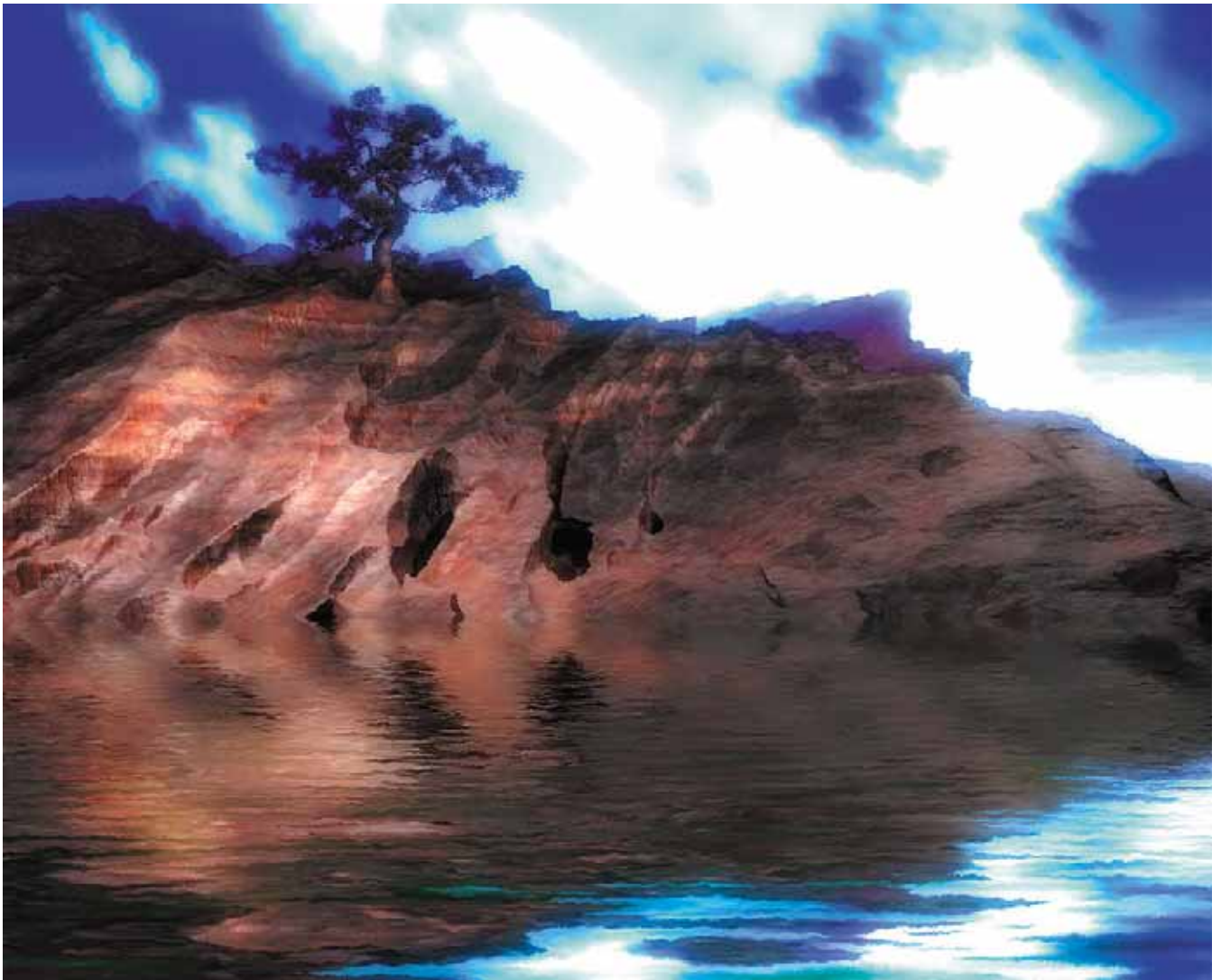


Inside Smartcards

# Der steinige Weg zur Interoperabilität



Studiert man die Datenblätter diverser Hersteller von Security-Anwendungen tauchen immer wieder dieselben Begriffe wie CSP, PKCS#11, PKCS#15, ISO7816-4, usw. auf. All dies sind Standards die sich mit kryptografischen Schnittstellen befassen.

Man könnte fast meinen, dass diese aus Marketinggründen gelistet werden, jedoch was verbirgt sich hinter diesen Standards und welchen Vorteil hat der Hersteller der Anwendung, als auch Anbieter von Hardware (Smartcards, Token) durch die Unterstützung dieser Standards?

## Die Schnittstellen zur Applikation

Damit eine Applikation die kryptografische Funktionalität einer Smartcard nutzen kann, hat diese mehrere Möglichkeiten der Implementierung. Zunächst kann natürlich direkt auf die Karte zugegriffen werden. Die Kommunikation findet dabei mittels kartenspezifischer Befehle (APDUs – Application Protocol Data Unit) statt. Für jeden weiteren Smartcard-Typ, der von der Applikation unterstützt werden soll, muss die Applikation um diesen Kartentyp erweitert werden. Aus Gründen der Wartung und Interoperabilität wird diese Möglichkeit nicht weiter betrachtet.

Soll die Applikation auf eine kryptografische Schnittstelle zugreifen, bietet sich zum einen ein CSP oder die Verwendung von PKCS#11 an. Beide Schnittstellen bieten auf eine abstrahierte allgemeine Weise kryptografische Funktionalitäten (ver-/entschlüsseln, signieren, ..), die auf Objekte (Schlüssel) angewendet werden können. Die Karten werden dadurch auf logischer Ebene einheitlich und brauchen keine unterschiedliche Behandlung (Treiber, ..). Die eigentlichen technischen Unterschiede und Details werden innerhalb der Implementierung der Schnittstelle realisiert. Die Wahl der Schnittstelle hängt hauptsächlich von der Art der Applikation ab und unterscheidet sich durch die Art der Anwendung dieser.

PKCS#11 ist der Cryptographic Token Interface (Cryptoki) Standard aus einer Reihe von Standards (Public-Key Cryptography Standards), die von RSA Security in Kooperation mit Industrie und Ämtern entwickelt wurde. Der Aufbau basiert auf Ob-

jekten, die man in drei Klassen unterteilen kann: Daten-, Zertifikats- und Schlüsselobjekte (Public Key, Private Key, Secret Key). Der Zugriff erfolgt über so genannte Slots, die den Smartcard-Lesern entsprechen, in denen jeweils ein Token vorhanden sein kann. In der Spezifikation wird der Begriff „Token“ als allgemeine Abstraktion kryptografischer Hardware verwendet, im weiteren Verlauf des Artikels liegt jedoch das Augenmerk speziell auf Smartcards. Der CSP (Cryptographic Service Pro-

men der Smartcard unterstützt werden, und dies natürlich auch nur dann wenn die Karte eingesteckt ist, oder aber auch Software-Tokens zu implementieren, die dann immer zur Verfügung stehen.

Der CSP findet in Microsoft-Anwendungen seinen Einsatz wie beispielsweise Outlook (Email Security), Internet Explorer (SSL), Smartcard Logon (zertifikatsbasierte Anmeldung über Microsoft GINA). Auf der anderen Seite ist PKCS#11 ein allgemeiner Standard, der auch bei

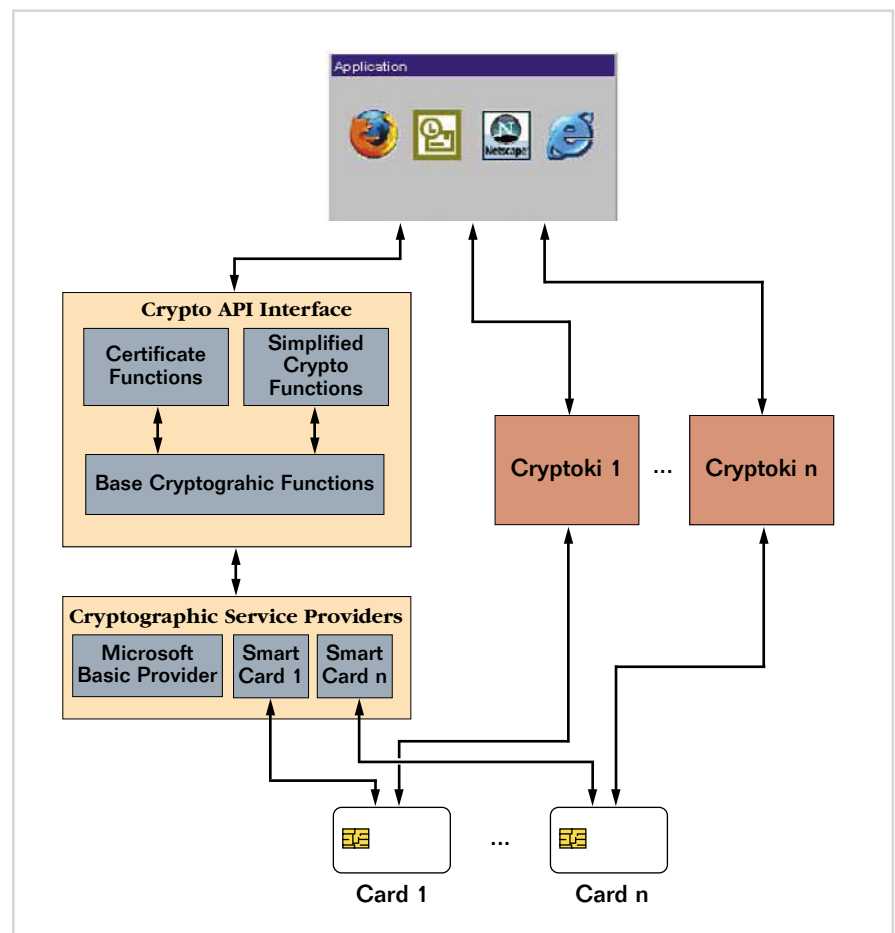


Bild 1: Schnittstellen der Applikation.

vider) ist ein installierbares Modul der Microsoft CryptoAPI, die wiederum die Schnittstelle bereitstellt. Wie auch in der Bild 1 erkennbar, bietet Microsoft einen „Microsoft Base Provider“, der kryptografische Operationen in Software realisiert und somit auch als kryptografische Bibliothek verwendet werden kann. In PKCS#11 bleibt es dem Ersteller überlassen, ob nur die Mechanis-

Nicht-Microsoft Betriebssystemen (Linux) und Applikationen (etwa Netscape, Mozilla, PGP, Lotus Notes, ...) seine Verwendung findet.

Wie so oft gibt es nun mehrere Standards oder Implementierungen zum selben Thema, die sich durchgesetzt haben. Es stellt sich die Frage, wofür der ganze Aufwand – für ein paar Objekte die auf einer Karte gespeichert sind? Kann es nicht einfach

wie zum Beispiel bei einem USB-Stick funktionieren, der die Mass Storage Class von USB unterstützt? Dazu sollte man unterscheiden, dass Smartcards ein eigenes Betriebssystem auf dem Chip implementiert haben (Card Operating System – COS), das den eigentlichen Unterschied der Smartcard Typen darstellt und – ja – es gibt dafür auch einen Standard über den Aufbau der Karte und der Kommandos. Dies ist in ISO/IEC 7816-4 .. 9 enthalten, jedoch lässt der Standard einige Interpretationsmöglichkeiten offen und es gibt auch Möglichkeiten für proprietäre Kommandos. Die Hersteller der Kartenbetriebssysteme erfüllen zumeist nur zum Teil die Spezifikation, sei es aus Gründen der Implementierung oder der Substituierbarkeit, um sich nicht direkten Preiskämpfen auszuliefern.

### Smartcards als Teil der Sicherheitslösung

Die Smartcard dient dazu, Daten und Schlüssel auf der Karte abzulegen und anzuwenden. Wie dies erreicht wird, soll ein kleiner Exkurs in den Standard ISO/IEC 7816-4 zeigen. Eine Smartcard hat eine ähnliche Verzeichnisstruktur wie man sie schon vom PC kennt. Es gibt ein Root-Verzeichnis, das Master File (MF), Unterverzeichnisse (Dedicated File – DF) und Dateien (Elementary Files –EF), die man in unterschiedlichen Ebenen anordnen kann.

Jedes dieser Files hat FCPs (File Control Parameters), die bei der Erstellung des Files angegeben werden. Darin enthalten sind Informationen wie die Größe, die Struktur, die File-ID und vor allem die Security Attributes, in denen die Zugriffsrechte

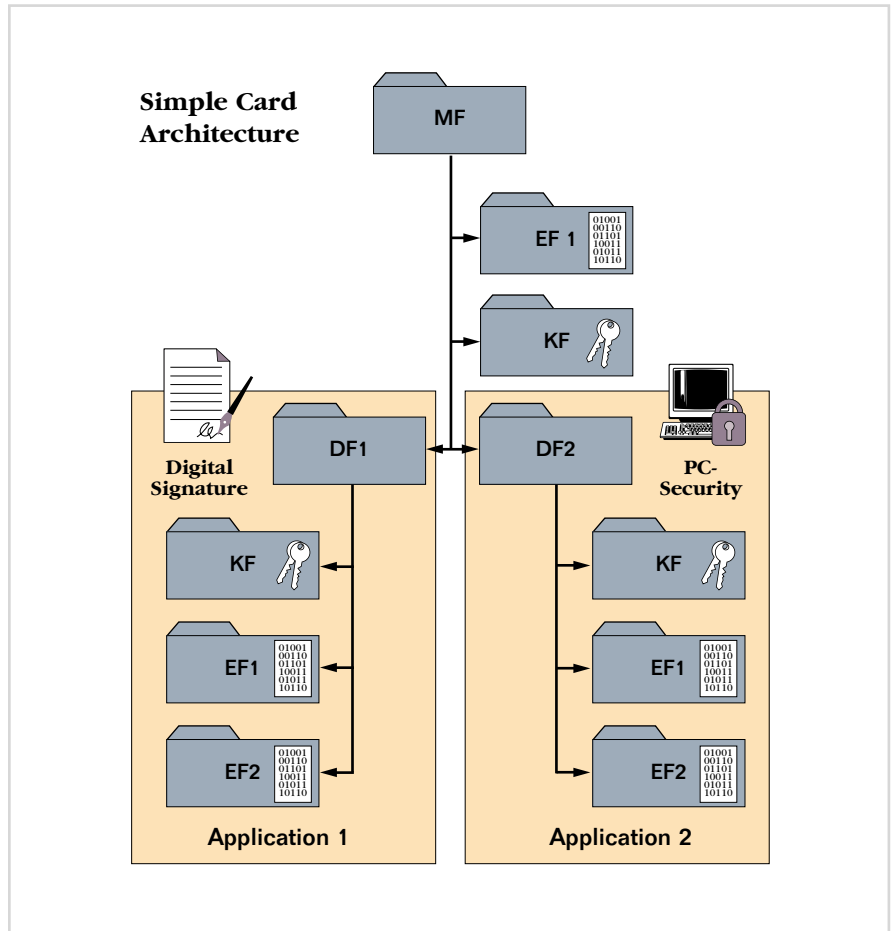


Bild 2: Verzeichnisstruktur einer Karte.

spezifiziert sind. Da eine Applikation den Datenhaushalt einer Karte nicht unbedingt kennen muss, werden diese FCPs auch wahlweise bei der Auswahl eines Files zurückgegeben.

EFs können verschiedene Strukturen haben. Bei transparenter Struktur steht ein Byte adressierbarer Datenspeicher zur Verfügung. Recordorientierte Files haben einen tabellarischen Aufbau bei dem jeder Record mit einer ID versehen wird und über diesen ansprechbar ist. Bei TLV (Tag Length Value) orientierten Files werden die Datenelemente über die Tags angesprochen.

Jedes File hat eine File-ID, die innerhalb des Verzeichnisses eindeutig sein muss. Wird ein File ausgewählt (mittels eines Select-Kommandos) wird diese File-ID angegeben. Die File-IDs sind bei dem Erstellen der Files frei wählbar, mit Ausnahme des MasterFiles, das immer die ID 0x3F00 besitzt.

Bei DFs gibt es zusätzlich noch den DF-Namen über den diese auch ausgewählt werden können. Dieser besteht aus einer Registered-ID, die beim ISO-Konsortium beantragt werden kann und einer Proprietary Extension, die frei wählbar ist. Enthält

„Wie die Schlüssel auf der Karte abzulegen sind und welche Rechte für Schlüssel vorhanden sind, ist in dem Standard ISO/IEC 7816-4 nicht beschrieben. Somit ist der Hersteller des Kartenbetriebssystems zu eigenen Interpretationen gezwungen.“

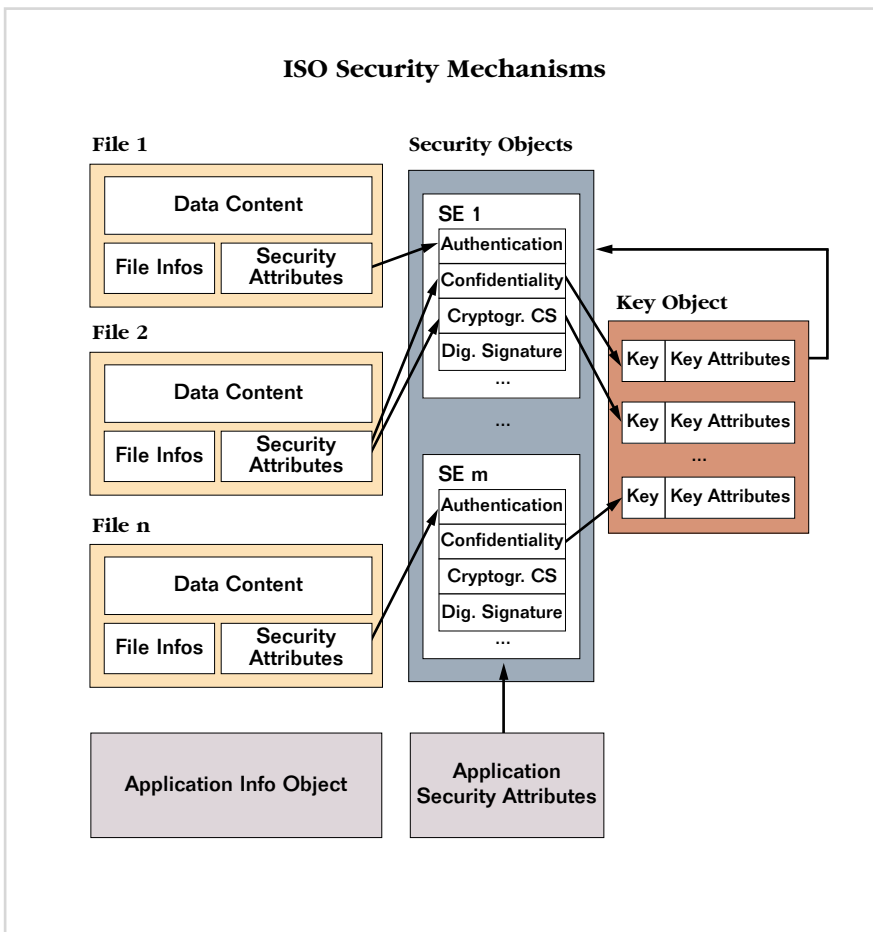


Bild 3: ISO Sicherheitsmechanismen.

das DF eine Applikation, kann diese über den DF-Name immer eindeutig identifiziert werden.

### Die eigentliche Sicherheitsstruktur

Innerhalb der Security Attributes (siehe Bild 3) liegt die eigentliche Sicherheitsstruktur, die auch eine Verkettung von mehreren Objekten erlaubt. Darin enthalten sind im einfachen (compact) Format so genannte Access Condition Bytes, die angeben, ob für eine bestimmte Aktion, die auf das File angewendet wird, eine Sicherheitsrichtlinie existiert. Solche Aktionen können beispielsweise das Erstellen, Löschen, Lesen und Schreiben von Verzeichnissen oder Files sein. Wenn keine Sicherheitsrichtlinie für das File existiert, dann darf diese Aktion nicht ausgeführt werden. Diese Sicherheitsrichtlinien sind in dem Security Condition Byte enthalten, wobei man hier die Aus-

wahl zwischen verschiedenen Sicherheitsmechanismen hat. Dabei kann man zwischen External Authentication, Secure Messaging und User Authentication wählen.

Bei External Authentication wird der Karte gezeigt, dass diese mit einer authentischen Gegenstelle (Applikation) zu tun hat, indem ein Geheimnis verifiziert wird. Angestoßen wird dies durch die Applikation, da der Kommunikationsablauf immer im Rahmen des Challenge Response Verfahrens ablaufen muss. Dabei wird von der Applikation eine Zufallszahl von der Karte angefordert (mittels GetChallenge) und diese dann mit einem geheimen Schlüssel verschlüsselt. Diese verschlüsselte Zahl wird dann wieder zurück an die Karte gesendet, die dann verifizieren kann ob der Schlüssel richtig ist, indem beispielsweise die selbe Operation in der Karte ausgeführt wird und die beiden Ergebnisse verglichen werden. User Authentication

stellt sicher, dass der Benutzer ein Geheimnis (PIN) kennt, das an die Karte (mittels des Kommandos Verify) gesendet wird.

### Secure Messaging

In diesem Zusammenhang ist der dritte Sicherheitsmechanismus sehr interessant. Secure Messaging dient dazu einen sicheren Transportkanal (ähnlich wie VPN) zur Karte zu ermöglichen. Wird dies beispielsweise bei der Übertragung der PINs nicht verwendet, erfolgt die Übertragung in Plaintext und kann einfach mitprotokolliert werden. Bei Secure Messaging hat man die Möglichkeit die Authentizität und die Vertraulichkeit der Daten sicherzustellen. Dies erfolgt durch die Bildung einer kryptografischen Prüfsumme und durch das Verschlüsseln der Daten. Für die Verwendung von Secure Messaging muss man zuvor der Karte mitteilen welche Schlüssel verwendet werden sollen. Dazu wählt man das Security Environment (SE) über das die Schlüssel referenziert sind und setzt im Kommando ein Flag, das anzeigt, dass Secure Messaging für dieses Kommando verwendet wird. Meist wird in der Prüfsumme noch ein Zähler hinzugefügt, der bei jedem Kommando inkrementiert wird um Replay Attacks zu verhindern.

Die zuvor angesprochenen Security Environments beinhalten CRTs (Control Reference Templates). Diese beschreiben für den jeweiligen Mechanismus (Authentication, Confidentiality und Cryptographic Checksum für Secure Messaging, ...) welcher Schlüssel (über eine KeyID) zu verwenden ist und wo dieser zu finden ist. Innerhalb jedes DFs befinden sich SEs und es kann selbstverständlich auch zu anderen DFs verwiesen werden. Jedes SE hat eine eigene ID, die innerhalb des Security Condition Bytes angegeben wird.

Wie die Schlüssel auf der Karte abzulegen sind und welche Rechte für Schlüssel vorhanden sind, ist leider in dem Standard nicht beschrieben. Somit ist man auch wieder bei

dem Problem, dass hier der Hersteller des Kartenbetriebssystems zu eigenen Interpretationen gezwungen ist. Meistens werden die Schlüssel in Files abgelegt. Je nach Schlüsselinhalt haben diese dann einen anderen Aufbau. Auch was die Rechte anbelangt sind diese etwas unterschiedlich, da beispielsweise auch die Verwendung eines Schlüssel restriktiert werden kann. Um das obige Beispiel nochmals heranzuziehen, könnte ein PIN in dem Execute Recht die Bedingung Secure Messaging angeben und somit muss ein sicherer Kanal für die Verifikation hergestellt werden. Eine Veranschaulichung und mögliche Verkettungen sind in Bild 3 dargestellt.

Ist einmal eine Sicherheitsbedingung erfüllt, so muss diese nicht für jedes Kommando neu durchgeführt werden. Der Status bleibt so lange erhalten bis ein anderes DF ausgewählt wird oder die Karte zurückgesetzt wird.

Es sei hier noch zur Vollständigkeit erwähnt, dass Files auch eine Lifecycle State haben. Auch die Security Environments beziehen sich auf einen bestimmten Lifecycles State eines Files. Somit können auch für verschiedene Lifecycle States eines Files unterschiedliche Sicherheitsrichtlinien vergeben werden. Dies ist sinnvoll, wenn beispielsweise die Personalisierung in mehreren Etappen erfolgt. Gewechselt wird der Lifecycle State mit eigenen Kommandos (activate, deactivate, terminate) die natürlich selbst auch der Rechtestruktur unterlegen sind.

Wie man sieht ist eine Smartcard ein äußerst sicherer Ort um Daten abzulegen. Es geht noch weiter, denn nicht nur das Kartenbetriebssystem,

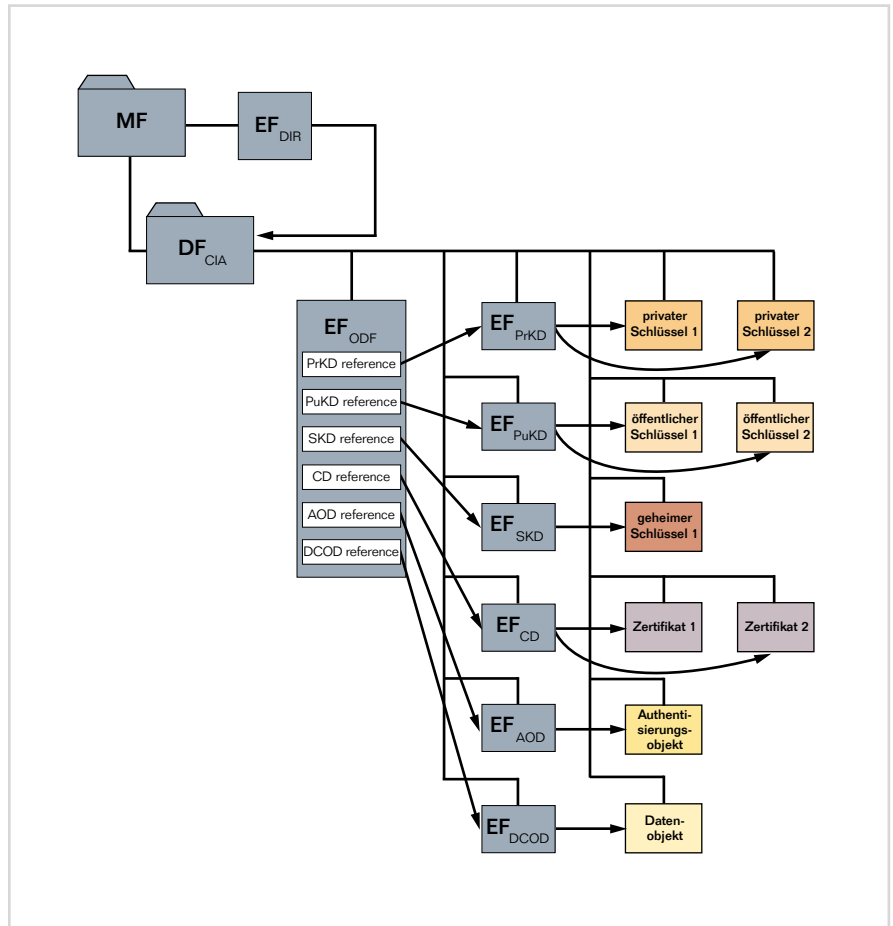


Bild 4: Struktur der ISO7816-15 Applikation.

sondern auch der Chip selbst verfügt über einige Sicherheitsmerkmale. So befinden sich Sensoren am Chip, die das Öffnen des Prozessors detektieren, wie auch der Stromverbrauch zufällig reguliert wird um DPA (Differential Power Analysis) Attacken entgegenzuwirken.

### ISO7816-15 : Die Umsetzung auf der Karte

Die Chipkartennorm ISO7816-15 mit dem Titel „Cryptographic Information Application“ beschreibt den Aufbau einer kryptografischen Applikation

mit dazugehörigen Datenformaten auf einer Smartcard. Diese Norm basiert auf dem Standard PKCS#15 wobei Komponenten die nicht auf Smartcards zutreffen entfernt wurden, jedoch auch um einige Bereiche erweitert wurde (etwa Authentifizierung). Ziel ist es verschiedene Schlüssel, Zertifikate und auch Datenobjekte innerhalb einer Applikation auf einer Karte zu verwalten und dabei auch mit Sicherheitsmechanismen zu schützen. Dazu muss die Struktur modular und erweiterbar sein, als auch die Daten mit einer ausreichenden Beschreibung verse-

„Eine Smartcard ist ein äußerst sicherer Ort um Daten abzulegen. Es geht noch weiter, denn nicht nur das Kartenbetriebssystem, auch der Chip selbst verfügt über einige Sicherheitsmerkmale.“

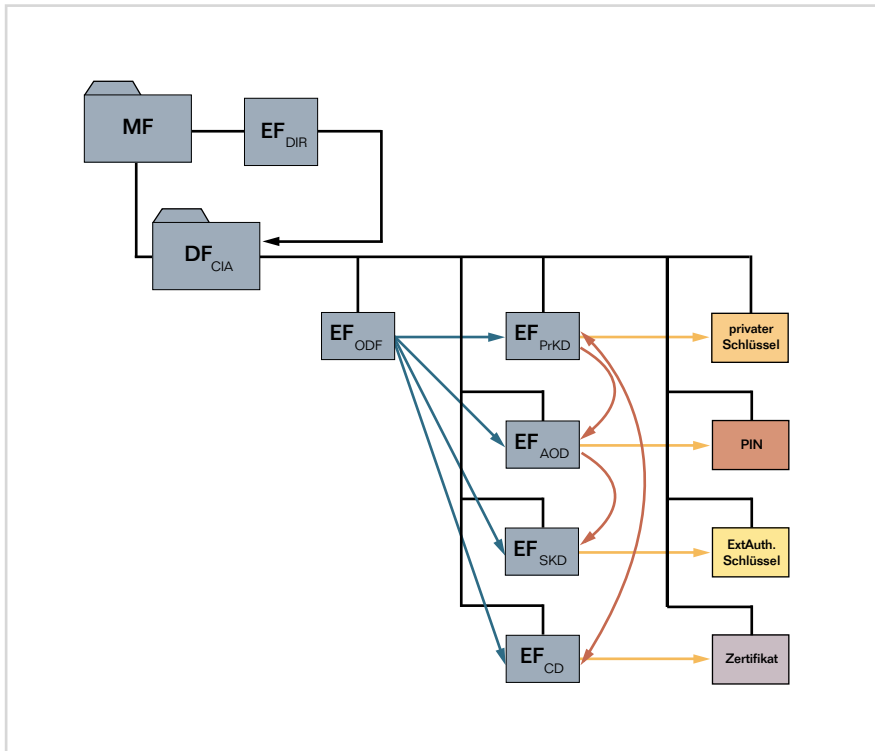


Bild 5: Beispiel einer Verketzung.

hen sein müssen. Als Grundlage für die Darstellung der Daten wurde die Formatbeschreibung ASN.1 gewählt.

Die Struktur der kryptografischen Applikation ist innerhalb eines DFs (siehe Bild 4), dessen DF-Name, für eine genaue Identifizierung, einen bestimmten Inhalt haben muss. Die erste Anlaufstelle innerhalb der Applikation ist das EF<sub>ODF</sub> (Object Directory File). Dieses hat eine bestimmte File-ID (0x5301) und darin befinden sich nicht anderes als Referenzen (File-IDs) auf Verzeichnisfiles. Man unterscheidet Verzeichnisfiles für PrivateKey, PublicKey, SecretKeys, Certificates, AuthenticationObjects und DataContainerObjects. Sie enthalten CIOs (Cryptographic Information Objects), die eine komplette Beschreibung des Objektes (und auch wie dieses gesichert ist) darstellen und mitunter auf das eigentliche Datenobjekt referenzieren, das wiederum in einem eigenen EF enthalten ist. Es ergibt sich somit eine Struktur, die beliebig erweiterbar ist und über die Referenzen die Objekte auch untereinander verknüpfbar sind, da natürlich Zertifikate in Verbindung mit privaten und öffentlichen Schlüs-

seln stehen und über Authentisierungsobjekte geschützt sein können.

Ein Beispiel eines Zusammenhanges der Objekte ist in Bild 5 veranschaulicht. Einem Zertifikat ist ein privater Schlüssel zugehörig. Für die Verwendung des privaten Schlüssels, wie beim Erstellen einer Signatur, wird auf ein Authentication Object referenziert, in dem die Verwendung ei-

nes PINs definiert ist. In dem Authentication Object kann wiederum eine Sicherheitsbedingung (External Authentication) für die Verwendung des PINs enthalten sein, wodurch auf ein Secret Key Object referenziert wird.

## Fazit

Schon wenn man bei den oben angeführten Themen an der Oberfläche kratzt, scheinen diese nicht ganz trivial zu sein und auch in der Umsetzung ist mit einigem Aufwand zu rechnen. Doch wie man sieht, ist jeder in der Kette daran angehalten, sich diesen Standards zu unterwerfen. Will ein Hersteller einer Anwendung diese auf breiter Ebene anbieten, sollte die Unterstützung von PKCS#11 und/ oder CSP nicht fehlen. Dementsprechend ist ein Hersteller von Smartcards angehalten für seine Produkte einen CSP und einen PKCS#11 Treiber zur Verfügung zu stellen. Auch bei der Einführung eines Smartcard Systems sollte man bedenken, ob weitere Zertifikate unterstützt werden sollen und die Verwendung einer ISO/IEC 7816-15 Struktur vorgesehen. Der Weg zur Interoperabilität ist steinig, jedoch zahlt es sich aus.

DI (FH) Alexander Chaloupka  
ac@cryptas.com

## Links

### Produkte:

<http://www.cryptoshop.com/de/products/cardtoken/index.php>  
<http://www.cryptoshop.com/de/products/software/pkcs11csp/index.php>  
<http://www.cryptoshop.com/de/service/comparison/index.php>

### Weitere Informationen:

Microsoft Smart Card Reference Guide

<http://www.microsoft.com/technet/security/topics/identitymanagement/scard.msp>

CryptoAPI

[http://msdn.microsoft.com/library/default.asp?url=/library/en-us/secrypto/security/cryptoapi\\_system\\_architecture.asp](http://msdn.microsoft.com/library/default.asp?url=/library/en-us/secrypto/security/cryptoapi_system_architecture.asp)

CSP

[http://msdn.microsoft.com/library/default.asp?url=/library/en-us/secrypto/security/cryptoapi\\_and\\_the\\_microsoft\\_base\\_cryptographic\\_provider.asp](http://msdn.microsoft.com/library/default.asp?url=/library/en-us/secrypto/security/cryptoapi_and_the_microsoft_base_cryptographic_provider.asp)

RSA PKCS

<http://www.rsasecurity.com/rsalabs/node.asp?id=2124>

ISO 7816 (Integrated circuit(s) cards with contacts

<http://www.iso.org>