

# Testsuite und Conformance Tests für Chipkarten im Gesundheitswesen

Detlef Kraus  
Maximilian Reinders

SRC Security Research & Consulting GmbH  
Graurheindorfer Str. 149A  
53117 Bonn

## 1 Übersicht

Das vorliegende Dokument beschreibt das Design eines Testsystems für nicht-personalisierte und personalisierte Chipkarten wie sie beispielsweise in der Kreditwirtschaft oder ab 2006 auch im Gesundheitswesen eingesetzt werden. Dazu werden im Folgenden die jeweiligen Anforderungen an den Test der Chipkarten und an das Testsystem beschrieben.

## 2 Anforderungen an den Chipkartentest für Gesundheitskarten

### 2.1 Testmethodik

Beim Testen von Programmen werden grundsätzlich die beiden Testmethoden *Whitebox Test* und *Blackbox Test* unterschieden. Bei einem Blackbox Test werden keine Details der Implementation des zu testenden Programms betrachtet. Die Testfälle werden ausschließlich aus der Spezifikation abgeleitet.

Beim Blackbox Test wird das gesamte Programm getestet ohne die Funktionsweise oder Implementierung zu berücksichtigen. Die Funktionsweise einer oder mehrerer zusammenhängender Methoden wird mit Hilfe von sog. Unit Tests überprüft. Es wird generell empfohlen Unit Tests vor der Implementierung (aber nach der Spezifikation) einer Methode zu entwerfen.

Ein Blackbox Test steht typischerweise am Ende einer Entwicklung vor der Auslieferung an den Kunden oder dem Beginn des Rollouts. Blackbox Tests sollten gerade auch unerwartete Eingaben und Randfälle berücksichtigen. Blackbox Tests zeigen, ob das Programm wirklich als Ganzes funktioniert.

Als Nachteil eines Blackbox Tests wird häufig die Tatsache betrachtet, dass Teile des Programms von den Testeingaben überhaupt nicht oder nur in einfachen Fällen berührt werden könnten.

Dagegen wird bei einem Whitebox Test die Programmstruktur berücksichtigt. Insbesondere können beim Whitebox Test einzelne Methoden getestet werden, noch bevor ihr Anwendungskontext fertig ist. Die Testfälle sollten den gewünschten Anwendungsbereich überstreichen und möglichst jeden Zweig des Methodenrumpfes tatsächlich erreichen.

Da Testsuiten für Chipkarten in der Regel unabhängig von der Entwicklung der Betriebssysteme und Anwendungen realisiert werden und gleichzeitig zur Zulassung von Produkten verschiedener Hersteller dienen, können Sie sinnvoller Weise nur auf Blackbox Tests basieren. Whitebox Tests werden von den Entwicklern selbst durchgeführt.

## 2.2 Kurze Begriffsdefinitionen

- Aufbau** bezeichnet den Prozess in der Testvorbereitung, in dem die Datenstrukturen und Dateiinhalte, die zur Durchführung eines Tests auf der Testkarte vorhanden sein müssen, auf diese aufgebracht werden.
- Fehlerfall** bezeichnet einen Testfall, in dem eine Fehlersituation hergestellt und überprüft wird, oder bei dem die externe Welt ein fehlerhaftes Kommando an die Karte absetzt. Der erwartete Wert des Returncodes bzw. eine Auswahl zulässiger erwarteter Werte ist für jeden Fehlerfall separat festgelegt. Eine Karte muss auf eine Fehlersituation oder ein fehlerhaftes Kommando immer mit dem gleichen Returncode reagieren, sofern Wahlmöglichkeiten offen stehen. Jeder nicht als zulässig festgelegte Returncode, insbesondere auch der positive Returncode '90 00', zeigt an, dass der Testfall nicht erfolgreich abgeschlossen wurde.
- Gutfall** bezeichnet einen Testfall, der die spezifikationsgemäße Ausführung eines Kommandos beschreibt, wobei der Returncode in der Regel den Wert '90 00' annimmt. In einigen speziellen Fällen, wo der externe Zugriff auf Kartendaten nicht möglich ist, kann das korrekte Verhalten der Karte jedoch nur dadurch gezeigt werden, dass die Karte das übermittelte Kommando mit einem Fehlercode zurückweist. In diesen Fällen wird der erwartete Returncode separat festgelegt. Jede andere Returncode zeigt an, dass der Testfall nicht erfolgreich abgeschlossen wurde.

<b>Karteneingangszustand</b>	Zustand der Karte bei Testbeginn, d. h. vor der Ausführung der Testvorbereitung zur Ausführung des zu prüfenden Kommandos
<b>Testausgangszustand</b>	Zustand der Karte unmittelbar nach der Ausführung des zu prüfenden Kommandos
<b>Testeingangszustand</b>	Zustand der Karte unmittelbar vor der Ausführung des zu prüfenden Kommandos
<b>Testfall</b>	bezeichnet den einzelnen Test einer Funktionskomponente (Gutfall oder Fehlerfall).
<b>Testkategorie</b>	Gruppe von Testszenarien
<b>Testnachbereitung</b>	Notwendige Kommandosequenz, um den Karteneingangszustand (wieder) zu erreichen.
<b>Testscenario</b>	Gruppe von Testfällen
<b>Testvorbereitung</b>	Notwendige Kommandosequenz, um den Testeingangszustand zu erreichen.

### 2.3 Allgemeine Festlegungen

Ein Test erfolgt immer in der folgenden Sequenz:

Zustand vor der Kommandoausführung	Ausgeführte Kommandosequenz	Zustand nach Kommandoausführung
Karteneingangszustand	Testvorbereitung	Testeingangszustand
Testeingangszustand	Testfall	Testausgangszustand
Testausgangszustand	Testnachbereitung	Karteneingangszustand

Dabei kann der Karteneingangszustand für unterschiedliche Testszenarien und –kategorien verschieden sein. Im Folgenden wird davon ausgegangen, das mindestens die Testvorbereitung und –nachbereitung herstellerneutral möglich sind (anderenfalls bräuchte man "Berge" von Testkarten und es wäre kein automatisierter Test möglich).

Die Testsuite muss in der Lage sein, den jeweiligen Testeingangszustand auf der Karte zu erzeugen. Im Falle einer inhomogenen Kartenlandschaft ist zu klären, wie eine einheitliche Testvorbereitung erfolgen kann.

Zum Erreichen des Karteneingangszustandes stehen zwei Varianten zur Alternative:

Variante 1: Die Testsuite ist in der Lage, die Karte geeignet zu manipulieren. Dadurch entsteht der Vorteil, dass automatisierte Tests möglich sind. Somit erzielt man eine hohe Flexibilität bei der Testerstellung, sobald eine Funktionsbibliothek vorhanden ist. Von diesem Zeitpunkt an ist keine Unterstützung durch den Hersteller mehr notwendig. Nachteilig ist, dass Funktionsbibliotheken pro Hersteller notwendig sind. Für später hinzukommende Hersteller liegen die Schnittstellen dann fest.

Variante 2: Die Hersteller liefern Testkarten mit verschiedenen Konfigurationen. Hierbei hat man den Vorteil, dass keine herstellereigenspezifischen Funktionsbibliotheken erforderlich sind. Nachteil hierbei ist jedoch, dass automatisierte Tests durch Kartenwechsel eingeschränkt werden. Insgesamt hat man eine geringere Flexibilität bei der Erstellung von Testfällen (Konfigurationen müssen zuerst festgelegt werden) und bei den Herstellern fällt wiederholt Aufwand zur Bereitstellung geeigneter Ladesequenzen (zur Erstellung von Karten mit geeignetem Testeingangszustand) an.

### 2.3.1 Unterscheidung der Testfälle

Testfälle werden wie folgt unterteilt und bezüglich ihrer Varianz beschrieben:

Gutfälle:

Test aller verschiedenen gültigen Parameter P1, P2 (Übersichtsmatrix: Welche Parametrisierung wird in welchem Testfall verwendet ?)

Test aller gültigen Varianten von Kommandodaten

Durchzuführende Prüfung pro mögliche Kommandovariante:

Ist alles, das verändert sein muss, korrekt verändert ?

Ist nichts, das nicht verändert werden soll, verändert ?

Fehlerfälle (immer nur eine gezielte Fehlereinstreuung):

Fehler in der Kommandosyntax (z.B. unzulässige Parameter P1, P2, unzulässig fehlende oder vorhandene Lc- und/oder Le-Bytes)

Fehler in den Kommandodaten (z.B. unzulässige Formate, Datenobjekte, Reihenfolgen, etc.)

Fehler in der Kartenstruktur (z.B. unzulässige Struktur oder Fehlen von Dateien, die das Kommando im Zuge der Kommandoausführung verwendet)

Fehler in Kartendaten (z.B. unzulässige Dateiinhalte in Dateien, die das Kommando im Zuge der Kommandoausführung verwendet)

Fehler im Kartenzustand (z.B. Kommando(-variante) ist an einen bestimmten Kartenzustand oder DF-Namen gebunden)

Fehlerfälle werden für personalisierte Karte nur teilweise verwendet, insbesondere nimmt man dort an, dass keine Fehler in Kartenstruktur und Kartendaten vorliegen. Die entsprechende Prüfung erfolgt durch geeignete Gutfälle (Inhalts- und Strukturprüfung)

### **2.3.2 Testvorbereitung**

Im Rahmen der Testvorbereitung sind gegebenenfalls Modifikationen an Kartendaten oder Kartenzuständen durchzuführen, z. B. wenn an einem EF andere Zugriffsregeln festzulegen sind oder wenn spezielle Sicherheitszustände für die Ausführung des Kommandos erforderlich sind.

Ferner wird der interne Status der Karte, d. h. der Inhalt der relevanten Dateien vor Testdurchführung, für eine spätere Überprüfung protokolliert.

Die Testsuite muss in der Lage sein, den jeweiligen Testeingangszustand auf der Karte zu erzeugen. In einer inhomogenen Kartenlandschaft ist zu klären, wie eine einheitliche Testvorbereitung erfolgen kann. Prinzipiell ist die Testvorbereitung auf die Verwendung der interoperablen Kommandos eingeschränkt.

### **2.3.3 Testdurchführung und Ergebnisauswertung**

Es wird eine Kommando APDU aufgebaut und an die Chipkarte geschickt. Die Response APDU der Karte wird vom Testsystem für eine folgende Auswertung in Empfang genommen.

Es wird überprüft, inwieweit die Testergebnisse mit den erwarteten Ergebnissen übereinstimmen:

- Der Returncode der Antwortnachricht wird überprüft.
- Bei Gutfällen wird die Korrektheit der Antwortdaten der Chipkarte überprüft. Bei Fehlerfällen wird geprüft, dass keine Antwortdaten von der Chipkarte ausgegeben werden.
- Sofern möglich, werden die durch das Kommando auf der Chipkarte modifizierten Daten geprüft.

### 2.3.4 Testnachbereitung

Grundsätzlich wird der vordefinierte Testaufbau nach Beendigung des Testfalles wiederhergestellt, sofern er im Ablauf modifiziert wurde. Ferner ist in dieser Phase darauf zu achten, dass die Karte nach Beendigung des Testfalles in einem Zustand ist, der den ordnungsgemäßen Beginn des nächsten Testes gestattet.

Die Testsuite muss in der Lage sein, den jeweiligen Karteneingangszustand auf der Karte (wieder) herzustellen. In einer inhomogenen Kartenlandschaft ist zu klären, wie eine einheitliche Testnachbereitung erfolgen kann. Prinzipiell ist die Testvorbereitung auf die Verwendung der interoperablen Kommandos eingeschränkt.

## 2.4 Systemtest (Testsuite für nicht-personalisierte Karten)

Die Kategorie Systemtest stellt den Plattformtest für die unterschiedlichen COS-Plattformen dar.

Dieser Test dient dem Nachweis der korrekten Funktionalität der zu implementierenden Kommandos sowie Prüfung der weiteren funktionalen Anforderungen, die sich aus Teil 1 der Spezifikation der elektronischen Gesundheitskarte<sup>1</sup> ergeben:

- Schnittstellenkommandos (Kapitel 4, Anhang C)

Es sind die jeweils zulässigen Kommandovarianten und Parametrisierungen zu testen. Insbesondere sind auch die vorgesehenen Returncodes für Fehlersituationen zu prüfen.

Ferner müssen geeignete Test für die optionalen Kommandos durchgeführt werden.

- Zu unterstützende Dateitypen (Kapitel 5)

Die beschriebenen Datei-Typen mit den zugehörigen Grenzbereichen (min./max. Länge, etc) müssen nutzbar sein. Die Prüfung erfolgt über geeignete Schnittstellenkommandos. Offen ist die Frage, wie das Anlegen und Löschen von Dateien durch das Testsystem bewerkstelligt werden kann.

- Sicherheitsattribute (Kapitel 6)

Kombinationen von Sicherheitsattributen sind intensiv zu testen, um Funktionseinschränkungen für die Zukunft zu vermeiden. Insbesondere ist "Kreativität" bei Design der zu testenden Zugriffsregeln erforderlich, da die zukünftig

---

<sup>1</sup> Electronic Health Card Specification, Part 1 Commands, Algorithms and Functions of the COS Platform, Version 1.2 Draft (Pre-Final-Version), 20.01.2005

benötigten Zugriffsbedingungen noch nicht bekannt sind. Der Test muss gerade in diesem Bereich eine sehr hohe Abdeckung haben.

- PIN-Management (Kapitel 7)  
Die geforderte Funktionalität ist zu verifizieren, insbesondere die Reaktion auf Syntaxfehler und PIN-Fehleingaben. Das Resetting Code Verfahren ist zu prüfen. Offen ist hier die Frage, wie ggf. interne Daten wie Fehlbedienungsähler manipuliert werden können.
- Security Environments (Kapitel 8)  
Mechanismus prüfen
- Secure Messaging (Kapitel 9, Anhang A)  
Funktionalität ist zu prüfen, kryptographische Operationen der Karte (MAC-Berechnung und/oder Verschlüsselung) sind zu verifizieren.
- Algorithmen (Kapitel 10)  
Algorithmen sind zu prüfen, kann im Rahmen der Kommando- und/oder Secure Messaging-Tests erfolgen.
- Control Reference Templates (Kapitel 11)  
Die Funktionalität, CRT's für ein Security Environment mit dem Kommando Manage Security Environment zu setzen ist zu verifizieren.
- Personalisierung, Kartenmanagement und Nachladen (Kapitel 12)  
Die Mechanismen zum Kartenmanagement sind zu verifizieren. Hier ist insbesondere ein sehr großer Aufwand zu erwarten, da die Mechanismen herstellerabhängig bleiben.
- Elektrische Eigenschaften und Übertragungsprotokolle (Kapitel 13)  
Zu unterstützende Spannungsbereiche, Einhaltung des Protokollparameter, geforderte Übertragungsraten sind zu verifizieren
- Command chaining (Kapitel 14)  
Die Unterstützung für die genannten Kommandos ist zu prüfen.
- Empfohlene Erweiterungen (Anhang E)  
Geeignete Test sind auszuführen.

Tests dieser Kategorie sollten bewusst auch mit anderen Dateiparametern ausgeführt werden, als derzeit für die realen Anwendungen vorgesehen. Dies ermöglicht eine größere Sicherheit für zukünftige Entwicklungen.

## 2.5 Funktionstest (Testsuite für personalisierte Karten)

Dieser Test dient der Prüfung der korrekten Ausführung einer im Produktionsprozess erzeugten Karte. Insbesondere sollen die Dateiinhalte der Anwendungsdateien auf Korrektheit geprüft werden, es soll die korrekte Funktionsfähigkeit dieser Karte in den definierten Terminalabläufen, welche sich aus den Teilen 2 bis 4 der Gesundheitskartenspezifikation<sup>2</sup> ergeben, verifiziert werden. Dies erfordert die Bereitstellung von Referenzdaten und Referenzschlüsseln der personalisierten Karte für das Testsystem.

Tests dieser Kategorie sind im wesentlichen Gutfall-Tests. Karteninhalte sollen nur in einer Weise manipuliert werden, wie sie für die Verwendung im Feld auch möglich sein soll.

Der Funktionstest muss mit jeder neuen Anwendung für die Karte erweitert werden.

Insbesondere sind die folgenden Bereiche abzudecken:

- Dateien auf MF-Level mit globalen Datenobjekten und CV-Zertifikaten
- Dateien der Health Care Application (DF.HCA)
- Dateien der Signatur Anwendung (DF.ESIGN)
- Dateien für kryptographische Informationen zur Signatur-Anwendung (DF.CIA.ESIGN)
- Prüfung der Einhaltung der vorgeschriebenen Zugriffsbedingungen, insbesondere Sicherstellung, dass auf bestimmte Daten nicht oder nur nach Authentisierung zugegriffen werden kann.
- Prüfung der Abläufe zur Card-to-Card-Authentisierung
- Prüfung der Abläufe zur Karteninhaber-Authentisierung (PIN-Prüfung)
- Prüfung der Abläufe zum Kartenmanagement (Update Versichertenstammdaten)
- Prüfung der Nachstrukturierung
- Prüfung des Rezept-Handlings für den Kartentransport
- Prüfung des Rezept-Handlings für den Servertransport

---

<sup>2</sup> Electronic Health Card Specification, Part 2: Basic Applications and Functions, Version 0.8 (Draft-Version), 02.01.2005

Electronic Health Card Specification, Part 3: Electronic Prescription, Version 0.7 (Draft), 09.01.2005

Electronic Health Card Specification, Part 4: Voluntary Applications (currently no public version available)

### 3 Anforderungen an das Testsystem

Ein Testsystem besteht aus den folgenden funktionalen Komponenten

- Oberfläche (GUI)
- Datenbank (für Protokolle, ggf. auch Testskripte und Funktionsbibliotheken)
- Testprogramm (idealerweise skriptbasiert, geeignet für automatisierten Betrieb)
- Kartenleser (High Level mit Möglichkeiten für Protokolltest)

#### 3.1 Anforderung GUI

Die GUI ist das Zentrale Programm, welches die Komponenten Datenbank und Testprogramm ansteuert, sie ermöglicht den Austausch von Daten zwischen verschiedenen Teststellen (sofern datenbanktechnisch möglich). An die GUI werden insbesondere die folgenden Anforderungen gestellt:

- Erfassung von herstellereigenen Daten:  
Pro Hersteller müssen die jeweils spezifischen Daten, wie z.B. Name, Adresse, Chip- und Maskenbezeichnung erfasst und in der Datenbank verwaltet werden.
- Erfassung von herstellerneutralen Referenzdaten und Referenzschlüsseln  
Personalisierte Testkarten enthalten Referenzdaten und –schlüssel. Diese müssen ebenfalls erfasst und in der Datenbank verwaltet werden.
- Eignung für automatisierten Betrieb  
Große Testsequenzen müssen automatisch ablaufen und die jeweiligen Ergebnisse protokolliert werden. Fehler in einem einzelnen Testfall sollten keine Auswirkung auf den oder die folgenden Testfälle haben. Daher muss zum Ende eines Testfalls immer ein definierter Zustand erzeugt werden.
- Erzeugung von Testberichten  
In der Datenbank werden Herstellerdaten und Protokolle verwaltet. Wünschenswert ist die automatische Generierung eines Testreports, der für ein dediziertes Kartenprodukt den ausgeführten Testumfang und die aufgetretenen Fehler beinhaltet.

- **Import-/Export-Schnittstelle**

Das Testsystem muss mindestens über eine Importschnittstelle verfügen, über die neue Testfallskripte aufgenommen werden können. Die Importschnittstelle muss die Funktionsfähigkeit der einzubringenden Testfallskripte verifizieren können.

Gegebenenfalls ist auch eine Exportschnittstelle zum Austausch von Daten zwischen zwei Testsystemen sinnvoll. Je nach Vernetzung von Datenbankinhalten kann jedoch der Import von exportierten Daten sehr aufwändig sein.

- **Benutzerfreundlichkeit**

Die GUI muss einfach zu bedienen sein. Mehrfacheingaben gleicher Daten sollten vermieden werden (z.B. identische Herstellerdaten für unterschiedliche Masken).

### **3.2 Anforderungen an die Datenbank**

Es gibt keine besonderen Anforderung an die Datenbank. Üblicherweise werden Standard-Datenbanken, wie z.B. SQL verwendet.

### **3.3 Anforderung an das Testprogramm**

Das Testprogramm kommuniziert über den Kartenleser mit einer Chipkarte. Dabei erhält es die Eingabedaten (Testabläufe) aus der Datenbank und stellt seinerseits Protokolldaten für die Datenbank bereit. Das Testprogramm muss interpreterbasiert sein, um Testabläufe in einer Skriptform und unabhängig vom Testprogramm erzeugen zu können.

Das Testprogramm muss daher über die folgenden Eigenschaften verfügen:

- **Nachbildung der Kartenfunktionalität**

Die Kartenfunktionalität muss vollständig nachgebildet werden, um die Aktionen der Chipkarte überprüfen zu können. Gleichzeitig müssen auch "Terminal"-Funktionalitäten verfügbar sein, um die Kommandonachrichten an die Karten erzeugen zu können.

Mindestens die folgenden Funktionalitäten sind bereitzustellen:

- Funktionen zur Generierung von Kommando - APDUs für Chipkarten
- Funktionen zur Prüfung von Antwort – APDUs von Chipkarten
- Kryptographische Funktionen

RSA (mindestens 2048 Bit) und Triple-DES als Standardalgorithmen sind zu unterstützen. Zusätzlich müssen Funktionen zur Bereitstellung von Kommandodaten z.B. Signaturen wie auch zur Prüfung von Antwortdaten der Chipkarte z.B. Signaturen vorhanden sein.

- Bereitstellung von Standardfunktionen  
z.B. TLV – Operationen, Bit – und Byte – Manipulationen, etc.
- Funktionen zur Einstellung von Betriebsspannung, Betriebsfrequenz und Übertragungsrate
- Erweiterbarkeit um neue Funktionen (ggf. auch Kryptoalgorithmen)
- ...

### **3.4 Anforderung an den Kartenleser**

Der Kartenleser muss in der Lage sein, die Transportprotokolleigenschaften zu überprüfen. Er muss ferner mit unterschiedlichen Spannungen (idealerweise 5V, 3V und 1,8 V) sowie verschiedene Übertragungsraten und –frequenzen unterstützen.

Die vorstehenden Eigenschaften müssen automatisch konfigurierbar sein, um den Kartentest automatisiert ablaufen lassen zu können.