1d 1f 6e 53 74
B (Hex)	66 2c 61 c4 30 d8 4e a4 fe 66 a7 73 3d 0b 76 b7 bf 93 eb c4 af 2f 49 25 6a e5 81 01 fe e9 2b 04
G (Hex) ECPoint	04 a3 e8 eb 3c c1 cf e7 b7 73 22 13 b2 3a 65 61 49 af a1 42 c4 7a af bc 2b 79 a1 91 56 2e 13 05 f4 2d 99 6c 82 34 39 c5 6d 7f 7b 22 e1 46 44 41 7e 69 bc b6 de 39 d0 27 00 1d ab e8 f3 5b 25 c9 be
N (Int)	76 88 49 56 39 70 45 34 42 20 80 97 46 62 90 01 64 90 92 73 75 31 78 44 14 52 95 38 75 55 19 06 30 63 53 63 59 07 9
H	1
OID	0.4.0.127.0.7.2.2.4.2.2 id_PACE_ECDH_GM_AES_CBC_CMAC_128

Tabelle 5.7.: Statische PACE-Parameter aus der EF.CardAccess

EF.CardAccess Informationen

In der *EF.CardAccess* werden statische Parameter festlegt. Diese Parameter bleiben bei jeder Ausführung unverändert. Die Parameter bestehen aus den folgenden Werten:

5.3.2. Algorithmusbeschreibung

Die folgende Algorithmusbeschreibung orientiert sich an einem einfachen und nach der Spezifikation durchgeführten PACE-Durchlauf ohne Zugriffsberechtigungen anzufordern. Die Beschreibung ist von Algorithmen und Parametern auf die Vorgaben aus diesem Testlauf eingeschränkt, damit Abweichungen in den Implementierungen einfacher erkannt werden können.

Nonce

Der nPA liefert im ersten Schritt eine verschlüsselte Nonce zurück. Diese muss entschlüsselt werden. Für die Entschlüsselung muss laut BSI TR-03110 v2.01 [BSI09b] AES im ECB-Modus ohne Padding verwendet werden: „**AES/ECB/NoPadding**“.

Key Generation for Nonce Encryption	
CAN encoded as ISO-8859-1	432866
03 in Big Endian & 32 Bit	0x00 0x00 0x00 x003
SHA-1 (160 Bit) Hex value	e9 73 b4 5f 64 d3 8f a6 07 20 b8 9a ad c1 f0 27 0b e5 87 49
We need only the first 128 Bit for the key	e9 73 b4 5f 64 d3 8f a6 07 20 b8 9a ad c1 f0 27

Tabelle 5.8.: Key Generation for Nonce Encryption based on the password (CAN)

Achtung: In der TR-03110 v2.02 wurde der Modus von ECB auf CBC geändert, welcher für die finalen Ausweise verwendet wird.

Um die Nonce zu entschlüsseln, muss ein 128 Bit langer Schlüssel generiert werden. Dieser Schlüssel wird aus der CAN und der Zahl 03 generiert. Die Zahl 03 ist 32 Bit lang und ist in Big Endian kodiert.

Mapping Data

Mapping Data ist der erste Teil von PACE. Die Zahl s sind die ersten 128 Bit der entschlüsselten Zufallszahl von der Smartcard, welche als gemeinsames Geheimnis in die Rechnung eingeht. Die Berechnung sieht wie folgt aus:

$$
\begin{aligned}
X_{1_{PCD}} &= G * x_{1_{PCD}} \\
Y_{1_{PICC}} &= G * y_{1_{PICC}} \\
P &= Y_1 * x_{1_{PCD}} = X_1 * y_{1_{PICC}} \\
G_{both} &= s * G + P
\end{aligned}
$$

G_{both} (Gs) ist der gemeinsame Punkt nach der Ausführung.

Achtung: Alle von Byte Array nach (Big)Integer umgewandelten Zahlen müssen positiv sein. Bei Java kann die Umwandlung wie folgt durchgeführt werden:

```
1 // 1 = always positive
2 BigInteger someBigInt = new BigInteger(1, byteArray);
```

Ephemeral Key Agreement

Danach wird ein anonymer Schlüsselaustausch durchgeführt:

$$X_{2_{PCD}} = G_{both} * x_{2_{PCD}}$$
$$Y_{2_{PICC}} = G_{both} * y_{2_{PICC}}$$
$$K_{both} = Y_{2_{PICC}} * x_{2_{PCD}} = X_{2_{PCD}} * H * y_{2_{PICC}}$$

Achtung: Alle von Byte Array nach (Big)Integer umgewandelten Zahlen müssen positiv sein. Bei Java kann die Umwandlung wie folgt durchgeführt werden:

```
1 // 1 = always positive
2 BigInteger someBigInt = new BigInteger(1, byteArray);
```

Mutual Authentication

Für Mutual Authentication muss nach BSI TR-03110 v2.01 [BSI09b] Seite 51 ein vollständiger Public Key inklusive der Parameter mit CMac signiert werden. Das Format für Elliptic Curve Public Keys sind ebenfalls im TR-03110 v2.01 auf Seite 90/91 zu finden.

Laut Seite 50 der TR-03110 v2.01 [BSI09b] muss für den CMac Schlüssel der x-Wert des Shared Secrtes K_{both} vom Ephemeral Key Agreement mit SHA-1 gehasht werden. Die ersten 128 Bit des Hash-Wertes stellen den Schlüssel für die Verschlüsselungsfunktion der CMac dar.

Wie im ObjectIdentifier angegeben, muss für die CMac Funktion AES im CBC-Modus ohne Padding verwendet werden. Die Schlüssellänge beträgt wie angegeben 128 Bit. Die folgende Konfiguration der Blockchiffre muss verwendet werden: **AES/CBC/NoPadding**.

Der Authentication Token stellt **die ersten 64 Bit der 128 Bit langen Signatur** dar.

Achtung: Beim Erstellen des öffentlichen Schlüssels kann es passieren, dass bei der Umwandlung von BigInteger in Byte Arrays $0x00$ am Anfang des Byte-Arrays steht, damit die Zahl nicht als negative Zahl interpretiert wird. Diese $0x00$ muss vorher entfernt werden, da die Smartcard nur positive Zahlen erwartet.

5.3.3. Positiv durchgeführter Testfall gegen einen nPA

Die Tabelle 5.9 enthält die Zahlen von einem positiven PACE-Durchlauf.

Name	Value
Encrypted Nonce	5d3cea82082ea582fef946b30fa6406f
Decrypted Nonce	b6d512001fe742044dfc47f1502d91ea
x1	23d475a140a93ecef3318b3bd35247c7db634cc18f1bc984b880c985fa79e5dc
X1	0486f7592a62d7f266cb08da13c96f65e732080cf8191243497fbc78e2ce06a2f718bc17b453c670cdd2bb943cd88b1f0a8c19b79fb8a5ac3196e79d220c739f78
Y1	0477976f4d04c9edaf4583e8f9c67c2e04ccac3829267bca5ac82bdf53f188a93b50f0f6098e5f7922da07ea6b88c39abf196010150c47b42ad345dd74b51949e8
Gs	04a3b38b59b7c89bb65aba162e6accf736d0d7f72b3ba97c6c1ec9c94a2b8814c61f0041f6d67d4de5990922572e616fd445d5eb13b058c9a9404a202d2099aadf
x2	da9af5c591bbc3c77ae00d590541b04726df95b5db4926894b92df852efd7f0f
X2	048140a31ae2f3609fee4815295ed8c55387d13bd77e668add027df54b0a0b9a381c262ce6d55835c7f32bea0106327fec8535f97b1756663ac7683c17410c4841
Y2	04a575811341de030d18855d8c8d397beecf1cb7108bdf713c547ef84ee417d54f6da82536d7678911ce311966fb4ef33eba4d8a87912343656543203505626ba4
K_{shared}	0428aed43e09b6f0bfb1c01f889333ccb260c081c399a74a5b6c913d24fcf0693995f99dab1fbd113bc91c72c3d4442ce259274e4b52833b6f28fd26fdebed8005
k_{mac}	f7906ae856bb3ee63855ea90486d9065
k_{enc}	fab0ff5290cbd4808b91bbbd5a4ba493
Public Key 152	7f49 82011d 060a04007f00070202040202 8120a9fb57dba1eea9bc3e660a909d838d726e3bf623d52620282013481d1f6e5377 8220a9fb57dba1eea9bc3e660a909d838d726e3bf623d52620282013481d1f6e5374 8320662c61c430d84ea4fe66a7733d0b76b7bf93ebc4af2f49256ae58101fee92b04 844104a3b38b59b7c89bb65aba162e6accf736d0d7f72b3ba97c6c1ec9c94a2b8814c61f0041f6d67d4de5990922572e616fd445d5eb13b058c9a9404a202d2099aadf 8520a9fb57dba1eea9bc3e660a909d838d718c397aa3b561a6f7901e0e82974856a7 864104a575811341de030d18855d8c8d397beecf1cb7108bdf713c547ef84ee417d54f6da82536d7678911ce311966fb4ef33eba4d8a87912343656543203505626ba4 870101
CMac Signature	facbda98df2e111aa650715e06684b1a
auth-token	facbda98df2e111a

Tabelle 5.9.: Testcase: PACE successful finished log

5.3.4. APDU-Kommunikation zwischen nPA und eID-Client

In der Tabelle 5.10 wird eine mitgeschnittene Kommunikation zwischen dem eID-Client vom BMI und dem nPA dargestellt. Die Kommunikation findet zwischen dem Terminal (T) und dem nPA als Smartcard (C) statt. Die APDUs wurden auf die relevanten Befehle beschränkt.

Richtung	APDU
SELECT EID Application	
$T \rightarrow C$	00 a4 04 0c 09 e8 07 04 00 7f 00 07 03 02
$C \rightarrow T$	90 00
SELECT MF	
$C \rightarrow T$	00 a4 00 0c 02 3f 00
$C \rightarrow T$	90 00
MSE (Manage Security Environment)	
$T \rightarrow C$	00 22 c1 a4 0f 80 0a 04 00 7f 00 07 02 02 04 02 02 83 01 03
$C \rightarrow T$	90 00
General Authenticate: Request encrypted nonce	
$T \rightarrow C$	10 86 00 00 02 7c 00 00
$C \rightarrow T$	7c 12 80 10 6d 98 9f d7 f0 f2 db d2 bc ac 1f fc 2f df 19 bb 90 00
General Authenticate: Mapping Data	
$T \rightarrow C$	10 86 00 00 45 7c 43 81 41 04 1b 3f 37 c6 ff b1 f8 0d 0f 0e a5 61 4a fa 66 42 a1 00 86 b0 d5 d6 6a 0a 15 2a ff c2 eb cf 09 52 41 24 00 69 d3 49 c9 c3 61 4a d1 79 28 c3 89 30 f9 74 9f e7 0d df 05 f7 f8 22 75 8f 0b 06 f9 68 00
$C \rightarrow T$	82 41 04 7a 4c 93 0a c9 75 de d0 ac 52 bc 9b f4 b1 b6 7b 1b 7e a6 7c 8e 85 bf d1 7f a9 1e 8d fa 6f 49 64 8d 6e 1b 1e 24 bd ef b2 96 f5 09 1c 85 9b 7c 50 88 9c 96 61 22 b5 3f 29 05 a0 1c f0 ab 4f 05 05 90 00
General Authenticate: Ephemeral Public Key	
$T \rightarrow C$	10 86 00 00 45 7c 43 83 41 04 0b df ea 9c 7e 0a 70 6c 54 09 ca b7 54 24 4f 53 ea 15 58 bf 8c b5 51 14 25 ee 5d b3 48 98 d8 32 3b 31 90 09 d7 85 d4 9c a4 88 cc 73 ca 24 8b a9 88 7c bb ab a0 c0 8c ed 3e e8 05 6e 63 1e 0d da 00
$C \rightarrow T$	7c 43 84 41 04 7a 18 aa 03 e5 7d 5c 4f 8a ad 26 3b a9 db ec 12 66 3b e3 9b 9b 4e 08 fd a1 83 59 c4 cc f8 39 0b a0 01 68 fe 29 7c db 58 b9 6c d5 92 e9 1e 12 b0 97 53 3d 40 97 c8 e3 c2 8a b2 7f 49 ab 72 b4 d8 90 00
General Authenticate: Mutual Authentication	
$T \rightarrow C$	00 86 00 00 0c 7c 0a 85 08 84 d3 35 b4 94 f1 e5 3b 00
$C \rightarrow T$	7c 0a 86 08 f6 7c 1b 4b 5f 5f 54 44 90 00
Secure Messaging: Verify Session Keys	
$T \rightarrow C$	0c 2c 02 03 1d 87 11 01 7d 9d 67 ef 57 97 a2 09 53 cf 2c d8 71 6e 05 ac 8e 08 93 ee 7d 10 3d bf fe 93 00
$C \rightarrow T$	8e 08 81 3b 30 1b 13 62 a2 e0 90 00

Tabelle 5.10.: Mitgeschnittener PACE-Durchlauf zwischen nPA und dem eID-Client vom BMI

5.4. Java

Für die Implementierung wurde Java aus Programmiersprache gewählt. Diese besitzt viele Komponenten für dieses Projekt und wurde wegen der möglichen Portabilität des Codes auf verschiedene Betriebssystem und Plattformen ausgewählt.

Im Folgenden werden wichtige Teile der Java Architektur erläutert, die für eine Implementierung eines eID-Clients notwendig sind.

5.4.1. Java Provider Architektur

Java stellt für bestimmte Funktionalitäten so genannte Provider zur Verfügung. Diese Provider erfüllen bestimmte Aufgaben und sind gegeneinander austauschbar, ohne den Code für die Nutzung der Provider zu ändern.

5.5. Kryptographische Provider

Cryptographic Provider Architecture (CPA) ist Bestandteil der Java6 Standard Edition und bietet die Möglichkeit Provider über vordefinierte Interfaces anzusprechen. Diese Provider sollte soweit standardisiert sein, so dass sie sich auch austauschen lassen. Für dieses Projekt wurde *Bouncy Castle* als CPA ausgewählt. Die Begründung für die Wahl, sowie die technischen Details werden im Folgenden Abschnitten beschrieben.

5.5.1. Bouncy Castle

Bouncy Castle ist ein Krypto-Provider für kryptographische Operationen und existiert unter anderem für Java und das Microsoft .NET Framework. Diese Beschreibung beschränkt sich lediglich auf die Java Version, da die Implementierung dieses Projektes auf Java basiert. Im folgenden wird die Lizenz und die technischen Möglichkeiten von *Bouncy Castle* beschrieben.

Lizenz

Bouncy Castle [bou] steht unter einer adaptierten MIT-Lizenz [mit], welche auf der Projektseite und der Dokumentation zur Software eingesehen werden kann.

Die Lizenz erlaubt eine uneingeschränkte Nutzung der Software in beliebigen Softwareprojekten. Es muss lediglich ein Verweis auf das Projekt und die Lizenz existieren. Dadurch ist die Verwendung von *Bouncy Castle* auch in kommerziellen Projekten ohne weitere Einschränkungen möglich.

Patentfreie Version

Bouncy Castle enthält in der "Extended„ Version den IDEA Algorithmus. Dieser ist durch Patente geschützt und ist nur für nicht kommerzielle Nutzung frei. Dieser Algorithmus wird für diese Arbeit nicht benötigt, so dass die patentfreie Version von *Bouncy Castle* eingesetzt werden kann.

5.5.2. Bouncy Castle für Java

Damit *Bouncy Castle* als Java Security Provider eingesetzt wird, muss dieser entsprechend initialisiert werden. Das kann durch die Datei 'java.security' oder direkt im Quellcode erfolgen.

Für diese Implementierung wird die Initialisierung direkt im Quellcode durchgeführt, um sicher zu stellen, dass der weniger sichere Sun-Provider nicht zum Einsatz kommt. Der Bouncy Castle Provider wird als JAR-Datei mitgeliefert und Standardprovider gesetzt.

5.6. Java Smartcard Kommunikation

Die Kommunikation mit dem Smartcard Lesegerät und weiter mit der Smartcard ist für Java auf zwei Arten möglich. Die erste und ältere Möglichkeit ist über einen betriebssystemspezifischen PC/SC Treiber [pcs11], der über einen Wrapper angesteuert wird (siehe auch Abschnitt 5.6.1).

Die zweite und für die Umsetzung dieser Arbeit verwendete Methode basiert auf der in Java mitgelieferten *Java Smartcard I/O*, welche wiederum auf dem Java Provider Konzept (siehe Abschnitt 5.4.1) basiert.

5.6.1. Java PC/SC

Java PC/SC [pcs11] ist ein Wrapper für die PC/SC Lite Schnittstelle, welcher die Kommunikation mit einer Smartcard erlaubt. Diese Schnittstelle war besonders für viele Linux Nutzer interessant, da die Hersteller oft keine APIs und Bibliotheken für Linux zur Verfügung stellten.

Durch den direkten Zugriff auf die PC/SC Bibliothek von Linux, ist Java PC/SC direkt von der Version der Bibliothek abhängig. Dadurch ist abhängig von der Bibliothekversion, eine passende Version notwendig.

Heute wird *Java PC/SC* nicht mehr in neuen Projekten eingesetzt. Mit der Java Smartcard I/O (siehe Abschnitt 5.6.2) ist eine Implementierung in der Java Runtime Environment direkt vorhanden und erfordert keinerlei Anpassungen am System. Durch die bereits vorhandenen und für alle Betriebssysteme gleiche Schnittstelle wurde *Java PC/SC* überflüssig und wird nicht mehr weiter entwickelt.

5.6.2. Java Smartcard I/O

Java Smartcard I/O basiert auf dem Java Provider Konzept (siehe Abschnitt 5.4.1). Es ist eine vordefinierte Schnittstelle, welche mehrere Provider nutzen kann. In der derzeitigen Version 1.6 von Java ist nur ein PC/SC Treiber vorhanden, über den die Kommunikation mit der Smartcard stattfindet. Weitere Treiber können eingebunden.

Der Vorteil gegenüber Java PC/SC (siehe Abschnitt 5.6.1) liegt vor allem in der Standardisierung über alle Plattformen und Betriebssysteme hinweg. Dadurch ist garantiert, dass die Software ohne zusätzliche Komponenten funktioniert und die Programmierer unabhängig von der Plattform auf die Smartcard zugreifen können.

Zusätzlich wird nicht nur eine Bibliothek erspart, sondern auch eine Lizenz, welche bei Java PC/SC zu beachten wäre.

5.7. Implementierung des Prototyps

5.7.1. EAC Bibliothek

Damit die Implementierung eines Prototypen möglichst effektiv und vor allem unabhängig vom Prototypen möglich ist, wurde zuerst eine Bibliothek entwickelt, welche EAC und die Smartcard Kommunikation abbildet.

Damit die Bibliothek weiter entwickelt und von anderen Entwicklern verwendet werden konnte, wurden für alle wichtigen Klassen Unittests geschrieben. Diese ermöglichen eine Qualitätsprüfung der Implementierung und dienen gleichzeitig als Beispiele für die Verwendung der Bibliothek.

Die Bibliothek löst ihre Abhängigkeiten über Maven auf und erstellt die Bibliothek mit dem Maven Builder. Weiterhin enthält das Projekt alle notwendigen Dateien, um in Eclipse als Projekt eingebunden zu werden.

Im Folgenden werden die einzelnen Pakete der Bibliothek kurz erläutert.

de.fraunhofer.sit.eacLib.asn1

Dieses Paket enthält eine Klasse mit Funktionen zur Konvertierung von ASN.1 kodierten ObjectIdentifiern in verschiedene Formate.

de.fraunhofer.sit.eacLib.bouncyCastle

Dieses Paket enthält eine Klasse, welche die Initialisierung des Bouncy Castle Providers vornimmt und dessen Einbindung überprüft. Die Implementierung verwendet weiterhin direkte Funktionen des Bouncy Castle Providers, daher ist die Verwendung des Providers zwingend notwendig.

de.fraunhofer.sit.eacLib.cert.cert

Dieses Paket enthält eine Klasse zum dekodieren von PKCS8 kodierten Zertifikaten.

de.fraunhofer.sit.eacLib.eac

Dieses Paket enthält eine smartcard-unabhängige Implementierung der drei EAC Protokolle PACE, TA und CA.

de.fraunhofer.sit.eacLib.eac.crypto

Dieses Paket enthält Hilfsklassen mit kryptographischen Funktionen, die in den verschiedenen Protokollen benötigt werden.

de.fraunhofer.sit.eacLib.smartcard

Dieses Paket enthält Interfaces und die StatusWord Klasse für die Smartcard Kommunikation. Die Interfaces werden in den Unterpaketen dieses Paketes verwendet.

de.fraunhofer.sit.eacLib.smartcard.apdu

Dieses Paket enthält Klassen, welche die Version des nPAs analysieren können und die richtige Factory für die APDUs laden. Weiterhin enthält es Klassen, um Dateien auf der Smartcard einfacher auszulesen.

de.fraunhofer.sit.eacLib.smartcard.cardaccess

Auf dem nPA sind diverse Dateien hinterlegt, die für die Durchführung von EAC ausgelesen und nach deren Format geparst werden müssen. Dazu zählt die EF.CardAccess und die EF.CardSecurty Datei. Dieses Paket enthält die dafür notwendigen Klassen.

de.fraunhofer.sit.eacLib.smartcard.channel

Dieses Paket enthält eine Proxy-Implementierung des CardChannels für Smartcards. Mit dessen Hilfe ist es möglich, die Kommunikation zwischen Smartcard und dem Programm zu protokollieren.

de.fraunhofer.sit.eacLib.smartcard.iso24727

Dieses Paket enthält eine Klasse, welche die Bedeutung der APDU-Statuswörter nach ISO/IEC-24727 auflöst.

de.fraunhofer.sit.eacLib.smartcard.iso7816

Dieses Paket enthält eine Klasse, welcher TLV Objekte nach ISO/IEC-7816 lesen und erstellen kann.

de.fraunhofer.sit.eacLib.smartcard.secureMessaging

Dieses Paket enthält den SecureMessaingProvider, welche APDUs für SecureMessaging ver- und entschlüsseln kann.

de.fraunhofer.sit.eacLib.utils

Dieses Paket enthält die Klasse ByteUtil, welche die Umwandlung von verschiedenen Formaten zwischen Bytes, Integer und String durchführt.

de.fraunhofer.sit.eacLib.test.*

Dieses Paket enthält Unit Tests für die wichtigsten Klassen der Bibliothek. Der Aufbau der Pakete entspricht dem Aufbau der Bibliothek.

5.7.2. eID-Server

Der eID-Server basiert auf dem Jetty Webserver, welcher einer sehr einfache Implementierung der Kommunikation über Websockets ermöglicht. Die Implementierung ist an die Version 7 des Websocketprotokolls gebunden, da sie zum Zeitpunkt der Entwicklung die aktuelle Version darstellte.

eac

Dieses Paket enthält die EAC Implementierung auf der Serverseite sowie das Auslesen einer Datengruppe aus der eID-Funktion des nPAs.

websocket

Dieses Paket enthält die Klasse zum Starten des Servers.

websocket.common

Dieses Paket enthält eine Klasse mit Websocketkonstanten, welche aus der Jetty-Implementierung rauskopiert wurden, da sie beim Jetty als protected deklariert waren und damit nicht in der Implementierung verwendet werden konnten. Sie sind von der Jetty Version abhängig.

websocket.servlet

Die Klasse EacWebsocketServlet ist das Servlet, welches die Kommunikation mit Hilfe von Websockets durchführt und die übertragenen Daten an die eigentliche Websocket-Implementierung übergibt.

websocket.websocket

Dieses Paket enthält die eigentliche Implementierung des Websockets für die Kommunikation mit der eID-Funktion des nPAs. Weiterhin enthält das Paket eine Websocketimplementierung, welche die Protokollierung der ganzen Kommunkation ermöglicht. Sie wurde aus der beispielhaften Implementierung eine serverseitigen Websockets aus Jetty extrahiert.

5.7.3. eID-ClientApp und eID-ClientApplet

Die eID-ClientApp sowie das Servlet basieren in Grundzügen auf einer beispielhaften Implementierung eines Websocket-Clients, wobei der Client sowie der Server an die Version 7 des Websocket Protokolls gebunden sind.

client

Dieses Paket enthält die Klassen, welche den Client als Anwendung und als Applet ausführen. Alle anderen Komponenten werden identisch in beiden Implementierungen verwendet.

client.eac

Dieses Paket enthält die clientseitige Implementierung von EAC und der darauf folgenden transparenten Kommunikation zwischen nPA und eID-Server.

client.gui

Dieses Paket enthält die sehr rudimentäre grafische Implementierung des eID-Clients, welches für das Applet und die Anwendung verwendet wird.

Teil V.

Fazit

KAPITEL 6

ZUSAMMENFASSUNG

In der vorliegenden Arbeit sollte die Authentifizierung mit dem nPA über die eID-Anwendung analysiert werden. Dafür wurde das Vorgehen nach IT-Grundschutz als die am besten passende Methode für die Sicherheitsanalyse ausgewählt. Der eID-Client wurde als kritische Softwarekomponente identifiziert. Als Ergebnis der Analyse wurden Sicherheitsanforderungen für die Entwicklung eines eID-Clients erarbeitet.

Im Implementierungsteil der Arbeit wurde EAC und die Smartcard-Kommunikation mit Hilfe von APDUs implementiert und in eine Bibliothek ausgelagert. Damit die Implementierung unabhängig von einem Ausweis getestet und auf deren Korrektheit überprüft werden konnte, wurden statische Testfälle in Form von Unittests entwickelt. Diese ermöglichen eine Überprüfung der einzelnen Protokolle (PACE, TA und CA) und wichtige Teile der Software. Die Testfälle sollen zudem die Verwendung der entwickelten EAC-Bibliothek erleichtern, da sie vollständige Abläufe inklusive der Smartcard-Kommunikation für die Ausweise aus den Anwendungstests enthalten.

Die statischen Testfälle für PACE wurden während des laufenden Anwendungstests an die HU Berlin und die Uni Koblenz übergeben, um die Entwicklung der dort jeweils laufenden Projekte zu fördern. Die Tests enthielten statische Werte mit allen Zwischenergebnissen, so dass die Implementierungen unabhängig von der verwendeten Programmiersprache getestet werden konnten. Solche Tests wurden ebenfalls vom BSI für alle Teile von EAC [Inf] veröffentlicht.

Die Implementierung von EAC stellte sich schwieriger heraus, als es am Anfang vermutet wurde. Zuerst mussten die kryptographische Protokolle nach Vorgabe des BSI Standards TR-03110 [BSI09b] implementiert werden. Anschließend musste die Kommunikation mit

dem nPA über APDUs implementiert und gegen einen Ausweis aus den Anwendungstests getestet werden. Der nPA verhält sich dabei als eine Blackbox[1]. Eine Fehleranalyse ist sehr schwer möglich, da die EAC-Protokolle Schutzmechanismen enthalten, die genau das verhindern sollen.

Weiterhin wurde eine prototypische Entwicklung einer Client-Server Software für eID entwickelt. Der eID-Client wurde als Anwendung und Java Applet entwickelt, der über Websockets mit einem selbst geschriebenen eID-Server kommuniziert. Der Server ist in der Lage, Datengruppen aus der eID-Funktion des nPAs auszulesen. Das Websocket-Protokoll wurde gewählt, weil es ein einfaches Protokol ist, welches fast kein Overhead besitzt und somit die Überprüfung der Kommunikation und der Sicherheitsmechanismen erleichtert.

[1] Ein geschlossenes System, welches nur durch äußeres Verhalten getestet werden kann.

KAPITEL 7

AUSBLICK

Die prototypische Implementierung des eID-Clients und des eID-Servers basiert auf der Kommunikation über Websockets. Der Server kommuniziert mit Hilfe von APDUs, welche vom eID-Client an den nPA weitergeleitet werden. Damit übernimmt der Server die Komplexität der verschiedenen Kartenlesegeräte und Ausweisversionen. Die Implementierung enthält keine vollständige Implementierung alle eID-Funktionen des nPAs. Restricted Identification ist in der Implementierung gar nicht enthalten.

Nach der derzeitigen Planung der EU werden in der nahen Zukunft viele Ausweise elektronische Funktionen erhalten, die eine entsprechende Software für die EU-Länder übergreifende Verwendung benötigen. Diese prototypische Implementierung könnte als Anfang dienen, welche eID-kompatible Ausweise unterstützt.

Eine weitere mögliche und aus meiner Sicht sehr interessante Weiterentwicklung dieser Masterarbeit wäre es auch, die auf XML, SOAP und PAOS basierenden Kommunikationsschnittstellen der eCard-API näher zu untersuchen. Aus den Ergebnissen kann eine neue und auf aktuelleren Techniken (wie JSON, REST und Websockets) basierende Schnittstelle entwickelt werden, da die Weiterentwicklung von SOAP eingestellt wurde (siehe Auflösung der XML Protocol Working Group am 10.07.2009 [W3C]). Sollte sie vom BMI akzeptiert oder adaptiert werden, so könnte meiner Meinung nach, die Entwicklung von eID-kompatiblen Anwendungen vereinfacht und dadurch die Verbreitung der eID-Anwendung gefördert werden.

Teil VI.

Anhang

ANHANG A

ANHANG

A.1. Auf dem Chip des nPAs gespeicherte Daten

Auf dem Chip befinden sich in ASN.1 gespeicherte Informationen, welche für die Durchführung von PACE und CA benötigt werden.

Der Chip stellt Sicherheitsinformationen (SecurityInfo) in den beiden Daten *CardAccess* und *CardSecurity* zur Verfügung. Diese sind in zwei EFs unter dem Master File (MF) gespeichert (siehe auch Tabelle A.1).

Die CardAccess kann immer ausgelesene werden, währen die CardSecurity erst nach einer erfolgreichen Durchführung von PACE und TA möglich ist.

Dateiname	**EF.CardAccess**	**EF.CardSecurity**
File ID	0x011C	0x011D
Short File ID	0x1C	0x1D
Lesezugriff	Immer möglich	nach PACE + TA
Schreibzugriff	nie	nie
Größe	variabel	variabel
Inhalte	DER encoded SecurityInfo	DER encoded SignedData

Tabelle A.1.: Daten für den ePerso Generation 1 (Quelle BSI TR-03110 [BSI09b])

A.1.1. CardAccess

Die Datei *CardAccess* enthält die folgenden Sicherheitsinformationen:

- PACEInfo
- PACEDomainInformation
- PACEAuthenticationInfo
- ChipAuthenticationInfo
- ChipAuthenticationDomainParameterInfo
- TerminalAuthenticationInfo
- CardInfoLocator

Diese bestehen aus vordefinierten Algorithmen, welche die Smartcard (in diesem Fall nPA) unterstützt. Sie sind auf der Karte in DER-kodiert (siehe Abschnitt 5.1.3) gespeichert. Das ist eine eindeutige Kodierung der Informationen nach dem ASN.1 (siehe Abschnitt 5.1.1) Standard.

Die CardAccess eines nPAs aus dem Anwendungstest sieht wie folgt aus:

Listing A.1: Inhalt der EF.CardAccess Datei ausgelesen aus einem Testausweis

```
1  DER Set
2      DER Sequence
3          Object Identifier(0.4.0.127.0.7.2.2.5.2) Integer(1)
4      DER Sequence
5          Object Identifier(0.4.0.127.0.7.2.2.3.2.1) Integer(2)
6      DER Sequence
7          Object Identifier(0.4.0.127.0.7.2.2.4.2.1) Integer(1)
8      DER Sequence
9          Object Identifier(0.4.0.127.0.7.2.2.2) Integer(2) DER Sequence
10             Integer(283) Integer(27)
11     DER Sequence
12         Object Identifier(0.4.0.127.0.7.2.2.1.2) DER Sequence
13             DER Sequence
14                 Object Identifier(1.2.840.10045.4.3.1)
15             DER Bit String[84, 0]
16         Integer(65)
17     DER Sequence
18         Object Identifier(0.4.0.127.0.7.2.2.3.2) DER Sequence
19             Object Identifier(0.4.0.127.0.7.1.1.5.1.2) DER Sequence
20                 Integer(1) DER Sequence
```

```
21                      Object Identifier(1.2.840.10045.1.1)
22                      Integer(22721622932454352787552537995910928
23                              07334073214594499230443547294131 1)
24                  DER Sequence
25                      DER Octet String[28] DER Octet String[28]
26                  DER Octet String[57]
27                  Integer(22721622932454352787552537995910923612 5
28                          67546342330757191396560966559)
29      DER Sequence
30          Object Identifier(0.4.0.127.0.7.2.2.4.2) DER Sequence
31              Object Identifier(0.4.0.127.0.7.1.1.5.1.2) DER Sequence
32                  Integer(1) DER Sequence
33                      Object Identifier(1.2.840.10045.1.1)
34                      Integer(22721622932454352787552537995910923
35                              80733407321459449923044354729413 11)
36                  DER Sequence
37                      DER Octet String[28] DER Octet String[28]
38                  DER Octet String[57]
39                  Integer(22721622932454352787552537995910923612 3
40                          25675463423307571913965609665 59)
```

A.2. nPA spezifische Object Identifier

Listing A.2: Object Identifier für das deutsche BSI

```
1 bsi-de OBJECT IDENTIFIER ::= {
2    itu-t(0) identified-organization(4) etsi(0)
3    reserved(127) etsi-identified-organization(0) 7
4 }
```

Für den nPA werden diverse *Object Identifier (OID)* verwendet. Sie sind vom BSI in der Spezifikation TR-03110 [BSI09a] definiert und haben eine festgelegte Struktur. Das Beispiel in der Tabelle A.2 zeigt den Aufbau eines *Object Identifiers* und die Bedeutung der einzelnen Zeilen in der Struktur.

Um die Object Identifier besser erkennen und nutzen zu können, werden Sie in den folgenden Tabellen aufgelistet, wobei sie nur die im Zusammenhang mit nPA benötigten Object Identifier enthalten.

A.3. Object Identifier für Algorithmen

Die Tabellen A.5 - A.11 beinhalten weitere Object Identifier für Algorithmen welche entweder im nPA direkt oder in den Anwendungen im Zusammenhang mit dem nPA verwendet werden.

Aufbau des Object Identifiers 0.4.0.127.0.7.2.2.3.2.2 nach ITU-T	
OID Teil	**Bedeutung**
0.	itu-t (0)
4.	identified-organization (4)
0.	etsi (0)
127.	reserved (127)
0.	etsi-identified-organization (0)
7.	bsi-de (7)
2.	protocols (2)
2.	smartcard (2)
3.	id-CA(3)
2.	id-CA-DH (2)
2	id-CA-DH-AES-CBC-CMAC-128 (2)

Tabelle A.2.: Aufbau des Object Identifiers 0.4.0.127.0.7.2.2.3.2.2 nach ITU-T

Object Identifier	Description
0.4.0.127.0.7.2.2.4	id-PACE (4)
0.4.0.127.0.7.2.2.4.1	**id-PACE-DH-GM** (1)
0.4.0.127.0.7.2.2.4.1.1	id-PACE-DH-GM-3DES-CBC-CBC
0.4.0.127.0.7.2.2.4.1.2	id-PACE-DH-GM-AES-CBC-CMAC-128
0.4.0.127.0.7.2.2.4.1.3	id-PACE-DH-GM-AES-CBC-CMAC-192
0.4.0.127.0.7.2.2.4.1.4	id-PACE-DH-GM-AES-CBC-CMAC-256
0.4.0.127.0.7.2.2.4.2	**id-PACE-ECDH-GM** (2)
0.4.0.127.0.7.2.2.4.2.1	id-PACE-ECDH-GM-3DES-CBC-CBC
0.4.0.127.0.7.2.2.4.2.2	id-PACE-ECDH-GM-AES-CBC-CMAC-128
0.4.0.127.0.7.2.2.4.2.3	id-PACE-ECDH-GM-AES-CBC-CMAC-192
0.4.0.127.0.7.2.2.4.2.4	id-PACE-ECDH-GM-AES-CBC-CMAC-256
0.4.0.127.0.7.2.2.4.3	**id-PACE-DH-IM** (3)
0.4.0.127.0.7.2.2.4.3.1	id-PACE-DH-IM-3DES-CBC-CBC
0.4.0.127.0.7.2.2.4.3.2	id-PACE-DH-IM-AES-CBC-CMAC-128
0.4.0.127.0.7.2.2.4.3.3	id-PACE-DH-IM-AES-CBC-CMAC-192
0.4.0.127.0.7.2.2.4.3.4	id-PACE-DH-IM-AES-CBC-CMAC-256
0.4.0.127.0.7.2.2.4.4	**id-PACE-ECDH-IM** (4)
0.4.0.127.0.7.2.2.4.4.1	id-PACE-ECDH-IM-3DES-CBC-CBC
0.4.0.127.0.7.2.2.4.4.2	id-PACE-ECDH-IM-AES-CBC-CMAC-128
0.4.0.127.0.7.2.2.4.4.3	id-PACE-ECDH-IM-AES-CBC-CMAC-192
0.4.0.127.0.7.2.2.4.4.4	id-PACE-ECDH-IM-AES-CBC-CMAC-256

Tabelle A.3.: Im nPA verwendete Object Identifier für id-PACE nach BSI TR-03110 [BSI09a]

Object Identifier	Description
0.4.0.127.0.7.2.2.3	id-CA (4)
0.4.0.127.0.7.2.2.3.1	**id-CA-DH** (1)
0.4.0.127.0.7.2.2.3.1.1	id-CA-DH-3DES-CBC-CBC
0.4.0.127.0.7.2.2.3.1.2	id-CA-DH-AES-CBC-CMAC-128
0.4.0.127.0.7.2.2.3.1.3	id-CA-DH-AES-CBC-CMAC-192
0.4.0.127.0.7.2.2.3.1.4	id-CA-DH-AES-CBC-CMAC-256
0.4.0.127.0.7.2.2.3.2	**id-CA-ECDH** (2)
0.4.0.127.0.7.2.2.3.2.1	id-CA-ECDH-3DES-CBC-CBC
0.4.0.127.0.7.2.2.3.2.2	id-CA-ECDH-AES-CBC-CMAC-128
0.4.0.127.0.7.2.2.3.2.3	id-CA-ECDH-AES-CBC-CMAC-192
0.4.0.127.0.7.2.2.3.2.4	id-CA-ECDH-AES-CBC-CMAC-256

Tabelle A.4.: Im nPA verwendete Object Identifier für id-CA (Chip Authentication) nach BSI TR-03110 [BSI09a]

Object Identifier	Description
1.3.14.3.2.26	SHA1
2.16.840.1.101.3.4.2.4	SHA224
2.16.840.1.101.3.4.2.1	SHA256
2.16.840.1.101.3.4.2.2	SHA348
2.16.840.1.101.3.4.2.3	SHA512

Tabelle A.5.: Kryptographische Hashfunktionen nach ICAO 9303 Part 3 Volume 1 [ICA08a]

Object Identifier	Description
1.2.840.113549.1.1.1	RSA Encryption
1.2.840.113549.1.1.5	RSASSA-PKCS1_v15 with SHA1
1.2.840.113549.1.1.10	RSASSA-PSS (PKCS #1 Version 2.1)
1.2.840.113549.1.1.14	RSASSA-PKCS1_v15 with SHA224
1.2.840.113549.1.1.11	RSASSA-PKCS1_v15 with SHA256
1.2.840.113549.1.1.12	RSASSA-PKCS1_v15 with SHA384
1.2.840.113549.1.1.13	RSASSA-PKCS1_v15 with SHA512
1.2.840.10040.4.3	DSA with SHA1
1.2.840.10045.1	ECDSA with SHA1 (ANSI X9.62)
1.2.840.10045.4.1	ECDSA with SHA1 (ANSI X9.62)
1.2.840.10045.4.3.1	ECDSA with SHA224 (ANSI X9.62)
1.2.840.10045.4.3.2	ECDSA with SHA256 (ANSI X9.62)
1.2.840.10045.4.3.3	ECDSA with SHA384 (ANSI X9.62)
1.2.840.10045.4.3.4	ECDSA with SHA512 (ANSI X9.62)
0.4.0.127.0.7.4.1.1	ECDSA with SHA1 (BSI)
0.4.0.127.0.7.4.1.2	ECDSA with SHA224 (BSI)
0.4.0.127.0.7.4.1.3	ECDSA with SHA256 (BSI)
0.4.0.127.0.7.4.1.4	ECDSA with SHA384 (BSI)
0.4.0.127.0.7.4.1.5	ECDSA with SHA512 (BSI)

Tabelle A.6.: Signaturalgorithmen nach ICAO 9303 part 3 Volume 1 [ICA08a]

Object Identifier	Description
1.2.840.10045.2.1	ECDH
1.2.840.10046.2.1	DH
1.2.840.113549.1.3.1	DH

Tabelle A.7.: Spezifikationen für Schlüsselarten nach BSI TR-03110 [BSI09a]

Object Identifier	Description
0.4.0.127.0.7.2.2.1.1	CA-DH
0.4.0.127.0.7.2.2.1.2	CA-ECDH

Tabelle A.8.: Im nPA verwendete Object Identifier für id-CA (Chipauthentisierung) nach BSI TR-03110 [BSI09a]

Object Identifier	Description
0.4.0.127.0.7.2.2.2.1.1	TA-RSA-v1_5-SHA-1
0.4.0.127.0.7.2.2.2.1.2	TA-RSA-v1_5-SHA-256
0.4.0.127.0.7.2.2.2.1.3	TA-RSA-PSS-SHA-1
0.4.0.127.0.7.2.2.2.1.4	TA-RSA-PSS-SHA-256
0.4.0.127.0.7.2.2.2.2.1	TA-ECDSA-SHA-1
0.4.0.127.0.7.2.2.2.2.2	TA-ECDSA-SHA-224
0.4.0.127.0.7.2.2.2.2.3	TA-ECDSA-SHA-256

Tabelle A.9.: Signaturalgorithmen für Terminalauthentisierung nach BSI TR-03110 [BSI09a]

Object Identifier	Description
1.2.840.113549.1.1.1	RSA-ISO9792-2 SHA-1
1.3.36.3.4.2.2.1	RSA-ISO9792-2 SHA-1
1.3.36.3.4.3.2.1	RSA-ISO9792-2 RND SHA-1
1.3.36.3.4.3.2.3	RSA-ISO9792-2 RND SHA-224
1.3.36.3.4.3.2.4	RSA-ISO9792-2 RND SHA-256
1.2.840.113549.1.1.5	RSA with SHA1
1.2.840.113549.1.1.10	RSA_PSS
1.2.840.113549.1.1.14	RSA with SHA224
1.2.840.113549.1.1.11	RSA with SHA256
1.2.840.113549.1.1.12	RSA with SHA384
1.2.840.113549.1.1.13	RSA with SHA512
1.2.840.10045.2.1	ECDSA with SHA1

Tabelle A.10.: Passive Authentisierung Algorithmen nach BSI TR-03110 [BSI09a]

Object Identifier	Description
0.4.0.127.0.7.2.2.5	id-RI
0.4.0.127.0.7.2.2.5.1	id-RI-DH
0.4.0.127.0.7.2.2.5.1.1	id-RI-DH-SHA-1
0.4.0.127.0.7.2.2.5.1.2	id-RI-DH-SHA-224
0.4.0.127.0.7.2.2.5.1.3	id-RI-DH-SHA-256
0.4.0.127.0.7.2.2.5.2	id-RI-ECDH
0.4.0.127.0.7.2.2.5.2.1	id-RI-ECDH-SHA-1
0.4.0.127.0.7.2.2.5.2.2	id-RI-ECDH-SHA-224
0.4.0.127.0.7.2.2.5.2.3	id-RI-ECDH-SHA-256

Tabelle A.11.: Restricted Identification nach BSI TR-03110 [BSI09a]

LITERATURVERZEICHNIS

[AL02] Adams, Carlisle ; Lloyd, Steve: *Understanding PKI: Concepts, Standards, and Deployment Considerations*. 4th. Boston, MA, USA : Addison-Wesley Longman Publishing Co., Inc., 2002. – ISBN 0672323915

[AM01] Ateniese, Giuseppe ; Mangard, Stefan: A new approach to DNS security (DNSSEC). In: *Proceedings of the 8th ACM conference on Computer and Communications Security*. New York, NY, USA : ACM, 2001 (CCS '01). – ISBN 1–58113–385–5, 86–95

[ans01] *Public Key Cryptography for the Financial Services Industry, Key Agreement and Key Transport Using Elliptic Curve Cryptography*. 11 2001

[bou] *Bouncy Castle Project*. `http://www.bouncycastle.org/`. – (abgerufen am 20.03.2012)

[BSI08a] BSI: *Standard 100-1: Managementsysteme für Informationssicherheit (ISMS)*. 2008

[BSI08b] BSI: *Standard 100-2: IT-Grundschutz-Vorgehensweise*. 2008

[BSI08c] BSI: *Standard 100-3: Risikoanalyse auf der Basis von IT-Grundschutz*. 2008

[BSI08d] BSI: *Standard 100-4: Notfallmanagement*. 2008

[BSI08e] BSI: *Technische Richtlinie 03112-1: eCard-API-Framework – Protokolle*. 2008

[BSI08f] BSI: *TR-02102 Kryptographische Verfahren: Empfehlungen und Schlussellängen.* 2008

[BSI09a] BSI: *Technical Guideline 03110: Advanced Security Mechanisms for Machine Readable Travel Documents – Extended Access Control (EAC), Password Authenticated Connection Establishment (PACE) and Restricted Identification (RI).* 2009

[BSI09b] BSI: *Technical Guideline 03110 v2.01: Advanced Security Mechanisms for Machine Readable Travel Documents – Extended Access Control (EAC), Password Authenticated Connection Establishment (PACE) and Restricted Identification (RI).* 2009

[BSI09c] BSI: *Technical Guideline 03111: Elliptic Curve Cryptography Identification (RI).* 2009

[BSI09d] BSI: *Technical Guideline 03128: EAC-PKI'n für den elektronischen Personalausweis: Rahmenkonzept für den Aufbau und den Betrieb von Document Verifiern.* 2009

[BSI09e] BSI: *Technische Richtlinie 03119: Anforderungen an Chipkartenleser mit ePA Unterstützung v1.1.* 2009

[BSI10] BSI: *Technische Richtlinie 03127: Architektur elektronischer Personalausweis und elektronischer Aufenthaltstitel.* 10 2010

[Buc10] Buchmann, Johannes: *Einführung in die Kryptographie.* 5. Auflage. Springer-Verlag Berlin Heidelberg, 2010

[DO10] Dominik Oepen, Frank M.: „Die gesamte Technik ist sicher" Besitz und Wissen: Relay-Angriffe auf den neuen Personalausweis / Humboldt-Universität zu Berlin, Institut für Informatik, Lehrstuhl für Systemarchitektur. Unter den Linden 6, 10099 Berlin, 12 2010. – Forschungsbericht

[Dwo04] Dworkin, Morris: *Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication.* `http://csrc.nist.gov/publications/nistpubs/800-38C/SP800-38C_updated-July20_2007.pdf`. Version: 05 2004. – (abgerufen am 20.03.2012)

[Eck09] Eckert, Claudia: *IT-Sicherheit - Konzepte, Verfahren, Protokolle.* 6. Oldenbourg Verlag München Wien, 2009

[Gon98a] Gong, Li: *Java Security Architecture (JDK1.2).* `http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.49.8545&rep=rep1&type=pdf`. Version: 1998

[Gon98b] Gong, Li: Secure Java Class Loading. In: *IEEE Internet Computing* 2 (1998), November, 56–61. `http://dx.doi.org/10.1109/4236.735987`. – DOI 10.1109/4236.735987. – ISSN 1089–7801

[Gro] Group, NPD: *Macromedia Flash and Shockwave Players.* `http://www.adobe.com/products/player_census/npd/`. – (abgerufen am 20.03.2012)

[ICA08a] ICAO: *Machine Readable Travel Documents - Part 3: Machine Readable Official Travel Documents, Volume 1, MRtds with Machine Readable DateStored in Optical Character Recognition Format.* 2008

[ICA08b] ICAO: *Machine Readable Travel Documents - Part 3: Machine Readable Official Travel Documents, Volume 2, Specifications for Electronically Enabled MRtds with Biometric Indentifcation Capability.* 2008

[Inf] Informationstechnik, Bundesamt für Sicherheit in d. (Hrsg.) ; Bundesamt für Sicherheit in der Informationstechnik (Veranst.): *Worked Example for Extended Access Control (EAC) PACE, Chip Authentication and Terminal Authentication*

[ISO94] ISO/IEC: *7498-1 Information technology - Open Systems Interconnection - Basic Reference Model: The Basic Model.* Association Française de Normalisation 11 avenue Francis de Pressensé 93571 Saint-Denis La Plaine Cedex France, 1994

[ISO95] ISO/IEC: *10536-4 Identification cards - Contactless integrated circuit(s) cards - Part 4: Answer to Reset and transmission Protocols.* Association Française de Normalisation 11 avenue Francis de Pressensé 93571 Saint-Denis La Plaine Cedex France, 1995

[ISO04a] ISO/IEC: *7816-3 Identification cards - Integrated circuit cards - Part 3: Cards with contacts - Electrical interface and transmission protocols.* Association Française de Normalisation 11 avenue Francis de Pressensé 93571 Saint-Denis La Plaine Cedex France, 04 2004

[ISO04b] ISO/IEC: *7816-4 Identification cards - Integrated circuit cards - Part 4: Organization, security and commands for interchange.* Association Française de Normalisation 11 avenue Francis de Pressensé 93571 Saint-Denis La Plaine Cedex France, 04 2004

[ISO04c] ISO/IEC: *7816-6 Identification cards - Integrated circuit cards - Part 6: Interindustry data elements for interchange*. Association Française de Normalisation 11 avenue Francis de Pressensé 93571 Saint-Denis La Plaine Cedex France, 04 2004

[ISO05a] ISO/IEC: *27001:2005 Information technology — Security techniques — Information security management systems — Requirements*. ISO copyright office Case postale 56 CH-1211 Geneva 20 Tel. + 41 22 749 01 11 Fax + 41 22 749 09 47 E-mail copyright@iso.org Web www.iso.org, 06 2005

[ISO05b] ISO/IEC: *27002:2005 Information technology — Security techniques — Code of practice for information security management*. ISO copyright office Case postale 56 CH-1211 Geneva 20 Tel. + 41 22 749 01 11 Fax + 41 22 749 09 47 E-mail copyright@iso.org Web www.iso.org, 06 2005

[IT02a] ITU-T: *X.680: Information technology – Abstract Syntax Notation One (ASN.1): Specification of basic notation*. `http://www.itu.int/ITU-T/studygroups/com17/languages/X.680-0207.pdf`. Version: 07 2002

[IT02b] ITU-T: *X.690: Information technology – ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)*. `http://www.itu.int/ITU-T/studygroups/com17/languages/X.690-0207.pdf`. Version: 07 2002

[IT04] ITU-T: *X.660: Information technology – Open Systems Interconnection – Procedures for the operation of OSI Registration Authorities: General procedures and top arcs of the ASN.1 Object Identifier tree*. `http://www.itu.int/ITU-T/studygroups/com17/oid/X.660-E.pdf`. Version: 08 2004

[JSF96] J. Steven Fritzinger, Marianne M.: Java Security. In: *Sun Microsystems, Inc.* (1996). `http://www.net.uom.gr/Books/Manuals/whitepaper.pdf`

[Kal98] Kaliski, B.: *PKCS 7: Cryptographic Message Syntax Standard*. 03 1998

[mit] *MIT License*. `http://www.opensource.org/licenses/mit-license.php`. – (abgerufen am 20.03.2012)

[NRM05] Nancy R. Mead, Tehodore R. S. Eric D. Hough H. Eric D. Hough: Security Quality Requirements Engneering (SQUARE) Methodology / Carnegie Mellon Software Engineering Institute, Pidsburgh, PA. USA. 2005. – Forschungsbericht

[pcs11] *M.U.S.C.L.E Project - PC/SC*. `http://www.linuxnet.com/middle.html`. Version: 09 2011. – (abgerufen am 20.03.2012)

[PH10] Pfitzmann, Andreas ; Hansen, Marit: *A terminology for talking about privacy by data minimization: Anonymity, Unlinkability, Undetectability, Unobservability, Pseudonymity, and Identity Management*. http://dud.inf.tu-dresden.de/literatur/Anon_Terminology_v0.34.pdf. Version: August 2010. – v0.34

[RA02] Rivest, Shamir ; Adleman: *PKCS 1: RSA Cryptography Standard*. ftp://ftp.rsasecurity.com/pub/pkcs/pkcs-1/pkcs-1v2-1.pdf. Version: 2002

[RE03] Rankl, Wolfgang ; Effing, Wolfgang: *Smart Card Handbook*. 3. New York, NY, USA : John Wiley & Sons, 2003. – 1088 S. – ISBN 0470856688

[Scha] Schejbal, Jan: *AusweisApp gehackt (Malware über Autoupdate)*. http://janschejbal.wordpress.com/2010/11/09/ausweisapp-gehackt-malware-uber-autoupdate/. – (abgerufen am 20.03.2012)

[Schb] Schejbal, Jan: *ePerso: PIN-Diebstahl ohne Malware*. https://janschejbal.wordpress.com/2011/01/17/eperso-pin-diebstahl-ohne-malware/. – (abgerufen am 20.03.2012)

[SN99] Standards, National I. ; (NIST), Technology: *DATA ENCRYPTION STANDARD (DES)*. http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf. Version: 10 1999

[SN02] Standards, National I. ; (NIST), Technology: *SECURE HASH STANDARD (FIPS PUBS 180-2)*. http://csrc.nist.gov/publications/fips/fips180-2/fips180-2withchangenotice.pdf. Version: 08 2002. – (abgerufen am 20.03.2012)

[Sta] Statowl: *Web Browser Plugin Market Share - Web Browser Plugin Market Penetration and Global Usage*. http://www.statowl.com/plugin_overview.php. – (abgerufen am 20.03.2012)

[W3C] W3C: *XML Protocol Working Group*. http://www.w3.org/2000/xp/Group/. – (abgerufen am 20.03.2012)