ZENTRALER KREDITAUSSCHUSS

Financial Transaction Services (FinTS)

- Security -

Sicherheitsverfahren HBCI

Herausgeber:

Bundesverband deutscher Banken e.V., Berlin

Deutscher Sparkassen- und Giroverband e.V., Bonn/Berlin

Bundesverband der Deutschen Volksbanken und Raiffeisenbanken e.V., Berlin

Bundesverband Öffentlicher Banken Deutschlands e.V., Berlin

Version: 3.0

Stand: 18.07.2013

Final Version

Die vorliegende Schnittstellenspezifikation für eine automatisiert nutzbare multibankfähige Homebanking-Schnittstelle (im Folgenden: Schnittstellenspezifikation) wurde im Auftrag des Zentralen Kreditausschusses entwickelt. Sie wird hiermit zur Implementation in Kunden- und Kreditinstitutssysteme freigegeben.

Dieses Dokument kann im Internet abgerufen werden unter http://www.fints.org.

Versionsführung

Das vorliegende Dokument wurde von folgenden Personen erstellt bzw. geändert:

Name	Organisation	Datum	Version	Dokumente	Anmerkungen
Stein	SIZ	15.11.2002	3.0	FinTS 3.0 Security - Sicherheitsverfahren HBCI.doc	Frühere Versionen wurden im Rahmen der HBCI-Spezifikation veröffentlicht
Haubner	für SIZ	21.06.2005	3.0	FinTS 3.0 Security - Sicherheitsverfahren HBCI Rel. 2005-06-21.doc	Enthält alle bekannt gewordenen Fehler und Klarstellungen bis zum Releasedatum 21.06.2005.
Haubner	für GAD	07.05.2007	3.0	FinTS 3.0 Security - Sicherheitsverfahren HBCI Rel. 2007-05-07 final version.doc	Enthält die Anpassungen im Zusammenhang mit der Einführung von SECCOS 6 Bankensignaturkarten
Haubner	für GAD	15.05.2008	3.0	FinTS 3.0 Security - Sicherheitsverfahren HBCI Rel. 2008-05-15 final version.doc	Korrekturen und Klarstellungen zur SECCOS 6 Unterstützung.
Haubner	für SIZ	14.10.2011	3.0	FinTS 3.0 Security - Sicherheitsverfahren HBCI Rel. 2011-09-23 final version.doc	Ergänzen RAH-Verfahren
Haubner	für SIZ	25.09.2012	3.0	FinTS 3.0 Security - Sicherheitsverfahren HBCI Rel. 2012-09-25 final version.doc	Einführen DK-Padding bei RAH-Verfahren
Haubner	Für SIZ	18.07.2013	3.0	FinTS 3.0 Security - Sicherheitsverfahren HBCI Rel. 2013-07-18 FV.doc	Klarstellungen und Fehlerkorrekturen, Verweise auf DK Kryptokatalog

Änderungen gegenüber der Vorversion

Hinzufügungen und Änderungen sind im Dokument in dieser Farbe und zusätzlich durch Unterstreichung und einen Randbalken markiert. Löschungen sind aufgrund der besseren Übersichtlichkeit nur durch einen Randbalken markiert. Hypertextlinks sind in dieser Farbe markiert. Falls sich die Kapitelnummerierung geändert hat, bezieht sich die Kapitelangabe auf die neue Nummerierung. Aufgrund der umfangreichen Textumstellungen wurden nicht alle Änderungen markiert.

lfd. Nr.	Kapitel	Seiten-nummer	Kennung[1]	Art[2]	Beschreibung
1	Diverse	Diverse	0408	E	Ergänzen des RAH-Verfahrens und der damit verbundenen Sicherheitsprofile RAH-7, RAH-9 und RAH-10
2	B.2.2.	S. 17ff	0408, 0425	Ä	Anpassen der Abbildungen im Zuge der Einführung des RAH-Verfahrens. Ergänzen des DK-Paddings. Ersetzen des Terminus „HBCI-Nachricht“ durch „FinTS-Nachricht“
3	B.1.1, S. 3			Ä	Anpassen des Passus zu verpflichtenden Sicherheitsprofilen
4	B.2.2.1			Ä	ZKA-Padding, einfügen der AES-Blocklänge=16 Byte für den Wert „L“ Fehlerbehebungen und Klarstellungen in den Abbildungen 1, 2 und 3
5	B.3.1.3.1			Ä	Löschen von Step 5, da nicht mehr relevant.
6	Diverse			Ä	Ersetzen der konkret angegebenen Schlüssellängen durch Referenz auf die Empfehlungen des DK Kryptokatalogs [DK Krypto]
7	C.1.3.2.4.1	S. 99		Ä	Wegfall der Prüfung, ob Ausgangswert = Verschlüsselungsergebnis ist.

[1] nur zur internen Zuordnung
[2] F = Fehler; Ä = Änderung; K = Klarstellung; E = Erweiterung

Inhaltsverzeichnis

Abbildungsverzeichnis

Abkürzungen

Abkürzung	Bedeutung
AC	Access Condition
AEF	Application Elementary File
AES	Advanced Encryption Standard
AID	Application Identifier
BPD	Bankparameterdaten
C	Datenstruktur ist konditional
CBC	Cipher Block Chaining
CID	Cardholders Information Data (Kartenidentifikationsdaten der ZKA-Chipkarte)
CLA	Class Byte
CR	Carriage-Return (Wagenrücklauf)
DDV	DES-DES-Verfahren
DE	Datenelement
DEG	Datenelementgruppe
DES	Data Encryption Standard
DF	Dedicated File
DFÜ	Synonym verwendet für "Datenkommunikation, die in Form von Filetransfer, E-Mail, Online-Nachrichtenaustausch etc. erfolgen kann
DK	Die Deutsche Kreditwirtschaft
ECB	Electronic Code Book
EDIFACT	Electronic Data Interchange for Administration, Commerce and Transport
EF	Elementary File
EU	Elektronische Unterschrift; basiert auf dem asymmetrischen RSA-Verfahren
FCI	File Control Information
FCP	File Control Parameters
FCS	Frame Check Sequence
FMD	File Management Data
GD	Gruppendatenelement
GDG	Gruppendatenelementgruppe
HBCI	Homebanking Computer Interface
I	Information (z.B. Schlüsselart)
ID	Identifikationsmerkmal (Nummer oder alphanumerischer Code)
ISO	International Organisation for Standardisation
IV	Initialisierungsvektor
KGK	Key Generating Key

Abkürzung	Bedeutung
LF	Line-Feed (neue Zeile)
M	Datenstruktur muss vorhanden sein und ist inhaltlich korrekt zu füllen
MAC	Message Authentication Code; Symmetrisches Verfahren zur Erzeugung einer elektronischen Signatur (derzeit für die ZKA-Chipkarte eingesetzt)
MF	Master File
MFC	Multifunktions-Chipkarte
MIME	Multipurpose Internet Mail Extensions
N	Nachricht
N	Nicht erlaubt (not allowed) (Datenstruktur darf nicht vorhanden sein)
O	Datenstruktur ist optional
OID	Object IDentifier
PKD	Public-Key-Daten
RAH	RSA-AES-Hybridverfahren
RDH	RSA-DES-Hybridverfahren
RFC	Request for Comment
RSA	Asymmetrischer Algorithmus für die elektronische Unterschrift (EU) (vgl. MAC), benannt nach den Erfindern Rivest, Shamir und Adleman.
SEG	Segment
SEQ	Sequenznummer
SF	Segmentfolge
SFI	Short File Identifier
SHA	Secure Hash Algorithm
SSL	Secure Socket Layer
T	Transaktion (z.B. Schlüsselart)
UN/EDIFACT	s. EDIFACT
UPD	Userparameterdaten
ZKA	Zentraler Kreditausschuss

Literaturhinweise

- **Allgemeines**

[DF_NOTEPAD] Genereller Aufbau der SECCOS-Applikation Notepad („DF_NOTEPAD“) für FinTS und das DFÜ-Abkommen , Version 3.0, 21.06.2005, Zentraler Kreditausschuss

[Formals] Financial Transaction Services (FinTS) – Formals (Allgemeine Festlegungen für multibankfähige Online-Verfahren der deutschen Kreditwirtschaft), Version 3.0, 14.06.2011, Zentraler Kreditausschuss

[HKAZS] Financial Transaction Services (FinTS) – Security, Alternative Sicherheitsverfahren, Version 3.0, 22.01.2013, Die Deutsche Kreditwirtschaft

[ISO 3166] ISO 3166-1:1996: Code for the representation of names of countries and their subdivisions - Part 1: Country code http://www.din.de/gremien/nas/nabd/iso3166ma/ oder http://www.unece.org/trade/lcode/loc99.zip)

- **Verfahrensbeschreibung**

[AES] Federal Information Processing Standards 197 v. 26. November 2001, National Institute of Standards and Technology (NIST)

[SigG] Gesetz über Rahmenbedingungen für elektronische Signaturen und zur Änderung weiterer Vorschriften v. 16. Mai 2001, Bundesgesetzblatt Jahrgang 2001, Teil I Nr. 22

[SigV] Verordnung zur elektronischen Signatur v. 16. November 2001, Bundesgesetzblatt Jahrgang 2001, Teil I Nr. 59

[EU-Richtlinie] Richtlinie 1999/93/EG des Europäischen Parlaments und des Rates vom 13. Dezember 1999 über gemeinschaftliche Rahmenbedingungen für elektronische Signaturen, Amtsblatt der Europäischen Gemeinschaften v. 19.01.2000

[FormAnpG] Gesetz zur Anpassung der Formvorschriften des Privatrechts und anderer Vorschriften an den modernen Rechtsgeschäftsverkehr, 13. Juli 2001, Bundesgesetzblatt Jahrgang 2001, Teil I Nr. 35

[DFÜ-Abkommen] Kryptographische Verfahren des deutschen Kreditgewerbes für die Elektronische Unterschrift und für die Verschlüsselung im Rahmen der Kunde-Bank-Kommunikation
in: Anlage 1 der Schnittstellenspezifikation für die Datenfernübertragung zwischen Kunde und Kreditinstitut gemäß DFÜ-Abkommen – Spezifikation für die EBICS-Anbindung, Version 2.5, 16.05.2011

[DK Krypto] ZKA Kryptographie – Teil 1: Empfohlene kryptographische Algorithmen, Version 1.0

[ISO 9735-5] ISO 9735-5:1999 Electronic data interchange for administration, commerce and transport - (EDIFACT) - Application level syntax rules; (Syntax version number: 4) - Part 5: Security rules for batch EDI (Authenticity; Integrity and Non-repudiation of origin)

[ISO 9735-7] ISO 9735-7:1999 Electronic data interchange for administration, commerce and transport - (EDIFACT) - Application level syntax rules; (Syntax version number: 4) – Part 7: Security rules for batch EDI (Confidentiality)

[ISO 9735-9] ISO 9735-9:1999 Electronic data interchange for administration, commerce and transport - (EDIFACT) - Application level syntax rules; (Syntax version number: 4) – Part 9: Security key and certificate management message (Message type - KEYMAN)

[ISO 9796] ISO 9796:1991: Information technology - Security techniques - Digital signature scheme giving message recovery

[ISO 9796-2] ISO 9796-2:1997: Information technology - Security techniques - Digital signature scheme giving message recovery – Part 2: Mechanisms using a hash-function

[ISO 9796-3] ISO 9796-3:2000 Information technology - Security techniques - Digital signature scheme giving message recovery – Part 3: Discrete logarithm based mechanisms

[ISO 10116] ISO 10116:1997 Information technology Security techniques - Modes of operation for an n-bit block cipher algorithm

[ISO 10118-2] ISO 10118-2:1994 Information technology - Security techniques - Hash functions Part 2: Hash functions using an n-bit block cipher algorithm

[ISO 10118-3] ISO 10118-3:1998 Information technology - Security techniques - Hash functions Part 3: Dedicated hash-functions, 1998

[ISO 10126-1] ISO 10126-1:1991: Banking - Procedures for message encipherment (wholesale) – Part 1: General principles

[ISO 10126-2] ISO 10126-2:1991 Banking - Procedures for message encipherment (wholesale) – Part 2: DEA algorithm

[X3.92] ANSI X3.92-1981 (R1987): Data Encryption Algorithm

[X3.106] ANSI X3.106-1983 (R1996): Data Encryption Algorithm, Modes of operation for the

[X9.19] ANSI X9.19-1996: Financial Institution Retail Message Authentication

[X9.23] ANSI X9.23-1995 (R1995): Financial Institution Encryption of Wholesale Financial Messages

[X509] RFC 3039: Internet X.509 Public Key Infrastructure Qualified Certificates Profile

[PKCS1] PKCS #1: RSA Cryptography Standard, Version 2.1, RSA Laboratories, June 2002
(ftp://ftp.rsasecurity.com/pub/pkcs/pkcs-1/pkcs-1v2-1.pdf)

[SHA-1] FIPS 180-1, Secure Hash Standard, Federal Information Processing Standards Publication 180-1, U. S. Department of Commerce / N.I.S.T., National Technical Information Service, 1995
(http://www.itl.nist.gov/fipspubs/fip180-1.htm)

[SHA-256] Federal Information Processing Standards Publication
180-2 2002 August 1,
(http://csrc.nist.gov/publications/fips/fips180-2/fips180-2.pdf)

[ALGO] Geeignete Kryptoalgorithmen gemäß Anlage 1, I 2, SigV vom 22. November 2001,
aktueller Stand siehe unter http://www.bsi.de/esig/kryptoalg.htm

[ISIS/MTT] ISIS/MTT (Industrial Signature Interoperability and MailTrusT Specification / MailTrusT) Version 1 – Part 1: Certificate and CRL Profiles.

[CIPHER] EDIFACT Message Implementation Guidelines: Ciphered Text Message. CIPHER, SJWG; Working Draft Version, Paris September 16th 1994

[EDIFACT SIG] EDIFACT Security Implementation Guidelines, Trade/WP.4/R.1026/Add.2, 22 February

[KEYMAN] MIG Handbook UN/EDIFACT Message KEYMAN (proposed draft), June 30, 1995

[RSA] R. Rivest, A. Shamir, L. Adleman: A method for obtaining digital signatures and public key cryptosystems, Communications of the ACM, vol. 21 no. 2, 1978.

[RIPEMD] H. Dobbertin, A. Bosselaers, B. Preneel: „RIPEMD-160, a strengthened version of RIPEMD“, Fast Software Encryption - Cambridge Workshop 1996, LNCS, Band 1039, D. Gollmann, Ed., Springer-Verlag, 1996, S. 71-82
(http://www.esat.kuleuven.ac.be/~bosselae/ripemd160.html)

- **Chipapplikationen**

[ISO PIN1] ISO 9564-1, Banking – Personal Identification Number Management and Security, Part 1: PIN protection principles and techniques, DIS 1999

[DAT-MF] Schnittstellenspezifikation für die ec-Karte mit Chip, Dateien des MF, Version 4.2, 01.12.1999

[LT] Schnittstellenspezifikation für die ec-Karte mit Chip, Ladeterminal, Version 3.0, 02.04.1998

[DATKOM] Schnittstellenspezifikation für die ZKA-Chipkarte, Datenstrukturen und Kommandos, Version 4.1, 01.07.1999

[KT-KONZEPT] Schnittstellenspezifikation für die ZKA-Chipkarte, Konzept für die Unterstützung der Signatur-Anwendung der ZKA-Chipkarte durch das Internet-Kundenterminal, Version 1.0, 15. Februar 2002

[KT-SIG] Schnittstellenspezifikation für die ZKA-Chipkarte, Spezifikation des Internet-Kundenterminals für die Unterstützung der Signatur-Anwendung der ZKA-Chipkarte (ZKA-SIG-API), Version 2.0, 10. März 2008

[SECCOS] Schnittstellenspezifikation für die ZKA-Chipkarte, Secure Chip Card Operating System (SECCOS), Version 5.0, 5. Juni 2001 mit Errata vom 13. Juni 2001

[SECCOS-6] Interface Specifications for the SECCOS ICC Secure Chip Card Operating System (SECCOS) Version 6.2.1, 11.11.2009

[ZKASIG] Schnittstellenspezifikation für die ZKA-Chipkarte, Digital Signature Application for SECCOS 6, Version 1.3.1, 10. März 2011

[DINSIG] Chipcards with digital signature application/function according to SigG and SigV, Part 4: Basic Security Services, DIN V66291-4 vom 14. September 2001

A. EINLEITUNG

In diesem Dokument wird das Sicherheitsverfahren HBCI („Homebanking Computer-Interface“) beschrieben Dieses Verfahren beruht auf modernen kryptographischen Methoden und Algorithmen, wie z.B. der Digitalen Signatur und Chipkartentechnologie.

Dieses Sicherheitsverfahren kann in multibankfähigen Onlinebanking-Verfahren der deutschen Kreditwirtschaft eingesetzt werden.

Informationen bzgl. Nachrichtenaufbau und Dialogablauf sind dem Dokument [Formals] zu entnehmen.

B. VERFAHRENSBESCHREIBUNG

B.1 Allgemeines

Im Rahmen von HBCI werden zeitgemäße Sicherheitsmechanismen und -methoden eingesetzt, welche den Missbrauch der im Bereich des Homebankings eingesetzten Systeme verhindern.

Das folgende Kapitel ist in sechs Abschnitte gegliedert, welche sich mit den verwendeten Sicherheitsmechanismen, den Abläufen, den bankfachlichen Anforderungen sowie den Segmentformaten für Signatur, Verschlüsselung und Key-Management beschäftigen.

Die Ausführungen lehnen sich an bestehende deutsche Kreditinstitutsstandards (ZKA-Abkommen, z.B. DFÜ-Abkommen, ec-Chipkarte), sowie an internationale Standards (z.B. ISO, UN/EDIFACT) an.

Grundsätzlich kommen im Rahmen von HBCI drei verschiedene Sicherheitslösungen zum Einsatz:

- zwei auf dem asymmetrischen RSA-Verfahren basierende Lösungen
- eine auf dem symmetrischen DES-Verfahren basierende Chipkartenlösung

Die drei Varianten werden mit RAH (RSA-AES-Hybridverfahren), RDH (RSA-DES-Hybridverfahren), bzw. DDV (DES-DES-Verfahren) gekennzeichnet. RAH und RDH signieren mit RSA-EU und chiffrieren den Nachrichtenschlüssel mittels RSA, während DDV den MAC als Signatur verwendet und den Nachrichtenschlüssel (nachrichtenbezogener Chiffrierschlüssel) mittels 2-Key-Triple-DES verschlüsselt.

Die in Version 3.0 neu aufgenommene einheitliche Chipkartenlösung für das RAH- respektive RDH-Verfahren ist das angestrebte Zielverfahren. Da diese Sicherheitskonzeption momentan aufgrund technischer Restriktionen noch nicht flächendeckend umzusetzen ist, kommt bis zur durchgehenden Verfügbarkeit der RSA-Chipkartenlösung zusätzlich sowohl die DDV-Lösung auf Chipkartenbasis als auch RAH-/RDH-Lösungen auf reiner Softwarebasis oder auf Basis proprietärer Chipkartenlösungen zum Einsatz.

♦ **RAH-Verfahren**

Realisierung Bank:	verpflichtend
Realisierung Kunde:	verpflichtend. Ausgenommen hiervon sind Endgeräte, die eine RSA-EU-Lösung oder RAH-Verschlüsselung noch nicht erlauben (z.B. Smartphones mit MAC-Chipkarte erlauben ggf. keine RSA-EU, PC-basierte Produkte müssen hingegen stets die RSA-EU unterstützen).

- **RDH-Verfahren**

  Realisierung Bank: verpflichtend, falls übergangsweise das RAH-Verfahren noch nicht angeboten werden kann

  Realisierung Kunde: verpflichtend,solange das RAH-Verfahren noch nicht flächendeckend eingeführt ist.
  Ausgenommen hiervon sind Endgeräte, die eine RSA-EU-Lösung oder RDH-Verschlüsselung noch nicht erlauben (z.B. Smartphones mit MAC-Chipkarte erlauben ggf. keine RSA-EU, PC-basierte Produkte müssen hingegen stets die RSA-EU unterstützen).

- **DDV-Verfahren**

  Realisierung Bank: optional (empfohlen)
  Realisierung Kunde: optional

B.1.1 Sicherheitsprofile

Die Sicherheitsverfahren RAH, RDH und DDV können unterschiedlich parametrisiert werden, wobei Sicherheitsprofile entstehen. Um Multibankfähigkeit zu gewährleisten, ist bei Kommunikation auf Basis von FinTS 3.0 kundenproduktseitig die Unterstützung der Sicherheitsprofile R**A**H-7 und R**D**H-7, sowie R**A**H-9 und R**D**H-9 verpflichtend. Aus Kompatibilitätsgründen sind die in den bisherigen FinTS-Versionen genutzten Profile RDH-1, RDH-2, RDH-3, RDH-5, RDH-6, RDH-7, RDH-8, RDH-10, und DDV-1 weiterhin gültig. Andere als die unten genannten Profile sind nicht zulässig.

Das Kreditinstitut teilt dem Kunden die bankseitig unterstützten Profile in den Bankparameterdaten mit. Der Kunde wählt aus diesen Verfahren das für ihn geeignete Verfahren aus und bildet auf diese Weise Signatur und Verschlüsselung. Das Kreditinstitut antwortet stets mit dem vom Kunden gewählten Verfahren.

Hier eine Übersicht der zugelassenen Sicherheitsprofile und deren Anwendungsspektrum:

Sicherheits-profil	Schlüssel-länge	Medium	Bemerkungen
RAH-7	gemäß [DK Krypto][1]	Bankensignaturkarte SECCOS 6	mit SHA-256, PKCS#1 PSS Padding, AES-Verschlüsselung
RAH-9	gemäß [DK Krypto]	Bankensignaturkarte SECCOS 6	wie RAH-7 ohne Zertifikate
RAH-10	gemäß [DK Krypto]	RSA-SW-Lösung	mit SHA-256, PKCS#1 PSS Padding, AES-Verschlüsselung
RDH-1	708 bis 768 bit	RSA-SW-Lösung RSA-Chipkarte	RIPEMD-160, ISO 9796-1 Padding, Triple-DES-Verschlüsselung
RDH-2	1024 bis 2048 bit	RSA-SW-Lösung	RIPEMD-160, ISO 9796-2 Padding, Triple-DES-Verschlüsselung
RDH-3	1024 bis 2048 bit	Bankensignaturkarte SECCOS 5	RIPEMD-160, ISO 9796-2 Padding, Triple-DES-Verschlüsselung bzw. SHA-1, PKCS#1 V15 Padding, Triple-DES-Verschlüsselung
RDH-4[2]	1024 bis 2048 bit	Bankensignaturkarte SECCOS 5	SHA-1, PKCS#1 V15 Padding, Triple-DES-Verschlüsselung
RDH-5	1024 bis 2048 bit	Bankensignaturkarte SECCOS 5	wie RDH-3 ohne Zertifikate
RDH-6	gemäß [DK Krypto]	Bankensignaturkarte SECCOS 5 / 6	mit SHA-256, PKCS#1 V15 Padding, Triple-DES-Verschlüsselung
RDH-7	gemäß [DK Krypto]	Bankensignaturkarte SECCOS 6	mit SHA-256, PKCS#1 PSS Padding, Triple-DES-Verschlüsselung
RDH-8	gemäß [DK Krypto]	Bankensignaturkarte SECCOS 5 / 6	wie RDH-6 ohne Zertifikate
RDH-9	gemäß [DK Krypto]	Bankensignaturkarte SECCOS 6	wie RDH-7 ohne Zertifikate
RDH-10	gemäß [DK Krypto]	RSA-SW-Lösung	mit SHA-256, PKCS#1 PSS Padding, Triple-DES-Verschlüsselung

Die Angaben zum SECCOS-Betriebssystem bzw. der Betriebssystemversion sind nur als beispielhaft anzusehen; es kann auch jede gleichwertige Signaturkarte verwendet werden, welche die geforderten Verfahren unterstützt.

Die Information über die Betriebssystemversion kann dem Byte 24 in EF_ID entnommen werden. Dort sind derzeit folgende Werte vorgesehen:

X'01': SECCOS 5

X'06': SECCOS 6

[1] Die Schlüssellängen sind gemäß den Empfehlungen des DK-Kryptokatalogs [DK Krypto] zu verwenden.

[2] RDH-4 ist als Verfahren obsolet, da SHA-1 als Hashverfahren für neue Einsatzzwecke nicht mehr als sicher einzustufen ist.

♦ RAH-7

Als Sicherheitsmedium für das Kundensystem ist nur die Bankensignaturkarte oder eine gleichwertige Signaturkarte zugelassen.

Parameter	Wert	Bedeutung
Signaturalgorithmus, kodiert	10	RSA
Operationsmodus bei Signatur	19	Signier- und Signaturschlüssel - RSASSA- PSS [PKCS1]
Verwendung des Signaturalgorithmus	6	Owner Signing
Hashalgorithmus, kodiert	6 3	Signierschlüssel - SHA-256 / SHA-256 [SHA-256] Signaturschlüssel - SHA-256 [SHA-256]
Verschlüsselungsalgorithmus, kodiert	14	AES-256 [AES]
Operationsmodus bei Verschlüsselung	18	RSAES-PKCS1-v1_5 [PKCS1]
Schlüsselart	S V D	Signierschlüssel Chiffrierschlüssel Schlüssel für Digitale Signaturen
Schlüssellänge	gemäß [DK Krypto]	
Zertifikatstyp	3	X.509
Zertifikatsinhalt	EF_X509.CH.DS	fortgeschritten oder qualifiziert abh. von der Sicherheitsklasse

Im Rahmen des Paddingverfahrens RSASSA-PSS wird als „Mask Generation Function“ MGF1 verwendet. Beim Signierschlüssel wird ein doppeltes Hashing (Software und Bankensignaturkarte) durchgeführt. Dies wird durch eine spezielle Ausprägung des „Hashalgorithmus, kodiert“ gekennzeichnet.

Als Salt-Länge (Länge des Initialwertes) ist die Länge des Hashwertes zu verwenden. Diese Festlegung ist z. B. auch Bestandteil der SECCOS 6 Spezifikation.

♦ RAH-9

Als Sicherheitsmedium für das Kundensystem ist nur die Bankensignaturkarte oder eine gleichwertige Signaturkarte zugelassen.

Parameter	Wert	Bedeutung/Anmerkung
Signaturalgorithmus, kodiert	10	RSA
Operationsmodus bei Signatur	19	RSASSA-PSS [PKCS1]
Verwendung des Signaturalgorithmus	6	Owner Signing
Hashalgorithmus, kodiert	6	SHA-256 / SHA-256 [SHA-256]
Verschlüsselungsalgorithmus, kodiert	14	AES-256 [AES]
Operationsmodus bei Verschlüsselung	18	RSAES-PKCS1-v1_5 [PKCS1]
Schlüsselart	S V	Signierschlüssel Chiffrierschlüssel
Schlüssellänge	gemäß [DK Krypto]	
Zertifikatstyp		ohne
Zertifikatsinhalt	nicht spezifiziert	

Im Rahmen des Paddingverfahrens RSASSA-PSS wird als „Mask Generation Function“ MGF1 verwendet. Beim Signierschlüssel wird ein doppeltes Hashing (Software und Bankensignaturkarte) durchgeführt. Dies wird durch eine spezielle Ausprägung des „Hashalgorithmus, kodiert“ gekennzeichnet.

Als Salt-Länge (Länge des Initialwertes) ist die Länge des Hashwertes zu verwenden. Diese Festlegung ist z. B. auch Bestandteil der SECCOS 6 Spezifikation.

♦ RAH-10

Als Sicherheitsmedium für das Kundensystem ist eine RSA-Softwarelösung zugelassen.

Parameter	Wert	Bedeutung/Anmerkung
Signaturalgorithmus, kodiert	10	RSA
Operationsmodus bei Signatur	19	RSASSA-PSS [PKCS1]
Verwendung des Signaturalgorithmus	6	Owner Signing
Hashalgorithmus, kodiert	6	SHA-256 / SHA-256 [SHA-256]
Verschlüsselungsalgorithmus, kodiert	14	AES-256 [AES]
Operationsmodus bei Verschlüsselung	2	CBC (0-Padding)
Schlüsselart	S V	Signierschlüssel Chiffrierschlüssel
Schlüssellänge	gemäß [DK Krypto]	
Zertifikatstyp	1 2 3	ZKA UN/EDIFACT X.509
Zertifikatsinhalt	nicht spezifiziert	

Im Rahmen des Paddingverfahrens RSASSA-PSS wird als „Mask Generation Function“ MGF1 verwendet. Beim Signierschlüssel wird ein doppeltes Hashing durchgeführt. Dies wird durch eine spezielle Ausprägung des „Hashalgorithmus, kodiert“ gekennzeichnet.

Als Salt-Länge (Länge des Initialwertes) ist die Länge des Hashwertes zu verwenden.

♦ **RDH-1**

Als Sicherheitsmedien für das Kundensystem sind RSA-Softwarelösungen und RSA-Chipkarten zugelassen.

Parameter	Wert	Bedeutung/Anmerkung
Signaturalgorithmus, kodiert	10	RSA
Operationsmodus bei Signatur	16	ISO 9796-1
Verwendung des Signaturalgorithmus	6	Owner Signing
Hashalgorithmus, kodiert	999	RIPEMD-160
Verschlüsselungsalgorithmus, kodiert	13	2-Key-Triple-DES
Operationsmodus bei Verschlüsselung	2	CBC (0-Padding)
Schlüsselart	S V	Signierschlüssel Chiffrierschlüssel
Schlüssellänge	708-768 Bit	
Zertifikatstyp	1 2 3	ZKA UN/EDIFACT X.509
Zertifikatsinhalt	nicht spezifiziert	

♦ **RDH-2**

Als Sicherheitsmedium für das Kundensystem ist eine RSA-Softwarelösung zugelassen.

Parameter	Wert	Bedeutung/Anmerkung
Signaturalgorithmus, kodiert	10	RSA
Operationsmodus bei Signatur	17	ISO 9796-2
Verwendung des Signaturalgorithmus	6	Owner Signing
Hashalgorithmus, kodiert	*999*	RIPEMD-160
Verschlüsselungsalgorithmus, kodiert	13	2-Key-Triple-DES
Operationsmodus bei Verschlüsselung	2	CBC (0-Padding)
Schlüsselart	S V	Signierschlüssel Chiffrierschlüssel
Schlüssellänge	1024-2048 Bit	
Zertifikatstyp	1 2 3	ZKA UN/EDIFACT X.509
Zertifikatsinhalt	nicht spezifiziert	

♦ **RDH-3**

Als Sicherheitsmedium für das Kundensystem ist nur die Bankensignaturkarte oder eine gleichwertige Signaturkarte zugelassen.

Parameter	Wert	Bedeutung/Anmerkung
Signaturalgorithmus, kodiert	10	RSA
Operationsmodus bei Signatur	18 (bei S) 17 (bei D)	Signierschlüssel: RSASSA-PKCS1-v1_5 [PKCS1] Signaturschlüssel: ISO 9796-2
Verwendung des Signaturalgorithmus	6	Owner Signing
Hashalgorithmus, kodiert	1 (bei S) 999 (bei D)	Signierschlüssel: SHA-1 Signaturschlüssel: RIPEMD-160
Verschlüsselungsalgorithmus, kodiert	13	2-Key-Triple-DES
Operationsmodus bei Verschlüsselung[4]	18	RSAES-PKCS1-V1_5 [PKCS1]
Schlüsselart	S V D	Signierschlüssel Chiffrierschlüssel Schlüssel für Digitale Signaturen
Schlüssellänge	1024-2048 Bit	
Zertifikatstyp	3	X.509
Zertifikatsinhalt	EF_X509.CH.DS	fortgeschritten oder qualifiziert abh. von der Sicherheitsklasse

- **RDH-4**

  Das Verfahren RDH-4 ist obsolet, da es SHA-1 als Hashwertverfahren einsetzt und daher für Neu-Anwendungen nicht mehr als sicher gelten kann.

- **RDH-5**

  Als Sicherheitsmedium für das Kundensystem ist eine Bankensignaturkarte oder eine gleichwertige Signaturkarte zugelassen.

Parameter	Wert	Bedeutung/Anmerkung
Signaturalgorithmus, kodiert	10	RSA
Operationsmodus bei Signatur	18	RSASSA-PKCS1-v1_5 [PKCS1]
Verwendung des Signaturalgorithmus	6	Owner Signing
Hashalgorithmus, kodiert	1	SHA1
Verschlüsselungsalgorithmus, kodiert	13	2-Key-Triple-DES
Operationsmodus bei Verschlüsselung	18	RSAES-PKCS1-v1_5 [PKCS1]
Schlüsselart	S V	Signierschlüssel Chiffrierschlüssel
Schlüssellänge	1024-2048 Bit	
Zertifikatstyp		ohne
Zertifikatsinhalt	nicht spezifiziert	

[4] Paddingverfahren für Ver-/Entschlüsselung des Session-Keys

♦ **RDH-6**

Als Sicherheitsmedium für das Kundensystem ist nur die Bankensignaturkarte oder eine gleichwertige Signaturkarte zugelassen.

Parameter	Wert	Bedeutung
Signaturalgorithmus, kodiert	10	RSA
Operationsmodus bei Signatur	18	Signier- und Signaturschlüssel – RSASSA -PKCS1-v1_5 [PKCS1]
Verwendung des Signaturalgorithmus	6	Owner Signing
Hashalgorithmus, kodiert	3	SHA-256 [SHA-256]
Verschlüsselungsalgorithmus, kodiert	13	2-Key-Triple-DES
Operationsmodus bei Verschlüsselung	18	RSAES-PKCS1-v1_5 [PKCS1]
Schlüsselart	S V D	Signierschlüssel Chiffrierschlüssel Schlüssel für Digitale Signaturen
Schlüssellänge	gemäß [DK Krypto]	
Zertifikatstyp	3	X.509
Zertifikatsinhalt	EF_X509.CH.DS	fortgeschritten oder qualifiziert abh. von der Sicherheitsklasse

- **RDH-7**

  Als Sicherheitsmedium für das Kundensystem ist nur die Bankensignaturkarte oder eine gleichwertige Signaturkarte zugelassen.

Parameter	Wert	Bedeutung
Signaturalgorithmus, kodiert	10	RSA
Operationsmodus bei Signatur	19	Signier- und Signaturschlüssel - RSASSA- PSS [PKCS1]
Verwendung des Signaturalgorithmus	6	Owner Signing
Hashalgorithmus, kodiert	6 3	Signierschlüssel - SHA-256 / SHA-256 [SHA-256] Signaturschlüssel - SHA-256 [SHA-256]
Verschlüsselungsalgorithmus, kodiert	13	2-Key-Triple-DES
Operationsmodus bei Verschlüsselung	18	RSAES-PKCS1-v1_5 [PKCS1]
Schlüsselart	S V D	Signierschlüssel Chiffrierschlüssel Schlüssel für Digitale Signaturen
Schlüssellänge	gemäß [DK Krypto]	
Zertifikatstyp	3	X.509
Zertifikatsinhalt	EF_X509.CH.DS	fortgeschritten oder qualifiziert abh. von der Sicherheitsklasse

Im Rahmen des Paddingverfahrens RSASSA-PSS wird als „Mask Generation Function“ MGF1 verwendet. Beim Signierschlüssel wird ein doppeltes Hashing (Software und Bankensignaturkarte) durchgeführt. Dies wird durch eine spezielle Ausprägung des „Hashalgorithmus, kodiert“ gekennzeichnet.

Als Salt-Länge (Länge des Initialwertes) ist die Länge des Hashwertes zu verwenden. Diese Festlegung ist z. B. auch Bestandteil der SECCOS 6 Spezifikation.

- **RDH-8**

  Als Sicherheitsmedium für das Kundensystem ist nur die Bankensignaturkarte oder eine gleichwertige Signaturkarte zugelassen.

Parameter	Wert	Bedeutung/Anmerkung
Signaturalgorithmus, kodiert	10	RSA
Operationsmodus bei Signatur	18	RSASSA-PKCS1-v1_5 [PKCS1]
Verwendung des Signaturalgorithmus	6	Owner Signing
Hashalgorithmus, kodiert	3	SHA-256 [SHA-256]
Verschlüsselungsalgorithmus, kodiert	13	2-Key-Triple-DES
Operationsmodus bei Verschlüsselung	18	RSAES-PKCS1-v1_5 [PKCS1]
Schlüsselart	S V	Signierschlüssel Chiffrierschlüssel
Schlüssellänge	gemäß [DK Krypto]	
Zertifikatstyp		ohne
Zertifikatsinhalt	nicht spezifiziert	

- **RDH-9**

  Als Sicherheitsmedium für das Kundensystem ist nur die Bankensignaturkarte mit oder eine gleichwertige Signaturkarte zugelassen.

Parameter	Wert	Bedeutung/Anmerkung
Signaturalgorithmus, kodiert	10	RSA
Operationsmodus bei Signatur	19	RSASSA-PSS [PKCS1]
Verwendung des Signaturalgorithmus	6	Owner Signing
Hashalgorithmus, kodiert	6	SHA-256 / SHA-256 [SHA-256]
Verschlüsselungsalgorithmus, kodiert	13	2-Key-Triple-DES
Operationsmodus bei Verschlüsselung	18	RSAES-PKCS1-v1_5 [PKCS1]
Schlüsselart	S V	Signierschlüssel Chiffrierschlüssel
Schlüssellänge	gemäß [DK Krypto]	
Zertifikatstyp		ohne
Zertifikatsinhalt	nicht spezifiziert	

Im Rahmen des Paddingverfahrens RSASSA-PSS wird als „Mask Generation Function“ MGF1 verwendet. Beim Signierschlüssel wird ein doppeltes Hashing (Software und Bankensignaturkarte) durchgeführt. Dies wird durch eine spezielle Ausprägung des „Hashalgorithmus, kodiert“ gekennzeichnet.

Als Salt-Länge (Länge des Initialwertes) ist die Länge des Hashwertes zu verwenden. Diese Festlegung ist z. B. auch Bestandteil der SECCOS 6 Spezifikation.

- **RDH-10**

  Als Sicherheitsmedium für das Kundensystem ist eine RSA-Softwarelösung zugelassen.

Parameter	Wert	Bedeutung/Anmerkung
Signaturalgorithmus, kodiert	10	RSA
Operationsmodus bei Signatur	19	RSASSA-PSS [PKCS1]
Verwendung des Signaturalgorithmus	6	Owner Signing
Hashalgorithmus, kodiert	6	SHA-256 / SHA-256 [SHA-256]
Verschlüsselungsalgorithmus, kodiert	13	2-Key-Triple-DES
Operationsmodus bei Verschlüsselung	2	CBC (0-Padding)
Schlüsselart	S V	Signierschlüssel Chiffrierschlüssel
Schlüssellänge	gemäß [DK Krypto]	
Zertifikatstyp	1 2 3	ZKA UN/EDIFACT X.509
Zertifikatsinhalt	nicht spezifiziert	

  Im Rahmen des Paddingverfahrens RSASSA-PSS wird als „Mask Generation Function“ MGF1 verwendet. Beim Signierschlüssel wird ein doppeltes Hashing durchgeführt. Dies wird durch eine spezielle Ausprägung des „Hashalgorithmus, kodiert“ gekennzeichnet.

  Als Salt-Länge (Länge des Initialwertes) ist die Länge des Hashwertes zu verwenden.

- **DDV-1**

  Als Sicherheitsmedium für das Kundensystem ist nur die ec-Karte mit Chip zugelassen.

Parameter	Wert	Bedeutung/Anmerkung
Signaturalgorithmus, kodiert	1	DES
Operationsmodus bei Signatur	999	Retail-MAC
Verwendung des Signaturalgorithmus	6	Owner Signing
Hashalgorithmus, kodiert	999	RIPEMD-160
Verschlüsselungsalgorithmus, kodiert	13	2-Key-Triple-DES
Operationsmodus bei Verschlüsselung	2	CBC (0-Padding)
Schlüsselart	S V	Signierschlüssel Chiffrierschlüssel
Schlüssellänge	128 Bit	
Zertifikatstyp	-	nicht zulässig
Zertifikatsinhalt	-	nicht zulässig

B.1.1.1 Sicherheitsprofile im Secoder-Applikationsmodus

Für den Einsatz der folgenden Sicherheitsprofle ist als Chipkartenleser ein Secoder mindestens in Version 2.1 Voraussetzung.

- **RAH-7 im Secoder-Applikationsmodus**

Als Sicherheitsmedium für das Kundensystem ist nur die Bankensignaturkarte oder eine gleichwertige Signaturkarte zugelassen. Als Chipkartenleser ist ein Secoder ab Version 2.1 zu verwenden. Im Applikationsmodus des Secoders haben Signaturen z. B. bei Sicherheitsfunktion 811 folgenden Aufbau:

Parameter	Wert	Bedeutung
Signaturalgorithmus, kodiert	10	RSA
Operationsmodus bei Signatur	19	Signier- und Signaturschlüssel - RSASSA- PSS [PKCS1]
Verwendung des Signaturalgorithmus	6	Owner Signing
Hashalgorithmus, kodiert	6 3	Signierschlüssel - SHA-256 / SHA-256 [SHA-256] Signaturschlüssel - SHA-256 [SHA-256]
Verschlüsselungsalgorithmus, kodiert	14	AES-256 [AES]
Operationsmodus bei Verschlüsselung	18	RSAES-PKCS1-v1_5 [PKCS1]
Schlüsselart	S V D	Signierschlüssel Chiffrierschlüssel Schlüssel für Digitale Signaturen
Schlüssellänge	gemäß [DK Krypto]	
Zertifikatstyp	3	X.509
Zertifikatsinhalt	EF_X509.CH.DS	fortgeschritten oder qualifiziert abh. von der Sicherheitsklasse

Im Rahmen des Paddingverfahrens RSASSA-PSS wird als „Mask Generation Function“ MGF1 verwendet. Beim Signierschlüssel wird ein doppeltes Hashing (Software und Bankensignaturkarte) durchgeführt. Dies wird durch eine spezielle Ausprägung des „Hashalgorithmus, kodiert“ gekennzeichnet.

Als Salt-Länge (Länge des Initialwertes) ist die Länge des Hashwertes zu verwenden. Diese Festlegung ist z. B. auch Bestandteil der SECCOS 6 Spezifikation.

♦ **RAH-9 im Secoder-Applikationsmodus**

Als Sicherheitsmedium für das Kundensystem ist nur die Bankensignaturkarte oder eine gleichwertige Signaturkarte zugelassen. Als Chipkartenleser ist ein Secoder ab Version 2.1 zu verwenden. Im Applikationsmodus des Secoders haben Signaturen z. B. bei Sicherheitsfunktion 811 folgenden Aufbau:

Parameter	Wert	Bedeutung/Anmerkung
Signaturalgorithmus, kodiert	10	RSA
Operationsmodus bei Signatur	19	RSASSA-PSS [PKCS1]
Verwendung des Signaturalgorithmus	6	Owner Signing
Hashalgorithmus, kodiert	6	SHA-256 / SHA-256 [SHA-256]
Verschlüsselungsalgorithmus, kodiert	14	AES-256 [AES]
Operationsmodus bei Verschlüsselung	18	RSAES-PKCS1-v1_5 [PKCS1]
Schlüsselart	S V	Signierschlüssel Chiffrierschlüssel
Schlüssellänge	gemäß [DK Krypto]	
Zertifikatstyp		ohne
Zertifikatsinhalt	nicht spezifiziert	

Im Rahmen des Paddingverfahrens RSASSA-PSS wird als „Mask Generation Function“ MGF1 verwendet. Beim Signierschlüssel wird ein doppeltes Hashing (Software und Bankensignaturkarte) durchgeführt. Dies wird durch eine spezielle Ausprägung des „Hashalgorithmus, kodiert“ gekennzeichnet.

Als Salt-Länge (Länge des Initialwertes) ist die Länge des Hashwertes zu verwenden. Diese Festlegung ist z. B. auch Bestandteil der SECCOS 6 Spezifikation.

B.1.2 Sicherheitsklassen

Die Sicherheitsklasse gibt für jede Signatur den erforderlichen Sicherheitsdienst an. Als Sicherheitsdienst gelten derzeit „Authentikation“ und „Non-Repudiation“.

Der Sicherheitsdienst „Authentikation“ erfordert die Signatur mit der Schlüsselart „S“ (Schlüssel auf Kundenseite: S_K.CH.AUT$_{C/S}$). Der Sicherheitsdienst „Non-Repudiation“ erfordert die Signatur mit der Schlüsselart „D“ (Schlüssel auf Kundenseite: S_K.CH.DS).

Derzeit sind folgende Sicherheitsklassen zulässig:

Code	Bedeutung
0	kein Sicherheitsdienst erforderlich
1	Sicherheitsdienst „Authentikation“
2	Sicherheitsdienst „Authentikation“ mit fortgeschrittener elektronischer Signatur gemäß §2, SigG und optionaler Zertifikatsprüfung unter Verwendung des S-Schlüssels (Schlüssel S_K.CH.$AUT_{C/S}$)
3	Sicherheitsdienst „Non-Repudiation“ mit fortgeschrittener elektronischer Signatur gemäß §2, SigG und optionaler Zertifikatsprüfung unter Verwendung des DS-Schlüssels (S_K.CH.DS)
4	Sicherheitsdienst „Non-Repudiation“ mit fortgeschrittener bzw. qualifizierter elektronischer Signatur gemäß §2, SigG und zwingender Zertifikatsprüfung unter Verwendung des DS-Schlüssels (S_K.CH.DS)

Zu einem späteren Zeitpunkt kann die Notwendigkeit einer weiteren Sicherheitsklasse überprüft werden, die qualifizierte Signaturen mit zwingender Zertifikatsprüfung erfordert.

Folgende Zuordnungen von Sicherheitsklassen auf Sicherheitsprofile sind möglich:

Sicherheitsprofil	Sicherheitsklasse(n)
DDV	1
RAH-7	1, 2, 3, 4
RAH-9	1, 2
RAH-10	1
RDH-1	1
RDH-2	1
RDH-3	1, 2, 3, 4
RDH-5	1, 2
RDH-6	1, 2, 3, 4
RDH-7	1, 2, 3, 4
RDH-8	1, 2
RDH-9	1, 2
RDH-10	1

Die Sicherheitsklasse gibt für jeden Geschäftsvorfall den erforderlichen Sicherheitsdienst an. Signaturen gemäß der Sicherheitsklasse 2 und höher entsprechen den Anforderungen des Signaturgesetzes und erlauben damit rechtsverbindliche Willenserklärungen unter der Voraussetzung, dass die außerhalb des HBCI-Protokolls liegenden Anforderungen (z.B. Anforderungen an die Zertifizierungsinfrastruktur und an die Endgeräte) ebenfalls erfüllt sind.

Jede Signatur, die im Rahmen von HBCI generiert wird, muss der festgelegten Sicherheitsklasse entsprechen:

- Technische Signaturen (Dialoginitialisierung, Dialogendenachricht) erfolgen generell mit Sicherheitsklasse 1 (Authentikation)
- Bei Geschäftsvorfällen kann das Kreditinstitut die Sicherheitsklasse individuell festlegen (Die Sicherheitsklasse wird dem Kunden in den Bankparameterdaten des betreffenden Geschäftsvorfalls mitgeteilt)

Hinweis:
Sicherheitsklassen werden nur in Verbindung mit dem Sicherheitsverfahren HBCI benutzt. Unterstützt ein Kreditinstitut ausschließlich das PIN/TAN-Verfahren, so ist in das DE ‚Sicherheitsklasse' des jeweiligen Geschäftsvorfallparametersegmentes als Füllwert ‚0' einzustellen. Die Sicherheitsklasse hat bei PIN/TAN für die Verarbeitung keine Bedeutung und darf vom Kundenprodukt für PIN/TAN nicht ausgewertet werden. Stattdessen sind bei PIN/TAN die Informationen aus HIPINS für die Festlegung benötigter Sicherheitsmerkmale zu verwenden.

B.2 Mechanismen

B.2.1 Elektronische Signatur

Die Bildung der elektronischen Signatur erfolgt durch die Vorgänge

- Bildung des Hashwerts
- Ergänzen des Hashwerts auf eine vorgegebene Länge und
- Berechnung der elektronischen Signatur über den Hashwert.

Je nach Sicherheitsverfahren sind die Verarbeitungsschritte jeweils verschieden.

B.2.1.1 Hashing

Als Hash-Funktion können im Rahmen von HBCI abhängig vom Sicherheitsprofil entweder RIPEMD-160 [RIPEMD], SHA-1 [SHA-1] oder SHA-256 [SHA-256] eingesetzt werden.

♦ **RIPEMD-160**

Der Hash-Algorithmus RIPEMD-160 bildet Eingabe-Bitfolgen beliebiger Länge auf einen als Bytefolge dargestellten Hash-Wert von 20 Byte (160 Bit) Länge ab. Teil des Hash-Algorithmus ist das Padding von Eingabe-Bitfolgen auf ein Vielfaches von 64 Byte. Das Padding erfolgt auch dann, wenn die Eingabe-Bitfolge bereits eine Länge hat, die ein Vielfaches von 64 Byte ist. RIPEMD-160 verarbeitet die Eingabe-Bitfolgen in Blöcken von 64 Byte Länge.

Als Initialisierungsvektor dient die binäre Zeichenfolge X'01 23 45 67 89 AB CD EF FE DC BA 98 76 54 32 10 F0 E1 D2 C3'[5].

♦ **SHA-1**

Der Hash-Algorithmus SHA-1 bildet Eingabe-Bitfolgen beliebiger Länge auf Bytefolgen von 20 Byte Länge ab. Teil des Hash-Algorithmus ist das Padding von Eingabe-Bitfolgen auf ein Vielfaches von 64 Byte. Das Padding erfolgt auch dann, wenn die Eingabe-Bitfolge bereits eine Länge hat, die ein Vielfaches von 64 Byte ist. SHA-1 verarbeitet die Eingabe-Bitfolgen in Blöcken von 64 Byte Länge.

♦ **SHA-256**

Der Hash-Algorithmus SHA-256 bildet Eingabe-Bitfolgen beliebiger Länge auf Bytefolgen von 32 Byte Länge ab. Teil des Hash-Algorithmus ist das Padding von Eingabe-Bitfolgen auf ein Vielfaches von 64 Byte. Das Padding erfolgt auch dann, wenn die Eingabe-Bitfolge bereits eine Länge hat, die ein Vielfaches von 64 Byte ist. SHA-256 verarbeitet die Eingabe-Bitfolgen in Blöcken von 64 Byte Länge.

B.2.1.2 Elektronische Signatur bei DDV (DES-basierend)

1. Hashing der Nachricht

Als Hash-Funktion kann, anhängig vom Sicherheitsprofil RIPEMD-160 oder SHA-256 eingesetzt werden.

---

[5] Little-Endian-Notation

2. Formatierung des Hashwerts

Formatierung des Hashwerts bei RIPEMD-160

Das Padding ist je nach Typ der eingesetzten Chipkarte (Typ 0 oder Typ 1) unterschiedlich:

Bei Typ 0-Karten erfolgt das Padding entsprechend der folgenden Abbildung mit X'00' auf das nächste Vielfache von 8 Byte:

Padding

Byte-Position:	24 ... 21	20 ... 1
	00 00 00 00	H a s h w e r t

Bei Typ 1-Karten erfolgt das Padding entsprechend der folgenden Abbildung auf das nächste Vielfache von 8 Byte: Sei der Hashwert = $Hash_L$ | $Hash_R$, wobei $Hash_L$ die linken 8 Byte und $Hash_R$ die rechten 12 Byte des Hashwerts bezeichnet.

Padding

Byte-Position:	24 23	22 ... 11	10 9	8 ... 1
	00 80	H a s h R	0C 81	H a s h L

Ob eine Karte vom Typ 0 oder Typ 1 vorliegt, kann anhand der Länge der Kartenidentifikationsdaten (CID) ermittelt werden. Für Typ 0-Karten hat die CID eine Länge von 22 Byte, für Typ 1-Karten mindestens eine Länge von 24 Byte.

Formatierung des Hashwerts bei SHA-256

Da der Hashwert bei SHA-256 mit 32 Byte bereits ein Vielfaches von 8 darstellt, muss bei diesem Verfahren kein Padding stattfinden.

3. Berechnung der elektronischen Signatur

Als Signatur wird ein Retail CBC-MAC gemäß ANSI X9.19 gebildet. Hierzu wird der gepaddete Hashwert zunächst in 3 Blöcke der Länge 8 Byte aufgeteilt. Als Zwischenresultat wird ein einfacher CBC-MAC über die ersten 2 Blöcke berechnet. Als Initialisierungsvektor kommt X'00 00 00 00 00 00 00 00' zum Einsatz. Dabei verwendet man als Schlüssel die linke Hälfte des Signierschlüssels. Anschließend erfolgt eine 2-Key-Triple-DES-Verschlüsselung mit dem Signierschlüssel des Kunden (muss beim Kreditinstitut hergeleitet werden) über die XOR-Summe des Zwischenergebnisses mit dem letzten Nachrichtenblock. Der so erhaltene 8 Byte(=64 bit)-Ausgabeblock ist der Retail CBC-MAC.

B.2.1.3 Elektronische Signatur bei RAH und RDH (RSA-basierend)

1. Hashing der Nachricht

Als Hash-Funktion kann abhängig vom Sicherheitsprofil entweder RIPEMD-160, SHA-1 oder SHA-256 eingesetzt werden.

2. Formatierung des Hashwerts

Die Formatierung des Hashwerts erfolgt auf folgende Art und Weise:

ISO 9796-2	RDH-2, RDH-3 und RDH-5
PKCS#1 V1.5	RDH-3, RDH-6 und RDH-8
PKCS#1 PSS	RAH-7, RAH-9 und RAH-10, RDH-7, RDH-9 und RDH-10
ISO 9796:1991 (Kap. 5.1 – 5.4)	Übergangsweise für das Altverfahren RDH-1, wobei der Hashwert wird für die nachfolgende Signaturbildung als Langzahl[6] interpretiert (s. auch die Beispiele in der Anlage zu ISO 9796:1991).

3. Berechnung der elektronischen Signatur

Der Hashwert wird mittels RSA entweder gemäß DIN/ISO 9796-2 (bei RDH-2, RDH-3 und RDH-5), gemäß PKCS#1 V1.5 (bei RDH-6 und RDH-8) oder gemäß PKCS#1 PSS (bei RAH-7, RAH-9 und RAH-10 bzw. RDH-7, RDH-9 und RDH-10) signiert. Übergangsweise ist für das Altverfahren RDH-1 auch die Signatur gemäß ISO 9796-1 zulässig.[7]

---

[6] Unter Langzahl wird dabei die kanonische Darstellung einer natürlichen Zahl in einem Feld [0..n] bezeichnet, wobei die Wertigkeit der Felder von 0 bis n abnimmt.

[7] Im Falle von ISO 9796-1 sind auch die dort in den Anhängen A.4 „Signature function“ und A.5 „Verification function“ beschriebenen Operationen durchzuführen und die Anhänge B und C zu berücksichtigen.

B.2.2 Verschlüsselung

Bei der Verschlüsselung wird für jede Nachricht ein separater Nachrichtenschlüssel verwendet. Die Verschlüsselung der HBCI-Nutzdaten erfolgt folgendermaßen:

- Bei RAH-7, RAH-9 und RAH-10
  Die Verschlüsselung erfolgt mittels AES-256 gemäß [AES]. Der Nachrichtenschlüssel wird mittels RSA (RAH) chiffriert und mit der verschlüsselten Nachricht mitgeliefert.
- Sonst:
  Die Verschlüsselung erfolgt mittels 2-Key-Triple-DES gemäß ANSI X3.92. Der Nachrichtenschlüssel wird entweder mittels 2-Key-Triple-DES (DDV) oder RSA (RDH) chiffriert und mit der verschlüsselten Nachricht mitgeliefert.

Der Nachrichtenschlüssel muss für jede Nachricht eines Dialoges individuell verschieden sein. Dies muss gewährleistet werden, indem das sendende System den Nachrichtenschlüssel dynamisch generiert.

Sollte bei der Verarbeitung des Nachrichtenschlüssels, insbesondere beim Padding ein Fehler auftreten, so sind außer dem negativen Prüfergebnis selbst keine weiteren Details an die aufrufende Funktion zurückzugeben, um keine Rückschlüsse über die Art des Fehlers und damit ggf. auf den Schlüssel selbst zu geben.

B.2.2.1 Verschlüsselung bei RAH-7, RAH-9 und RAH-10:

Die Verschlüsselung und Entschlüsselung erfolgt bei den RAH-Verfahren in den folgenden drei Schritten:

1. Der Sender erzeugt eine Zufallszahl als Nachrichtenschlüssel.
2. Dieser Nachrichtenschlüssel wird verwendet, um die Daten mittels AES im CBC Modus gemäß ISO 10116 (ANSI X3.106) zu verschlüsseln (vgl. Abbildung 1). Das Padding der Nachricht erfolgt gemäß den Vorgaben des Kryptokatalogs der Deutschen Kreditwirtschaft (vgl. [DK Krypto], Kapitel 4.3.1) (vgl.Abbildung 2 und Abbildung 3).

„ZKA-Padding“ (vgl. [DK Krypto], Kapitel 4.3.1 auf S. 20):

Für die Verarbeitung von Daten durch einen kryptographischen Algorithmus kann deren Darstellung als Folge von Byte-Blöcken mit einer vorgegebenen Länge L erforderlich sein. Das ZKA-Padding ist eine Methode zur Formatierung des letzten, möglicherweise unvollständigen Datenblocks auf die Länge von L Byte. Die den Daten zugehörigen Bytes können eindeutig von den durch das Padding hinzugefügten Bytes unterschieden werden.

An die Daten M wird zunächst das Byte '80' angehängt. Falls M || '80' nun eine Byte-Länge besitzt, die kein Vielfaches von L ist, werden weitere Bytes '00' angehängt, bis das Ergebnis der Operation eine Byte-Länge besitzt, die ein Vielfaches von L ist.

$$\text{ZKA-Padding}(M) = M \,||\, '80' \,||\, \underbrace{'00' \,||\ldots||\, '00'}_{\text{Verkettung bis zur Gesamtlänge der Byte-Folge als Vielfaches von L Byte (hier: AES-Blocklänge = 16 Byte)}}$$

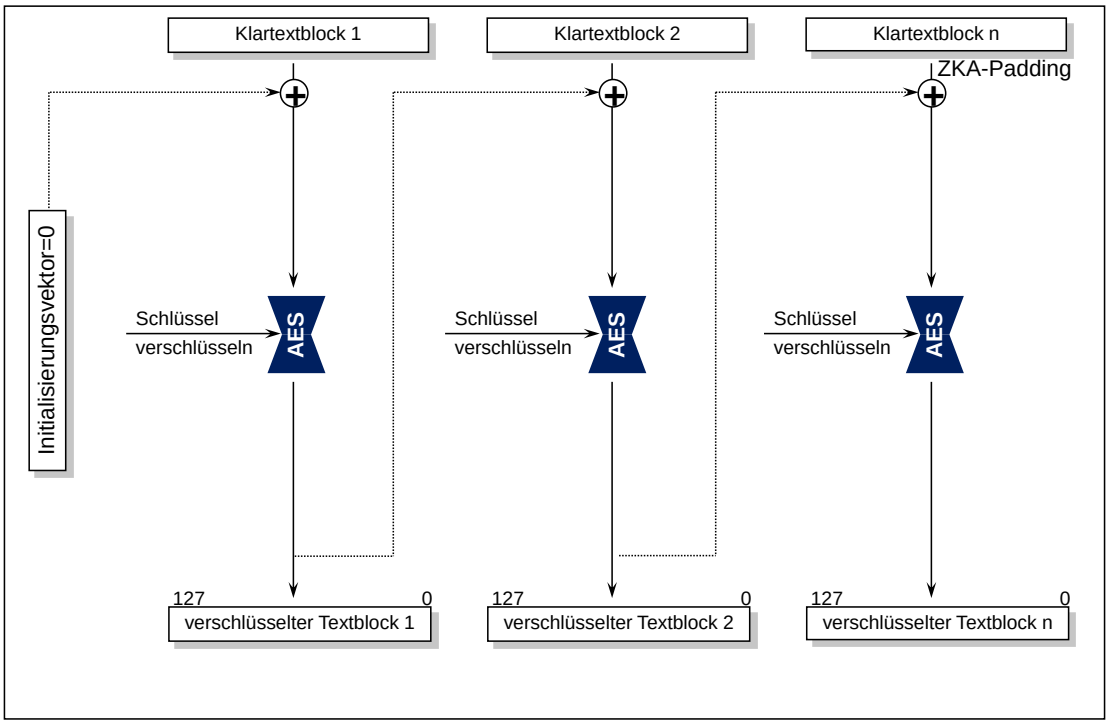


Abbildung 1: Nachrichtenverschlüsselung mit AES im CBC-Mode für RAH-Verfahren

3. Der aktuelle Nachrichtenschlüssel wird mit dem öffentlichen Schlüssel des Empfängers chiffriert. Da die Länge des Nachrichtenschlüssels bei AES nur 32 Byte, d.h. 256 Bit beträgt, muss er auf die Moduluslänge des verwendeten öffentlichen Chiffrierschlüssels ergänzt werden. Das Padding wird abhängig vom Sicherheitsprofil auf unterschiedliche Art und Weise vorgenommen, wie in den folgenden Abbildungen gezeigt.

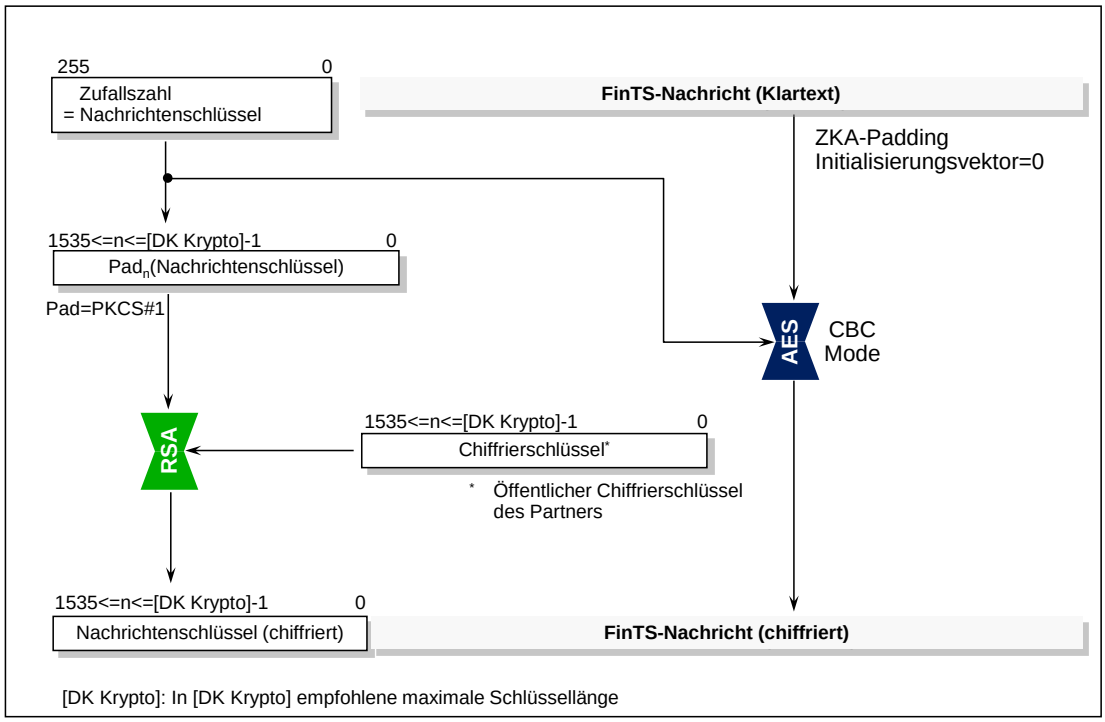


Abbildung 2: Verschlüsselung bei RAH-7 und RAH-9

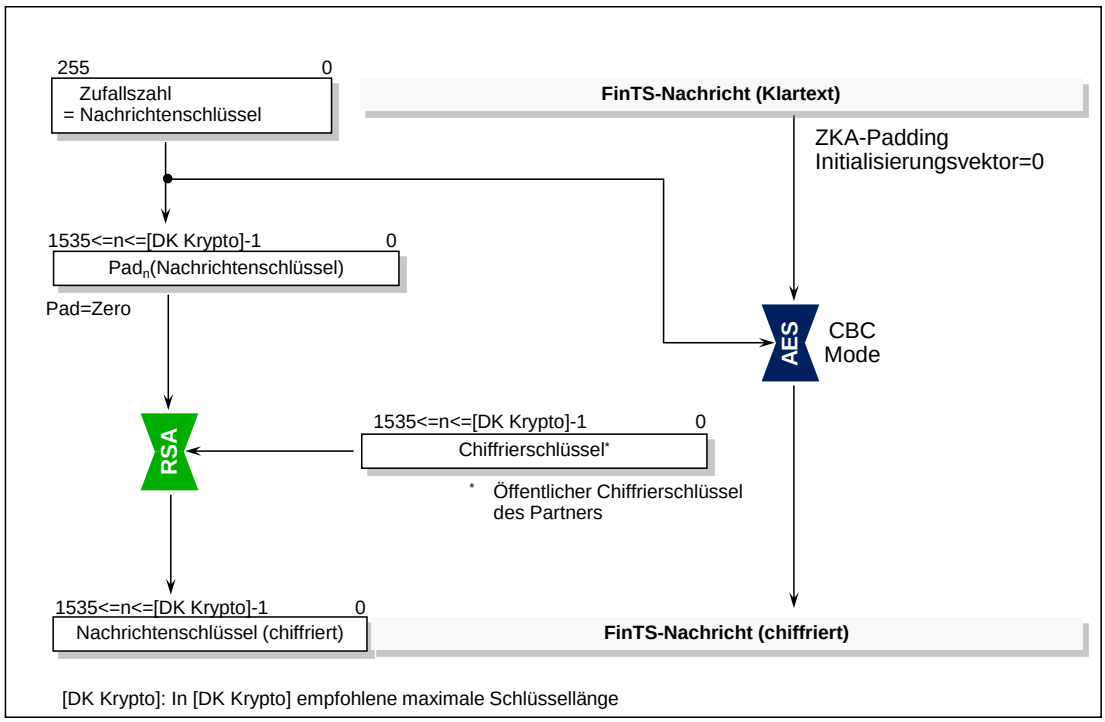


Abbildung 3: Verschlüsselung bei RAH-10

B.2.2.2 Verschlüsselung bei RDH und DDV:

Die ersten zwei Schritte sind für die beiden Verfahren RDH und DDV identisch:

1. Der Sender erzeugt eine Zufallszahl als Nachrichtenschlüssel und stellt ungerade Parität sicher. Bei der Auswahl der Zufallszahl ist darauf zu achten, dass keiner der folgenden schwachen oder halbschwachen Schlüssel[9] gewählt wird (vgl. Kapitel B.3.1.1).

   Die schwachen Schlüssel des DES-Algorithmus:

   ```
   X'01 01 01 01 01 01 01 01'
   X'FE FE FE FE FE FE FE FE'
   X'1F 1F 1F 1F 0E 0E 0E 0E'
   X'E0 E0 E0 E0 F1 F1 F1 F1'
   ```

   Die halbschwachen Schlüssel des DES-Algorithmus:

   ```
   X'01 FE 01 FE 01 FE 01 FE'
   X'FE 01 FE 01 FE 01 FE 01'
   X'1F E0 1F E0 0E F1 0E F1'
   X'E0 1F E0 1F F1 0E F1 0E'
   X'01 E0 01 E0 01 F1 01 F1'
   X'E0 01 E0 01 F1 01 F1 01'
   X'1F FE 1F FE 0E FE 0E FE'
   X'FE 1F FE 1F FE 0E FE 0E'
   X'01 1F 01 1F 01 0E 01 0E'
   X'1F 01 1F 01 0E 01 0E 01'
   X'E0 FE E0 FE F1 FE F1 FE'
   X'FE E0 FE E0 FE F1 FE F1'
   ```

2. Dieser Nachrichtenschlüssel wird verwendet, um die Daten mittels 2-Key-Triple-DES im CBC Modus gemäß ISO 10116 (ANSI X3.106) zu verschlüsseln (vgl. Abbildung 4). Das Padding der Nachricht erfolgt oktettorientiert gemäß ISO 10126 (ANSI X9.23), der Initialisierungsvektor ist X'00 00 00 00 00 00 00 00' (vgl. Abbildung 5 und Abbildung 6).

[9] Die schwachen und halbschwachen Schlüssel entsprechen denen des DFÜ-Abkommens.

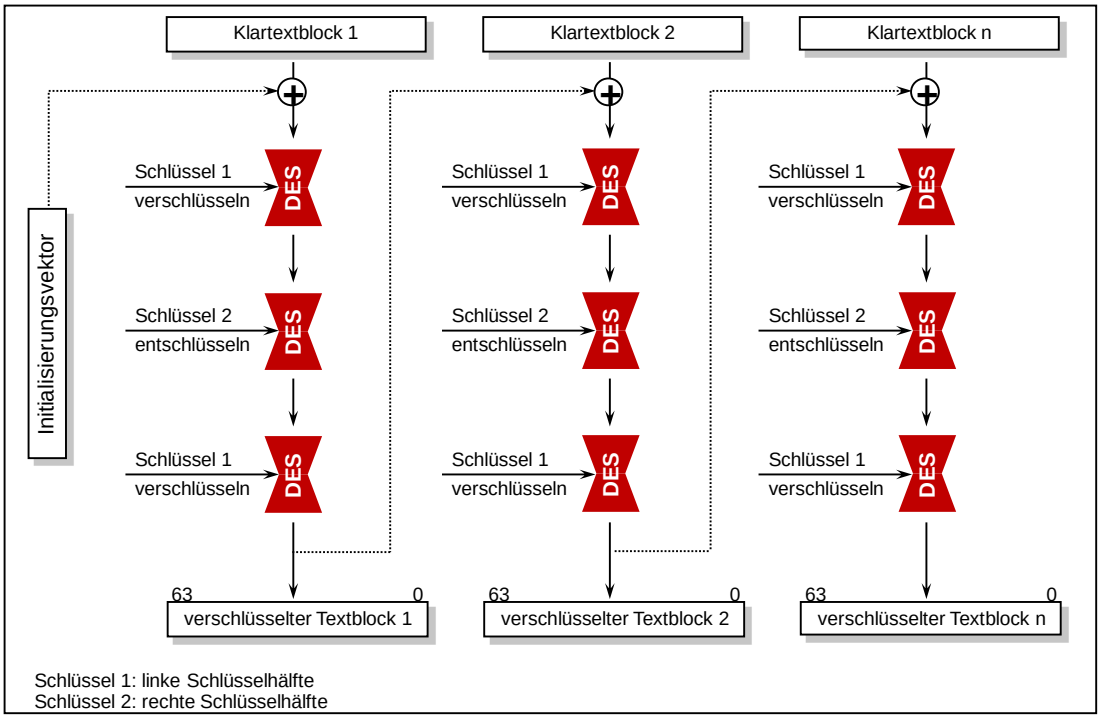


Abbildung 4: Nachrichtenverschlüsselung generell mit 2-Key-Triple-DES im CBC-Mode für RDH und DDV

Die weitere Verarbeitung ist bei DDV und RDH unterschiedlich:

B.2.2.2.1 Verschlüsselung bei DDV (DES-basierend)

3. Der aktuelle Nachrichtenschlüssel für die Chiffrierung der Daten wird vom Kundenprodukt mit dem kundenindividuellen Chiffrierschlüssel der Chipkarte mittels 2-Key-Triple-DES im ECB-Mode (ISO 10116) verschlüsselt (vgl. Abbildung 5 und Abbildung 6).

   Aufgrund vorgegebener Verfahren bei der ZKA-Chipkarte wird zum Chiffrieren und Dechiffrieren des Nachrichtenschlüssels, unabhängig von der Übertragungsrichtung, kundensystemseitig immer die Routine „Encrypt“ benutzt, kreditinstitutsseitig immer die Routine „Decrypt“ (vgl. Kapitel C.2.5.2).

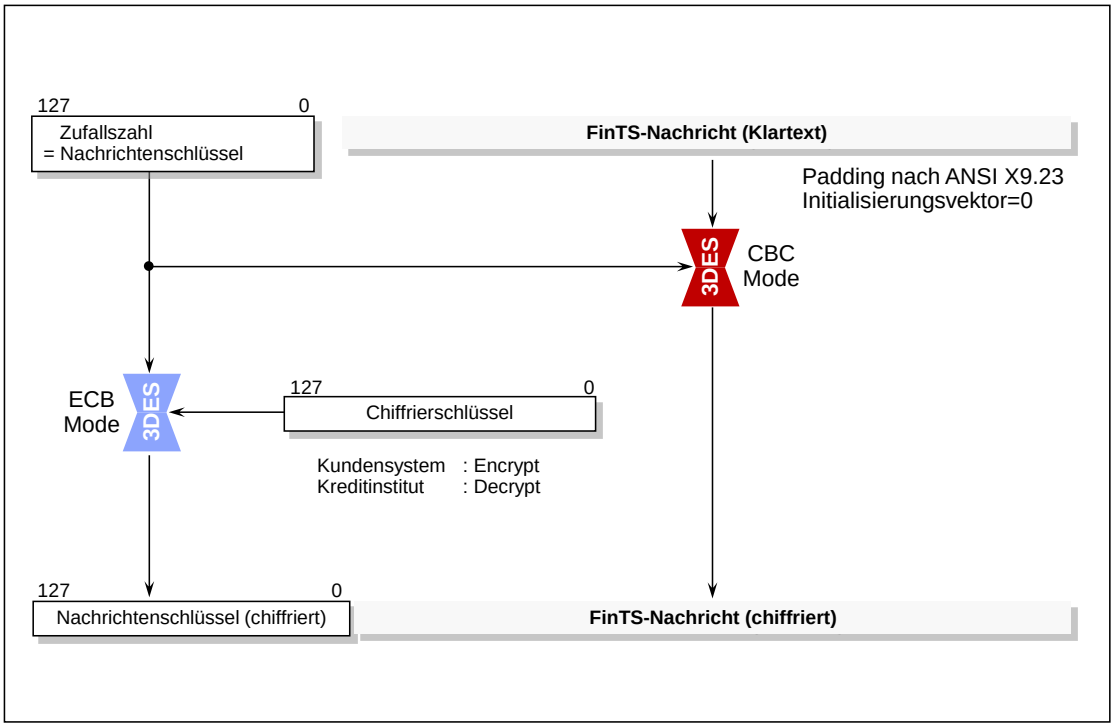


Abbildung 5: Verschlüsselung bei 2-Key-Triple-DES im DDV-Verfahren

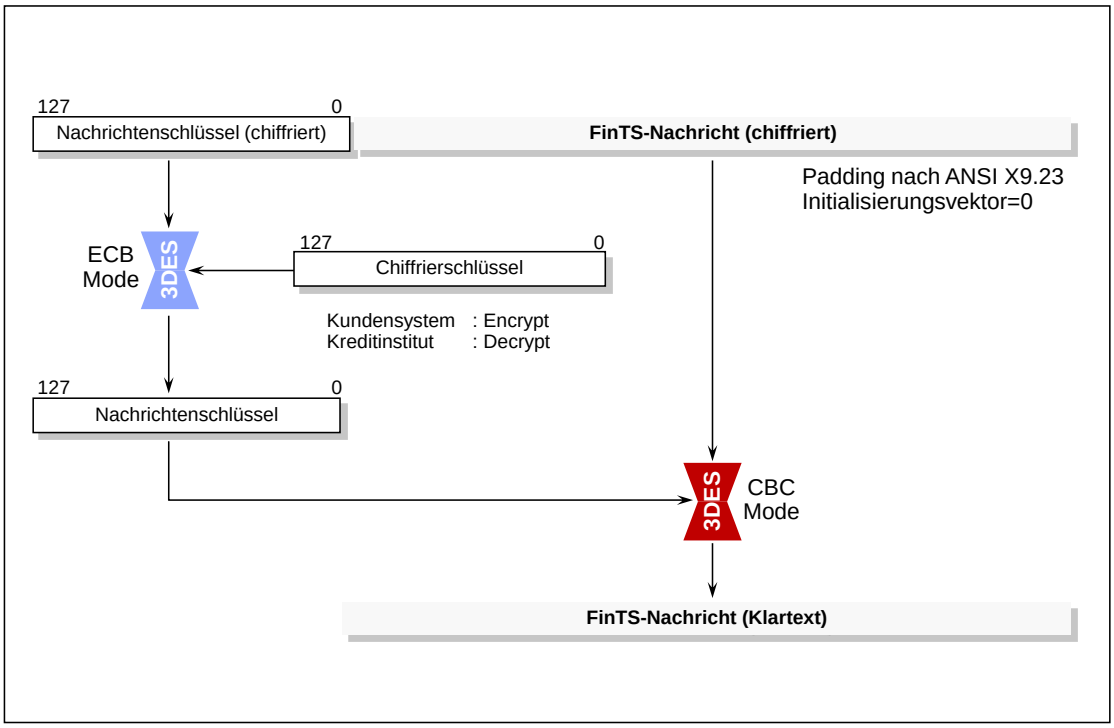


Abbildung 6: Entschlüsselung bei 2-Key-Triple-DES im DDV-Verfahren

B.2.2.2.2 Verschlüsselung bei RDH (RSA-basierend)

3. Der aktuelle Nachrichtenschlüssel wird mit dem öffentlichen Schlüssel des Empfängers chiffriert. Da die Länge des Nachrichtenschlüssels nur 16 Byte, d.h. 128 Bit bei 2-Key-Triple-DES beträgt, muss er auf die Moduluslänge des verwendeten öffentlichen Chiffrierschlüssels ergänzt werden. Das Padding wird abhängig vom Sicherheitsprofil auf unterschiedliche Art und Weise vorgenommen, wie in den folgenden Abbildungen gezeigt.

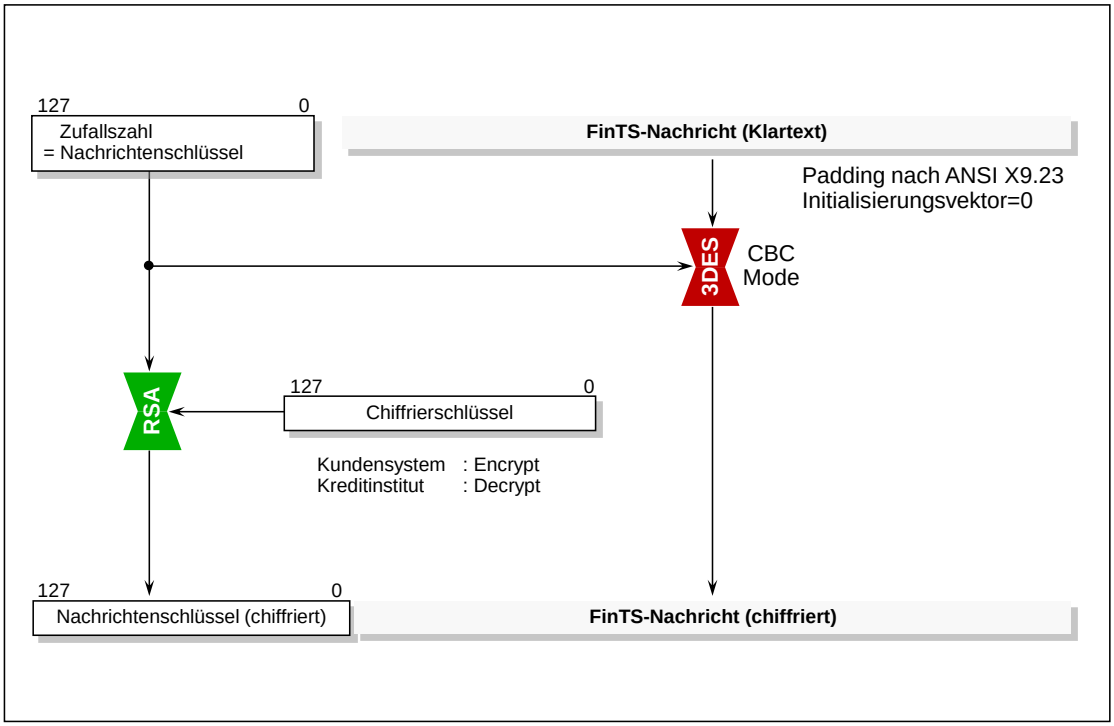


Abbildung 7: Verschlüsselung bei 2-Key-Triple-DES im RDH-Verfahren

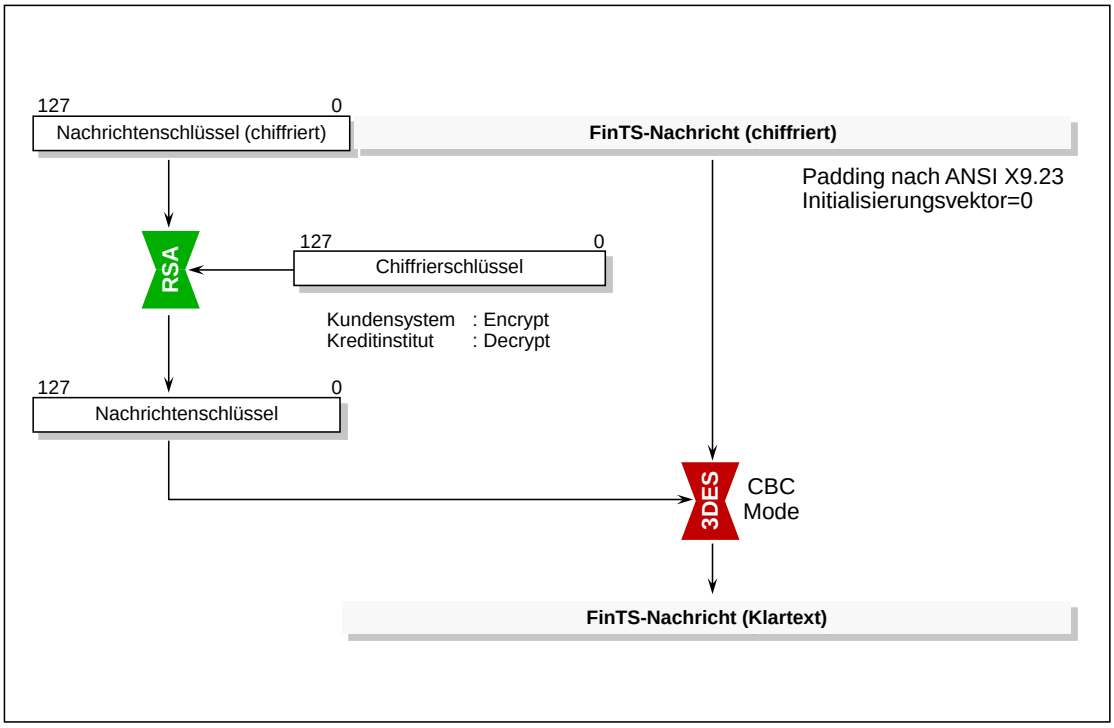


Abbildung 8: Entschlüsselung bei 2-Key-Triple-DES im RDH-Verfahren

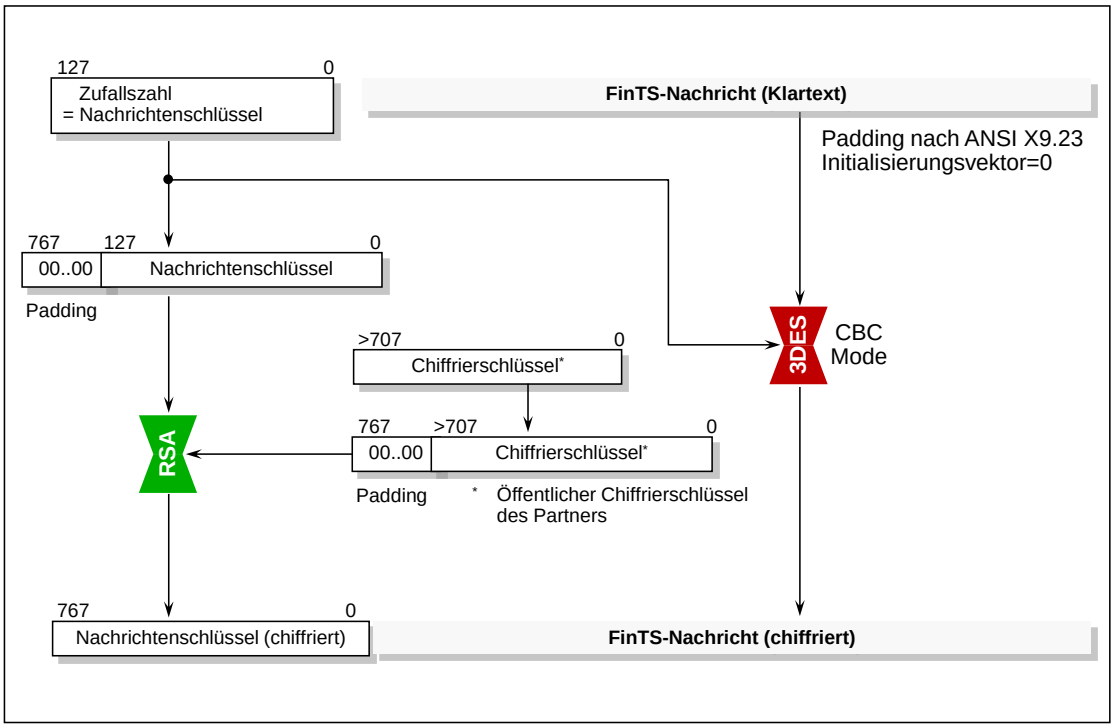


Abbildung 9: Verschlüsselung bei RDH-1

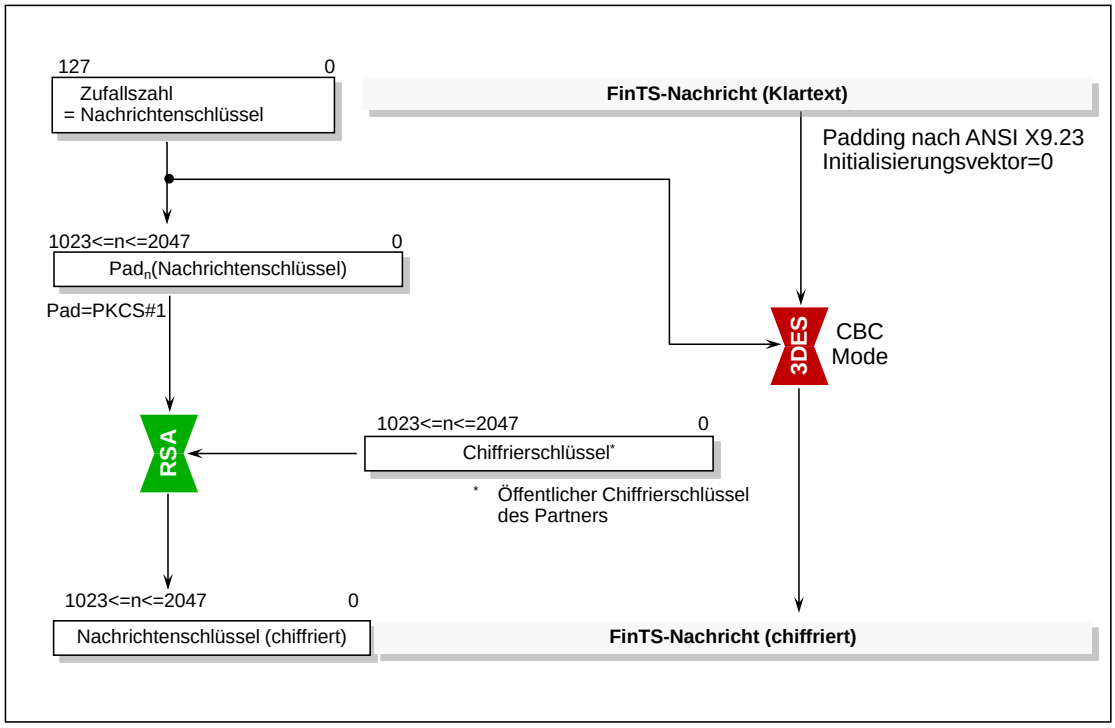


Abbildung 10: Verschlüsselung bei RDH-3 und RDH-5

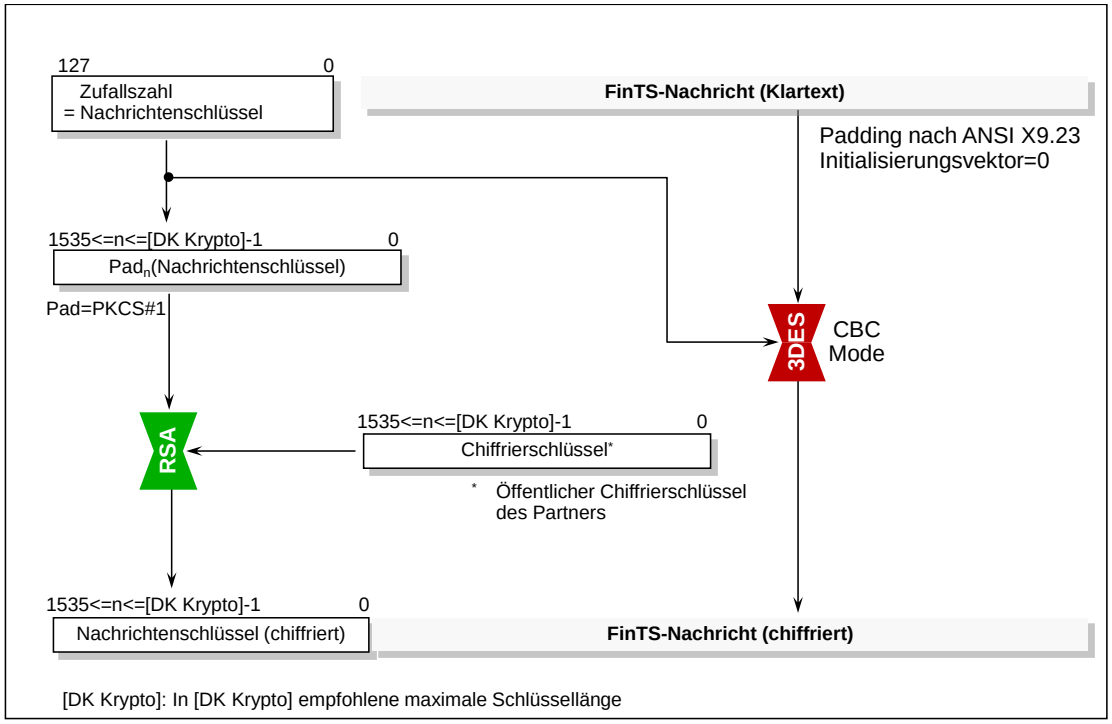


Abbildung 11: Verschlüsselung bei RDH-6 bis RDH-9

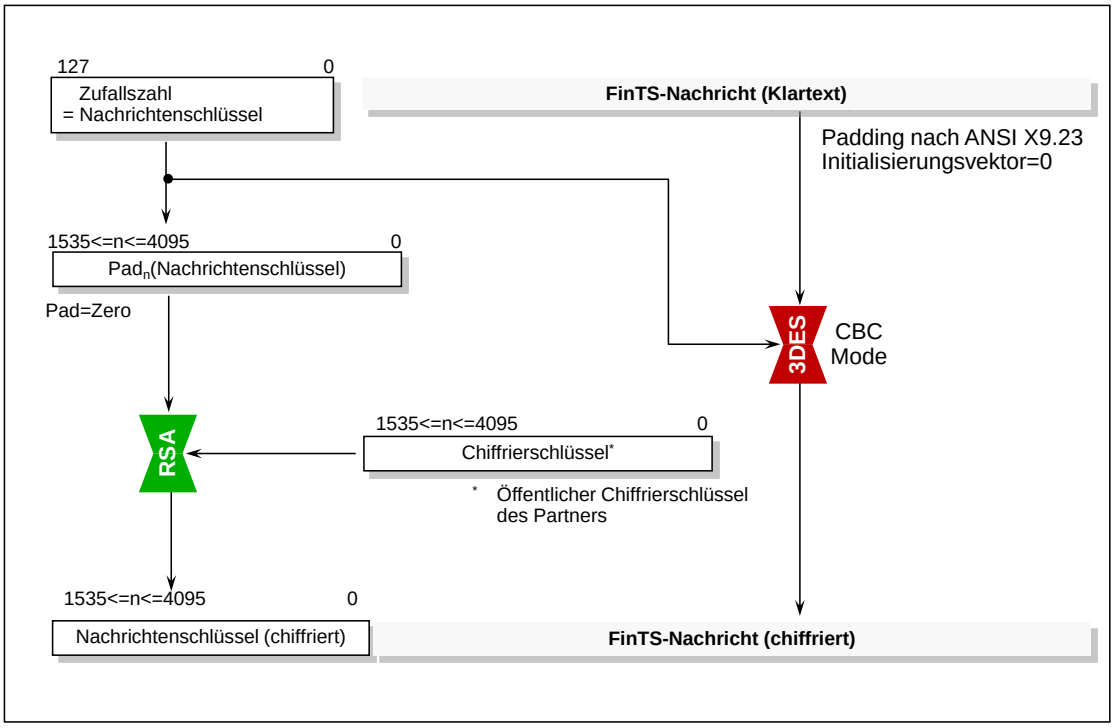


Abbildung 12: Verschlüsselung bei RDH-10

B.2.3 Sicherheitsmedien beim Kundenprodukt

Bei Verwendung des symmetrischen Verfahrens (DDV) muss eine vom Kreditinstitut ausgegebene ZKA-Chipkarte eingesetzt werden, welche die Berechnung der kryptographischen Funktionen so durchführt, dass die kartenindividuellen Schlüssel niemals die Chipkarte verlassen.

Werden asymmetrische Verfahren (RAH oder RDH) eingesetzt, so kann als Sicherheitsmedium eine vom Kreditinstitut ausgegebene RSA-Chipkarte oder eine Schlüsseldatei dienen.[10] Falls eine Chipkarte zum Einsatz kommen soll, wird die in Kap. C.1 beschriebene Bankensignaturkarte empfohlen. Auf dem Sicherheitsmedium wird unter anderem der private Schlüssel des Kunden gespeichert. Es ist aber auch möglich, öffentliche Schlüssel des Kreditinstitutes darauf abzulegen oder aber im Falle einer Chipkarte die kryptographischen Operationen damit durchzuführen. Bei Einsatz einer RSA-Chipkarte müssen die geheimen Daten (z.B. private Schlüssel, Passworte) gegen unberechtigtes Auslesen geschützt sein.

Es ist zwingend erforderlich, die Daten auf dem Sicherheitsmedium (kryptographisch) zu schützen. Speziell ist im Rahmen der Speicherung der Schlüsselpaare in der Schlüsseldatei sicherzustellen, dass die Daten unter Einbeziehung eines Passwortes (Banking-PIN o.ä.) verschlüsselt werden und der Zugriff auf die verschlüsselten Daten nur über die manuelle Eingabe des entsprechenden Passwortes möglich ist.

[10] Der Aufbau des Dateiformats ist bei Bedarf bei den auf www.fints.org in der Rubrik „Impressum“ gelisteten Adressen erhältlich.

B.3 Abläufe

B.3.1 Schlüsselverwaltung

Bei der Schlüsselverwaltung muss zwischen der Verwendung von symmetrischen Schlüsseln für DDV und asymmetrischen Schlüsseln für RAH und RDH unterschieden werden.

Gemeinsam gültig sind hingegen für beide Verfahren die verwendeten Schlüsselarten, Schlüsselnamen und die Generierung von Nachrichtenschlüsseln.

B.3.1.1 Gemeinsam verwendete Verfahren zur Schlüsselverwaltung

♦ **Schlüsselarten**

Bei den Sicherheitsverfahren DDV-1, RAH-9, RAH-10, RDH-1, RDH-2, RDH-5, RDH-8, RDH-9 und RDH-10 können Kunde und Kreditinstitut über zwei Schlüssel bzw. Schlüsselpaare verfügen:

- einen Signierschlüssel bzw. -schlüsselpaar
- einen Chiffrierschlüssel bzw. -schlüsselpaar

Der Signierschlüssel wird zum Unterzeichnen von Transaktionen verwendet, während der Chiffrierschlüssel zum Verschlüsseln von Nachrichten dient.

Bei den Verfahren RAH-7, RDH-3, RDH-6 und RDH-7 können Kunde und Kreditinstitut über bis zu drei Schlüssel bzw. Schlüsselpaare verfügen:

- einen Schlüssel für digitale Signaturen
- einen Signierschlüssel
- einen Chiffrierschlüssel

Abhängig von der Personalisierung der Chipkarte können Signier- und Chiffrierschlüssel identisch sein.

Der Signierschlüssel und der DS-Schlüssel werden zum Unterzeichnen von Transaktionen verwendet, während der Chiffrierschlüssel zum Verschlüsseln von Nachrichten dient. Falls kreditinstitutsseitig nur Geschäftsvorfälle angeboten werden, für die gemäß Bankparameterdaten die Unterzeichnung mit dem Signierschlüssel ausreichend ist, ist der DS-Schlüssel nicht erforderlich.

Bei Verwendung von Schlüsseldateien (Sicherheitsprofil RAH,10, RDH-1, RDH-2 und RDH-10) wird dringend empfohlen, dass getrennte Signier- und Chiffrierschlüssel zum Einsatz kommen.

♦ **Schlüsselnamen**

Der Schlüsselname bei den 2-Key-Triple-DES- und RSA-Schlüsseln setzt sich aus den folgenden alphanumerischen Komponenten zusammen:

- Ländercode
  (max. 3 Byte, es wird gemäß ISO 3166 der numerische Ländercode verwendet)
- Kreditinstitut
  (max. 30 Byte, normalerweise Bankleitzahl, vgl. [Formals], Kapitel II.5.3.2)

- Benutzerkennung
  (max. 30 Byte, kann vom Kreditinstitut festgelegt werden, vgl. [Formals], Kapitel III.1.1)
- Schlüsselart
  (1 Byte, D: DS-Schlüssel; S: Signierschlüssel; V: Chiffrierschlüssel)
- Schlüsselnummer
  (max. 3 Byte)
- Versionsnummer
  (max. 3 Byte)

Falls kein öffentlicher Schlüssel des Kreditinstituts vorliegt, so ist als Versionsnummer der Wert „999“ einzustellen. Damit wird kreditinstitutsseitig auf den aktuell gültigen Schlüssel referenziert (Ein Kreditinstitut kann während einer Übergangszeit evtl. mehrere Schlüssel bis zu einem Verfallsdatum vorhalten. Aktuell gültig ist jeweils der neueste Schlüssel).

♦ **Generierung von Nachrichtenschlüsseln**

Zur Chiffrierung von Nachrichten wird ein dynamisch erzeugter Nachrichtenschlüssel verwendet, der folgendermaßen gebildet wird:

RAH-Verfahren:

1. Generieren einer 32 Byte langen Zufallszahl

RDH-Verfahren:

1. Generieren einer 16 Byte langen Zufallszahl
2. Erzeugung von ungerader Parität (optional)
3. Testen, ob erste und zweite Schlüsselhälfte unterschiedlich (optional)
4. Testen nach schwachen und semi-schwachen Schlüsseln (optional) (s. Kap. B.2.2)

B.3.1.2 Symmetrische Schlüssel für DDV

Für Verschlüsselung und MAC-Berechnung werden, wie unter VI.3.1.1 beschrieben, unterschiedliche Schlüssel für Signatur und Chiffrierung verwendet.

B.3.1.2.1 Schlüsselgenerierung

Beim symmetrischen Verfahren (DDV) sind zur Bildung eines kundenindividuellen Schlüssels beim Kreditinstitut zwei Voraussetzungen zu erfüllen:

- Generierung eines ZKA-weit eindeutigen 2-Key-Triple-DES-Masterkey pro Schlüsselart und Ablegen in einer sicheren Umgebung (Hardwareeinrichtung) als Key Generating Key (KGK).
- Herleiten des jeweiligen kundenindividuellen Schlüssels mittels CID-Feld (Cardholders Information Data = Feld „EF_ID“) auf der ZKA-Chipkarte und entsprechendem 2-Key-Triple-DES-Masterkey.

♦ **Generierung eines 2-Key-Triple-DES-Masterkey:**

Für die Generierung von ZKA-weit einheitlichen 2-Key-Triple-DES-Masterkeys (KGK = Key Generating Key), die als Basis für die Herleitung der kundenindividuellen Signier- und Chiffrierschlüsseln dienen, ist folgendes Verfahren, analog der ZKA-Chipkarte, zu verwenden:

1. Generieren einer 16 Byte langen Zufallszahl
2. Erzeugung von ungerader Parität (optional)
3. Testen, ob erste und zweite Schlüsselhälfte unterschiedlich
4. Testen nach schwachen und semi-schwachen Schlüsseln (s. Kap. B.2.2)

♦ **Herleitung von Kartenschlüsseln:**

Zur eindeutigen Herleitung der symmetrischen Signier- und Chiffrierschlüssel wird das Feld „EF_ID“ im Master File (MF) der ZKA-Chipkarte (Cardholders Information Data (CID) ohne Padding) zusätzlich übertragen (s. DEG „Sicherheitsidentifikation, Details“).

Ein kartenindividueller Schlüssel KK von 16 Byte Länge wird aus

- KGK (Key Generating Key, 16 Byte)
- CID (vollständiger Inhalt von EF_ID, mit X’00’ auf das nächste Vielfache von 8 Byte Länge aufgefüllt) und
- dem öffentlich bekannten Initialwert I = X’52 52 52 52 52 52 52 52 25 25 25 25 25 25 25 25’ (16 Byte)

zu

KK = P(d * KGK(H(I, CID)))

berechnet.

Hierbei bezeichnen

- ‘P’ die Funktion "Parity Adjustment" auf ungerade Parität, die wie folgt definiert ist:
  Sei b_1,..,b_8 die Darstellung eines Byte als Folge von 8 bit. Dann setzt P das niedrigstwertige bit b_8 jedes Byte auf ungerade Parität, d.h. b_8 wird in jedem Byte so gesetzt, dass es eine ungerade Anzahl von 1 enthält.

- ‘d * KGK’ die 2-Key-Triple-DES-Entschlüsselung im ECB-Mode (ISO 10116) mit dem Schlüssel KGK.
- ‘H’ die in ISO 10118-2 definierte Hash-Funktion.

B.3.1.2.2 Initiale Schlüsselverteilung

Die initiale Schlüsselverteilung erfolgt implizit mit der Verteilung der Chipkarte.

B.3.1.2.3 Schlüsseländerungen

Beim symmetrischen Verfahren (DDV) ist wegen der Verknüpfung mit der Chipkarte auf elektronische Weise keine Änderung einzelner kartenindividueller Schlüssel möglich. Im Falle einer vermuteten Kompromittierung muss daher ein Kartenaustausch oder ein Ersatz aller Schlüssel und des Feldes „EF_ID“ erfolgen.

Bei einer Schlüsseländerung wird die Signatur-ID (Sequenzzähler der Chipkarte) auf 1 zurückgesetzt. Die im Kreditinstitut geführte Liste der eingereichten bzw. noch nicht eingereichten Signatur-IDs (s. Doppeleinreichungskontrolle) wird gelöscht.

B.3.1.2.4 Schlüsselverteilung nach Kompromittierung

Die Schlüsselverteilung nach einer Kompromittierung erfolgt ebenfalls mittels Vergabe einer neuen Chipkarte bzw. Ersatz aller Schlüssel und des EF-ID-Feldes. Die alte Chipkarte bzw. deren Schlüssel werden gesperrt.

B.3.1.3 Asymmetrische Schlüssel für RAH und RDH

Grundsätzlich können Kunde und Kreditinstitut beim asymmetrischen Verfahren (RAH und RDH) über maximal drei Schlüsselpaare verfügen:

- ein Signierschlüsselpaar
- ein Chiffrierschlüsselpaar
- ein Schlüsselpaar für die Erzeugung Digitaler Signaturen (DS)

Der Signierschlüssel sowie der DS-Schlüssel werden zum Unterzeichnen von Nachrichten verwendet, während der Chiffrierschlüssel zum Verschlüsseln von Nachrichten dient (vgl. Kapitel B.1.1).

Falls ein Kreditinstitut seine Nachrichten nicht signiert, kann es auf das Signier-Schlüsselpaar verzichten.

B.3.1.3.1 Schlüsselgenerierung

Die Schlüsselpaare des Kunden sind vom Kundenprodukt bzw. von der Chipkarte zu erzeugen. Die Schlüsselpaare des Kreditinstituts sind vom Kreditinstitut zu erzeugen. Die privaten Schlüssel sind jeweils geheim zu halten.

Die Schlüsselgenerierung hat gemäß dem folgenden Ablauf stattzufinden:[11]

1. Es wird ein konstanter öffentlicher Exponent e und ein für jeden Kunden individueller Modulus n für jedes eingesetzte RSA-Schlüsselsystem verwendet.
2. Der konstante öffentliche Exponent e wird auf die 4. Fermat'sche Primzahl festgelegt: $e = 2^{16} + 1$
3. Der Modulus n eines jeden RSA-Schlüsselsystems hat eine Länge von N Bit. Es sind keine führenden 0-Bits erlaubt, so dass auf jeden Fall gilt: $2^{N-1} \leq n < 2^N$
4. Der Zielwert für N ist bei RDH-1 768, wobei eine aus der Suche nach starken Primzahlen resultierende Unterschreitung dieses Wertes um maximal 60 Bit zulässig ist. Bei RDH-2, RDH-3 und RDH-5 liegt der Zielwert für N zwischen 1024 und 2048. Bei RAH-7, RAH-9 und RAH-10 sowie RDH-6, RDH-7, RDH-8, RDH-9 und RDH-10 ergibt sich der Zielwert für N gemäß den Empfehlungen aus [DK Krypto].

Schlüsselgenerierung bei RAH10 und RDH-10:
Das Kundensystem muss sicher stellen, dass die Schlüssellänge eines neu generierten Schlüsselpaares des Kunden gleich der Länge des öffentlichen Signierschlüssels des Instituts ist, falls das Institut Instituts signaturen unterstützt. Anderenfalls ist die Länge des Chiffrierschlüssels maßgebend.

~~5. n ist das Produkt zweier großer, zufällig ausgewählter Primzahlen p und q. Folgende Anforderungen werden an die Faktoren p und q gestellt:~~

- ~~p hat eine vorher festgelegte minimale Länge~~

[11] Das Verfahren entspricht dem des DFÜ-Abkommens.

- ~~p - 1 hat einen großen Primteiler[12] r~~
- ~~p + 1 hat einen großen Primteiler s~~
- ~~r - 1 hat einen großen Primteiler~~

~~Die entsprechenden Forderungen werden an q gestellt.~~

~~Die Längen von p und q sollen sich um höchstens 12 Bits unterscheiden.~~

~~Bei der Wahl von p und q ist sicherzustellen, dass e kein Primfaktor von p - 1 oder q - 1 ist.~~

B.3.1.3.2 Behandlung von Zertifikaten

In FinTS ist die Verwendung von Zertifikaten durch die vorgesehenen Elemente unterstützt, es existieren jedoch keine Prozesse für das Zertifikatsmanagement. Diese sollen zu einem späteren Zeitpunkt auf Basis einer standardisierten Zertifizierungsinfrastruktur übernommen werden.

Folgende Festlegungen gelten für die Belegung der Zertifikatsfelder in den FinTS-Segmenten:

1. Allgemein
   Bei Verwendung des Signaturschlüssels (D-Schlüssel) wird grundsätzlich in allen Nachrichten ein Zertifikat im Signaturkopf mitgeschickt.

   Bei Verwendung des Authentifikationsschlüssels (S-Schlüssel) kann ein Zertifikat in den Signaturkopf eingestellt werden.
   Im Verschlüsselungskopf kann ebenfalls ein Zertifikat eingestellt werden. Ggf. dort eingestellte Zertifikate können vom Institut ignoriert werden.

2. Erstmalige Übermittlung Kundenschlüssel bzw. Schlüsseländerung

   Bei der Erstmaligen Übermittlung der Kundenschlüssel bzw. bei der Schlüsseländerung wird grundsätzlich der Authentifikationsschlüssel (S-Schlüssel) und wahlweise das zugehörige Zertifikat verwendet. Das Zertifikat wird nur in das vorgesehene Element im Geschäftsvorfall (HKSAK bzw. HKISA) eingestellt (nicht in den Signaturkopf).

3. Signaturkarten-Profil mit drei unterschiedlichen Schlüsseln

   Wenn ein Signaturkarten-Profil mit 3 unterschiedlichen Schlüsseln verwendet wird, muss bei der Erstmaligen Übermittlung der Kundenschlüssel bzw. der Schlüsseländerung auch die Möglichkeit bestehen, das Zertifikat für den eigenen Verschlüsselungsschlüssel im jeweiligen Geschäftsvorfall (HKSAK bzw. HKISA) mitzuschicken.

---

[12] Der Primteiler sollte dabei ungefähr der Länge des Schlüssels entsprechen.

♦ **Ausgewählte Beispiele für Rückmeldungscodes**

Code	Beispiel
9351	Zertifikat noch nicht gültig
9352	Zertifikat zurückgezogen bzw. gesperrt
9353	Zertifikatssignatur falsch
9354	Zertifizierungsinstanz (Herausgeber) nicht akzeptiert
9355	Fehler im Zertifikatsaufbau
9356	Zertifikatstyp nicht akzeptiert

B.3.1.3.3 Initiale Schlüsselverteilung

Der Kunde benötigt für das Einrichten eines neuen Zugangs folgende Initialinformationen:

- seine Benutzerkennung
- Informationen zum Kommunikationszugang

Die Übermittlung dieser Informationen ist auf folgenden Wegen denkbar:

- Schriftstück des Kreditinstitutes (Benutzerkennung und Zugangsdaten müssen manuell vom Kunden eingegeben werden)
- Schlüsseldatei des Kreditinstitutes mit folgendem Inhalt:
  - Segment HIUPA der UPD inkl. Benutzerkennung
  - Aktuelle Version der Zugangsdatenbank des jeweiligen Verbandes bzw. Segment HIKOM mit den Kommunikationszugangsdaten des jeweiligen Instituts
- Chipkarte des Kreditinstitutes, die die Kommunikationszugangsdaten in der Applikation EF_NOTEPAD enthält.

Zu Beginn muss ein gegenseitiger Austausch der öffentlichen Schlüssel von Kunde und Kreditinstitut erfolgen. Zukünftig soll dieser Austausch durch eine Anforderung der Zertifikate bei den jeweiligen Zertifizierungsinstanzen erfolgen. Dieser Prozess findet außerhalb des HBCI-Protokolls statt und wird daher hier nicht näher beschrieben. Übergangsweise kann der Schlüsselaustausch auch im Rahmen eines HBCI-Dialoges erfolgen.

Hierzu ist folgender Ablauf vorgesehen:

1. Das Kreditinstitut übermittelt seinen öffentlichen Chiffrierschlüssel an den Kunden. Falls es Nachrichten signiert, übermittelt es ebenfalls seinen öffentlichen Signierschlüssel. Hierzu gibt es zwei Möglichkeiten:
   - Zusenden bzw. Aushändigung der Schlüssel und anderer relevanter Daten auf einem Medium (z.B. Schlüsseldatei[13], Chipkarte) bei Vertragseröffnung.

   Falls dem Kunden eine Schlüsseldatei zugesendet wird, hat diese folgende Daten zu enthalten:

[13] Es kann sich hierbei um dieselbe Schlüsseldatei handeln, mit der dem Kunden seine Benutzerkennung mitgeteilt wird (s.o.).

- Datei mit bis zu drei Segmenten vom Typ HIISA, die jeweils einen öffentlichen Schlüssel des Kreditinstitutes enthalten
- BPD des Kreditinstitutes

- Übertragung der Schlüssel beim Erstzugang

  (1) Der Kunde fordert die öffentlichen Schlüssel und die BPD mit Hilfe der Key-Management-Nachricht „Erstmalige Anforderung der Schlüssel des Kreditinstituts“ (s. Kap. B.6.2.1) an. Diese Nachricht ist weder signiert noch chiffriert.

  (2) Der weitere Ablauf ist abhängig davon, ob das Kreditinstitut seine Antwortnachrichten signiert.

  Fall A: Das Kreditinstitut signiert

  Der Kunde erhält die öffentlichen Schlüssel des Kreditinstituts zurückgemeldet. Während die Authentizität des Chiffrierschlüssels dabei durch die Signatur gesichert ist, ist die Authentizität des Signierschlüssels nicht gesichert, da das Kundensystem die Echtheit der Signatur noch nicht prüfen kann.

  Fall B: Das Kreditinstitut signiert nicht

  Der Kunde erhält nur den öffentlichen Chiffrierschlüssel zurückgemeldet. Dessen Authentizität ist dabei nicht gesichert.

  (3) Die Sicherung der Authentizität dieser Schlüssel kann über folgende Mechanismen erfolgen:

  Fall A: Ini-Brief

  Diese Nachricht wird von einem Ini-Brief an den Kunden begleitet. Die Gestaltung ist dem Kreditinstitut freigestellt, sollte sich aber am Muster in Abbildung 14 bzw. Abbildung 15 orientieren. Der Ini-Brief enthält für den Fall A Exponent und Modulus des Signierschlüssels sowie dessen Hashwert und für den Fall B Exponent und Modulus des Chiffrierschlüssels sowie dessen Hashwert.

  Bei RDH-1 sind dabei Exponent und Modulus mit führenden Nullen (X’00’) auf 768 Bit zu ergänzen (in den Abbildungen nicht mehr berücksichtigt).

  Bei RAH-7, RAH-9 und RAH-10 sowie RDH-2, RDH-3, RDH-5, RDH-6, RDH-7, RDH-8, RDH-9 und RDH-10 ist hierbei der Exponent mit führenden Nullen (X’00’) auf die reale Länge des Modulus zu ergänzen.

  Für die Auswahl des zu verwendenden Hashwertverfahrens gelten folgende Regeln:

  RIPEMD-160:
  Bei RDH-1, RDH-2, RDH-3 und RDH-5 generell;
  bei RDH-6, RDH-7, RDH-8, RDH-9 oder RDH-10, wenn EF_NOTEPAD, HBCI-Version C0=001 (vgl. Abschnitt C.1.2.2.2)

SHA-256:
bei RAH-7, RAH-9 und RAH-10 sowie RDH-6, RDH-7, RDH-8, RDH-9 oder RDH-10, wenn EF_NOTEPAD, HBCI-Version C0=002 (vgl. Abschnitt C.1.2.2.2)

Ferner enthält der Ini-Brief den jeweiligen Schlüsselnamen.

Bei der Hashwertbildung ist wie folgt vorzugehen:

a) RDH-1: Padding der höchstwertigen Bits von Exponent und Modulus des Schlüssels mit Nullen (X'00') auf 1024 Bit

   sonst: Padding der höchstwertigen Bits des Exponenten mit Nullen (X'00') auf die reale Länge des Modulus

b) Konkatenierung von Exponent und Modulus (Exponent || Modulus)

c) Bildung des Hashwerts mittels RIPEMD-160 bzw. SHA-256 gemäß Kap. B.2.1.1 über diesen Ausdruck

Nach Erhalt des Ini-Briefs führt der Kunde einen Vergleich des im Ini-Brief aufgeführten Hashwerts mit dem Hashwert des vom Kreditinstitut übermittelten Schlüssels durch.

Bei Übereinstimmung der Hashwerte gelten der bzw. die öffentlichen Schlüssel des Kreditinstituts als authentisiert.

Das Kundenprodukt sollte den Hashwertvergleich für den Kunden in geeigneter Weise unterstützen.

Fall B: Übermittlung des Hashwerts auf der Chipkarte

Auf der Karte befindet sich in der Applikation EF_NOTEPAD (s. Kap. C.1.1) für Fall A der Hashwert des öffentlichen Signierschlüssels des Kreditinstituts und für Fall B der Hashwert des öffentlichen Chiffrierschlüssels des Kreditinstituts. Die Hashwertbildung erfolgt wie in Fall A.

Dieser Hashwert wird vom Kundenprodukt mit dem Hashwert des in der Nachricht übermittelten Schlüssels verglichen.

Das Kundenprodukt sollte den Kunden über das Ergebnis des Hashwertvergleichs informieren.

Bei Übereinstimmung der Hashwerte gelten der bzw. die öffentlichen Schlüssel des Kreditinstituts als authentisiert.

Fall C: Prüfung des übermittelten Zertifikates

Falls das Kreditinstitut über zertifikatsbasierte Schlüssel verfügt, übermittelt es das jeweilige Zertifikat in der Nachricht zusammen mit dem öffentlichen Schlüssel.

Somit ist der Kunde in der Lage, das Zertifikat bei der jeweiligen Zertifizierungsinstanz zu verifizieren. Diese Verifikation findet außerhalb des FinTS-Protokolls statt und wird daher hier nicht näher beschrieben.

Ein Hashwertvergleich wie in den beiden anderen Fällen ist nicht erforderlich.

Das Kundenprodukt sollte den Kunden über das Ergebnis der Zertifikatsprüfung informieren.

2. Der Kunde übermittelt alle seine öffentlichen Schlüssel, die mit dem privaten Signierschlüssel unterzeichnet wurden, im Rahmen der Key-Management-Nachricht „Erstmalige Übermittlung der Schlüssel des Kunden“ an das Kreditinstitut (vgl. Kapitel B.8.1.3). Diese Nachricht muss sowohl signiert als auch chiffriert sein.

3. Um die Authentizität der Schlüssel zu gewährleisten, sind folgende Mechanismen möglich:

   Fall A: Ini-Brief

   Der Kunde erfährt anhand des Rückmeldungscodes 3310 („Ini-Brief erforderlich“) in der Kreditinstitutsnachricht, dass diese Nachricht durch einen Ini-Brief gemäß dem in Abbildung 14 bzw. Abbildung 15 aufgeführten Muster begleitet werden muss. Im Ini-Brief bestätigt der Kunde ausschließlich den öffentlichen Signierschlüssel mit handschriftlicher Unterschrift. Eine Bestätigung des öffentlichen Chiffrierschlüssels ist nicht erforderlich, da dieser mit dem Signierschlüssel signiert wird und damit authentifiziert ist. Neben dem Schlüssel und dem Schlüsselnamen wird im Ini-Brief der Hashwert des Schlüssels aufgeführt. Dieser wird ebenso gebildet wie der Hashwert im Ini-Brief des Kreditinstituts (s.o.).

   Im Kreditinstitut findet ein Vergleich zwischen dem im Ini-Brief aufgeführten Hashwert und dem Hashwert des vom Kunden übermittelten öffentlichen Signierschlüssels statt.

   Falls dieser Vergleich positiv verläuft, werden die öffentlichen Schlüssel des Kunden freigeschaltet.

   Fall B: Prüfung des übermittelten Zertifikates

   Der Kunde erfährt anhand des Rückmeldungscodes 3320 („Ini-Brief nicht erforderlich“) in der Kreditinstitutsnachricht, dass das Kreditinstitut die Prüfung der Authentizität der Schlüssel auf Basis eines Zertifikates vornehmen kann.

   Falls der Kunde über zertifikatsbasierte Schlüssel verfügt, übermittelt er daher das jeweilige Zertifikat in der Nachricht zusammen mit dem öffentlichen Schlüssel. Somit ist das Kreditinstitut in der Lage, das Zertifikat bei der jeweiligen Zertifizierungsinstanz zu verifizieren.

Diese Verifikation findet außerhalb des FinTS-Protokolls statt und wird daher hier nicht näher beschrieben.

Ein Hashwertvergleich wie in Fall A ist nicht erforderlich.

4. Im nächsten Schritt wird der Kunde freigeschaltet.
5. Es hat eine Synchronisierung der Kundensystem-ID zu erfolgen (s. [Formals], Kap. III.8).
6. Hiermit ist die Erstinitialisierung abgeschlossen und der Kunde kann Auftragsnachrichten senden.

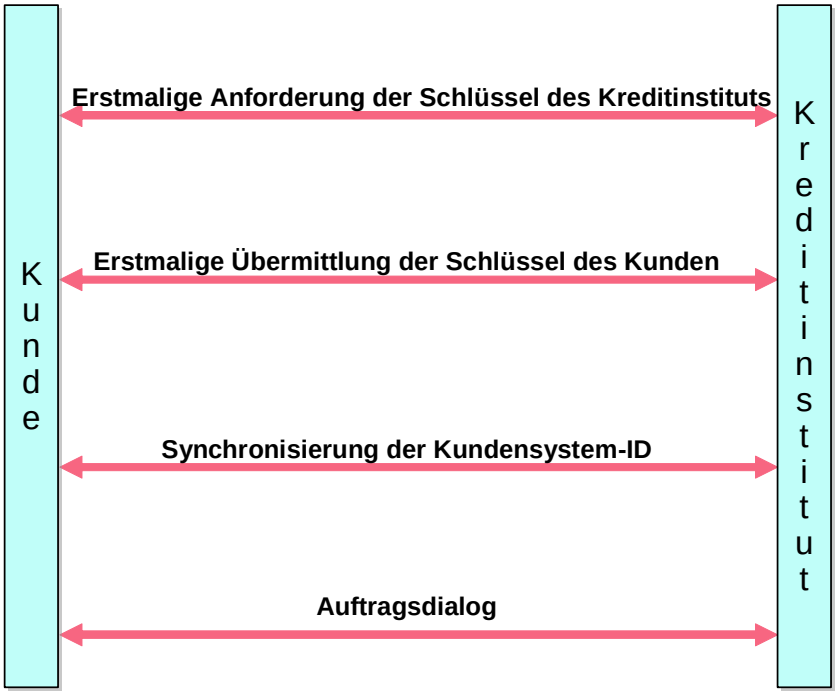


Abbildung 13: Ablauf der Erstinitialisierung bei RDH

Um die Multibankfähigkeit verschiedener Kundenprodukte zu sichern, gelten für die Ini-Schlüsseldatei folgende Namenskonventionen:

- Segment HIUPA: <Benutzerkennung>.UPA
- Datei mit den öffentlichen Schlüsseln: <Benutzerkennung>.PKD
- BPD: <Bankleitzahl>.BPD
- Segment mit Kommunikationszugang: <Bankleitzahl>.KOM
- Zugangsdatenbank des Verbandes: BDB.KOM, BVR.KOM, DSGV.KOM bzw. VOEB.KOM

Falls die Benutzerkennung nicht im Dateisystem darstellbar ist, ist sie entsprechend zu kürzen. Die Schlüsseldatei muss im Standardformat des jeweiligen Betriebssystems formatiert sein. Die Dateien sind im Stammverzeichnis der Schlüsseldatei abzulegen.

Ini-Brief HBCI

Benutzername	_____________	Kundensoftware-interner Name (Angabe freigestellt)
Datum	_____________	Datum der Erstellung des Initialisierungsauftrags (TT.MM.JJJJ)
Uhrzeit	_____________	Uhrzeit der Erstellung des Initialisierungsauftrags (hh:mm)
Empfänger	_____________	Kreditinstitutskennung (wird vom jeweiligen Kreditinstitut mitgeteilt)
Benutzerkennung	_____________	max. 30 Stellen alphanumerisch (wird vom jeweiligen Kreditinstitut mitgeteilt)
Schlüsselnummer	_____________	Nummer des Signierschlüssels (max. 3 Stellen)
Schlüsselversion	_____________	Version des Signierschlüssels (max. 3 Stellen)
HBCI-Version	_____________	derzeit 3.0
Sicherheitsprofil	_____________	RDH-2, RDH-3 oder RDH-5

Öffentlicher Schlüssel für die elektronische Signatur:

Exponent <Reale Moduluslänge>

```
00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00
00  00  00  00  00  00  00  00  00  00  00  00  00  01  00  01
```

Modulus ..2048

```
97  A9  D8  FD  68  46  BA  8C  5E  DB  E0  F5  1C  35  41  41
3C  4A  B3  7C  95  6A  CC  11  29  E3  A7  63  A6  7B  A8  6D
E8  1D  99  D2  74  24  6E  6C  C6  FD  A9  9C  A2  19  15  44
BF  2A  5D  B6  93  5B  FA  FB  88  19  FE  B2  1C  96  FB  4F
5E  66  6A  2D  F8  80  5E  84  9B  42  8C  E6  9A  76  AB  52
5A  36  C1  6D  81  D1  65  0A  BC  86  35  86  E7  8E  5B  B5
EF  E0  84  A1  44  76  69  DE  34  D5  31  A2  D0  25  02  78
B5  BE  AB  C0  D9  30  B1  04  91  DF  0D  3B  E1  33  3D  E7
24  57  CD  E5  A2  13  EF  95  4B  23  81  58  91  60  85  71
39  3E  00  92  7C  F9  06  3A  97  3D  3F  87  52  80  AF  4E
37  39  F1  9E  14  A5  71  30  33  6C  E4  DD  38  66  E8  B2
29  6B  BF  F2  25  36  3E  0F  A0  55  39  4A  31  7A  69  88
9B  D6  26  B6  B6  53  CA  C7  40  4A  60  79  29  C4  97  DC
43  3B  54  A6  B6  44  33  E0  E8  05  17  43  50  13  EE  D4
F2  EF  F2  BD  B2  2A  B9  87  52  95  FC  56  93  24  75  A7
E7  D1  CD  90  9A  D3  D8  78  99  1D  C3  21  4D  2F  5A  A3
```

Hashwert (RIPEMD-160):

```
0B  AD  D5  D5  FD  57  0D  34  E7  84  7C  ED  AE  4B  3D  EE
6B  EB  98  AC
```

Ich bestätige hiermit den obigen öffentlichen Schlüssel für meine elektronische Signatur.

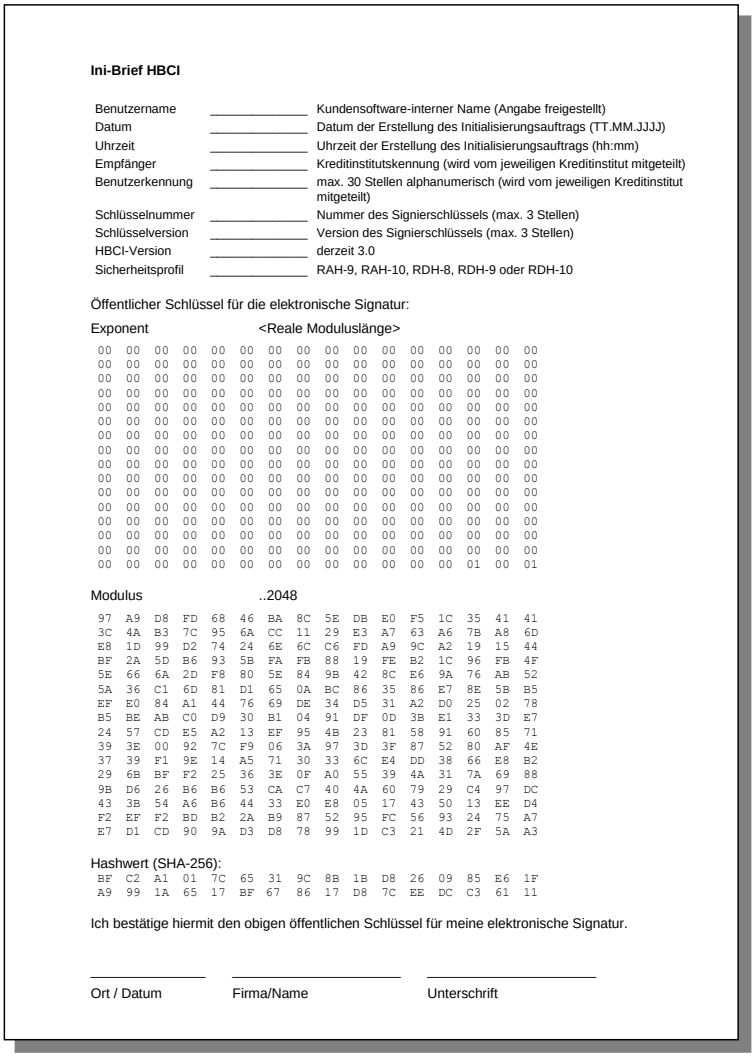
_______________	______________________	______________________
Ort / Datum	Firma/Name	Unterschrift

Abbildung 14: Beispiel für die Gestaltung des Ini-Briefs bei RDH-2 oder RDH-5

Die folgende Abbildung zeigt ein Beispiel für einen Ini-Brief bei Verwendung der Sicherheitsprofile RAH-9, RAH-10, RDH-8, RDH-9 oder RDH-10 unter Verwendung von SHA-256 als Hashwertverfahren[14].

---

[14] SHA-256 als Haswertverfahren kann nur bei Bankensignaturkarten zum Einsatz kommen, die über die EF_NOTEPAD HBCI-Version C0=002 verfügen (vgl Abschnitt C.1.2.2.2).



Ini-Brief HBCI

Benutzername	______________	Kundensoftware-interner Name (Angabe freigestellt)
Datum	______________	Datum der Erstellung des Initialisierungsauftrags (TT.MM.JJJJ)
Uhrzeit	______________	Uhrzeit der Erstellung des Initialisierungsauftrags (hh:mm)
Empfänger	______________	Kreditinstitutskennung (wird vom jeweiligen Kreditinstitut mitgeteilt)
Benutzerkennung	______________	max. 30 Stellen alphanumerisch (wird vom jeweiligen Kreditinstitut mitgeteilt)
Schlüsselnummer	______________	Nummer des Signierschlüssels (max. 3 Stellen)
Schlüsselversion	______________	Version des Signierschlüssels (max. 3 Stellen)
HBCI-Version	______________	derzeit 3.0
Sicherheitsprofil	______________	RAH-9, RAH-10, RDH-8, RDH-9 oder RDH-10

Öffentlicher Schlüssel für die elektronische Signatur:

Exponent <Reale Moduluslänge>

```
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 01 00 01
```

Modulus ..2048

```
97 A9 D8 FD 68 46 BA 8C 5E DB E0 F5 1C 35 41 41
3C 4A B3 7C 95 6A CC 11 29 E3 A7 63 A6 7B A8 6D
E8 1D 99 D2 74 24 6E 6C C6 FD A9 9C A2 19 15 44
BF 2A 5D B6 93 5B FA FB 88 19 FE B2 1C 96 FB 4F
5E 66 6A 2D F8 80 5E 84 9B 42 8C E6 9A 76 AB 52
5A 36 C1 6D 81 D1 65 0A BC 86 35 86 E7 8E 5B B5
EF E0 84 A1 44 76 69 DE 34 D5 31 A2 D0 25 02 78
B5 BE AB C0 D9 30 B1 04 91 DF 0D 3B E1 33 3D E7
24 57 CD E5 A2 13 EF 95 4B 23 81 58 91 60 85 71
39 3E 00 92 7C F9 06 3A 97 3D 3F 87 52 80 AF 4E
37 39 F1 9E 14 A5 71 30 33 6C E4 DD 38 66 E8 B2
29 6B BF F2 25 36 3E 0F A0 55 39 4A 31 7A 69 88
9B D6 26 B6 B6 53 CA C7 40 4A 60 79 29 C4 97 DC
43 3B 54 A6 B6 44 33 E0 E8 05 17 43 50 13 EE D4
F2 EF F2 BD B2 2A B9 87 52 95 FC 56 93 24 75 A7
E7 D1 CD 90 9A D3 D8 78 99 1D C3 21 4D 2F 5A A3
```

Hashwert (SHA-256):

```
BF C2 A1 01 7C 65 31 9C 8B 1B D8 26 09 85 E6 1F
A9 99 1A 65 17 BF 67 86 17 D8 7C EE DC C3 61 11
```

Ich bestätige hiermit den obigen öffentlichen Schlüssel für meine elektronische Signatur.

______________	____________________	____________________
Ort / Datum	Firma/Name	Unterschrift

Abbildung 15: Beispiel für die Gestaltung des Ini-Briefs bei RAH-9, RAH-10, RDH-8, RDH-9 oder RDH-10

B.3.1.3.4 Schlüsseländerungen

- **Routinemäßige Schlüsseländerung des Kunden**

  Bei Speicherung der Schlüssel auf einer Chipkarte ist i.d.R. auf elektronische Weise keine Änderung einzelner kartenindividueller Schlüssel möglich. Im Falle einer routinemäßigen Schlüsseländerung (z.B. bei Ablauf des Zertifikates) oder einer vermuteten Kompromittierung muss daher ein Kartenaustausch oder ein Ersatz aller Schlüssel erfolgen.

  Falls die Karte die Generierung neuer Schlüssel zulässt oder im Falle anderer Speichermedien (Schlüsseldatei) ändert der Kunde seine Schlüsselpaare unabhängig voneinander.

  Der Kunde sendet je Kreditinstitut im Rahmen eines HBCI-Dialoges eine Nachricht, in welcher dieses über einen neuen öffentlichen Schlüssel informiert wird (vgl. Kapitel B.6.2.1). Die Nachricht ist mit dem alten (bei Wechsel des Signierschlüssels), respektive dem aktuellen (bei Wechsel des DS-Schlüssels oder des Chiffrierschlüssels) privaten Signierschlüssel des Kunden zu signieren und mit dem aktuellen Chiffrierschlüssel des Kreditinstituts zu chiffrieren.

Das Kreditinstitut speichert diesen neuen öffentlichen Schlüssel des Kunden und verwendet ihn ab sofort (d.h. bereits in der Antwortnachricht) für alle Verschlüsselungen bzw. Verifikationen von Signaturen. Gleichzeitig kann der alte Schlüssel gesperrt werden. Zusätzlich ist es jedoch bei kartengestützten Verfahren – unabhängig von der Nutzung von Zertifikaten – erlaubt, einen Schlüssel für die Laufzeit der Karte weiter aktiv zu halten und somit zwei Schlüssel parallel zu unterstützen.

Falls die Übermittlung der neuen Schlüssel aus irgendeinem Grunde fehlschlägt, kann der Kunde den Vorgang beliebig wiederholen.

Bei einer Schlüsseländerung wird die Signatur-ID auf 1 zurückgesetzt. Die Liste der eingereichten bzw. noch nicht eingereichten Signatur-IDs (s. Doppeleinreichungskontrolle) wird gelöscht.

- **Routinemäßige Schlüsseländerung des Kreditinstituts**

  Ein Kreditinstitut generiert bei Bedarf ein neues Schlüsselpaar.

  Der Kunde sendet jeweils bei der Dialoginitialisierung die Referenz auf die öffentlichen Schlüssel des Kreditinstitutes mit (vgl. [Formals], Kapitel III.3.1). Falls das Kreditinstitut über aktuellere öffentliche Schlüssel verfügt, werden diese in der Kreditinstitutsnachricht mitübertragen (vgl. [Formals], Kapitel III.3.2 respektive B.6.1.3). Die neuen Schlüssel gelten ab sofort, d.h. bereits für die erste Auftragsnachricht nach der Dialoginitialisierung. Da das Kreditinstitut i.d.R. aber auch noch die alten Schlüssel aktiv hält, werden für einen begrenzten Zeitraum auch noch Nachrichten akzeptiert, die mit den alten Kreditinstitutsschlüsseln chiffriert wurden.

  Zur Verifikation des kreditinstitutsseitigen öffentlichen Schlüssels auf dem Kundensystem kann das entsprechende Kreditinstitut die Kreditinstitutsnachricht mit dem alten Signierschlüssel signieren (wenn eine kreditinstitutsseitige Signatur vorgesehen ist) oder den Hashwert des öffentlichen Schlüssels analog der initialen Schlüsselverteilung an den Kunden übermitteln. Die Verifikation ist grundsätzlich optional.

  Für den Fall, dass der alte Kreditinstitutsschlüssel nicht mehr zur Verfügung steht oder gesperrt werden musste, wird dem Kunden - falls er den alten Kreditinstitutsschlüssel zur Chiffrierung der Dialoginitialisierung verwendet – der Rückmeldungscode "9030" mit dem Hinweis "Fehler beim Entschlüsseln" gesendet. Ggf. kann die Dialoginitialisierung vom Kreditinstitutssystem auch gar nicht verarbeitet werden, so dass keine Antwort gesendet wird. Daraufhin sollte das Kundenprodukt über den anonymen Dialog mit Hilfe der Nachricht „Erstmalige Anforderung der Schlüssel des Kreditinstituts“ (s. Kap. B.6.2.1) die neuen Kreditinstitutsschlüssel anfordern. Zur Verifikation der neuen Schlüssel muss dem Kunden in diesem Fall zusätzlich ein Ini-Brief mit dem Hashwert des neuen Kreditinstitutsschlüssels zugeschickt werden.

B.3.1.3.5 Schlüsselverteilung nach Kompromittierung

Die Verteilung der Schlüssel nach einer Kompromittierung erfolgt analog der Schlüsselverteilung bei der Initialisierung. Es findet immer ein Austausch aller Schlüssel statt, auch dann, wenn nur einer der Schlüssel kompromittiert wurde.

B.3.2 Schlüsselsperrung

Bei der Schlüssel- bzw. Benutzersperrung muss zwischen folgenden Fällen unterschieden werden:

- Kompromittierung des eigenen Schlüssels
- Verlust des eigenen Schlüssels
- Überschreiten der Anzahl der Falschsignaturen

Zusätzlich müssen bei der Sperrung noch folgende Punkte berücksichtigt werden:

- Information des Kunden
- Entsperrung

Die Sperrung anderer Benutzer wird als eigenständiger Auftrag behandelt und zu einem späteren Zeitpunkt realisiert.

♦ **Kompromittierung des eigenen Schlüssels**

Bei Verdacht auf Kompromittierung des eigenen Schlüssels kann die Sperrung mittels einer speziellen Nachricht (vgl. Kapitel B.8.1.4) erfolgen, welche signiert sein muss.

♦ **Verlust des eigenen Schlüssels**

Bei einem Verlust (inkl. Diebstahl) des eigenen Schlüssels (respektive des Speichermediums) muss der Kunde Schlüssel bzw. Medium sperren und beim Kreditinstitut ein anderes Medium inkl. Schlüssel beantragen.

Eine nicht-signierungspflichtige Sperrmöglichkeit ist optional, da hierdurch die Gefahr des Mißbrauchs gegeben ist (absichtliche Sperrung fremder Anschlüsse). Der Segmentaufbau erfolgt analog der oben beschriebenen Nachricht, jedoch ist keine Signatur nötig (möglich). Die Steuerung hierfür erfolgt über das Feld „Anzahl benötigter Signaturen“ in der UPD.

Eine Sperrung auf anderem Weg (z.B. telefonische Sperrung über Servicezentralen) muss immer möglich sein (z.B. Verlust der eigenen Infrastruktur).

♦ **Überschreiten der Anzahl der Falschsignaturen**

Wird beim Einreichen von Aufträgen durch fehlerhafte Signaturen die festgelegte Anzahl von n Falschsignaturen in Folge überschritten, werden kreditinstitutsseitig die Schlüssel gesperrt. Als Falschsignaturen werden dabei fehlgeschlagene kryptographische Operationen, jedoch z.B. keine fehlerhaften Berechtigungen verstanden.

Bei einer Sperrung aufgrund zu vieler Fehlsignaturen werden alle Kundenschlüssel gesperrt. Sofern die Nachricht lediglich von einem einzigen Benutzer signiert wurde oder falls bei einer mehrfach signierten Nachricht der Dialogführer von der Fehlsignaturensperre betroffen ist, wird der Dialog beendet. Der Dialogabbruch erfolgt dabei kreditinstitutsseitig im Anschluss an die Antwortnachricht, d.h. ein Austausch von Dialogbeendigungsnachrichten findet nicht statt. Die Antwort ist beim DDV-Verfahren weder signiert noch verschlüsselt. Beim RAH- bzw. RDH-Verfahren ist die Antwort signiert (sofern kreditinstitutsseitig signiert wird) aber nicht verschlüsselt. In der Antwortnachricht teilt das Kreditinstitut lediglich den Grund des Dialogendes mit. Antworten auf Aufträge dürfen nicht mitgesendet werden, da diese aufgrund der Sperrung nicht abgesichert werden können.

- **Information des Kunden**

  Im Falle einer Sperrung aufgrund von Schlüsselkompromittierung oder Schlüsselverlust erhält der Kunde auf die Sperrnachricht eine Antwortnachricht (vgl. Kapitel B.8.1.4 b), welche ihm die Sperrung bestätigt. Bei einer Sperrung wegen Überschreitung des Maximalwertes möglicher Falschsignaturen erhält er lediglich einen entsprechenden Rückmeldungscode. In jedem Fall erhält er jedoch entsprechende Fehlermeldungen bei der Einreichung nachfolgender Nachrichten.

- **Entsperrung der Benutzerkennung**

  Eine Entsperrung erfolgt nur gegen handschriftliche Unterschrift des Kunden.

  Ist der Schlüssel kompromittiert oder nicht mehr auffindbar, so wird für den Benutzer eine neue Chipkarte, respektive neue Schlüssel und ein neues EF_ID (DDV), oder ein neues Schlüsselpaar (RAH bzw. RDH) erzeugt und der alte Schlüssel bleibt gesperrt. Es werden in jedem Falle alle Schlüsselpaare neu vergeben, auch wenn nur ein Schlüsselpaar kompromittiert sein sollte. Damit ein Benutzer nach einer Sperrung wieder zum Zugang zum System autorisiert werden kann, darf er in diesem Fall ausnahmsweise einer erneute Erstinitialisierung durchführen und seine Schlüssel über einen Ini-Brief freischalten lassen.

  In den übrigen Fällen kann der Schlüssel einfach durch das Kreditinstitut entsperrt werden.

B.4 Bankfachliche Anforderungen

- **Zu signierende Nachrichten**

  Grundsätzlich sind alle Kundennachrichten zu signieren, bei Sicherheitsprofil RAH-7, RDH-3, RDH-6 und RDH-7 gemäß den in den BPD vorgegebenen Sicherheitsklassen. Ausnahmen gelten beim anonymen Zugang, bei der Erstinitialisierung und der Schlüsselsperrung.

  Die Signatur von Kreditinstitutsnachrichten ist optional.

- **Doppeleinreichungskontrolle**

  Die Doppeleinreichungskontrolle wird mittels eines Zählers pro Signatur realisiert (Signatur-ID), dessen Inhalt jeweils in die Signatur(en) der Nachricht einfließt. Falls als Sicherheitsmedium keine Chipkarte verwendet wird, wird zur Doppeleinreichungskontrolle zusätzlich zur Signatur-ID die Kundensystem-ID benötigt.

  Bei der Doppeleinreichungskontrolle (Verhinderung von Replay-Attacken) ist zu berücksichtigen, dass die sequentiell erzeugten Referenznummern (=Signatur-IDs) beim Kreditinstitut nicht in derselben Reihenfolge eintreffen müssen, da diese kundenseitig auch offline (d.h. zeitlich voneinander unabhängig) generiert werden können. Das Kreditinstitut muss deshalb sicherstellen, dass innerhalb eines bestimmten Zeitraums keine Sequenznummer mehrfach erscheint.

  Aus diesem Grund muss beim Kreditinstitut eine Liste mit den eingereichten (Positivliste) oder noch nicht eingereichten (Negativliste) Signatur-IDs geführt werden. Nach einer festgelegten Aufbewahrungsfrist wird eine Referenznummer nicht mehr akzeptiert. (Konkret wird ein Kreditinstitut eine Nachricht abweisen, welche länger als die vereinbarte Frist nach einer Nachricht mit höherer Signatur-ID eintrifft). Diese Liste muss je Signaturschlüsselpaar geführt werden, d.h., falls der Benutzer sowohl mit dem Signierschlüssel- als auch mit dem DS-Schlüssel unterschreibt, sind zwei Listen erforderlich.

- **Mehrfachsignaturen**

  Bei Mehrfachsignaturen kann unterschieden werden, ob die Reihenfolge der Unterzeichnung bedeutungslos oder relevant ist. Diese Unterscheidung muss nicht nur im Kundenprodukt gemacht werden können, sondern hat auch Einfluss auf die Verarbeitung und Kontrolle im Kreditinstitut. In der vorliegenden FinTS-Version ist die Reihenfolge der Signaturen bedeutungslos.

  Sind die Berechtigungsprofile mehrerer signierender Benutzer zueinander inkonsistent, so liegt es im Ermessen des Kreditinstituts, ob es die Nachricht annimmt oder ablehnt (Beispiel: Der Erfasser einer Nachricht, für deren Aufträge drei Signaturen erforderlich sind, liefert nur eine zweite Signatur eines Benutzers mit, der über das Recht verfügt, die Aufträge alleine zu signieren).

  Ob es zulässig ist, dass bei Mehrfachsignaturen verschiedene Signaturverfahren eingesetzt werden, gibt das Kreditinstitut in den BPD im Segment „Sicherheitsverfahren“ ([Formals], Kap. IV.4) an.

B.5 Formate für Signatur und Verschlüsselung

Für die Speicherung der Sicherheitsinformationen für die Signatur(en) werden unmittelbar nach dem Nachrichtenkopf das (die) Segment(e) „Signaturkopf“ (HNSHK) und unmittelbar vor dem Nachrichtenabschluss das (die) Segment(e) „Signaturabschluss“ (HNSHA) in die bestehende Nachricht eingeschoben.

Dies entspricht dem in UN/EDIFACT definierten Vorgehen und kann folgendermaßen visualisiert werden:

HNHBK	HNSHK	HBCI-Nutzdaten	HNSHA	HNHBS

(Die grau hinterlegten Bereiche gehen in die Signatur mit ein.)

Falls mehrere Signaturen für HBCI-Nachrichten erforderlich sind, so wiederholen sich Signaturkopf und -abschluss entsprechend:

HNHBK	$HNSHK_2$	$HNSHK_1$	HBCI-Nutzdaten	$HNSHA_1$	$HNSHA_2$	HNHBS

(Die grau hinterlegten Bereiche bezeichnen die Daten für die Zweit-Signatur bei beliebiger Reihenfolge der Signaturen (vgl. Kapitel B.4)).

Bei der Verschlüsselung wird nach dem Nachrichtenkopf ein Verschlüsselungskopf-Segment (HNVSK) eingefügt. Dies bedeutet, dass alle Daten nach dem Segmentendekennzeichen des Nachrichtenkopfes bis zum letzten Byte vor dem Nachrichtenabschluss inklusive aller Signaturen in die Verschlüsselung eingehen:

HNHBK	HNVSK	e_k(HNSHK$_n$ | HBCI-Nutzdaten | HNSHA$_n$)	HNHBS

Grundsätzlich erfolgt die Reihenfolge der Sicherheitsverarbeitung in folgender Reihenfolge:

1. elektronische Signatur
2. evtl. Zweit- und Drittsignatur
3. (Komprimierung) und Verschlüsselung

Für die Übermittlung der sicherheitsrelevanten Informationen werden die folgenden Segmente und Datenelementgruppen übertragen.

B.5.1 Signaturkopf

- **Beschreibung**

  Der Signaturkopf enthält Informationen über den damit verbundenen Sicherheitsservice, sowie über den Absender.

- **Format**

  Name: Signaturkopf
  Typ: Segment
  Segmentart: Administration
  Kennung: HNSHK
  Bezugssegment: -
  Version: 4
  Sender: Kunde/Kreditinstitut

Nr.	Name	Typ	Format	Länge	Status	Anzahl	Restriktionen
1	Segmentkopf	DEG			M	1	
2	Sicherheitsprofil	DEG			M	1	
3	Sicherheitsfunktion, kodiert	DE	code	..3	M	1	1, 2
4	Sicherheitskontrollreferenz	DE	an	..14	M	1	<>0
5	Bereich der Sicherheitsapplikation, kodiert	DE	code	..3	M	1	1
6	Rolle des Sicherheitslieferanten, kodiert	DE	code	..3	M	1	1, 3, 4
7	Sicherheitsidentifikation, Details	DEG			M	1	
8	Sicherheitsreferenznummer	DE	num	..16	M	1	
9	Sicherheitsdatum und -uhrzeit	DEG			M	1	
10	Hashalgorithmus	DEG			M	1	
11	Signaturalgorithmus	DEG			M	1	
12	Schlüsselname	DEG			M	1	
13	Zertifikat	DEG			C	1	M: bei RAH-7, RDH-3, RDH-6 und RDH-7 in Verbindung mit mindestens einem zu signierenden Geschäftsvorfall, der Sicherheitsklasse 2, 3 oder 4 erfordert. O: bei RAH-9, RDH-1, RDH-5, RDH-8 und RDH-9 in Verbindung mit zu signierenden Geschäftsvorfällen, die Sicherheitsklasse 1 bis 2 erfordern N: bei DDV-1, RAH-10, RDH-2 und RDH-10

- **Belegungsrichtlinien**

Sicherheitsfunktion, kodiert

Abhängig von Sicherheitsprofil und Schlüsseltyp und HBCI-Version ist folgender Wert einzustellen:

Sicherheitsprofil	Schlüsseltyp	HBCI V2.x	FinTS V3.0
DDV-1	S	2	2
RAH-7	S	-	2
RAH-7	D	-	1
RAH-9	S	-	2
RAH-10	S	-	2
RDH-1	S	1	1
RDH-2	S	-	2
RDH-3	S	-	2
RDH-3	D	-	1
RDH-5	S	-	2
RDH-6	S	-	2
RDH-6	D	-	1
RDH-7	S	-	2
RDH-7	D	-	1
RDH-8	S	-	2
RDH-9	S	-	2
RDH-10	S	-	2

RDH-1 bleibt 1 aus Kompatibilitätsgründen zu HBC V2.x.

Weitere Erläuterungen sind im Data Dictionary zu finden.

Bereich der Sicherheitsapplikation, kodiert

Der einzig zugelassene Wert ist "1", d.h. SHM (nur Signaturkopf und HBCI-Nutzdaten).

Rolle des Sicherheitslieferanten, kodiert

Der Inhalt dieses Feldes sollte derzeit nicht ausgewertet werden. Optional können aber die nachfolgenden Festlegungen angewendet werden, sofern dies zwischen Kunde und Kreditinstitut zuvor vereinbart wurde:

1. **Dialoginitialisierung und -ende:**

   Die Rolle wird durch den Dialogführenden bestimmt. Es ist nur eine Signatur erlaubt. Erlaubt ist nur der Wert ISS/wert1[15].

2. **Auftragsnachricht:**

   Grundsätzlich gilt: Sobald die Rolle „WIT“ verwendet wird, muss dieser Benutzer mit der Benutzerkennung aus der Dialoginitialisierung arbeiten. Auch der Benutzer „WIT“ muss bankseitig entsprechend der Auftragsart am Konto des Benutzers „ISS“ berechtigt sein.

   Die Reihenfolge der Signaturen ist beliebig.

Anzahl Signaturen	Erlaubte Kombinationen		
	1. Signatur	2. Signatur	3. Signatur
1	ISS/wert1	-	-
2	ISS/wert1	CON/beliebig	-
	WIT/wert1	ISS/beliebig	-
3	WIT/wert1	ISS/beliebig	CON/beliebig

Auch bei Belegung dieses Feldes kann das Kundenprodukt nicht davon ausgehen, dass das Feld kreditinstitutsseitig ausgewertet wird.

Sicherheitsidentifikation, Details

Wenn eine Synchronisierung der Kundensystem-ID durchgeführt wird, ist als Identifizierung der Partei ‚0’ einzustellen.

Sicherheitsdatum und -uhrzeit

Als Bezeichner wird „1“ eingestellt, da es sich um einen Sicherheitszeitstempel handelt.

Zertifikat

Im Falle der Bankensignaturkarte ist je nach Signaturanforderung der Geschäftsvorfälle entweder das Zertifikat C_X509.CH.DS oder das Zertifikat C_X509.CH.AUT$_{C/S}$[&KE] anzugeben.

[15] Die Notation gibt die Rolle gefolgt von der Benutzerkennung an.

B.5.2 Signaturabschluss

- **Beschreibung**

  Der Signaturabschluss stellt die Verbindung mit dem dazugehörigen Signaturkopf her und enthält als "Validierungsresultat" die elektronische Signatur.

- **Format**

  Name: Signaturabschluss
  Typ: Segment
  Segmentart: Administration
  Kennung: HNSHA
  Bezugssegment: -
  Version: 2
  Sender: Kunde/Kreditinstitut

Nr.	Name	Typ	Format	Länge	Status	Anzahl	Restriktionen
1	Segmentkopf	DEG			M	1	
2	Sicherheitskontrollreferenz	DE	an	..14	M	1	<>0
3	Validierungsresultat	DE	bin	..512	C	1	M: bei HBCI N: bei PINTAN
4	Benutzerdefinierte Signatur	DEG			C	1	N: bei HBCI M/N/O bei anderen Verfahren

B.5.3 Verschlüsselungskopf

♦ Beschreibung

Der Verschlüsselungskopf enthält Informationen über die Art des Sicherheitsservice, die Verschlüsselungsfunktion und die zu verwendenden Chiffrierschlüssel.

Zum Abgleich mit den in den BPD definierten Verschlüsselungsverfahren DDV bzw. RAH und RDH wird das Feld „Bezeichner für Algorithmusparameter, Schlüssel“ in der DEG „Verschlüsselungsalgorithmus“ herangezogen.

♦ Format

Name: Verschlüsselungskopf
Typ: Segment
Segmentart: Administration
Kennung: HNVSK
Bezugssegment: -
Version: 3
Sender: Kunde/Kreditinstitut

Nr.	Name	Typ	Format	Länge	Status	Anzahl	Restriktionen
1	Segmentkopf	DEG			M	1	
2	Sicherheitsprofil	DEG			M	1	
3	Sicherheitsfunktion, kodiert	DE	code	..3	M	1	4
4	Rolle des Sicherheitslieferanten, kodiert	DE	code	..3	M	1	1, 4
5	Sicherheitsidentifikation, Details	DEG			M	1	
6	Sicherheitsdatum und -uhrzeit	DEG			M	1	
7	Verschlüsselungsalgorithmus	DEG			M	1	
8	Schlüsselname	DEG			M	1	
9	Komprimierungsfunktion	DE	code	..3	M	1	
10	Zertifikat	DEG			C	1	O: kreditinstitutsseitig bei RAH-7, RAH-9, sowie RDH-1, RDH-2, RDH-3, RDH-5 RDH-6, RDH-7, RDH-8 und RDH-9 (vgl. B.3.1.3.2) N: sonst

♦ Belegungsrichtlinien

Sicherheitsdatum und -uhrzeit

Als Bezeichner (DE Datum- und Zeitbezeichner, kodiert) wird „1“ (Sicherheitszeitstempel) eingestellt.

Zertifikat

Im Falle der Bankensignaturkarte ist das Zertifikat EF_C_X509.CH.KE anzugeben.

B.5.4 Verschlüsselte Daten

- **Beschreibung**

  Dieses Segment enthält die verschlüsselten (und komprimierten) Daten.

- **Format**

  Name: Verschlüsselte Daten
  Typ: Segment
  Segmentart: Administration
  Kennung: HNVSD
  Bezugssegment: -
  Version: 1
  Sender: Kunde/Kreditinstitut

Nr.	Name	Typ	Format	Länge	Status	Anzahl	Restriktionen
1	Segmentkopf	DEG			M	1	
2	Daten, verschlüsselt	DE	bin	..	M	1	

B.6 Key-Management

B.6.1 Formate für Key-Management

Für die Schlüsseländerung, die Schlüsselverteilung sowie die Schlüsselsperrung sind die nachfolgenden Segmente vorgesehen. Diese dürfen nur im Rahmen der speziellen Key-Management-Nachrichten verwendet werden.

B.6.1.1 Änderung eines öffentlichen Schlüssels

- **Beschreibung**

  Dieses Segment enthält einen neuen öffentlichen Schlüssel des Kunden.

- **Format**

  Name: Schlüsseländerung
  Typ: Segment
  Segmentart: Administration
  Kennung: HKSAK
  Bezugssegment: -
  Version: 3
  Sender: Kunde

Nr.	Name	Typ	Format	Länge	Status	Anzahl	Restriktionen
1	Segmentkopf	DEG			M	1	
2	Nachrichtenbeziehung, kodiert	DE	code	1	M	1	2
3	Bezeichner für Funktionstyp	DE	code	..3	M	1	112
4	Sicherheitsprofil	DEG			M	1	
5	Schlüsselname	DEG			M	1	
6	Öffentlicher Schlüssel	DEG			M	1	
7	Zertifikat	DEG			O	1	

- **Belegungsrichtlinien**

Nachrichtenbeziehung, kodiert

Im Zusammenhang mit der Schlüsseländerung ist immer folgender Wert vorgesehen: "2" (Key-Management-Nachricht erwartet Antwort)

Bezeichner für Funktionstyp

Im Zusammenhang mit der Schlüsseländerung ist folgender Wert vorgesehen: "112" (Certificate Replacement)

Sicherheitsprofil

Es wird das den Schlüsseln entsprechende Sicherheitsprofil eingestellt.

Schlüsselname

Es ist der Name des neuen öffentlichen Schlüssels des Kunden einzustellen.

Zertifikat

Falls für den neuen öffentlichen Schlüssel ein Zertifikat verfügbar ist, kann es dem Kreditinstitut auf diese Weise eingereicht werden.

B.6.1.2 Anforderung eines öffentlichen Schlüssels

- **Beschreibung**

  Dieses Segment enthält die Anfrage nach einem öffentlichen Schlüssel des Kreditinstituts. Im Feld „Sicherheitsprofil“ gibt der Kunde an, für welches Profil er die Schlüssel anfordert. Das Segment wird entweder innerhalb der Dialoginitialisierung (vgl. [Formals], Kapitel III.3.1) oder im Rahmen der erstmaligen Schlüsselanforderung (vgl. Kapitel B.6.2.1) gesendet.

- **Format**

  Name: Anforderung eines öffentlichen Schlüssels
  Typ: Segment
  Segmentart: Administration
  Kennung: HKISA
  Bezugssegment: -
  Version: 3
  Sender: Kunde

Nr.	Name	Typ	Format	Länge	Status	Anzahl	Restriktionen
1	Segmentkopf	DEG			M	1	
2	Nachrichtenbeziehung, kodiert	DE	code	1	M	1	2
3	Bezeichner für Funktionstyp	DE	code	..3	M	1	124
4	Sicherheitsprofil	DEG			M	1	
5	Schlüsselname	DEG			M	1	
6	Zertifikat	DEG			O	1	

- **Belegungsrichtlinien**

Nachrichtenbeziehung, kodiert

Im Zusammenhang mit der Anfrage nach einem öffentlichen Schlüssel ist immer folgender Wert vorgesehen: "2" (Key-Management-Nachricht erwartet Antwort)

Bezeichner für Funktionstyp

Im Zusammenhang mit der Anfrage für einen öffentlichen Schlüssel ist folgender Wert vorgesehen: "124" (Certificate Status Request)

Schlüsselname

In den Schlüsselnamen ist die Schlüsselnummer und -version des Schlüssels einzustellen, den das Kundenprodukt als aktuellen öffentlichen Schlüssel des Kreditinstituts kennt. Falls dieser noch nicht vorliegt, ist in beide Felder der Wert „999“ einzustellen.

B.6.1.3 Übermittlung eines öffentlichen Schlüssels

♦ **Beschreibung**

Dieses Segment wird zum einen innerhalb der Dialoginitialisierungsantwort (vgl. [Formals], Kapitel III.3.2) an den Kunden übertragen, falls sich der öffentliche Schlüssel des Kreditinstituts geändert hat. Es enthält dann jeweils einen öffentlichen Schlüssel des Kreditinstituts.

Zum anderen wird das Segment im Rahmen der erstmaligen Anforderung der öffentlichen Schlüssel des Kreditinstituts (vgl. Kapitel B.6.2.1) benötigt.

♦ **Format**

Name: Übermittlung eines öffentlichen Schlüssels
Typ: Segment
Segmentart: Administration
Kennung: HIISA
Bezugssegment: HKISA
Version: 3
Sender: Kreditinstitut

Nr.	Name	Typ	Format	Länge	Status	Anzahl	Restriktionen
1	Segmentkopf	DEG			M	1	
2	Nachrichtenbeziehung, kodiert	DE	code	1	M	1	1
3	Austauschkontrollreferenz	DE	id	#	M	1	
4	Nachrichtenreferenz-nummer	DE	num	..4	M	1	>0
5	Bezeichner für Funktions-typ	DE	code	..3	M	1	224
6	Schlüsselname	DEG			M	1	
7	Öffentlicher Schlüssel	DEG			M	1	
8	Zertifikat	DEG			O	1	

♦ **Belegungsrichtlinien**

Nachrichtenbeziehung, kodiert

Es ist folgender Wert vorgesehen: "1" (Key-Management-Nachricht ist Antwort)

Austauschkontrollreferenz

Dialog-ID der Anfragenachricht des Kunden nach einem öffentlichen Schlüssel (vgl. [Formals], Kapitel II.6.2).

Wird das Segment HIISA in einer Schlüsseldatei auf einem Medium abgelegt, so kann dieses Feld mit dem Wert "0" belegt werden.

Nachrichtenreferenznummer

Nachrichtennummer der Anfragenachricht des Kunden nach einem öffentlichen Schlüssel (vgl. [Formals], Kapitel II.6.2).

Wird das Segment HIISA in einer Schlüsseldatei auf einem Medium abgelegt, so kann dieses Feld mit einem beliebigen gültigen Wert belegt werden.

Bezeichner für Funktionstyp

Es ist folgender Wert vorgesehen: "224" (Certificate Status Notice)

Schlüsselname

Der zurückgemeldete Schlüsselname enthält insbesondere die zugehörige Schlüssel- und Versionsnummer, die das Kundenprodukt für die Referenzierung des in der DEG „Öffentlicher Schlüssel“ übertragenen neuen öffentlichen Schlüssels verwendet.

Öffentlicher Schlüssel

Diese Datenelementgruppe enthält den neuen öffentlichen Schlüssel des Kreditinstitutes.

B.6.1.4 Schlüsselsperrung

- **Beschreibung**

  Dieses Segment enthält die Anforderung für das Sperren eines Schlüssels.

- **Format**

  Name: Schlüsselsperrung
  Typ: Segment
  Segmentart: Administration
  Kennung: HKSSP
  Bezugssegment: -
  Version: 3
  Sender: Kunde

Nr.	Name	Typ	Format	Länge	Status	Anzahl	Restriktionen
1	Segmentkopf	DEG			M	1	
2	Nachrichtenbeziehung, kodiert	DE	code	1	M	1	2
3	Bezeichner für Funktionstyp	DE	code	..3	M	1	130
4	Sicherheitsprofil	DEG			M	1	
5	Schlüsselname	DEG			M	1	
6	Sperrenkennzeichen	DE	code	..3	M	1	1, 501, 999
7	Sicherheitsdatum und -uhrzeit	DEG			O	1	
8	Zertifikat	DEG			O	1	

- **Belegungsrichtlinien**

Nachrichtenbeziehung, kodiert

Im Zusammenhang mit der Schlüsselsperrung ist folgender Wert vorgesehen: "2" (Key-Management-Nachricht erwartet Antwort)

Bezeichner für Funktionstyp

Im Zusammenhang mit der Schlüsselsperrung ist folgender Wert vorgesehen: "130" (Certificate Revocation)

Sicherheitsprofil

Es wird das den Schlüsseln entsprechende Sicherheitsprofil eingestellt.

Schlüsselname

Es sind die Identifikationsmerkmale des zu sperrenden Signierschlüssels einzustellen, unabhängig davon, dass grundsätzlich immer sowohl Signier- als auch Chiffrierschlüssel gesperrt werden (s. Kap. B.8.1.4).

Sicherheitsdatum und -uhrzeit

Enthält optional Datum und Uhrzeit, ab welcher der Schlüssel nicht mehr gültig ist. Als Bedeutung wird „6“ (für CRT, Certificate Revocation Time) eingestellt.

Es ist zu beachten, dass eine terminierte Sperre nicht von allen Kreditinstituten unterstützt wird. Das Kundenprodukt sollte den Kunden auf diesen Sachverhalt hinweisen.

B.6.1.5 Bestätigung der Schlüsselsperrung

- **Beschreibung**

  Dieses Segment enthält die Bestätigung für eine Schlüsselsperrung.

- **Format**

  Name: Bestätigung der Schlüsselsperrung
  Typ: Segment
  Segmentart: Administration
  Kennung: HISSP
  Bezugssegment: HKSSP
  Version: 3
  Sender: Kreditinstitut

Nr.	Name	Typ	Format	Länge	Status	Anzahl	Restriktionen
1	Segmentkopf	DEG			M	1	
2	Nachrichtenbeziehung, kodiert	DE	code	1	M	1	1
3	Austauschkontrollreferenz	DE	id	#	M	1	
4	Nachrichtenreferenz-nummer	DE	num	..4	M	1	>0
5	Bezeichner für Funktions-typ	DE	code	..3	M	1	231
6	Schlüsselname	DEG			M	1	
7	Sperrenkennzeichen	DE	code	..3	M	1	1, 501, 999
8	Sicherheitsdatum und -uhrzeit	DEG			M	1	
9	Zertifikat	DEG			O	1	

- **Belegungsrichtlinien**

Nachrichtenbeziehung, kodiert

Im Zusammenhang mit der Bestätigung der Schlüsselsperrung ist folgender Wert vorgesehen: "1" (Key-Management-Nachricht ist Antwort)

Austauschkontrollreferenz

Dialog-ID der Sperranforderung des Kunden (vgl. [Formals], Kapitel II.6.2).

Nachrichtenreferenznummer

Nachrichtennummer der Sperrenanforderung des Kunden (vgl. [Formals], Kapitel II.6.2).

Bezeichner für Funktionstyp

Im Zusammenhang mit der Bestätigung der Schlüsselsperrung ist folgender Wert vorgesehen: "231" (Revocation Confirmation)

Schlüsselname

Es sind die Identifikationsmerkmale des gesperrten Signierschlüssels einzustellen, unabhängig davon, dass grundsätzlich immer sowohl Signier- als auch Chiffrierschlüssel gesperrt werden (s. Kap. B.8.1.4).

Sicherheitsdatum und -uhrzeit

Enthält optional Datum und Uhrzeit, ab welcher der Schlüssel nicht mehr gültig ist. Als Bedeutung wird „6“ (für CRT, Certificate Revocation Time) eingestellt.

B.6.2 Key-Management-Nachrichten

Aufträge des Key-Managements dürfen nur in den folgenden separaten Nachrichten übertragen werden.

Hiervon abweichend wird der Auftrag „Anforderung eines öffentlichen Schlüssels des Kreditinstituts“ nicht als eigene Nachricht, sondern innerhalb der Dialoginitialisierung übertragen.

Die Nachrichten für das Key-Management müssen zum Teil kryptographisch geschützt werden. Alternativ können auch Offline-Sicherungsverfahren (z.B. Brief) zum Einsatz kommen (vgl. Kapitel B.3.1.3).

Es sind folgende Key-Management-Nachrichten vorgesehen:

- Änderung eines öffentlichen Schlüssels des Kunden
- Erstmalige Anforderung der Schlüssel des Kreditinstituts
- Erstmalige Übermittlung der Schlüssel des Kunden
- Schlüsselsperrung durch den Kunden

Mit Ausnahme der Sperrnachricht sind alle Key-Management-Nachrichten nur bei Einsatz des RAH- und RDH-Verfahrens möglich.

B.6.2.1 Änderung eines öffentlichen Schlüssels des Kunden

Realisierung Bank: verpflichtend
Realisierung Kunde: verpflichtend

a) Kundennachricht

- **Beschreibung**

Diese Nachricht ist nur bei Verwendung des RAH- bzw. RDH-Verfahrens möglich. Der Nachricht muss eine Dialoginitialisierung vorausgehen. Der Auftrag muss mit dem alten Signierschlüssel signiert werden.

Es muss unterschieden werden, ob die Schlüsseländerung auch das Sicherheitsprofil wechselt oder nicht.

Die folgenden Wechselmöglichkeiten bestehen, falls Sicherheitsprofilwechsel unterstützt sind:

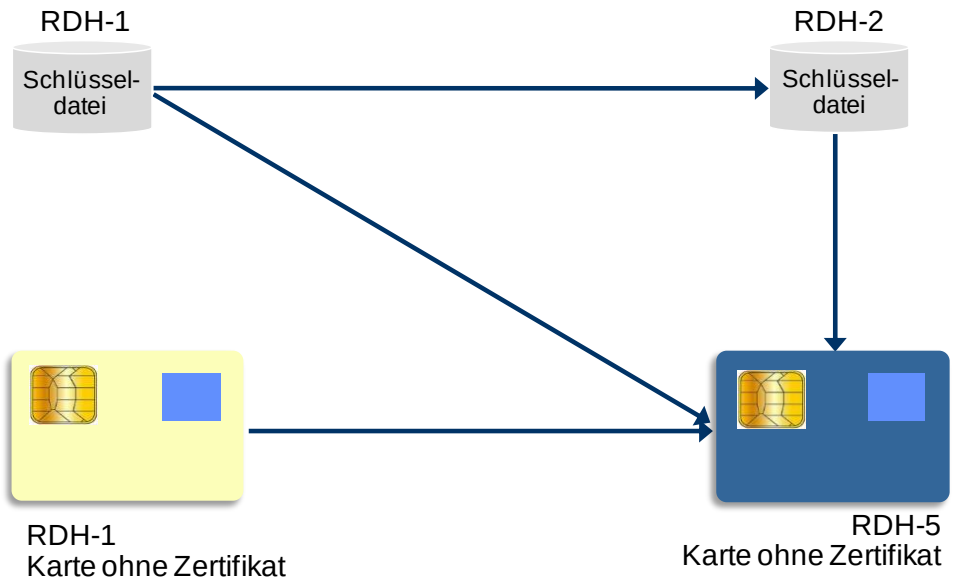


Abbildung 16: Unterstützte Sicherheitsprofilwechsel RDH-1, RDH-2 und RDH-5

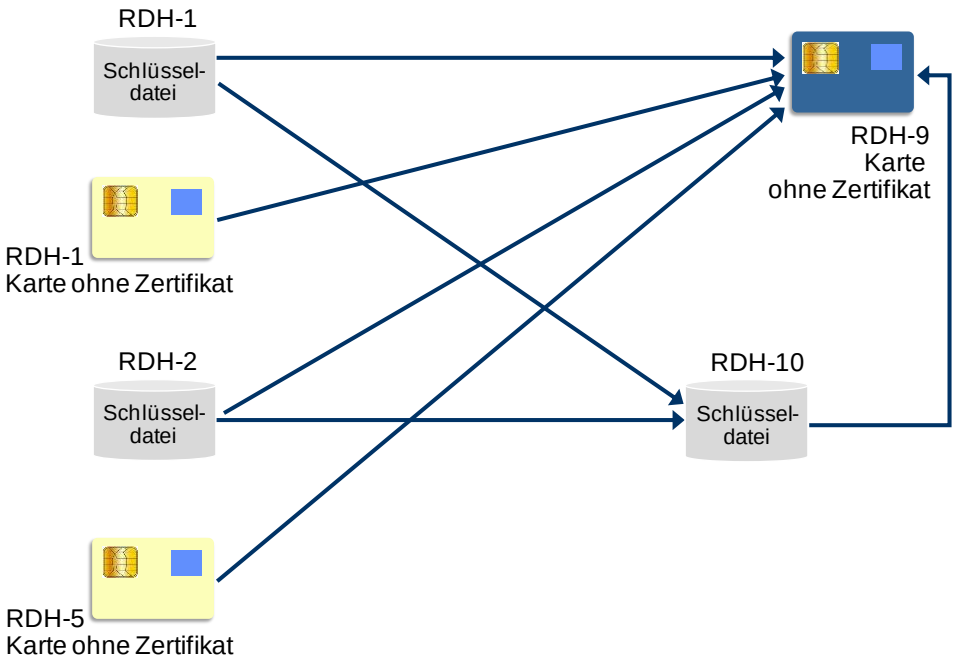


Abbildung 17: Unterstützte Sicherheitsprofilwechsel RDH-1, RDH-2 RDH-5, RDH-9 und RDH-10

Zusammengefasst ergeben sich folgende Wechselmöglichkeiten:

B.7 RDH-x / RAH-y (aktuelles Verfahren)	B.8 RDH-y / RAH-y (neues Verfahren)
RDH-9 Bankensignaturkarte ohne Zertifikat	RAH-9 Bankensignaturkarte ohne Zertifikat
RDH-10 Schlüsseldatei	RAH-10 Schlüsseldatei
RDH-10 Schlüsseldatei	RAH-9 Bankensignaturkarte ohne Zertifikat
RAH-10 Schlüsseldatei	RAH-9 Bankensignaturkarte ohne Zertifikat
RDH-1 Schlüsseldatei	RDH-2 Schlüsseldatei
RDH-1 Schlüsseldatei	RDH-5 Bankensignaturkarte ohne Zertifikat
RDH-1 Schlüsseldatei	RDH-9 Bankensignaturkarte ohne Zertifikat
RDH-1 Schlüsseldatei	RDH-10 Schlüsseldatei
RDH-1 Bankensignaturkarte ohne Zertifikat	RDH-5 Bankensignaturkarte ohne Zertifikat
RDH-1 Bankensignaturkarte ohne Zertifikat	RDH-9 Bankensignaturkarte ohne Zertifikat
RDH-2 Schlüsseldatei	RDH-5 Bankensignaturkarte ohne Zertifikat
RDH-2 Schlüsseldatei	RDH-9 Bankensignaturkarte ohne Zertifikat
RDH-2 Schlüsseldatei	RDH-10 Schlüsseldatei
RDH-5 Bankensignaturkarte ohne Zertifikat	RDH-9 Bankensignaturkarte ohne Zertifikat
RDH-10 Schlüsseldatei	RDH-9 Bankensignaturkarte ohne Zertifikat

1. ohne Wechsel des Sicherheitsprofils:

   Nach der erfolgreichen Durchführung der Schlüsseländerung wird der vorher aktuelle Schlüssel automatisch gesperrt. Es ist darauf zu achten, dass die Version des neuen Schlüssels höher ist als die des alten Schlüssels.

2. mit Wechsel des Sicherheitsprofils
   (vgl. Abbildung 16 und Abbildung 17):

   Bei einem Sicherheitsprofilwechsel muss der Kunde immer beide HKSAK-Segmente einstellen. Nach der erfolgreichen Durchführung der Schlüsseländerung wird durch das Kreditinstitut mitgeteilt, ob der vorher aktuelle RAH-x bzw. RDH-x-Schlüssel automatisch gesperrt wurde. Diese Nachricht wird mit den RAH-x bzw. RDH-x-Schlüsseln abgesichert. Wurden die RAH-x bzw. RDH-x-Schlüssel institutsseitig nicht gesperrt, wird der Dialog unter Absicherung der RAH-x bzw. RDH-x-Schlüssel beendet. Es ist darauf zu achten, dass die Nummer der RDH-2-Schlüssel 2 ist, die Version kann mit 1 beginnen. Ab RDH-5 und bei RAH-x sind Schlüsselnummer und –version vorgegeben.

Falls das Kreditinstitut nicht in der Lage ist, zwei Schlüsselpaare zu einem Kunden gleichzeitig zu halten und somit die Endenachricht mit den RAH-x bzw. RDH-x-Schlüsseln nicht mehr bedienen kann, ist dies dem Kundenprodukt durch den Rückmeldungscode 3250 mitzuteilen. Das Kundenprodukt soll dann keine Endenachricht mehr senden und den Bankdatensatz von der RAH-x bzw. RDH-x-Schlüsseldatei löschen.

Es empfiehlt sich, die RAH-x bzw. RDH-x-Schlüssel nach einem erfolgreichen Abschluss des Dialoges durch einen Sperrdialog ungültig zu machen.

Falls der Kunde eine Schlüsseländerungsnachricht sendet, diese aber aus kreditinstitutsinternen Verarbeitungsgründen nicht beantwortet wird, sollte das Kundenprodukt zunächst einen neuen Dialog auf Basis eines der Schlüsselpaare aufbauen. Falls diese Nachricht abgelehnt wird ist ein erneuter Versuch auf Basis eines anderen Schlüsselpaares vorzunehmen. Aus der Reaktion des Kreditinstituts ist für das Kundenprodukt ersichtlich, ob die Schlüsseländerung erfolgreich war oder wiederholt werden muss.
Da es nicht möglich ist, einen DS-Schlüssel, der ja eine natürliche Person identifiziert, über die HBCI-Schlüsseländerung zu ändern, dürften nur "1..2" HKSAK-Segmente eingestellt werden.

B.8.1.1.1 Wechsel des Sicherheitsprofils ohne Schlüsselwechsel

Diese Situation tritt bei der Migration von RDH- nach gleichrangigen RAH-Verfahren auf. Beim Übergang von gleichartigen Sicherheitsprofilen (z. B. RDH-9 auf RAH-9 oder RDH-10 auf RAH-10) muss zwar eine erneute Übermittlung der bestehenden öffentlichen Schlüssel durch entsprechende HKSAK-Segmente erfolgen, diese dienen jedoch nur dazu, die Änderung des Sicherheitsprofils bzgl. des Verschlüsselungsalgorithmus (RDH: 2-Key-Triple-DES nach RAH: AES-256) mitzuteilen. Die Schlüsselpaare selbst bleiben unverändert, d. h. weder im Kundenprodukt noch im Kreditinstitut werden Änderungen an den bestehenden Schlüsseln vorgenommen.

Beim Übergang von RDH- auf RAH-Verfahren ergeben sich folgende Möglichkeiten des Schlüsselwechsels (RDH-10 auf RAH-9) bzw. des Wechsel des Verschlüsselungsverfahrens von RDH auf RAH:

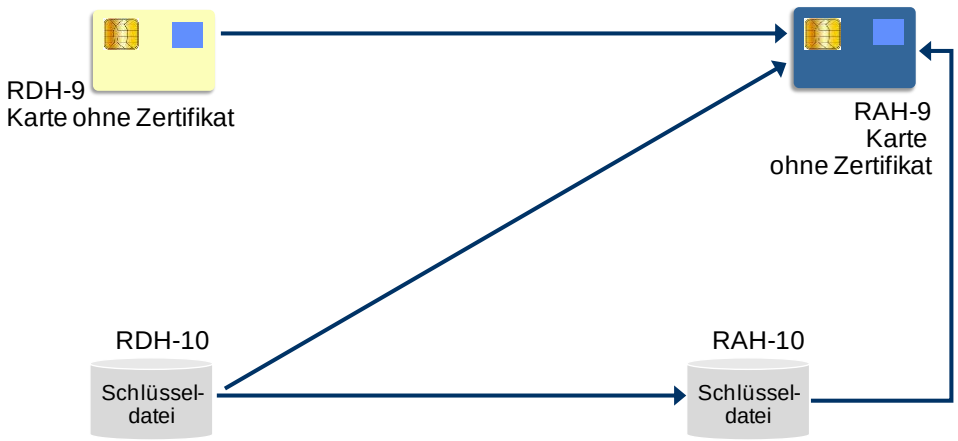


Abbildung 18: Unterstützte Sicherheitsprofilwechsel beim Übergang von RDH- auf RAH-Verfahren

Zum Verfahren s. Kap. B.3.1.3.4.

♦ **Format**

Name: Änderung eines öffentlichen Schlüssels des Kunden
Typ: Nachricht
Version: 4
Sender: Kunde

Nr.	Name	Typ	Kennung	Status	Anzahl	Anmerkungen
1	Nachrichtenkopf	SEG	HNHBK	M	1	s. [Formals], Kap. II.5.1
2	Signaturkopf	SEG	HNSHK	M	1	
3	Schlüsseländerung	SEG	HKSAK	M	1..3	
4	Signaturabschluss	SEG	HNSHA	M	1	
5	Nachrichtenabschluss	SEG	HNHBS	M	1	s. [Formals], Kap. II.5.2

♦ **Belegungsrichtlinien**

Der Kunde stellt entweder seinen neuen öffentlichen Signierschlüssel, seinen neuen öffentlichen Chiffrierschlüssel oder beide Schlüssel ein.

a) Kreditinstitutsnachricht

♦ **Format**

Name: Kreditinstitutsnachricht allgemein
Typ: Nachricht
Format: s. [Formals], Kap. II.8.1

♦ **Erläuterungen**

Es werden keine Datensegmente zurückgemeldet.

♦ **Ausgewählte Beispiele für Rückmeldungscodes**

Code	Beispiel
0020	Öffentlicher Schlüssel wurde geändert
3250	RDH-1-Schlüssel wurden gesperrt. Endenachricht nicht mehr möglich.
3260	RDH-1-Schlüssel weiterhin gültig. Schlüsselsperre wird empfohlen.
9210	Schlüsseländerung von RDH-1 auf RDH-2 zur Zeit nicht möglich
9010	Schlüsseländerung zur Zeit nicht möglich
9010	Sicherheitsverfahren unterstützt keine öffentlichen Schlüssel
9210	Eingereichter Schlüssel ist mit dem aktuellen Schlüssel identisch

B.8.1.2 Erstmalige Anforderung der Schlüssel des Kreditinstituts

Mit Hilfe dieser Nachricht fordert der Kunde erstmalig den öffentlichen Signier- und Chiffrierschlüssel des Kreditinstituts an. Gleichzeitig erhält er die aktuellen Bankparameterdaten, die er benötigt, um die unterstützten Verschlüsselungsverfahren des Kreditinstituts in Erfahrung zu bringen. Mit Hilfe dieser Informationen wird der Kunde in die Lage versetzt, beliebige Nachrichten zu verschlüsseln.

Realisierung Bank: optional
Realisierung Kunde: verpflichtend

a) Kundennachricht

- **Beschreibung**

Diese Nachricht wird an Stelle einer Dialoginitialisierung gesendet. Es dürfen keine Auftragsnachrichten folgen. Der Dialog ist vom Kunden nach Erhalt der Antwortnachricht mit einer Dialogendenachricht zu beendigen. Die Nachricht wird weder signiert noch verschlüsselt.

- **Format**

Name: Erstmalige Anforderung der Schlüssel des Kreditinstituts
Typ: Nachricht
Version: 4
Sender: Kunde

Nr.	Name	Typ	Kennung	Status	Anzahl	Anmerkungen
1	Nachrichtenkopf	SEG	HNHBK	M	1	s. [Formals], Kap. II.5.1
2	Identifikation	SEG	HKIDN	M	1	s. [Formals], Kap. III.3.1.2
3	Verarbeitungsvorbereitung	SEG	HKVVB	M	1	s. [Formals], Kap. III.3.1.3
4	Anforderung eines öffentlichen Schlüssels	SEG	HKISA	M	3	
5	Nachrichtenabschluss	SEG	HNHBS	M	1	s. [Formals],Kap. II.5.2

- **Belegungsrichtlinien**

Identifikation

Die Datenelemente des Segments sind wie beim anonymen Zugang zu belegen (s. [Formals], Kap. III.5).

Verarbeitungsvorbereitung

Mit diesem Segment fordert der Kunde die Bankparameterdaten an.

Anforderung eines öffentlichen Schlüssels

Mit diesen Segmenten fordert der Kunde jeweils den öffentlichen Signierschlüssel und den öffentlichen Chiffrierschlüssel des Kreditinstituts an. Es sind stets alle Schlüssel eines Sicherheitsprofils anzufordern, auch wenn das Kreditinstitut nicht signiert.

In die DEG „Schlüsselname“ ist für die Benutzerkennung der Standardwert ‘999’ einzustellen. In der Rückmeldung wird dem Kunden die korrekte Benutzerkennung des Kreditinstituts mitgeteilt.

Da bei der Erstinitialisierung noch keine BPD vorliegt, ist es für das Kundenprodukt evtl. problematisch, zu ermitteln welche Sicherheitsprofile das Kreditinstitut anbietet und - wenn mehrere möglich sind - welches Profil für den Kunden gilt. Falls dem Kunden diese Information nicht von seinem Kreditinstitut mitgeteilt wurde, sollte das Kundenprodukt versuchen, das Sicherheitsmedium zu lesen und daraus das richtige Sicherheitsprofil zu erschließen.
Da ein Kreditinstitut über keinen D-Schlüssel verfügt bzw. verfügen kann (Voraussetzung ist eine "natürliche Person"), dürfen nur zwei HKISA-Segmente eingestellt werden.

b) Kreditinstitutsnachricht

- **Format**

Name: Erstmalige Übermittlung der Schlüssel des Kreditinstituts
Typ: Nachricht
Version: 4
Sender: Kreditinstitut

Nr.	Name	Typ	Kennung	Status	Anzahl	Anmerkungen
1	Nachrichtenkopf	SEG	HNHBK	M	1	s. [Formals], Kap. II.5.1
2	Signaturkopf	SEG	HNSHK	O	1	
3	Rückmeldungen zur Gesamtnachricht	SEG	HIRMG	M	1	s. [Formals], Kap. II.7.2
4	Rückmeldungen zu Segmenten	SEG	HIRMS	O	n	s. [Formals], Kap. II.7.3
5	Bankparameterdaten	SF	#	O	1	s. [Formals], Kap. III.3.2.2
6	Übermittlung eines öffentlichen Schlüssels	SEG	HIISA	M	1..3	
7	Kreditinstitutsmeldung	SEG	HIKIM	O	n	s. [Formals], Kap. III.3.2.5
8	Signaturabschluss	SEG	HNSHA	O	1	
9	Nachrichtenabschluss	SEG	HNHBS	M	1	s. [Formals], Kap. II.5.2

- **Belegungsrichtlinien**

Signaturkopf

Falls das Kreditinstitut einen Signierschlüssel besitzt, d.h. seine Nachrichten grundsätzlich signiert, hat es auch diese Nachricht zu signieren, um die Authentizität des Chiffrierschlüssels zu sichern (s.u.).

Übermittlung eines öffentlichen Schlüssels

In diesen Segmenten werden dem Kunden die öffentlichen Schlüssel des Kreditinstituts mitgeteilt.

Falls das Kreditinstitut seine Nachrichten nicht signiert, erhält der Kunde nur den öffentlichen Chiffrierschlüssel zurückgemeldet. Auf die Anforderung des Signierschlüssels erhält er einen entsprechenden Rückmeldungscode der Kategorie „Warnungen und Hinweise“, der ihm anzeigt, dass das Kreditinstitut seine Nachrichten nicht signiert.

Da die Authentizität des Chiffrierschlüssels nicht gesichert ist, muss diese Nachricht durch einen Ini-Brief an den Kunden mit dem Hashwert des Chiffrierschlüssels begleitet werden (s. Kap. B.3.1.3.2).

Falls das Kreditinstitut seine Nachrichten signiert, erhält der Kunde sowohl den öffentlichen Chiffrier- als auch Signierschlüssel zurückgemeldet. Die Authentizität des Chiffrierschlüssels ist dabei durch die Signatur gesichert. Die Authentizität des Signierschlüssels ist jedoch nicht gesichert, da das Kundensystem die Echtheit der Signatur nicht prüfen kann. Daher muss in diesem Fall die Nachricht durch einen Ini-Brief mit dem Hashwert des Signierschlüssels begleitet werden.

- **Ausgewählte Beispiele für Rückmeldungscodes**

Code	Beispiel
0020	Auftrag ausgeführt
3310	Kein Schlüssel verfügbar, da Kreditinstitutsnachrichten nicht signiert werden

B.8.1.3 Erstmalige Übermittlung der Schlüssel des Kunden

Mit Hilfe dieser Nachricht übermittelt der Kunde erstmalig seinen öffentlichen Signier- und Chiffrierschlüssel an das Kreditinstitut („Erstinitialisierungsnachricht“).

Da der Absender des öffentlichen Schlüssels den Beweis erbringen muss, dass er auch im Besitz des zugehörigen privaten Schlüssels ist, muss die Nachricht des Kunden signiert sein.

Das Kreditinstitut darf eine Nachricht nicht ablehnen, nur weil für den Kunden noch kein öffentlicher Schlüssel in der Schlüsselverwaltung existiert. Falls die normale Signaturprüfung aus diesem Grund negativ verläuft, muss zunächst geprüft werden, ob es sich um eine Erstinitialisierung handelt. In diesem Fall ist der öffentliche Schlüssel aus der Erstinitialisierungsnachricht zu extrahieren und die Signaturprüfung auf der Basis dieses Schlüssels erneut vorzunehmen.

Die Erstinitialisierungsnachricht des Kunden ist zu verschlüsseln, da die darin enthaltenen benutzerbezogenen Daten (Kunden-ID, Benutzerkennung) als vertraulich einzustufen sind. Dies erfordert, dass sich der öffentliche Chiffrierschlüssel des Kreditinstituts schon vor dem Senden der Erstinitialisierung im Besitz des Kunden befinden muss. Ferner muss dem Kunden das Verschlüsselungsverfahren bekannt sein, das ihm in den Bankparameterdaten mitgeteilt wird. Um dem Kunden diese Daten vorab zukommen zu lassen bieten sich folgende Lösungen an:

- Das Kreditinstitut sendet dem Kunden eine Schlüsseldatei zu, die die Schlüssel und die aktuelle BPD enthält, wie in VI.3.1.3.2 beschrieben.
- Der Kunde sendet die Key-Management-Nachricht „Erstmalige Anforderung der Schlüssel des Kreditinstituts“ (s. Kap. B.6.2.1). Diese Nachricht wird begleitet von einem Ini-Brief.

Um die wiederholte Ausführung unberechtigter Initialisierungsversuche zu verhindern, sind kreditinstitutsseitig folgende Vorkehrungen zu treffen:

- Die Benutzerkennung sollte bei Verwendung des RAH- bzw. RDH-Verfahrens nicht durch benutzerindividuelle Merkmale (z.B. Kontonummer) hergeleitet werden können.
- Eine erneute Erstinitialisierung ist nur zulässig, wenn zuvor eine Sperrung der Schlüssel des Benutzers erfolgt ist. In allen anderen Fällen ist eine erneute Erstinitialisierungsnachricht abzulehnen.

Auf der Chipkarte können Kommunikationszugänge abgelegt werden (s. Kap. C). Da pro Institut jedoch mehrere Kommunikationszugänge gespeichert sein können (z.B. TCP/IP und HTTPS), muss ein Kundenprodukt zunächst prüfen, ob für dieses Institut bereits die Schlüssel eingereicht wurden, bevor eine erstmalige Übermittlung der Schlüssel des Kunden durchgeführt wird. Für den Fall, dass das Kundenprodukt die Schlüssel dennoch sendet, sollte das Institut die Warnung 3330 „Schlüssel liegen bereits vor“ zurückmelden.

Realisierung Bank: verpflichtend
Realisierung Kunde: verpflichtend

a) Kundennachricht

♦ **Beschreibung**

Diese Nachricht wird an Stelle einer Dialoginitialisierung gesendet. Es dürfen keine Auftragsnachrichten folgen. Die Nachricht muss signiert und verschlüsselt werden. Der Dialog ist vom Kunden nach Erhalt der Antwortnachricht mit einer Dialogendenachricht zu beendigen. Die Dialogendenachricht ist nicht zu signieren, da der übermittelte Kundenschlüssel zu diesem Zeitpunkt i.d.R. noch nicht freigeschaltet ist.

♦ **Format**

Name: Erstmalige Übermittlung der Schlüssel des Kunden
Typ: Nachricht
Version: 4
Sender: Kunde

Nr.	Name	Typ	Kennung	Status	Anzahl	Anmerkungen
1	Nachrichtenkopf	SEG	HNHBK	M	1	s. [Formals], Kap. II.5.1
2	Signaturkopf	SEG	HNSHK	M	1	
3	Identifikation	SEG	HKIDN	M	1	s. [Formals], Kap. III.3.1.2
4	Schlüsseländerung	SEG	HKSAK	M	2-3	
5	Signaturabschluss	SEG	HNSHA	M	1	
6	Nachrichtenabschluss	SEG	HNHBS	M	1	s. [Formals], Kap. II.5.2

♦ **Belegungsrichtlinien**

Identifikation

Der Benutzer hat die ihm zur Initialisierung mitgeteilten Daten einzustellen. Wenn die Erstinitialisierung mit der alten Benutzerkennung durchgeführt wird, ist – sofern noch vorhanden – die alte Kundensystem-ID anzugeben, andernfalls ist als Kundensystem-ID der Wert ‚0' anzugeben. Falls zu diesem Zeitpunkt noch keine Synchronisierung durchgeführt wurde, ist als Kundensystem-ID der Wert '0' einzustellen.

Schlüsseländerung

Der Kunde stellt seine öffentlichen Schlüssel ein. Dies können Signier-, Chiffrier- oder Authentikationsschlüssel sein.

Die Authentizität des Chiffrierschlüssels ist dabei durch die Signatur gesichert. Die Authentizität des Signierschlüssels ist jedoch nicht gesichert, da das Kreditinstitut die Echtheit der Signatur nicht prüfen kann. Daher muss die Nachricht durch einen Ini-Brief an das Kreditinstitut mit dem Hashwert des Signierschlüssels begleitet werden (s. Kap. B.3.1.3.2).

b) Kreditinstitutsnachricht

- **Beschreibung**

Die Ablehnung der Erstinitialisierungsnachricht darf aus sicherheitstechnischen Aspekten im Rahmen der Rückmeldungscodes nicht inhaltlich begründet werden. Fehlermeldungen, die sich auf den syntaktischen Aufbau der Nachricht bzw. der Segmente beziehen, sind hiervon unberührt.

- **Format**

Name:	Kreditinstitutsnachricht allgemein
Typ:	Nachricht
Format:	s. [Formals], Kap. II.8.1

- **Erläuterungen**

Es werden keine Datensegmente zurückgemeldet.

- **Ausgewählte Beispiele für Rückmeldungscodes**

Code	Beispiel
0010	Öffentlicher Schlüssel wurde entgegengenommen
0020	Öffentlicher Schlüssel wurde freigeschaltet
0020	Kunde wurde freigeschaltet
9010	Auftrag abgelehnt

B.8.1.4 Schlüsselsperrung durch den Kunden

Diese Nachricht beschreibt die Anforderung zum Sperren der Schlüssel durch den Kunden und die Bestätigung der Schlüsselsperrung durch das Kreditinstitut (vgl. Kapitel B.3.2).

Realisierung Bank: verpflichtend
Realisierung Kunde: verpflichtend

a) Kundennachricht

- **Beschreibung**

Es werden immer alle Schlüssel gesperrt. Eine selektive Schlüsselsperrung (z.B. nur Chiffrierschlüssel) ist gegenwärtig nicht zulässig.

Der Nachricht muss eine Dialoginitialisierung vorausgehen. Die Nachricht muss bei Kompromittierung signiert sein. Es liegt in der Entscheidung des Kreditinstituts, ob es auch nicht signierte (anonyme) Schlüsselsperrungen erlaubt (z.B. bei Verlust des Sicherheitsmediums). Die Steuerung erfolgt in den Userparameterdaten über das Feld „Anzahl benötigter Signaturen“. Die Nachricht darf maximal eine Signatur tragen.

Bei Verlust des Sicherheitsmediums liegen dem Benutzer u.U. die zur Durchführung der Sperrung erforderlichen Daten (Schlüsselnummer und -version) nicht vor. In diesem Fall ist zur Referenzierung auf den aktuellen Schlüssel jeweils der Wert ‘999’ einzustellen. Es ist daher darauf zu achten, dass dieser Wert reserviert ist und nicht im Rahmen der Versionszählung belegt wird.

Falls das Kreditinstitut unsignierte Sperrungen zulässt, muss dem Benutzer darüber hinaus explizit seine Benutzerkennung mitgeteilt werden. Beim RAH- bzw. RDH-Verfahren erfolgt dies im Rahmen des Ini-Briefs. Beim DDV-Verfahren kann diese dem Benutzer bei der Aushändigung der Chipkarte mitgeteilt werden.

Beim DDV-Verfahren wird der Dialog im Anschluss an die Sperrnachricht ungesichert beendet, d.h. die Kreditinstitutsantwortnachricht sowie die Dialogbeendigungsnachrichten werden weder signiert noch verschlüsselt.

Beim RAH- sowie RDH-Verfahren wird im Anschluss an die Sperrnachricht

- die Antwortnachricht sowie die Dialogendenachricht des Kreditinstituts nicht chiffriert, aber signiert (sofern das Kreditinstitut grundsätzlich signiert) und
- die Dialogendenachricht des Kunden chiffriert, aber nicht signiert

Diese Verfahren gelten nur bei einer erfolgreichen Sperrung. Bei einer fehlgeschlagenen Sperrung ist der Dialog gesichert zu Ende zu führen, da die Schlüssel des Kunden weiterhin aktiv sind.

Beim RAH- und RDH-Verfahren muss der Kunde nach einer Schlüsselsperrung zur Entsperrung eine erneute Erstinitialisierung durchführen.

♦ **Format**

Name: Sperrung eines Schlüssels durch den Kunden
Typ: Nachricht
Version: 4
Sender: Kunde

Nr.	Name	Typ	Kennung	Status	Anzahl	Anmerkungen
1	Nachrichtenkopf	SEG	HNHBK	M	1	s. [Formals], Kap. II.5.1
2	Signaturkopf	SEG	HNSHK	O	1	
3	Schlüsselsperrung	SEG	HKSSP	M	1	
4	Signaturabschluss	SEG	HNSHA	O	1	
5	Nachrichtenabschluss	SEG	HNHBS	M	1	s. [Formals], Kap. II.5.2

♦ **Belegungsrichtlinien**

Schlüsselsperrung

Dieses Segment enthält die Anforderung für die Schlüsselsperrung.

Eine selektive Schlüsselsperrung ist gegenwärtig nicht zulässig, d.h. es werden immer alle Kundenschlüssel gleichzeitig gesperrt. In der DEG „Schlüsselname“ sind die Merkmale des Signierschlüssels einzustellen (s. Kap. B.6.1.4).

b) Kreditinstitutsnachricht

♦ **Format**

Name: Bestätigung der Schlüsselsperrung durch das Kreditinstitut
Typ: Nachricht
Version: 4
Sender: Kreditinstitut

Nr.	Name	Typ	Kennung	Status	Anzahl	Anmerkungen
1	Nachrichtenkopf	SEG	HNHBK	M	1	s. [Formals], Kap. II.5.1
2	Signaturkopf	SEG	HNSHK	O	1	
3	Rückmeldungen zur Gesamtnachricht	SEG	HIRMG	M	1	s. [Formals], Kap. II.7.2
4	Rückmeldungen zu Segmenten	SEG	HIRMS	O	n	s. [Formals], Kap. II.7.3
5	Bestätigung der Schlüsselsperrung	SEG	HISSP	M	1	
6	Signaturabschluss	SEG	HNSHA	O	1	
7	Nachrichtenabschluss	SEG	HNHBS	M	1	s. [Formals], Kap. II.5.2

♦ **Ausgewählte Beispiele für Rückmeldungscodes**

Code	Beispiel
0020	Schlüssel wurde erfolgreich gesperrt
9010	Schlüssel ist bereits gesperrt
9010	Terminierte Sperren werden nicht unterstützt
9210	Unbekanntes Sperrenkennzeichen
9210	Sperrdatum liegt zu weit in der Zukunft

C. CHIPAPPLIKATIONEN

C.1 Chipapplikation für RAH / RDH

Kapitel C.1.1 dient als Überblick für die Datenstrukturen und Zugriffsregeln der Chipapplikation ”DF_NOTEPAD” für SECCOS-Chipkarten [SECCOS] bzw. [SECCOS-6]. Die Spezifikation des DF_NOTEPAD selbst und die Terminalabläufe sind im Dokument [DF_NOTEPAD] enthalten.

Im Verlauf dieses Kapitels ist mit ”Bankensignaturkarte” eine Chipkarte mit SECCOS-Betriebssystem und Signaturanwendung gemeint, die u.U. auch die Notepad-Applikation aus Kap. C.1.1 enthält. Weitere Applikationen, wie z.B. die elektronische Geldbörse, sind nicht notwendigerweise auf der Chipkarte enthalten. Ebenso kann die Bankensignaturkarte mit oder ohne Zertifikat ausgeliefert werden.

C.1.1 Applikation Notepad

Die Anwendung „Notepad“ dient als „Notizbuch“ zur Aufnahme von Daten anderer Anwendungen. Durch das Notizbuch wird somit ein mobiler Datenspeicher geschaffen, in dem bestimmte anwendungs- bzw. kundenspezifische Parameter abgelegt werden können, z.B. für die Bankverbindungsdaten in HBCI.

Wenn eine Anwendung auf die Karte zugreift, wird geprüft, ob auf der Chipkarte das Notizbuch DF_NOTEPAD vorhanden ist. Falls ja werden die Daten ausgelesen, falls nein, muss der Benutzer die Zugangsdaten selbst eingeben bzw. die Zugangsdaten werden im Kundenprodukt selber verwaltet.

Im Datenspeicher EF_NOTEPAD kann jeder Record durch eine Anwendung belegt werden. Die Unterscheidung der Zugehörigkeit bestimmter Dateninhalte erfolgt an Hand der Tags eines Records:

- ’00‘ bedeutet, dass der Record nicht belegt ist
- ’F0‘ bedeutet, dass der Record HBCI-Bankverbindungsdaten (HBCI-Parameterblock) enthält.
- ’F1‘ bedeutet, dass der Record Bankverbindungsdaten analog dem DFÜ-Abkommen enthält.

Weitere Kennungen sind für den späteren Gebrauch durch andere Anwendungen vorgesehen (Tag ‘F2‘ bis ‘FE‘).

Somit können mehrere HBCI-Bankverbindungsdaten (im Sinne der Multibankfähigkeit) in unterschiedlichen Records, jeweils mit Kennung/Tag ’F0‘ abgelegt werden. Jede HBCI-Bankverbindung belegt dabei einen Record analog der im Folgenden beschriebenen Struktur EF_NOTEPAD.

C.1.2 EF_NOTEPAD

Bei dem EF_NOTEPAD handelt es sich um ein lineares EF mit einer variablen Recordlänge, die aus technischen Gründen auf maximal 239[1] Byte begrenzt ist. Es dient der Ablage beliebiger Daten.

---

[1] Nach ISO 7816-4 ist eine APDU maximal 255 Bytes lang. Nach Abzug der Protokolldaten steht eine netto Datenlänge von maximal 239 Byte zur Verfügung.

Die HBCI Anwendung nutzt das EF_NOTEPAD zur Speicherung von Zugangsspezifischen Daten, den HBCI-Parameterblöcken. So kann ein Online-Banking-Kundenprodukt in einem HBCI-Parameterblock und damit in einem Record des EF_NOTEPADs Informationen wie z.B. die HBCI-Benutzerkennung ablegen. Darüber hinaus können vom Kundenprodukt in einem separaten weiteren Record aber auch (produktspezifische) Informationen zu Kundenpräferenzen und –einstellungen (z.B. Sprache, Anzeigeparameter etc.) abgelegt werden.

Den Herstellern von Kundensystemen wird vorgeschlagen, beim EF_NOTEPAD neben einer Länge von 239 Byte auch Karten mit einer Maximallänge von nur 200 Byte zu unterstützen. Zur Ermittlung der Maximallänge soll der Tag „82“ des Bereiches FCP ausgelesen werden.

Der Inhalt des Notepad kann im Wesentlichen nur nach vorhergehender, erfolgreicher CSA-Passwort-Verifizierung gelesen und verändert werden. Somit ist der Inhalt insbesondere vor unberechtigtem Auslesen geschützt (z.B. wenn die Kontonummer als Bestandteil der Benutzerkennung gespeichert ist).

Das Auslesen der Records erfolgt über ein *Read Record* auf alle vorhandenen Records. Wird ein HBCI-Parameterblock gesucht so ist anschließend ein Vergleich durchzuführen, ob der TAG des Records den Inhalt ’F0‘ enthält.

Alternativ können mit dem Kommando SEARCH RECORD mit dem Suchmuster ’F0‘ für das erste Byte des Recordinhalts genau die für HBCI relevanten Records ausgelesen werden.

- **FCP**

Für das EF_NOTEPAD sind die folgenden FCP festzulegen:

Tag	Länge	Wert	Erläuterung
'62'	'1C'		Tag und Länge für FCP
'82'	'05'	'14 41 00 EF XX'	Datei-Deskriptor für lineares EF mit variabler Recordlänge bis zu 239 ('EF') Byte und XX Records
'83'	'02'	'A6 11'	Datei-ID des EF_NOTEPAD
'85'	'02'	'YY YY'	für Nutzdaten allokierter Speicherplatz in Byte (XX Records mal 239 Byte)[2]
'88'	'01'	'D0'	SFI '1A' für das EF_NOTEPAD
'A1'	'08'	'8B 06 00 30 01 04 02 05'	Zugriffsregel-Referenzen

Die maximale Anzahl der Records und deren maximale Länge wird bei der Produktion der Karte festgelegt.

[2] Beispiel: für XX = ‘05‘ a 239 Byte ist ein Datenbereich von 1195 Byte anzulegen →YY YY = ’04 AB‘.

Im SE #1 dürfen READ, SEARCH und UPDATE RECORD nur ausgeführt werden, wenn zuvor eine Karteninhaberauthentikation mit dem globalen Passwort 3 (CSA-Passwort) erfolgt ist. Der Returncode wird nicht MAC-gesichert (Zugriffsregeln im Record 4 des EF_RULE).

Im SE #2 dürfen die Kommandos READ, SEARCH und UPDATE RECORD nur ausgeführt werden, wenn sie mit Secure Messaging durchgeführt werden. **Entweder** ist zuvor eine Karteninhaberauthentikation mit dem globalen Passwort 3 (CSA-Passwort) erfolgt und die MAC-Bildung im Secure Messaging erfolgt für Kommando- und Antwortnachricht mit dem Sessionkey SK2; **oder** (ohne vorherige Karteninhaberauthentikation) die MAC-Bildung erfolgt für Kommando- und Antwortnachricht mit dem $K_{Notepad\_Admin}$ (Zugriffsregel im Record 5 des EF_RULE).

Im SE #2 darf das Kommando APPEND RECORD nur mit Secure Messaging durchgeführt werden. Die MAC-Bildung erfolgt für Kommando- und Antwortnachricht mit dem $K_{Notepad\_Admin}$.

Im SE #2 darf das Kommando SELECT FILE (EF) ohne Einhaltung von Zugriffsbedingungen oder mit Secure Messaging durchgeführt werden. Die MAC-Bildung im Secure Messaging erfolgt für Kommando- und Antwortnachricht mit dem Sessionkey SK2.

- **Aufbau eines Records**

POS	Länge	Wert	Erläuterung
1	1	'XX'	Tag
2	1 oder 2	'XX' oder '81 XX'	Länge (bei Längen über 127 Byte ist die Kodierung '81' 'xx' zu verwenden)
3	L	'XX..XX'	Nutzdaten

Als Tags werden festgelegt:

Byte 1	Bedeutung
'00'	freier Record
'F0'	Belegung mit HBCI-Parameterblock
'F1'-'FE'	RFU

Durch den Tag 'F0' wird ein Recordeintrag als HBCI-Parameterblock für die HBCI-Anwendung gekennzeichnet. Für Belegungen der EF_NOTEPAD-Records durch andere Anwendungen stehen die Tags 'F1' bis 'FE' zur Verfügung. Die Kennungen werden durch den ZKA vergeben.

Initial werden alle Records mit '00..00' belegt und so als leere Records gekennzeichnet.

- **Beispiel eines EF_NOTEPADs**

In der folgenden Tabelle ist die beispielhafte Belegung eines EF_NOTEPAD mit 7 Records angegeben.

Record	Eintrag	Erläuterung
1	’F0 XX...XX‘	Erste HBCI-Bankverbindung
2	’F0 XX...XX‘	Zweite HBCI-Bankverbindung
3	’F0 XX...XX‘	Dritte HBCI-Bankverbindung
4	’00..00‘	frei
5	’F1 XX..XX‘	belegt durch Anwendung mit Kennung ’F1‘
6	’00..00‘	frei
7	’F0 XX...XX‘	Vierte HBCI-Bankverbindung

- **Umgang mit variablen Recordlängen**

  Durch die Definition des EF_NOTEPAD als lineares EF mit variabler Recordlänge werden beim Lesen eines Records nur die tatsächlich vorhandenen Daten von der Karte zurückgegeben.

  Command APDU eines READ RECORD:

Byte	Wert	Erläuterung
1-2	'00 B2'	CLA, INS
3	'0X'	P1, Recordnummer X
4	'D4'	P2, Reference Control Byte
5	'00'	L_e

  Wenn das READ RECORD erfolgreich ausgeführt wird, gibt die Chipkarte eine Antwortnachricht mit der folgenden Struktur zurück:

Byte	Länge	Wert	Erläuterung
1-L	L	'XX ...XX'	Recordeintrag
(L+1)-(L+2)	2	‘SW1 SW2‘	Positiver Returncode SW1 SW2

  Ein HBCI-Recordeintrag beginnt in diesem Fall mit dem Tag ‘F0‘ und einem Längenbyte.

C.1.2.1 Recordbelegung des EF_NOTEPAD mit einem HBCI-Parameterblock, Version 001

Bei Verwendung von SECCOS 6 muss mindestens die Version V002 des EF_NOTEPAD eingesetzt werden.

Ein HBCI-Recordeintrag hat bei V001 folgenden prinzipiellen Aufbau:

Tag		Länge (Byte)	Wert	Format	Status	Erläuterung
'F0'		Var. max 'EC'[3]				HBCI-Parameterblock
	'C0'	'03'	'30' '30' '31'	3an	O	Version 001 des HBCI-Parameterblocks
'E1'		Var. max. '5B'			M	HBCI-Institutsparameterblock
	'C1'	'01'-'14'	Kreditinstituts-bezeichnung	..20an	O	
	'C2'	'03'	Länderkenn-zeichen	3an	M	ISO 3166 numerisch in 3 ASCII-Zeichen codiert
	'C3'	'01'-'1E'	Kreditinstitutscode	..30an	M	in jeweils national bekannter Notation
	'C4'	'1B'	Hashwert Instituts-schlüssel	27bin	O	
	'C5'	'01'	Schlüsselstatus	1bin	M	8 Statusflags
'E2'		Var. max. '37'			M	HBCI-Kommunikations-parameterblock
	'C6'	'01'	Kommunikations-dienst	1n	M	2 = TCP/IP
	'C7'	'01'-'32'	Kommunikations-adresse	..50an	M	
'E2'		Var. max. '37'			O	2. HBCI-Kommunikations-parameterblock
	'C6'	'01'	Kommunikations-dienst	1n	M	2 = TCP/IP
	'C7'	'01'-'32'	Kommunikations-adresse	..50an	M	
'E3'		Var. max. '54'			O	HBCI-Kundenparameterblock
	'C8'	'01'-'1E'	Benutzerkennung	..30an	M	
	'C9'	'01'-'1E'	Kunden-ID	..30an	O	
	'CA'	'0C' oder '12"	Info Inhaber-schlüssel	12an oder 18an	M	Schlüsselnummer und Schlüsselversion jeweils für den Signierschlüssel, den Chiffrierschlüssels und optional für den Signaturschlüssel des Karteninhabers

Die Längen der einzelnen Records werden wie folgt nach ASN.1 BER (Basic Encoding Rules) kodiert:

[3] Nettodatenlänge ‚EC'=236 Byte + 3 Byte Längenfeld ergibt die maximale Recordlänge von 239 Byte

Längen 'XX', wobei 'XX' die hexadezimale Darstellung eines Wertes zwischen 0 und 127 ist, werden als 'XX' in ein Byte kodiert werden.

Längen 'XX', wobei 'XX' die hexadezimale Darstellung eines Wertes zwischen 128 und 255 ist, müssen als '81 XX' in zwei Byte kodiert werden

Ausnahme ist hier die Länge des TAG 'F0', dieser wird immer in der Form 'F0' '81 XX' kodiert.

Ist der Record länger als die tatsächliche ASN.1 Struktur so kann der überschüssige Speicherplatz im Record mit '00' belegt (z.B. ASN.1 Struktur 170 Byte, Recordlänge 239 Byte → Filler 69 Byte mit '00'). Das Kundenprodukt soll nur die Nutzdaten übertragen,

C.1.2.2 Recordbelegung des EF_NOTEPAD mit einem HBCI-Parameterblock, Version 002

Bei Verwendung von SECCOS 6 muss mindestens die Version V002 des EF_NOTEPAD eingesetzt werden.

Ein HBCI-Recordeintrag hat bei V002 folgenden prinzipiellen Aufbau:

Tag		Länge (Byte)	Wert	For-mat	Sta tus	Erläuterung
'F0'		Var. max 'EC'[4]				HBCI-Parameterblock
	'C0'	'03'	'30' '30' '32'	3an	M	Version 002 des HBCI-Parameterblocks
'E1'		Var. max. '5B'			M	HBCI-Institutsparameterblock
	'C1'	'01'-'14'	Kreditinstituts-bezeichnung	..20an	O	
	'C2'	'03'	Länderkenn-zeichen	3an	M	ISO 3166 numerisch in 3 ASCII-Zeichen codiert
	'C3'	'01'-'1E'	Kreditinstitutscode	..30an	M	in jeweils national bekannter Notation
	'C4'	'27'	Hashwert Instituts-schlüssel	39bin	O	
	'C5'	'01'	Schlüsselstatus	1bin	M	8 Statusflags
'E2'		Var. max. '37'			M	HBCI-Kommunikations-parameterblock
	'C6'	'01'	Kommunikations-dienst	1n	M	2 = TCP/IP
	'C7'	'01'-'32'	Kommunikations-adresse	..50an	M	
'E2'		Var. max. '37'			O	2. HBCI-Kommunikations-parameterblock
	'C6'	'01'	Kommunikations-dienst	1n	M	2 = TCP/IP
	'C7'	'01'-'32'	Kommunikations-adresse	..50an	M	
'E3'		Var. max. '54'			O	HBCI-Kundenparameterblock
	'C8'	'01'-'1E'	Benutzerkennung	..30an	M	
	'C9'	'01'-'1E'	Kunden-ID	..30an	O	
	'CA'	'0C' oder '12"	Info Inhaber-schlüssel	12an oder 18an	M	Schlüsselnummer und Schlüsselversion jeweils für den Signierschlüssel, den Chiffrierschlüssels und optional für den Signaturschlüssel des Karteninhabers

[4] Nettodatenlänge ‚EC'=236 Byte + 3 Byte Längenfeld ergibt die maximale Recordlänge von 239 Byte

C.1.2.2.1 Tag ‘F0‘: HBCI-Parameterblock

Durch den Tag ‘F0‘ wird ein Record mit HBCI-Parameterblock für die HBCI-Anwendung gekennzeichnet. Für Belegungen der EF_NOTEPAD-Records durch andere Anwendungen stehen die Tags ‘F1‘ bis ‘FE‘ zur Verfügung.

Ein HBCI-Parameterblock enthält in der angegebenen Reihenfolge:

- **optional** ein Versionskennzeichen
- genau einen HBCI-Institutsparameterblock mit **Tag ‘E1‘**
- genau einen HBCI-Kommunikationsparameterblöcke mit **Tag ‘E2‘**
- **optional** einen weiteren HBCI-Kommunikationsparameterblöcke mit **Tag ‘E2**[5]
- **optional** einen HBCI-Kundenparameterblock mit **Tag ‘E3‘**

Die maximale Länge des HBCI-Parameterblocks wird beschränkt durch die maximale Recordlänge von 239 Byte[6].

C.1.2.2.2 Tag ‘C0‘: HBCI-Version

In jedem ‘F0‘ Record kann zur Kennzeichnung der Version des EF-NOTEPAD ein Sub-Record (z. B ’C0’ ’03’ ’30’ ’30’ ’30’) aufgenommen werden. Die Zählung der Version beginnt bei 1. Ist kein Sub-Record ’C0’ vorhanden, so bedeutet dieses, dass die Belegung des EF-NOTEPAD gemäß der Version 1 erfolgt.

Anmerkung: In der vorhergehenden Version des Dokumentes wurde fälschlicherweise ’E0’ als Tag verwendet. ’E0’ kann für die erste HBCI-Version ’000’ weiter verwendet werden. Seit der aktuellen HBCI-Version des EF_NOTEPAD wird durchgängig ’C0’ verwendet.

C.1.2.2.3 Tag ‘E1‘: HBCI-Institutsparameterblock

Durch den Tag ‘E1‘ wird der Block der institutsspezifischen Parameter gekennzeichnet. Ein HBCI-Institutsparameterblock enthält in der angegebenen Reihenfolge:

- **optional** eine Kreditinstitutsbezeichnung mit **Tag ‘C1‘**, alphanumerisch mit bis zu 20 Zeichen
- genau ein Länderkennzeichen des kontoführenden Instituts mit **Tag ‘C2‘**. Verwendet wird der numerische ISO 3166-Code als 3-stellige alphanumerische Zeichenkette (z.B. Deutschland = “280“)
- genau eine Kreditinstitutskennung mit **Tag ‘C3‘**, in einer jeweils national bekannten Notation mit bis zu 30 Stellen. Für deutsche Kreditinstitute wird hier die 8-stellige Bankleitzahl verwendet.

---

5 Somit ist der erste HBCI-Kommunikationsparameterblock ist also verpflichtend, der zweite optional.

6 In einer konkreten Umsetzung ist es nicht möglich einen HBCI-Parameterblock mit allen Felder in der maximalen Länge zu nutzen. Dabei würde die maximale Recordlänge von 239 Byte überschritten.

- **V001: optional** einen Hashwert des öffentlichen Signierschlüssels des Instituts mit **Tag 'C4'**, binär mit genau 27 Byte. Der Eintrag besteht aus

  [3 Byte Schlüsselnummer | 3 Byte Schlüsselversion | 1 Byte Kennzeichen Hashverfahren | 20 Byte Hashwert].

  Als Kennzeichen für das Hashverfahren werden festgelegt:[7]

  - '02' = RIPEMD-160

  Die Parameter Schlüsselnummer und Schlüsselversion des Institutsschlüssels werden in je 3 Byte rechtsbündig mit führenden Nullen codiert (z.B. Schlüsselnummer 1 → die Bytefolge '30' '30' '31'.

- **V002: optional** einen Hashwert des öffentlichen Signierschlüssels des Instituts mit **Tag 'C4'**, binär mit genau 39 Byte für die Hashwertverfahren RIPEMD-160 und SHA-256. Das Verfahren ist abhängig vom Sicherheitsprofil zu wählen.

  Der Eintrag besteht bei RIPEMD-160 aus

  [3 Byte Schlüsselnummer | 3 Byte Schlüsselversion | 1 Byte Kennzeichen Hashverfahren | 32 Byte Hashwert].
  Der Hashwert ist hierbei folgendermaßen aufgebaut:
  [12 Byte '00' | 20 Byte RIPEMD-160 Hashwert]

  Der Eintrag besteht bei SHA-256 aus

  [3 Byte Schlüsselnummer | 3 Byte Schlüsselversion | 1 Byte Kennzeichen Hashverfahren | 32 Byte Hashwert].

  Als Kennzeichen für das Hashverfahren werden festgelegt:

  - '02' = RIPEMD-160 für RDH-3 und RDH-5
  - '03' = SHA-256 für RAH-7, RAH-9 sowie RDH-6 bis RDH-9

  Die Parameter Schlüsselnummer und Schlüsselversion des Institutsschlüssels werden in je 3 Byte rechtsbündig mit führenden Nullen codiert (z.B. Schlüsselnummer 1 → die Bytefolge '30' '30' '31'.

- genau ein Schlüsselstatus mit **Tag 'C5'**, binär von genau 1 Byte Länge. Der Schlüsselstatus enthält acht Flags mit folgender Bedeutung:

---

[7] Aus folgenden Gründen wird nur ein fest zugeordnetes Hashverfahren verwendet:
Generell könnte im Tag C4 jedes gültige Hashverfahren zum Einsatz kommen, wobei nur bei einer ZKA-Bankensignaturkarte mit zuvor aufgebrachtem Zertifikat im Grundsatz im Tag C4 beide Hashverfahren denkbar sind. Sollte hierbei z. B. die automatische Hashwertprüfung fehlschlagen (z.B. weil das Institut zwischenzeitlich die Schlüssel geändert hat), so wird clientseitig auf das INI-Briefverfahren (und damit in V001 auf das Hashverfahren RIPEMD-160) gewechselt. Auch beim Aufbringen neuer zusätzlicher Bankverbindungen auf die Chipkarte wird das INI-Briefverfahren (und damit in V001 RIPEMD-160) verwendet. Bei ZKA-Bankensignaturkarten ohne Zertifikat wird der Eintrag neuer Bankverbindungen immer über das INI-Brief-Verfahren (und damit bei V001 über RIPEMD-160) abgesichert.

Bit1	Erstmalige Übermittlung der Kundenschlüssel notwendig	'1'b - Ja '0'b - Nein
Bit2	Institutsrechner erwartet Signaturen nach ISO9796 mit AnnexA	'1'b - Ja '0'b - Nein
Bit3	Institutsschlüssel validiert	'1'b - Ja '0'b - Nein
Bit4	Ausstehende Übermittlung des neuen öffentlichen Chiffrierschlüssels des Kunden bei Schlüsseländerung[8]	'1'b - Ja '0'b - Nein
Bit5	Ausstehende Übermittlung des neuen öffentlichen Signierschlüssels des Kunden bei Schlüsseländerung[9]	'1'b - Ja '0'b - Nein
Bit6	Schlüsselsperre mit Erfolg durchgeführt (Info, da terminierte Sperrung erst in der Zukunft wirksam werden kann)	'1'b - Ja '0'b - Nein
Bit7	Leitungsprobleme bei Übermittlung neuer Schlüssel	'1'b - Ja '0'b - Nein
Bit8	Reserviert	'0'b

Bei der Personalisierung muss als Initialisierungswert '01' aufgebracht werden.

Ein HBCI-Institutsparameterblock belegt inklusive der Tag- und Längenbytes somit maximal 93 Byte.

C.1.2.2.4 Tag ‘E2‘: HBCI-Kommunikationsparameterblock

Durch das Tag ‘E2‘ wird der Block der generellen Kommunikations-Parameter gekennzeichnet. Ein HBCI-Kommunikationsparameterblock enthält in der angegebenen Reihenfolge:

- genau einen Kommunikationsdienst mit **Tag ‘C6‘**, 1 Stelle numerisch. Zurzeit definiert ist der numerische Wert 2 (TCP/IP)
- genau eine Kommunikationsadresse mit **Tag ‘C7‘**, alphanumerisch mit bis zu 50 Zeichen

Ein HBCI-Kommunikationsparameterblock belegt inklusive der Tag- und Längenbytes somit maximal 57 Byte.

C.1.2.2.5 Tag ‘E3‘: HBCI-Kundenparameterblock

Durch den Tag ‘E3‘ wird der **optional** vorhandene Block der kundenspezifischen Parameter gekennzeichnet. Ist der Block nicht vorhanden, so handelt es sich um eine im Rahmen der HBCI-Anwendung Bankensignaturkarte ohne Zertifikat. Ein HBCI-Kundenparameterblock enthält in der angegebenen Reihenfolge:

- genau eine Benutzerkennung mit **Tag ‘C8‘**, alphanumerisch mit bis zu 30 Zeichen
- **optional** eine Kunden-ID mit **Tag ‘C9‘**, alphanumerisch mit bis zu 30 Zeichen

---

[8] Nicht zu belegen, da die ZKA-Bankensignaturkarte keinen Wechsel der Kundenschlüssel unterstützt.

[9] Nicht zu belegen, da die ZKA-Bankensignaturkarte keinen Wechsel der Kundenschlüssel unterstützt.

- genau ein Info Inhaberschlüssel mit Tag 'CA', von genau 12 oder 18 numerischen Zeichen**.**

  Bei 12 Byte Länge des Blocks ist der Inhalt wie folgt definiert:

Schlüsselnummer Signierschlüssel [3n]
Schlüsselversion Signierschlüssel [3n]
Schlüsselnummer Chiffrierschlüssel [3n]
Schlüsselversion Chiffrierschlüssel [3n]

  Bei 18 Byte Lange ist der Inhalt wie folgt definiert:

Schlüsselnummer Signierschlüssel [3n]
Schlüsselversion Signierschlüssel [3n]
Schlüsselnummer Chiffrierschlüssel [3n]
Schlüsselversion Chiffrierschlüssel [3n]
Schlüsselnummer Signaturschlüssel [3n]
Schlüsselversion Signaturschlüssel [3n]

Die Parameter Schlüsselnummer und Schlüsselversion werden in je 3 Byte numerisch rechtsbündig mit führenden Nullen angegeben. (z.B. Schlüsselnummer 1 → "001" → die Bytefolge '30' '30' '31'.

Fehlen die Angaben für den Signaturschlüssel (CA Record der Länge 12 Byte) so werden als Schlüsselnummer und Schlüsselversion des Signaturschlüssels die Schlüsselnummer und Schlüsselversion des Signierschlüssels übernommen.

Fehlt der Teilrecord mit dem Tag 'CA' (nicht vorhandener Record E3 oder Record CA oder fehlendes EF_NOTEPAD) und liegen somit weder für den Signierschlüssel und den Chiffrierschlüssel noch für den Signaturschlüssel Schlüsselnummer und Schlüsselversion vor so sind vom FinTS-Client die Schlüsselnummern und Schlüsselversionen aller Schlüssel nach folgenden Mechanismen vorzubesetzen.

Die Schlüsselnummer wird gemäß dem genutzten RAH- bzw. RDH-Verfahren besetzt. Die Schlüsselversion wird gängigerweise im ersten Ausgabejahr mit "001" vorbesetzt und anschließend im jährlichen Turnus um 1 erhöht.

RDH Verfahren	Schlüsselnummer	Schlüsselversion
RDH3	"003" → '30' '30' 33'	"001" → '30' '30' '31'
RDH5	"005" → '30' '30' 35'	"001" → '30' '30' '31'
RDH6	"006" → '30' '30' 36'	"001" → '30' '30' '31'
RAH7, RDH7	"007" → '30' '30' 37'	"001" → '30' '30' '31'
RDH8	"008" → '30' '30' 38'	"001" → '30' '30' '31'
RAH9, RDH9	"009" → '30' '30' 39'	"001" → '30' '30' '31'

RDH Verfahren	Schlüsselnummer	Schlüsselversion
RAH10, RDH10	"010" → '30' '31' 30'	"001" → '30' '30' '31'

Über die Schlüsselnummer im EF_NOTEPAD kann das zu verwendende Sicherheitsprofil ermittelt werden.

Wichtiger Hinweis:
Bei allen Verfahren ab RDH-3 müssen für die Schlüsselnummer die entsprechenden Werte aus der obigen Tabelle verwendet werden. Die Nutzung von Schlüsselnummer „001“ ist nicht erlaubt.

Ein HBCI-Kundenparameterblock belegt inklusive der Tag- und Längenbytes somit maximal 86 Byte.

C.1.2.2.6 Beispiel

Beispiel für eine Recordbelegung für V001 (Tags und Längenbytes sind fett markiert)

Inhalt	Erläuterung
F0 81 76	HBCI-Parameterblock
E1 3D	Institutsparameterblock
C1 0C 54 45 53 54 49 4E 53 54 49 54 55 54	Institutsbezeichnung “TESTINSTITUT“
C2 03 32 38 30	Länderkennzeichen “280“
C3 08 31 32 33 34 35 36 37 38	BLZ 12345678
C4 1B 30 30 31 30 30 31 02 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 14	Schlüsselnummer 1, Schlüsselversion 1, Hashverfahren RIPEMD-160, Hashwert
C5 01 01	Schlüsselstatus '01'
E2 12	Kommunikationsparameterblock
C5 01 02	Kommunikationsdienst TCP/IP
C6 0D 31 39 32 2E 31 36 38 2E 31 31 2E 32 32	Kommunikationsadresse 192.168.11.22
E3 21	Kundenparameterblock
C8 0A 31 32 33 34 35 36 37 38 39 30	Benutzerkennung "1234567890"
C9 05 31 32 33 34 35	Kunden-ID "12345"
CA 0C 30 30 31 30 30 31 30 30 31 30 30 31	Info Inhaberschlüssel Schlüsselnummer SIG 1, Schlüsselversion SIG 1 Schlüsselnummer CHIF 1, Schlüsselversion CHIF 1

C.1.2.2.7 Erreichen der maximalen Recordlänge

Bei Ausnutzung aller Maximallängen und Aufnahme aller optionalen Felder und Angabe zweier Kommunikationsparameterblöcke und eines Kundenparameterblocks ergibt sich ein maximaler Platzbedarf von 297 Byte. Dieser Platzbedarf ist aber in einem Record nicht abbildbar. Normalerweise wird aber nur ein Kommunikationsparameterblock verwendet sowie

selten alle Maximallängen ausgereizt, so dass meistens die maximale Recordlänge von 239 Byte genügt. Bei älteren bereits ausgegebenen Bankensignaturkarten ist nur eine maximale Recordlänge von 200 Byte vorgesehen.

C.1.3 Terminalabläufe

Dieses Kapitel spezifiziert die Terminalabläufe im Umgang mit dem RAH- bzw. RDH-Verfahren auf SECCOS-Chipkarten [SECCOS] bzw. [SECCOS-6]. Ein Online-Banking-Kundenprodukt nutzt

- zur Verschlüsselung und Signierung von HBCI-Nachrichten die auf der Chipkarte zur Verfügung stehende Signatur-Anwendung (DF_SIG, [ZKASIG]) und die durch das Betriebssystem bereitgestellten Signatur-Funktionen,
- als Sequenzzähler (Signatur-ID) interne Bedienungszähler der Signatur-Anwendung (siehe Kap. C.1.3.1),
- als Datenspeicher für die Zugangsdaten ein auf der Chipkarte optional vorhandenes DF_NOTEPAD ([DF_NOTEPAD], siehe Kap. C.1.1).

C.1.3.1 Verfahren zur Ermittlung der Sicherheitsreferenznummern

Auf der Bankensignaturkarte wird kein eigenständiger Sequenzzähler (wie das Element EF_SEQ im HBCI DDV-Verfahren) verwaltet, sondern es werden jeweils chipkarteninterne „Usage Counter“ der beiden zur Signatur verwendeten Schlüssel S_K.CH.DS und S_K.CH.AUT$_{C/S}$ herangezogen.

Für jedes Signaturschlüsselpaar wird ein separater Usage Counter verwaltet.Dieser kann jeweils zwei, drei oder vier Byte lang sein.

Da die Usage Counter auf der Chipkarte dekrementiert werden, als Sicherheitsreferenznummer („Signatur-ID“) aber ein streng monoton aufsteigender Zähler gefordert ist, wird die konkrete Sicherheitsreferenznummer nach folgendem Algorithmus ermittelt:

1. Auslesen des 2 bis 4 Byte langen Usage Counter (UC) UC_{DS} des Schlüssels S_K.CH.DS bzw. UC_{AUT} des Schlüssels S_K.CH.AUT$_{C/S}$.
2. Sei **neg**(UC) die bitweise logische Negation von UC. Dann ist die Sicherheitsreferenznummer (SRN)

$$SRN_{DS} = \textbf{neg}(UC_{DS})$$

$$SRN_{AUT} = \textbf{neg}(UC_{AUT})$$

Die einzelnen Usage Counter haben folgende Wertebereiche:

von 0 bis 65.535	bei Länge(UC) = 2 Byte
von 0 bis 16.777.215	bei Länge(UC) = 3 Byte
von 0 bis 4.294.967.295	bei Länge(UC) = 4 Byte

Damit muss die Sicherheitsreferenznummer SRN über die entsprechenden Wertebereiche verfügen und benötigt zur Darstellung ebenfalls mindestens 2, 3 oder 4 Byte.

Ein Wrap-Around bei Erreichen des jeweiligen Maximalwerts findet nicht statt, da das Erreichen eines Usage Counter 0 den Schlüssel der Chipkarte für die weitere Verwendung sperrt.

Beispiel:

UC_{DS} = ’00 0A‘ (dezimal 10) $\Rightarrow$ SRN_{DS} = **neg**(UC_{DS}) = ‘FF F5‘ (dezimal 65.525)

UC_{AUT} = ‘FA 1D‘ (dezimal 64.029) $\Rightarrow$ SRN_{AUT} = **neg**(UC_{AUT}) = ’05 E2‘(dezimal 1506)

Dieser Algorithmus ist in der jeweiligen Anwendungssoftware zu realisieren.

C.1.3.2 Beschreibung der Terminalabläufe

Nachfolgend werden die Anwendungsabläufe aus Endgerätesicht an einem privaten Signaturterminal [KT-KONZEPT] spezifiziert. Hierbei werden ausschließlich die chipkartenbezogenen Aspekte berücksichtigt. Anwendungsbezogene Details sind nicht Bestandteil dieser Spezifikation.

Um die Abläufe möglichst einfach beschreiben zu können, werden in der nachfolgenden Beschreibung Befehle der ZKA-SIG-API [KT-SIG] verwendet. Hiermit ist jedoch die Verwendung der ZKA-SIG-API für technische Implementierungen nicht zwingend vorgeschrieben. Wird die ZKA-SIG-API nicht verwendet, so sind die in [KT-SIG] angegebenen Abläufe zum Aufruf der KT-Kommandos zu berücksichtigen.

Die Anwendungsabläufe lassen sich auch auf öffentliche Signaturterminals (Geschäftsterminals) erweitern. Zu beachten ist dabei insbesondere, dass in diesem Fall zusätzlich eine

- Komponenten-Authentikation zwischen Chipkarte und Geschäftsterminal mit Aushandlung eines Sessionkey-Paares (SK1, SK2) stattfindet;
- alle Befehle an die Chipkarte im Secure Messaging mit einem SK2-MAC durchgeführt werden müssen.

Falls bei der Ausführung der Kommandos ein Fehler auftritt, bricht das Terminal den Vorgang ab, es sei denn, es ist ein abweichendes Verhalten spezifiziert.

In den hier beschriebenen Abläufen ist das Kundenterminal durch ein *zka_sig_open* (zu Beginn des Ablaufs „Signatur einleiten“) und ein *zka_sig_close* (Am Ende des Ablaufs „Signatur beenden“) für die gesamte Zeitdauer exklusiv für die Kundenanwendung reserviert.

Um zwischenzeitlich anderen Anwendungen die Möglichkeit zu geben, die Signaturdienste der Karte zu nutzen (z.B. für die Zeitdauer der Nachrichtengenerierung), können die im Folgenden beschriebenen Teilabläufe jeweils auch durch ein *zka_sig_open* und ein *zka_sig_close* gekapselt werden. Dadurch wird die exklusive Reservierung des Kundenterminals aufgehoben, die internen Zwischenwerte der ZKA-SIG-API (insbes. der Chipdaten) bleiben jedoch erhalten. Erst durch Aufruf des *zka_sig_fini_signature_application* im Ablauf „Signatur beenden“ werden die internen Zwischenwerte der ZKA-SIG-API gelöscht.

Zur Administration der Signaturkarten (z.B. Freischalten eines Zertifikates, Rücksetzen des Fehlbedienungszählers) werden von den Kreditinstituten bzw. den Kartenemittenten Softwarekomponenten zur Verfügung gestellt werden, die in der privaten Kundenumgebung zum Einsatz kommen sollen. In Kundenprodukten, die nicht von den Kartenemittenten herausgegeben werden, sollen diese Administrationsfunktionen nicht realisiert werden.

Für die kreditinstitutsseitige Realisierung dieser Softwarekomponenten hat der ZKA Anforderungen und Festlegungen formuliert, die bei Bedarf über die jeweiligen Ansprechpartner der Standards erhältlich sind.

C.1.3.2.1 Signatur einleiten

Chipkarte			Endgerät	
			M1	Aufruf der ZKA-SIG-API-Funktion *zka_sig_open*
R2	OK	← →	M2	Aufruf der ZKA-SIG-API-Funktion *zka_sig_init_signature_application*
R3	OK	← →	M3	Aufruf der ZKA-SIG-API-Funktion *zka_sig_verify_CSA_password*
R4	OK / „File not found“	← →	C4	SELECT FILE DF_NOTEPAD
R5	Bankverbindung	← →	C5	ggf. READ RECORD EF_NOTEPAD
			A5	Daten prüfen und speichern

- **Erläuterung**

1. Die ZKA-SIG-API-Funktion *zka_sig_open* wird ausgeführt. Diese Funktion stellt eine exklusive Verbindung zum Kundenterminal her.
2. Die ZKA-SIG-API-Funktion *zka_sig_init_signature_application* wird ausgeführt. Diese sorgt insbesondere für ein Reset der Karte und das Auslesen der relevanten Basisinformationen der Karte.
3. Die ZKA-SIG-API-Funktion *zka_sig_verify_CSA_password* wird ausgeführt. Diese Funktion liest das CSA-Passwort ein und führt eine Verifikation gegenüber der Chipkarte durch.
4. Die Applikation „Notepad“ wird geöffnet, indem das ADF der Applikation, DF_NOTEPAD durch das Terminal mittels des Kommandos SELECT FILE ausgewählt wird.

- **Command APDU**

Byte	Wert	Erläuterung
1-2	'00 A4'	CLA, INS
3	'04'	P1, Selektion mit DF-Name
4	'0C'	P2, Keine Antwortdaten
5	'09'	L_C
6-14	'D2 76 00 00 25 4E 50 01 00'	AID der Notepad-Applikation

Wenn die Notepad-Applikation auf der Karte nicht vorhanden ist, wird der folgende Schritt übersprungen. In diesem Fall müssen die Zugangsdaten von einer anderen Stelle gelesen oder vom Benutzer eingegeben werden.

4. Das Terminal liest mittels READ RECORD sukzessive die Bankverbindungsdaten in den Records des EF_NOTEPAD (SFI '1A'), bis der oder die ”passenden” Einträge gefunden wurden. Das Lesen von Einträgen ist erst nach erfolgreicher CSA-Passwort-Verifikation (Schritt 2) möglich.

♦ **Command APDU**

Byte	Wert	Erläuterung
1-2	'00 B2'	CLA, INS
3	'0X'	P1, Recordnummer X
4	'D4'	P2, Reference Control Byte
5	'00'	L_e

Wenn das READ RECORD erfolgreich ausgeführt wird, gibt die Chipkarte eine Antwortnachricht mit der folgenden Struktur zurück:

Byte	Länge	Wert	Erläuterung
1-2	2	'XX LL'	Kennung und Länge
3-LL	LL	'XX..XX'	Nutzdaten
(LL+1)-(LL+2)	2	'XX XX'	Positiver Returncode SW1 SW2

Ist die Kennung ungleich ’00‘, so sind Parameterdaten gemäß Kap. C.1.1 enthalten. Es werden alle weiteren Records gelesen, bis die Chipkarte das Ende der Datei (keine weiteren Records) signalisiert.

Anstatt alle Records auszulesen und auf Übereinstimmung mit der Kennung zu überprüfen, kann alternativ auch das Kommando SEARCH RECORD verwendet werden, um mittels eines übergebenen Suchmusters vorab die ”passenden” Recordnummern in einem Schritt zu finden. Anschließend müssen dann nur diese Recordnummern mittels READ RECORD ausgelesen werden.

♦ **Command APDU**

Byte	Wert	Erläuterung
1-2	'00 A2'	CLA, INS für SEARCH RECORD
3	'01'	P1, Start mit Recordnummer 1
4	'D7'	P2, spezifische Suche im SFI ‘1A‘
5	'04'	L_C
6	'04'	CTRLB
7	'00'	Offset Indicator Byte
8	'02'	Konfigurationsbyte
9	’F0‘	Suchmuster
10	'00'	L_e

Wenn das SEARCH RECORD erfolgreich ausgeführt wird, gibt die Chipkarte eine Antwortnachricht mit der folgenden Struktur zurück:

Byte	Länge	Wert	Erläuterung
1-n	n	'XX XX'	Recordnummer(n)
n+1	1	'XX'	Statusbyte SW1
n+2	1	'XX'	Statusbyte SW2

Es können nun gezielt nur die in der Antwortnachricht angegebenen Records ausgelesen werden.

C.1.3.2.2 Nachrichten generieren

Dieser Teil des Gesamtablaufs ist nur insofern chipkartenrelevant, als (optional) Bankverbindungsdaten, die für die Auftragsgenerierung benötigt werden, aus der Chipkarte entnommen werden. Dies ist bereits im Schritt „Signatur einleiten“ (Kap. C.1.3.2.1) geschehen. Für die folgende Ablaufbeschreibung wird angenommen, dass die Anwendung bereits Auftrags-Nachrichten generiert hat. Diese Nachrichten müssen jetzt ggf. noch kryptographisch gesichert werden, d.h. es werden Segmente für die elektronische(n) Signatur(en) und für die Verschlüsselung entsprechend den jeweiligen Spezifikationen eingefügt.

C.1.3.2.3 Nachrichten signieren

C.1.3.2.3.1 Nachrichten signieren bei HBCI

Die folgenden Abläufe können im Falle von HBCI offline, d.h. außerhalb des Übertragungsdialogs vollzogen werden. Dies gilt für alle Nachrichten mit Ausnahme der Dialoginitialisierung. Der Grund besteht darin, dass für die Absicherung aller Kreditinstitutsnachrichten der Schlüssel des Senders der Dialoginitialisierungsnachricht erforderlich ist. Daher muss auch die Chipkarte des Senders während des gesamten Dialogs im Endgerät stecken.

Die Abläufe für die Signatur der Dialoginitialisierungsnachricht sind grundsätzlich identisch mit den im Folgenden beschriebenen Abläufen für die Signatur von Auftragsnachrichten. Da aber für die Dialoginitialisierung anwendungsseitig noch weitere Chipkartendaten (Benutzerkennung, Dialog-ID, Kommunikationszugang etc.) benötigt werden, wird der komplette Ablauf einschließlich der Signatur der Dialoginitialisierung im Kap. C.1.3.2.6 "Übertragungsdialog" noch einmal beschrieben.

Chipkarte			Endgerät	
R1	BZ	→ ←	M1	Sequenzzähler (Signatur-ID) ermitteln durch Aufruf der ZKA-SIG-API-Funktion *zka_sig_read_key_usage_counter* und anschließende Invertierung des Rückgabewerts gemäß Abschnitt C.1.3.1)
			A2	Signaturkopf aufbauen und in HBCI-Nachricht einfügen
			A3	Daten (Signaturkopf, HBCI-Nutzdaten) für Signatur bereitstellen
		→ ←	M4	Signaturerstellung (siehe Kap. C.1.3.3.1)
			A5	Signaturabschluss aufbauen und in HBCI-Nachricht einfügen
			A6	ggf. M1 bis A5 für weitere Nachrichten wiederholen
			A7	signierte HBCI-Nachrichten zur Weiterverarbeitung speichern

♦ **Erläuterung**

1. Der Sequenzzähler (Signatur-ID) wird durch Auslesen der Bedienungszähler der Signaturanwendung und anschließende Berechnung ermittelt. Das Auslesen erfolgt durch Aufruf der ZKA-SIG-API-Funktion *zka_sig_read_key_usage_counter* mit der Parameterbelegung

   - counter_type = ’00‘ bei Verwendung des S_K.CH.DS, bzw.
   - counter_type = ‘02‘ bei Verwendung des S_K.CH.$AUT_{C/S}$

   Das Ergebnis BZ wird gemäß Kap. C.1.3.1 zu SZ = **neg**(BZ) invertiert und als Sequenzzähler gespeichert.

2. Der Signaturkopf wird aufgebaut und in die HBCI-Nachricht eingefügt.
3. Die Daten (Signaturkopf, HBCI-Nutzdaten) für die Signaturerstellung werden bereitgestellt.
4. Die Signatur wird berechnet (siehe hierzu Kap. C.1.3.3).
5. Der Signaturabschluss wird aufgebaut und in die HBCI-Nachricht eingefügt.
6. Ggf. können die Schritte 1 bis 5 für weitere Nachrichten wiederholt werden.
7. Die signierten HBCI-Nachrichten können zur Weiterverarbeitung gespeichert werden.

Anmerkung: Für Mehrfachsignaturen wird jeweils die Abfolge „Signatur einleiten“ – „Nachrichten signieren“ – „Signatur beenden“ wiederholt. Dies kann auch zu einem späteren Zeitpunkt geschehen. Mehrfachsignaturen müssen jedoch abgeschlossen sein, bevor die Verschlüsselung der Nachricht (Kap. C.1.3.2.4) durchgeführt wird.

C.1.3.2.4 Nachrichten verschlüsseln

C.1.3.2.5 Nachrichten verschlüsseln bei RAH

Die Chipkarte ist bei der eigentlichen Nachrichtenverschlüsselung nicht involviert. Die Software berechnet einen Einmalschlüssel, verschlüsselt das Dokument und verschlüsselt den Einmalschlüssel zur Übertragung mit dem öffentlichen Key-Encryption-Schlüssel $P_K.RECV_{INST}.KE$ des empfangenden Kreditinstituts, welches dem entsprechenden Zertifikat des Empfängers entnommen wurde[10].

Allerdings wird die Chipkarte zur Berechnung von Zufallszahlen herangezogen, welche den Einmalschlüssel bilden.

[10] [DIN-SIG4, Kapitel 6.10.1]: „If an enciphered document is sent, the card is not involved: the software computes the content encryption key, enciphers the document and finally enciphers the content encryption key by applying the receiver’s public key taken from the receiver’s KE certificate.“

Chipkarte			Endgerät	
			A1	Daten (FinTS-Nutzdaten und ggf. Signatur) für die Verschlüsselung bereitstellen
R2	RND	←	C2	Aufruf der ZKA-SIG-API-Funktion *zka_sig_get_challenge*
		→	A2	RND als Einmalschlüssel-Fragment KS_{LL} speichern
R3	RND	←	C3	Aufruf der ZKA-SIG-API-Funktion *zka_sig_get_challenge*
		→	A3	RND als Einmalschlüssel-Fragment KS_{LR} speichern
R4	RND	←	C4	Aufruf der ZKA-SIG-API-Funktion *zka_sig_get_challenge*
		→	A4	RND als Einmalschlüssel-Fragment KS_{RL} speichern
R5	RND	←	C5	Aufruf der ZKA-SIG-API-Funktion *zka_sig_get_challenge*
		→	A5	RND als Einmalschlüssel-Fragment KS_{RR} speichern
			A6	KS_{LL}, KS_{LR}, KS_{RL} und KS_{RR} zu KS konkatenieren und speichern
			A7	Daten mit KS (symmetrisch) verschlüsseln
			A8	KS mit P_K.$RECV_{INST}$.KE (asymmetrisch) verschlüsseln
			A9	Verschlüsselungsdaten aufbauen und in FinTS-Nachricht einfügen
			A10	Verschlüsselte Daten als Binärdaten in Verschlüsselungsdaten einfügen
			A11	ggf. A1 bis A10 für weitere Nachrichten wiederholen
			A12	Verschlüsselte und signierte FinTS-Nachrichten zur weiteren Bearbeitung speichern

♦ **Erläuterung**

1. Die Daten (FinTS-Nutzdaten und ggf. Signatur) für die Verschlüsselung werden bereitgestellt.
2. Mit dem Aufruf der ZKA-SIG-API-Funktion *zka_sig_get_challenge* lässt sich das Terminal eine Zufallszahl von der HBCI-Karte geben.

   Wenn das Kommando erfolgreich ausgeführt wurde, gibt die HBCI-Karte eine 8 Byte lange Zufallszahl als Antwortdatum aus, die als Einmalschlüssel-Fragment KS_{LL} gespeichert wird.
3. Mit dem Aufruf der ZKA-SIG-API-Funktion *zka_sig_get_challenge* lässt sich das Terminal eine zweite Zufallszahl von der HBCI-Karte geben, die als Einmalschlüssel-Fragment KS_{LR} gespeichert wird.
4. Mit dem Aufruf der ZKA-SIG-API-Funktion *zka_sig_get_challenge* lässt sich das Terminal eine dritte Zufallszahl von der HBCI-Karte geben, die als Einmalschlüssel-Fragment KS_{RL} gespeichert wird.
5. Mit dem Aufruf der ZKA-SIG-API-Funktion *zka_sig_get_challenge* lässt sich das Terminal eine vierte Zufallszahl von der HBCI-Karte geben, die als Einmalschlüssel-Fragment KS_{LL} gespeichert wird.
6. KS_{LL}, KS_{LR}, KS_{RL}, und KS_{RR}, werden zu KS konkateniert und gespeichert.
7. Die zu übertragenden Daten werden mit KS symmetrisch verschlüsselt (AES CBC-Mode, IV=0, ZKA-Padding).
8. Der Einmalschlüssel KS wird linksbündig mit Nullbits auf die Schlüssellänge aufgefüllt und anschließend mit dem öffentlichen Key-Encryption-Schlüssel

P_K.$RECV_{INST}$.KE des empfangenden Instituts, welches dem entsprechenden Zertifikat des Empfängers entnommen wurde, verschlüsselt. Das Ergebnis wird mit führenden Nullbits auf die Schlüssellänge erweitert.

9. Die Verschlüsselungsdaten werden aufgebaut und in die FinTS-Nachricht eingefügt.
10. Die verschlüsselten Daten als Binärdaten in die Verschlüsselungsdaten eingefügt.
11. Ggf. werden die Schritte 1 bis 10 für weitere Nachrichten wiederholt.
12. Die verschlüsselten und signierten FinTS-Nachrichten werden zur weiteren Bearbeitung gespeichert.

C.1.3.2.5.1 Nachrichten verschlüsseln bei DDV und RDH

Die Chipkarte ist bei der eigentlichen Nachrichtenverschlüsselung nicht involviert. Die Software berechnet einen Nachrichtenschlüssel, verschlüsselt das Dokument und verschlüsselt den Nachrichtenschlüssel zur Übertragung mit dem öffentlichen Key-Encryption-Schlüssel P_K.$RECV_{INST}$.KE des empfangenden Instituts, welches der übermittelten Kreditinstitutsnachricht entnommen wurde[11].

Allerdings wird die Chipkarte zur Berechnung von Zufallszahlen herangezogen, welche den Nachrichtenschlüssel bilden.

Chipkarte			Endgerät	
			A1	Daten (FinTS-Nutzdaten und ggf. Signaturkopf/-abschluss) für die Verschlüsselung bereitstellen
R2	RND	←	C2	Aufruf der ZKA-SIG-API-Funktion *zka_sig_get_challenge*
		→	A2	RND als Nachrichtenschlüssel-Hälfte KS_L speichern
R3	RND	←	C3	Aufruf der ZKA-SIG-API-Funktion *zka_sig_get_challenge*
		→	A3	RND als Nachrichtenschlüssel-Hälfte KS_R speichern
			A4	KS_L mit KS_R zu KS konkatenieren und speichern
			A5	KS auf Eigenschaft „(halb-)schwacher Schlüssel“ überprüfen und ggfs. Schritte 2-4 wiederholen.
			A6	Herstellung der Parität für KS (Parity Adjustment)
			A7	Daten mit KS (symmetrisch) verschlüsseln
			A8	KS mit P_K.$RECV_{INST}$.KE (asymmetrisch) verschlüsseln
			A9	Verschlüsselungskopf aufbauen und in FinTS-Nachricht einfügen
			A10	Verschlüsselte Daten als Binärdaten in FinTS-Nachricht einfügen
			A11	ggf. A1 bis A10 für weitere Nachrichten wiederholen
			A12	Verschlüsselte und signierte FinTS-Meldungen zur weiteren Bearbeitung speichern

♦ **Erläuterung**

1. Die Daten (FinTS-Nutzdaten und ggf. Signaturkopf/-abschluss) für die Verschlüsselung werden bereitgestellt.

[11] [DIN-SIG4, Kapitel 6.10.1]: „If an enciphered document is sent, the card is not involved: the software computes the content encryption key, enciphers the document and finally enciphers the content encryption key by applying the receiver’s public key taken from the receiver’s KE certificate.“

2. Mit dem Aufruf der ZKA-SIG-API-Funktion *zka_sig_get_challenge* lässt sich das Terminal eine Zufallszahl von der HBCI-Karte geben.

   Wenn das Kommando erfolgreich ausgeführt wurde, gibt die HBCI-Karte eine Zufallszahl als Antwortdatum aus, die als Nachrichtenschlüssel-Hälfte KS_L gespeichert wird.

3. Mit dem Aufruf der ZKA-SIG-API-Funktion *zka_sig_get_challenge* lässt sich das Terminal eine weitere Zufallszahl von der HBCI-Karte geben, die als Nachrichtenschlüssel-Hälfte KS_R gespeichert wird.

4. KS_L wird mit KS_R zu KS konkateniert und gespeichert.

5. KS wird auf die Eigenschaft „(halb-)schwacher Schlüssel“ überprüft. Liegt ein (halb-)schwacher Schlüssel vor, so wird Schritt 2-4 wiederholt.

Schwache Schlüssel des DES:

01	01	01	01	01	01	01	01
FE	FE	FE	FE	FE	FE	FE	FE
1F	1F	1F	1F	0E	0E	0E	0E
E0	E0	E0	E0	F1	F1	F1	F1

Halbschwache Schlüssel des DES:

01	FE	01	FE	01	FE	01	FE
FE	01	FE	01	FE	01	FE	01
1F	E0	1F	E0	0E	F1	0E	F1
E0	1F	E0	1F	F1	0E	F1	0E
01	E0	01	E0	01	F1	01	F1
E0	01	E0	01	F1	01	F1	01
1F	FE	1F	FE	0E	FE	0E	FE
FE	1F	FE	1F	FE	0E	FE	0E
01	1F	01	1F	01	0E	01	0E
1F	01	1F	01	0E	01	0E	01
E0	FE	E0	FE	F1	FE	F1	FE
FE	E0	FE	E0	FE	F1	FE	F1

6. Für KS wird ein Parity Adjustment durchgeführt. Das Resultat ist der zu verwendende Nachrichtenschlüssel.

7. Die zu übertragenden Daten werden mit KS symmetrisch verschlüsselt.

8. Der Nachrichtenschlüssel KS wird gemäß Paddingverfahren für das entsprechende Sicherheitsprofil auf die Länge des öffentlichen Key-Encryption-Schlüssels $P_K.RECV_{INST}.K$ des empfangenden Instituts, welches der übermittelten Kreditinstitutsnachricht entnommen wurde, aufgefüllt und anschließend mit dem $P_K.RECV_{INST}.K$ verschlüsselt.. ~~Stimmt das Verschlüsselungsergebnis mit dem Ausgangswert überein, werden die Schritte 2 bis 8 wiederholt (Generierung eines neuen Schlüssels); ansonsten wird~~ Das Ergebnis wird mit führenden Nullbits auf die Schlüssellänge erweitert. ~~und es wird mit dem folgenden Schritt 9 fortgefahren.~~

9. Der Verschlüsselungskopf wird aufgebaut und in die FinTS-Nachricht eingefügt.

10. Die verschlüsselten Daten als Binärdaten in die FinTS-Nachricht eingefügt.

11. Ggf. werden die Schritte 1 bis 10 für weitere Nachrichten wiederholt.

12. Die verschlüsselten und signierten FinTS-Meldungen werden zur weiteren Bearbeitung gespeichert.

C.1.3.2.6 Übertragungsdialog

Chipkarte			Endgerät			Kreditinstitut	
			A1	Benutzerkennung aus der bereits gelesenen Bankverbindung extrahieren			
		→ ←	M2	Nachricht signieren (s. Kap. C.1.3.2.3)			
			A3	Kommunikationszugang aus Bankverbindung herstellen			
			C4	Nachricht (beginnend mit Dialoginitialisierungsnachricht) senden	→ ←	R4	Antwortnachricht
			A5	falls Antwortnachricht verschlüsselt: Daten (Binärdaten nach dem Verschlüsselungskopf) und verschlüsselten Session-Key enc(KS) aus dem Signaturkopf für die Entschlüsselung bereitstellen			
		→ ←	M6	Ausführung der ZKA-SIG-API-Funktion *zka_sig_decrypt* zur Session-Key-Entschlüsselung, Resultat ist der Session-Key KS			
			A7	Daten mit Session-Key KS entschlüsseln.			
			A8	falls Kreditinstitutsnachricht signiert: Daten (Signaturkopf, Nutzdaten, Signatur) für Signatur-Prüfung bereitstellen			
		→ ←	M9	Signatur-Prüfung (siehe KapC.1.3.3.2)			
			A10	C4 bis M9 für alle weiteren HBCI-Nachrichten wiederholen			

C.1.3.2.7 Signatur beenden

Chipkarte	

Endgerät	
M1	Aufruf der ZKA-SIG-API-Funktion *zka_sig_fini_signature_application*
M2	Aufruf der ZKA-SIG-API-Funktion *zka_sig_close*

♦ **Erläuterung**

1. Die ZKA-SIG-API-Funktion *zka_sig_fini_signature_application* wird ausgeführt. Diese Funktion setzt die ZKA-SIG-API in den Zustand „passiv“ und löscht die darin gespeicherten Werte.
2. Die ZKA-SIG-API-Funktion *zka_sig_close* gibt die Verbindung zum Kundenterminal wieder frei.

C.1.3.3 Makros

C.1.3.3.1 Signatur-Berechnung

Signaturen mit der Chipkarte werden im Rahmen der beiden Sicherheitsdienste „Authentication“ und „Non-Repudiation“ erzeugt.

- Sicherheitsdienst Authentication: Signatur mit Schlüssel S_K.CH.AUT$_{C/S}$ (Client-Server-Authentikations-Schlüssel)
- Sicherheitsdienst Non-Repudiation: Signatur mit Schlüssel S_K.CH.DS (Digitaler Signatur-Schlüssel)

Die tatsächliche Durchführung der Signatur durch die Chipkarte ist insbesondere an die Erfüllung von Zugriffsbedingungen geknüpft, hier ist dies insbesondere eine vorhergehende Benutzer-Authentikation in Form der Verifikation

- des CSA-Passworts für die Erlaubnis zur Signatur mit dem Schlüssel S_K.CH.AUT$_{C/S}$
- der Signatur-PIN für die Erlaubnis zur Signatur mit dem Schlüssel S_K.CH.DS

Durch einen in der Chipkarte personalisierten Parameter der Signatur-Anwendung [ZKASIG] wird dabei festgelegt, nach wie vielen elektronischen Signaturen spätestens die Benutzer-Authentikation zu wiederholen ist. Eine Benutzer-Authentikation wird bei Bedarf innerhalb der ZKA-SIG-API-Funktionen *zka_sig_digital_signature* bzw. *zka_sig_cs_authentication* durchgeführt.

Chipkarte			Endgerät	
R1	evtl. Hashwert	← → ←	M1	Hashwert HASH berechnen, optional unter Verwendung der ZKA-SIG-API-Funktion *zka_sig_hash*
R2a	Signatur	→	M2a	Sicherheitsdienst Non-Repudiation: Aufruf der ZKA-SIG-API-Funktion *zka_sig_digital_signature* **oder:**
R2b	Signatur	← →	M2b	Sicherheitsdienst Authentication: Aufruf der ZKA-SIG-API-Funktion *zka_sig_cs_authentication*

♦ **Erläuterung**

1. Die Berechnung des Hashwertes erfolgt in der Regel außerhalb der Chipkarte (Hashalgorithmus gemäß Vorgabe für den Sicherheitsdienst bzw. vom Institut übermittelter BPD). Optional ist es auch möglich, den letzen Schritt oder alle Schritte der Hashwert-Berechnung durch die Chipkarte durchführen zu lassen. Diese Berechnung ist dann Bestandteil des Ablaufs der ZKA-SIG-API-Funktion *zka_sig_hash*. Der zu verwendende Hash-Algorithmus wird dabei in Form der zugehörigen OID übergeben:
   - OID = 1.3.14.3.2.26 für SHA-1
   - OID = 1.3.36.3.2.1 für RIPEMD-160
   - OID = 2.16.840.1.101.3.4.2.1 für SHA-256

2a. Bei Verwendung des Schlüssels S_K.CH.DS (Sicherheitsdienst Non-Repudiation) wird die Signatur durch Aufruf der ZKA-SIG-API-Funktion *zka_sig_digital_signature* erzeugt. Die Auswahl des Signaturalgorithmus und Paddingverfahrens erfolgt gemäß Vorgabe für den Sicherheitsdienst bzw. vom Institut übermittelter BPD. Die Signaturanwendung der Chipkarte bietet die Verfahren „sha-1-WithRSAEncryption“ (PKCS#1-Signaturverfahren, Standard-RSA, SHA-1) und „sigS_ISO9796-2rndWithripemd160“ (DIN-Signaturverfahren, Standard-RSA, RIPEMD-160) an. Zusätzlich bieten Banken-Signaturkarten mit SECCOS 6 auch das Signaturverfahren RSASSA-PSS an.

Falls der Hashwert im vorangegangenen Schritt 1 durch die Chipkarte berechnet wurde, ist er noch in der Chipkarte gespeichert und braucht nicht erneut als Parameter des *zka_sig_digital_signature* übergeben zu werden.

2b. Bei Verwendung des Schlüssels S_K.CH.AUT$_{C/S}$ (Sicherheitsdienst Authentication) wird die Signatur durch Aufruf der ZKA-SIG-API-Funktion *zka_sig_cs_authentication* erzeugt. Die Chipkarte verwendet dabei intern ein Paddingformat gemäß PKCS#1 ([SECCOS, Kapitel 8.3.2.1][12]), wobei die Digest-Info nicht von der Chipkarte selbst erzeugt wird, sondern als aufbereiteter „Authentication-Input“ (= zu signierendes Datenfeld) übergeben werden muss.

Der Authentication-Input ist wie folgt aufgebaut ([SECCOS, Kapitel 8.1.8.3.1]):

Tag	Länge	Wert	Erläuterung
'30'	'21' bzw. '31'		Tag und Länge von SEQUENCE (SHA-1/RIPEMD-160 bzw. SHA-256)
'30'	'09' bzw. '0D'		Tag und Länge von SEQUENCE (SHA-1/RIPEMD-160 bzw. SHA-256)
'06'	'05' bzw. '09'	'2B 0E 03 02 1A' bzw. '2B 24 03 02 01' bzw. '60 86 48 01 65 03 04 02 01'	OID des SHA-1 (1 3 14 3 2 26) bzw. OID des RIPEMD-160 (1 3 36 3 2 1) bzw. OID des SHA-256 (2 16 840 1 101 3 4 2 1)
'05'	'00'	-	TLV-Kodierung von NULL
'04'	'14'	'XX..XX'	Hash-Wert

Anmerkung: Die direkte Weiterverwendung eines eventuell im Chip berechneten und dort zwischengespeicherten Hashwerts ist bei der Signatur im Sicherheitsdienst „Authentication“ nicht möglich. Der Hashwert (als Ergebnis von Schritt 1) muss daher explizit als Aufrufparameter in der oben beschriebenen Form in Schritt 2 übergeben werden.

[12] Auszug aus [SECCOS, Kapitel 8.3.2.1]: Falls der Authentication Input nicht zu lang ist, wird er zu einer Folge von N-1 Byte wie folgt formatiert:

Bezeichnung	Byte-Länge	Wert
Blocktyp	1	'01'
Paddingfeld (PS)	N-3-L	'FF...FF'
Separator	1	'00'
Datenfeld	L	Authentication Input (AI)

C.1.3.3.2 Signatur-Prüfung

Die ZKA-Chipkarte selbst unterstützt zurzeit keine Signatur-Prüfung[13]. Die Prüfung einer Signatur wird vom Kundenterminal-Makro “Überprüfen der Korrektheit der elektronischen Unterschrift” durchgeführt.

Die (mathematische) Korrektheit einer elektronischen Unterschrift wird überprüft, in dem sie mit dem entsprechenden öffentlichen Schlüssel entschlüsselt wird und das Ergebnis mit dem Hashwert über die signierten Daten verglichen wird. Der für die Überprüfung der elektronischen Signatur eingesetzte öffentliche Schlüssel liegt in dem Kundenterminal authentisch vor, falls die zu ihm gehörende Zertifikatshierarchie vorher ebenfalls in dem Kundenterminal überprüft wurde [KT-KONZEPT].

[13] [ZKASIG, Kapitel 1.1]: „Die ZKA-Chipkarte unterstützt [die] Signaturprüfung zur Zeit aus dem folgenden Grund nicht: Die Prüfung digitaler Signaturen, die mit beliebigen privaten Schlüsseln und/oder Algorithmen berechnet sind, würde voraussetzen, dass die Chipkarte X.509-Zertifikate auswertet. Dies ist gemäß Kapitel 16.1 von [DINSIG] zur Zeit nicht möglich.“

C.2 Chipapplikation für DDV

Im Folgenden wird für das in Kap. B beschriebene DDV-Verfahren eine entsprechende Chipanwendung namens „Banking“ synonym „HBCI-Banking“ spezifiziert. Voraussetzung ist neben den nachfolgend beschriebenen Datenelementen zusätzlich das Vorhandensein des Datenelements EF_ID sowie des Kryptoalgorithmus Triple-DES, wie sie in der „Schnittstellenspezifikation für die ec-Karte mit Chip“ vom ZKA festgelegt wurden. Die Spezifikation bezieht sich allein auf die für HBCI erforderlichen Datenelemente.

Die Anwendung „Banking“ kann auf einer dedizierten Chipkarte („HBCI-Karte“) oder auf beliebigen multifunktionalen Chipkarten implementiert werden, sofern sie das Betriebssystem der ec-Karte mit Chip einsetzen. Für die HBCI-Anwendung ist kein ausführbarer Code über die Spezifikationen in ISO 7816-4 bzw. der ec-Karte mit Chip hinaus erforderlich.

In diesem Kapitel werden die Datenstrukturen und Zugriffsregeln der Chipapplikation ”DF_BANKING_20” für Chipkarten vom Typ 1 („altes ZKA-Betriebssystem”) und Typ SECCOS 6 („neues ZKA-Betriebssystem”) spezifiziert. Die Kommandoabläufe im Terminal sind gemeinsam für Chipkarten vom Typ 1[14] und SECCOS 6 spezifiziert.

In Kap. C.2.1 wird explizit auf die Beschreibung für Typ 1 eingegangen. Im weiteren Verlauf dieses Dokuments ist mit ”*HBCI-Chipkarte*” eine Chipkarte mit neuem ZKA-Betriebssystem gemäß [DATKOM] und [DAT-MF] gemeint, die die HBCI-Applikation enthält. Weitere Applikationen, wie z.B. die elektronische Geldbörse, sind nicht notwendigerweise auf der Chipkarte enthalten. Ebenso kann die Bankensignaturkarte mit oder ohne Zertifikat ausgeliefert werden.

Das ADF der Applikation HBCI-Banking wird mit DF_BANKING_20 bezeichnet. In der vorliegenden Spezifikation ist es direkt im MF enthalten. Die für die Applikation relevanten DF-spezifischen Schlüssel sind im EF_KEY abgelegt, das direkt im DF_BANKING_20 enthalten ist.

In der vorliegenden Spezifikation werden im Kontext von Typ 1-Karten zwei Security-Environments verwendet:

1 Das Security-Environment mit der Nummer 1 (SE #1) als Standard-SE legt die Zugriffsregeln für die Dateien der Applikation HBCI-Banking für den Anwendungsfall, d.h. für den Zugriff im Feld an HBCI-fähigen Terminals fest.

2 Das Security-Environment mit der Nummer 2 (SE #2) als Administrations-SE legt die Zugriffsregeln für die Dateien und das Applikationsverzeichnis der Applikation HBCI-Banking für den Fall von Administrationsvorgängen, z.B. Kontrolle, Änderungen oder Erweiterungen, fest.

Die Selektion von SEs erfolgt, wie in [DATKOM] beschrieben, mit dem Kommando MANAGE SECURITY ENVIRONMENT. Für den Anwendungsfall, d.h. an HBCI-fähigen Terminals, ist eine Selektion des SE nicht notwendig, da mit der Selektion einer Applikation implizit das SE #1 aktiviert wird.

[14] In den Abläufen befinden sich an wenigen Stellen aus Gründen der Vollständigkeit noch Verweise auf das Vorläuferbetriebssystem für Chipkarten vom Typ 0.

C.2.1 Daten der Applikation HBCI-Banking für Typ 1

Die folgende Grafik gibt eine Übersicht über die Dateien einer HBCI-Karte mit der Applikation HBCI-Banking für Typ 1.

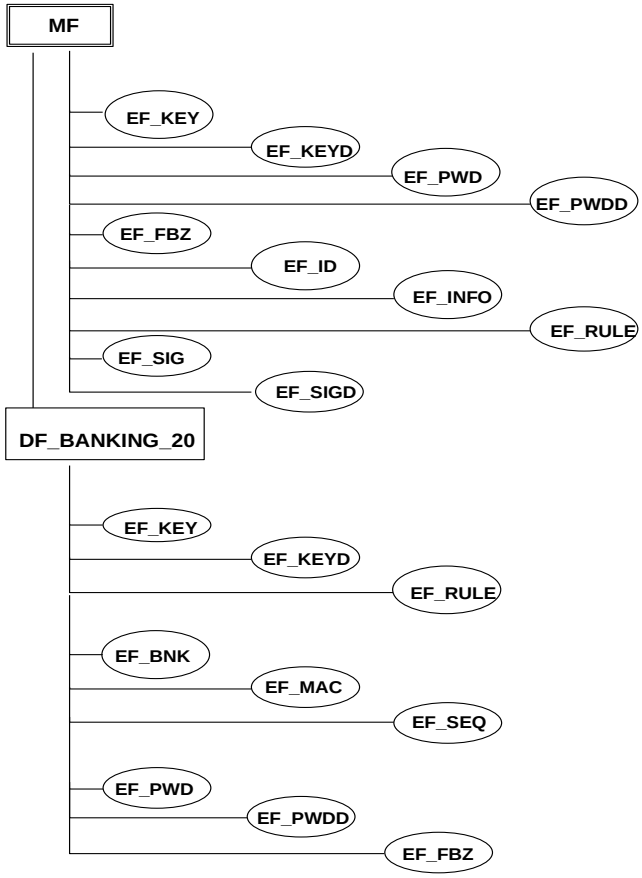


Abbildung 19: Datenelemente der Applikation "HBCI", Bankensignaturkarte mit Zertifikat

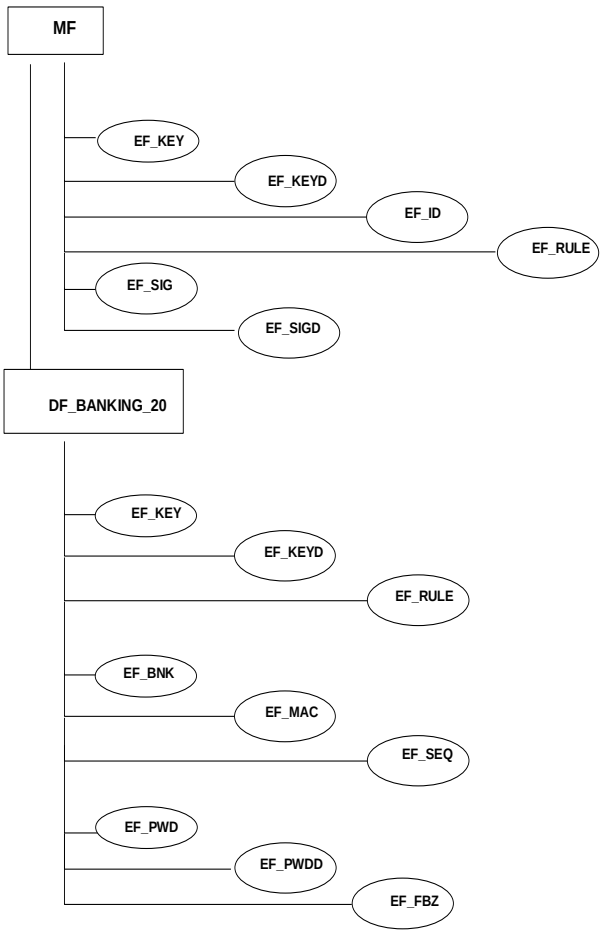


Abbildung 20: Datenelemente der Applikation "HBCI", Bankensignaturkarte ohne Zertifikat

C.2.1.1 ADF der Applikation HBCI-Banking

Für das ADF der Applikation HBCI-Banking (DF_BANKING_20) sind beim Anlegen die folgenden FCP festzulegen:

Tag	Länge	Wert	Erläuterung
'62'	'1A'		Tag und Länge für FCP
'82'	'01'	'38'	Datei-Deskriptor für DF
'83'	'02'	'A6 00'	Datei-ID des DF_BANKING_20
'84'	'09'	'D2 76 00 00 25 48 42 02 00'	DF-Name (AID) des DF_BANKING_20
'A1'	'06'	'8B 04 00 30 02 01'	Zugriffsregel-Referenzen

Der DF-Name (die AID) des DF_BANKING_20 bestehend aus der nationalen RID des ZKA ('D2 76 00 00 25'), der ASCII-kodierten Kennung "HB" ('48 42') sowie der Version der Applikation 2.0 ('02 00').

Die Zugriffsregeln für das DF_BANKING_20 stehen in der zugeordneten Regeldatei EF_RULE. Durch die Zugriffsregeln werden für die DF-spezifischen Kommandos die folgenden Festlegungen getroffen:

Wenn das DF_BANKING_20 selektiert ist, darf ein CREATE FILE (EF), DELETE FILE (self), INCLUDE oder EXCLUDE nur ausgeführt werden, wenn die Kommandonachricht mit Secure Messaging ausgeführt wird und mit einem korrekten MAC versehen ist, der unter Verwendung des Schlüssels K_{HBCI_Admin} aus dem EF_KEY des DF_BANKING_20 gebildet ist. Der Returncode wird für jedes dieser Kommandos durch die Karte mit einem MAC mit dem Schlüssel K_{HBCI_Admin} versehen. Die Kommandos CREATE FILE (DF) und DELETE FILE (child DF) dürfen nie ausgeführt werden. Alle zulässigen Administrationskommandos dürfen nur im SE #2 ausgeführt werden (Zugriffsregeln im Record 1 des EF_RULE).

Der Applikation HBCI-Banking sind 10 Dateien als AEF zuzuordnen:

SFI '01': EF_RULE im DF_BANKING_20

SFI '02': EF_KEY im DF_BANKING_20,

SFI '03': EF_PWD im DF_BANKING_20,

SFI '04': EF_PWDD im DF_BANKING_20,

SFI '05': EF_FBZ im DF_BANKING_20,

SFI '19': EF_ID im MF,

SFI '1A': EF_BNK im DF_BANKING_20,

SFI '1B': EF_MAC im DF_BANKING_20,

SFI '1C': EF_SEQ im DF_BANKING_20,

SFI '1E': EF_KEYD im DF_BANKING_20.

Wenn das DF_BANKING_20 mittels SELECT FILE selektiert wird und die entsprechende Option im Parameterbyte P2 des Kommandos gesetzt ist, wird die folgende FCI ausgegeben:

Tag	Länge	Wert	Erläuterung
'6F'	'0D'		Tag und Länge für FCI
'84'	'09'	'D2 76 00 00 25 48 42 02 00'	DF-Name (AID) des DF_BANKING_20
'A5'	'00'		keine proprietären Informationen

Wird das DF_BANKING_20 mittels SELECT FILE selektiert und die entsprechende Option im Parameterbyte P2 des Kommandos gesetzt, werden die folgenden FMD mit den Pfaden der AEFs ausgegeben (vorausgesetzt, das DF_BANKING_20 befindet sich direkt im MF):

Tag	Länge	Wert	Erläuterung
'64'	'44'		Tag und Länge für FMD
'85'	'03'	'C8 00 03'	Pfad für AEF mit SFI '19' (EF_ID im MF)
'85'	'05'	'08 A6 00 00 30'	Pfad für AEF mit SFI '01' (EF_RULE im DF_BANKING_20)
'85'	'05'	'10 A6 00 00 10'	Pfad für AEF mit SFI '02' (EF_KEY im DF_BANKING_20)
'85'	'05'	'18 A6 00 00 12'	Pfad für AEF mit SFI '03' (EF_PWD im DF_BANKING_20)
'85'	'05'	'20 A6 00 00 15'	Pfad für AEF mit SFI '04' (EF_PWDD im DF_BANKING_20)
'85'	'05'	'28 A6 00 00 16'	Pfad für AEF mit SFI '05' (EF_FBZ im DF_BANKING_20)
'85'	'05'	'D0 A6 00 03 01'	Pfad für AEF mit SFI '1A' (EF_BNK im DF_BANKING_20)
'85'	'05'	'D8 A6 00 03 02'	Pfad für AEF mit SFI '1B' (EF_MAC im DF_BANKING_20)
'85'	'05'	'E0 A6 00 03 03'	Pfad für AEF mit SFI '1C' (EF_SEQ im DF_BANKING_20)
'85'	'05'	'F0 A6 00 00 13'	Pfad für AEF mit SFI '1E' (EF_KEYD im DF_BANKING_20)

C.2.1.2 EF_RULE

- **Beschreibung**

  Die Datei EF_RULE enthält die Zugriffsregeln für die Applikation DF_BANKING_20. In den FCP von Dateien und Verzeichnissen wird auf diese Zugriffsregeln referenziert.

- **Format**

  Für das EF_RULE des DF_BANKING_20 sind die folgenden FCP festzulegen:

Tag	Länge	Wert	Erläuterung
'62'	'1C'		Tag und Länge für FCP
'82'	'05'	'14 41 00 24 08'	Datei-Deskriptor für lineares EF mit variabler Recordlänge (max. 36 Byte), 8 Records
'83'	'02'	'00 30'	Datei-ID des EF_RULE
'85'	'02'	'00 7D'	für Nutzdaten allokierter Speicherplatz in Byte
'88'	'01'	'08'	SFI '01' für das EF_RULE
'A1'	'08'	'8B 06 00 30 01 02 02 03'	Zugriffsregel-Referenzen

  Im SE #1 dürfen nur die Kommandos READ / SEARCH RECORD mit ungesicherter Kommando und Antwortnachricht ausgeführt werden (Zugriffsregel im Record 2 des EF_RULE).

  Im SE #2 darf APPEND RECORD nur ausgeführt werden, wenn es mit Secure Messaging ausgeführt wird. Die MAC-Bildung erfolgt für Kommando- und Antwortnachricht mit dem K_{HBCI_Admin}. UPDATE RECORD darf nie ausgeführt werden (Zugriffsregel im Record 3 des EF_RULE).

- **Daten**

  Das EF_RULE im DF_BANKING_20 enthält 8 Records mit den Zugriffsregeln für das Verzeichnis und die Datenfelder des Verzeichnisses.

  Die folgende Tabelle zeigt die Belegung dieser Records für eine HBCI-Chipkarte:

Rec.Nr.	Record-Inhalt	Byte
1	'80 01 DA B4 05 83 03 80 01 FF'	10
2	'80 01 81 90 00'	5
3	'80 01 84 B4 05 83 03 80 01 FF'	10
4	'80 01 86 AF 11 B4 05 83 03 80 01 FF B8 08 95 01 10 83 03 80 01 FF'	22
5	'80 01 86 B4 05 83 03 80 01 FF'	10
6	'80 01 82 A4 07 95 01 08 83 02 80 01 80 01 81 90 00'	17
7	'80 01 82 A4 07 95 01 08 83 02 80 01 80 01 81 AF 13 B4 08 95 01 20 83 03 80 02 FF A4 07 95 01 08 83 02 80 01'	36
8	'80 01 83 90 00 80 01 84 B4 05 83 03 80 01 FF'	15

  Die Records 1 bis 5 enthalten jeweils eine, die Records 6 bis 8 jeweils zwei Zugriffsregeln.

  Im folgenden werden die einzelnen Records des EF_RULE näher erläutert.

  Record 1 wird referenziert als Zugriffsregel von DF_BANKING_20 in SE #2.

CREATE FILE (EF), DELETE FILE (self), INCLUDE, EXCLUDE: MAC-SM-AC für Kommando- und Antwortnachricht mit K_{HBCI_Admin}:

Tag	Länge	Wert	Erläuterung
'80'	'01'	'DA'	Zugriffsart für CREATE FILE (EF), DELETE FILE (self), INCLUDE, EXCLUDE
'B4'	'05'		CCT - Tag und Länge
'83'	'03'	'80 01 FF'	Schlüsselreferenz für K_{HBCI_Admin}

Record 2 wird referenziert als Zugriffsregel von EF_RULE, EF_KEYD, EF_PWDD und EF_FBZ in SE #1.

READ / SEARCH RECORD: ALW

Tag	Länge	Wert	Erläuterung
'80'	'01'	'81'	Zugriffsart für READ / SEARCH RECORD
'90'	'00'		Zugriffsbedingung ALW

Record 3 wird referenziert als Zugriffsregel von EF_RULE, EF_BNK und EF_MAC in SE #2.

APPEND RECORD: MAC-SM-AC für Kommando- und Antwortnachricht mit dem Schlüssel K_{HBCI_Admin}.

Tag	Länge	Wert	Erläuterung
'80'	'01'	'84'	Zugriffsart für APPEND RECORD
'B4'	'05'		CCT - Tag und Länge
'83'	'03'	'80 01 FF'	Schlüsselreferenz für K_{HBCI_Admin}

Record 4 wird referenziert als Zugriffsregel von EF_KEY und EF_PWD in SE #2.

APPEND RECORD, UPDATE RECORD: MAC-ENC-SM-AC für Kommandonachricht und MAC-SM-AC für Antwortnachricht mit K_{HBCI_Admin}.

Tag	Länge	Wert	Erläuterung
'80'	'01'	'86'	Zugriffsart für APPEND RECORD, UPDATE RECORD
'AF'	'11'		AND- Template, Tag und Länge
'B4'	'05'		CCT - Tag und Länge
'83'	'03'	'80 01 FF'	Schlüsselreferenz für K_{HBCI_Admin}
'B8'	'08'		CT - Tag und Länge
'95'	'01'	'10'	Usage Qualifier: Nur für Kommandonachricht
'83'	'03'	'80 01 FF'	Schlüsselreferenz für K_{HBCI_Admin}

Record 5 wird referenziert als Zugriffsregel von EF_KEYD, EF_SEQ, EF_PWDD und EF_FBZ in SE #2.

APPEND RECORD, UPDATE RECORD: MAC-SM-AC für Kommando- und Antwortnachricht mit dem Schlüssel K_{HBCI_Admin}.

Tag	Länge	Wert	Erläuterung
'80'	'01'	'86'	Zugriffsart für APPEND RECORD, UPDATE RECORD
'B4'	'05'		CCT - Tag und Länge
'83'	'03'	'80 01 FF'	Schlüsselreferenz für K_{HBCI_Admin}

Record 6 wird referenziert als Zugriffsregel von EF_BNK und EF_SEQ in SE #1.

UPDATE RECORD: Karteninhaber-Authentikation (PWD) mit lokalem Passwort 1.

READ / SEARCH RECORD: ALW

Tag	Länge	Wert	Erläuterung
'80'	'01'	'82'	Zugriffsart für UPDATE RECORD
'A4'	'07'		AT - Tag und Länge
'95'	'01'	'08'	Usage Qualifier für Karteninhaber-Authentikation
'83'	'02'	'80 01'	Passwort-Referenz, lokales Passwort mit der Nummer 1
'80'	'01'	'81'	Zugriffsart für READ / SEARCH RECORD
'90'	'00'		ALW

Record 7 wird referenziert als Zugriffsregel von EF_MAC in SE #1.

UPDATE RECORD: Karteninhaber-Authentikation (PWD) mit lokalem Passwort 1.

READ / SEARCH RECORD: Karteninhaber-Authentikation (PWD) mit lokalem Passwort 1 und MAC-SM-AC für die Antwortnachricht mit dem Schlüssel K_{DAK}.

Tag	Länge	Wert	Erläuterung
'80'	'01'	'82'	Zugriffsart für UPDATE RECORD
'A4'	'07'		AT - Tag und Länge
'95'	'01'	'08'	Usage Qualifier für Karteninhaber-Authentikation
'83'	'02'	'80 01'	Passwort-Referenz, lokales Passwort mit der Nummer 1
'80'	'01'	'81'	Zugriffsart für READ / SEARCH RECORD
'AF'	'13'		AND - Template, Tag und Länge
'B4'	'08'		CCT - Tag und Länge
'95'	'01'	'20'	Usage Qualifier: Nur Antwortnachricht
'83'	'03'	'80 02 FF'	Schlüsselreferenz für K_{DAK}
'A4'	'07'		AT - Tag und Länge
'95'	'01'	'08'	Usage Qualifier für Karteninhaber-Authentikation
'83'	'02'	'80 01'	Passwort-Referenz, lokales Passwort mit der Nummer 1

Record 8 wird referenziert als Zugriffsregel im EF_PWDD.

VERIFY, CHANGE REFERENCE DATA: ALW

RESET RETRY COUNTER: MAC-SM-AC für Kommando- und Antwortnachricht mit K_{HBCI_Admin}

Tag	Länge	Wert	Beschreibung
'80'	'01'	'83'	Zugriffsart für VERIFY, CHANGE REFERENCE DATA
'90'	'00'		ALW
'80'	'01'	'84'	Zugriffsart für Kommando: RESET RETRY COUNTER
'B4'	'05'		CCT - Tag und Länge
'83'	'03'	'80 01 FF'	Schlüsselreferenz für K_{HBCI_Admin}

C.2.1.3 EF_KEY

♦ **Beschreibung**

Die applikationsspezifischen Schlüssel der Applikation HBCI-Banking sind im EF_KEY des Applikationsverzeichnisses DF_BANKING_20 gespeichert. Dies sind

- ein 16 Byte langer kartenindividueller Schlüssel K_{HBCI_Admin} mit der Schlüsselnummer '01' zur Administration der Applikation DF_BANKING_20,
- ein 16 Byte langer kartenindividueller Schlüssel K_{DAK} mit der Schlüsselnummer '02' als kundenindividueller Daten-Authentikationsschlüssel (DAK = Data Authentication Key)[15], sowie
- ein 16 Byte langer kartenindividueller Schlüssel K_{ENC} mit der Schlüsselnummer '03' als kundenindividueller Chiffrierschlüssel.

Die Schlüssel KHBCI_Admin, KDAK und KENC sind nur der HBCI-Chipkarte und dem für sie zuständigen Hintergrundsystem bekannt. Sie werden jeweils aus einem KGK (Key Generating Key) unter Verwendung der Kartenidentifikationsdaten im EF_ID des MF abgeleitet (vgl. Kapitel 8.4.1 von [DATKOM]). Das zuständige Hintergrundsystem kennt die jeweiligen KGK und leitet die kartenindividuellen Schlüssel bei Bedarf ab.

Es können pro logischer Schlüsselnummer verschiedene KGK verwendet werden. Ein KGK wird wie alle daraus abgeleiteten Schlüssel anhand der Schlüsselversion identifiziert. Die Schlüsselversion zur jeweiligen logischen Schlüsselnummer im zugehörigen EF_KEYD zeigt an, aus welchem KGK der jeweilige kartenindividuelle Schlüssel abgeleitet ist.

♦ **Format**

Für das EF_KEY des DF_BANKING_20 sind die folgenden FCP festzulegen:

Tag	Länge	Wert	Erläuterung
'62'	'16'		Tag und Länge für FCP
'82'	'05'	'12 41 00 12 03'	Datei-Deskriptor für lineares EF mit fester Recordlänge (18 Byte), 3 Records
'83'	'02'	'00 10'	Datei-ID des EF_KEY
'88'	'01'	'10'	SFI '02' für das EF_KEY
'A1'	'06'	'8B 04 00 30 02 04'	Zugriffsregel-Referenzen

Auf das EF_KEY darf nur im SE #2 zugegriffen werden.

Die Kommandos APPEND RECORD und UPDATE RECORD dürfen nur ausgeführt werden, wenn sie mit Secure Messaging durchgeführt werden, der Record-Inhalt verschlüsselt (ENC) ist und die Kommandonachricht mit einem MAC abgesichert ist. Verschlüsselung und MAC-Bildung erfolgen mit dem K_{HBCI_Admin}. Der Returncode eines APPEND RECORD oder UPDATE RECORD wird mit dem K_{HBCI_Admin} MAC-gesichert. Das Kommando READ RECORD darf nie ausgeführt werden. (Zugriffsregel im Record 4 des EF_RULE)

---

[15] Um den Begriff „Signierschlüssel“ für Anwendungen nach SigG bzw. EU-Richtlinie freizuhalten, wurde hier der Begriff „Daten-Authentikationsschlüssel“ gewählt. Im weiteren Text wird jedoch zur besseren Lesbarkeit weiterhin davon gesprochen, dass eine Nachricht mit diesem Schlüssel signiert wird.

♦ **Daten**

Das EF_KEY im DF_BANKING_20 enthält 3 Records mit den DF-spezifischen Schlüsseln des DF_BANKING_20.

Logische Schlüsselnummer	Schlüssel-Version	Schlüssel
'01'	'XX'	16 Byte langer K_{HBCI_Admin}
'02'	'XX'	16 Byte langer K_{DAK}
'03'	'XX'	16 Byte langer K_{ENC}

Es werden die Schlüsselversionen 1 bis 127 verwendet.

C.2.1.4 EF_KEYD

- **Beschreibung**

  Das EF_KEYD im DF_BANKING_20 enthält die Zusatzinformationen zu den DF-spezifischen Schlüsseln des DF_BANKING_20.

- **Format**

  Für das EF_KEYD sind die folgenden FCP festzulegen:

Tag	Länge	Wert	Erläuterung
'62'	'1C'		Tag und Länge für FCP
'82'	'05'	'14 41 00 1C 03'	Datei-Deskriptor für lineares EF mit variabler Recordlänge (max. 28 Byte) und 3 Records
'83'	'02'	'00 13'	Datei-ID des EF_KEYD
'85'	'02'	'00 48'	für Nutzdaten allokierter Speicherplatz in Byte
'88'	'01'	'F0'	SFI '1E' für das EF_KEYD
'A1'	'08'	'8B 06 00 30 01 02 02 05'	Zugriffsregel-Referenzen

  Im SE #1 dürfen nur die Kommandos READ / SEARCH RECORD mit ungesicherter Kommando und Antwortnachricht ausgeführt werden (Zugriffsregel im Record 2 des EF_RULE).

  Im SE #2 dürfen die Kommandos APPEND RECORD und UPDATE RECORD nur ausgeführt werden, wenn sie mit Secure Messaging durchgeführt werden. Die MAC-Bildung erfolgt für Kommando- und Antwortnachricht mit dem K_{HBCI_Admin} (Zugriffsregel im Record 5 des EF_RULE).

- **Daten**

  Das EF_KEYD enthält 3 Records, die die Zusatzinformation zu den DF-spezifischen Schlüsseln des DF_BANKING_20 enthalten.

  Das Datenobjekt mit Tag '93' enthält im Wertfeld als zweites Byte die Version des entsprechenden Schlüssels.

  Im folgenden wird der Aufbau der Schlüsselzusatzinformation dargestellt:

  Eintrag 1 (K_{HBCI_Admin}):

Tag	Länge	Wert	Erläuterung
'93'	'02'	'01 XX'	Schlüsselnummer und Schlüssel-Version
'C0'	'02'	'81 10'	Symmetrischer Schlüssel der Länge 16 Byte
'90'	'01'	'FF'	Fehlbedienungszähler
'7B'	'0F'		SE-Datenobjekt
'80'	'01'	'02'	Festlegung für SE #2
'B4'	'04'		CCT - Tag und Länge (Usage Qualifier '30' ist Defaultwert)
'89'	'02'	'12 22'	Algorithmus-ID: Schlüssel darf zur Bildung eines Retail-MAC im CFB-Mode verwendet werden
'B8'	'04'		CT - Tag und Länge (Usage Qualifier '10' ist Defaultwert)
'89'	'02'	'11 23'	Algorithmus-ID: Schlüssel darf zur Verschlüsselung als Triple-DES Schlüssel im CBC-Mode mit ICV ≠ 0 und ICV-Variante verwendet werden

Eintrag 2 (K_{DAK}):

Tag	Länge	Wert	Erläuterung
'93'	'02'	'02 XX'	Schlüsselnummer und Schlüssel-Version
'C0'	'02'	'81 10'	Symmetrischer Schlüssel der Länge 16 Byte
'7B'	'0C'		SE-Datenobjekt
'80'	'01'	'01'	Festlegung für SE #1
'B4'	'07'		CCT - Tag und Länge
'95'	'01'	'20'	Usage Qualifier: Nur SM-Antwortnachricht
'89'	'02'	'12 22'	Algorithmus-ID: Schlüssel darf zur Bildung eines Retail-MAC im CFB-Mode verwendet werden

Eintrag 3 (KENC):

Tag	Länge	Wert	Erläuterung
'93'	'02'	'03 XX'	Schlüsselnummer und Schlüssel-Version
'C0'	'02'	'81 10'	Symmetrischer Schlüssel der Länge 16 Byte
'7B'	'0C'		SE-Datenobjekt
'80'	'01'	'01'	Festlegung für SE #1
'A4'	'07'		AT - Tag und Länge
'95'	'01'	'40'	Usage Qualifier: Nur interne Authentikation
'89'	'02'	'21 12'	Algorithmus-ID: Schlüssel darf zur Authentikation der Chipkarte mit Triple-DES verwendet werden

C.2.1.5 EF_PWD

- **Beschreibung**

  Das lokale EF_PWD im DF_BANKING_20 enthält in dem 9 Byte langen Record '01' die Länge der HBCI-PIN und einen Referenzwert der HBCI-PIN der ZKA-Chipkarte. Die HBCI-PIN hat eine Mindestlänge von 5 Ziffern und darf maximal 12 Ziffern lang sein.

- **Format**

  Für das EF_PWD des DF_BANKING_20 sind die folgenden FCP festzulegen:

Tag	Länge	Wert	Erläuterung
'62'	'16'		
'82'	'05'	'12 41 00 09 01'	Datei-Deskriptor für lineares EF mit fester Recordlänge von 9 Byte
'83'	'02'	'00 12'	Datei-ID des EF_PWD
'88'	'01'	'18'	SFI '03' für das EF_PWD
'A1'	'06'	'8B 04 00 30 02 04'	Zugriffsregel-Referenz

  Auf das EF_PWD darf nur im SE #2 zugegriffen werden: Die Kommandos APPEND RECORD und UPDATE RECORD dürfen nur ausgeführt werden, wenn sie mit Secure Messaging durchgeführt werden, der Recordinhalt verschlüsselt (ENC) ist und die Kommandonachricht mit einem MAC abgesichert ist. Verschlüsselung und MAC-Bildung erfolgen dabei mit dem K_{HBCI_Admin}. Der Returncode eines APPEND RECORD oder UPDATE RECORD wird MAC-gesichert. Die MAC-Bildung erfolgt für die Antwortnachricht mit dem K_{HBCI_Admin}. Das Kommando READ RECORD darf nie ausgeführt werden (Zugriffsregel im Record 4 des EF_RULE).

- **Daten**

  Der Record '01' des EF_PWD enthält einen Referenzwert der HBCI-PIN.

Byte	Inhalt	Beschreibung
1	'05'	Länge der PIN
2 - 9	'XX..XX'	Referenzwert der PIN

  Zur Erzeugung des Referenzwertes wird aus der HBCI-PIN zunächst der 8 Byte lange 'Format 2 PIN Block' gemäß [ISO PIN1] wie folgt gebildet:

C	L	P	P	P	P	P	P/F	P/F	P/F	P/F	P/F	P/F	P/F	F	F

  Erläuterung:

  Jedes Feld repräsentiert ein Halbbyte.

C:	Kontroll-Feld, binär kodiert	hat immer den Wert '2'
L:	PIN-Länge, binär kodiert	mögliche Werte von '5' bis 'C'
P:	PIN-Ziffer, BCD-kodiert	
F:	Filler, binär kodiert	hat immer den Wert 'F'
P/F:	PIN-Ziffer/Filler	Belegung abhängig von der PIN-Länge

Der erzeugte Format 2 PIN Block wird mit PB bezeichnet. Aus diesem PIN Block wird der zu speichernde Referenzwert durch DES-Verschlüsselung mit sich selbst erzeugt:

PIN-Referenzwert: ePB(PB)

Falls erforderlich, wird vor der Verwendung von PB als DES-Schlüssel ein Parity Adjustment vorgenommen. PB wird als Klartext unverändert verwendet.

C.2.1.6 EF_PWDD

- **Beschreibung**

  Das EF_PWDD im DF_BANKING_20 enthält in Record '01' die Zusatzinformationen zu der im EF_PWD des DF_BANKING_20 abgelegten HBCI-PIN.

- **Format**

  Für das EF_PWDD sind die folgenden FCP festzulegen:

Tag	Länge	Wert	Erläuterung
'62'	'1C'		
'82'	'05'	'14 41 00 15 01'	Datei-Deskriptor für lineares EF mit variabler Recordlänge (max. 21 Byte) und 1 Record
'83'	'02'	'00 15'	Datei-ID des EF_PWDD
'85'	'02'	'00 15'	Für Nutzdaten allokierter Speicherplatz in Byte
'88'	'01'	'20'	SFI '04' für das EF_PWDD
'A1'	'08'	'8B 06 00 30 01 02 02 05'	Zugriffsregel-Referenzen

  Im SE #1 dürfen nur die Kommandos READ / SEARCH RECORD mit ungesicherter Kommando und Antwortnachricht ausgeführt werden (Zugriffsregel im Record 2 des EF_RULE).

  Im SE #2 dürfen APPEND RECORD und UPDATE RECORD nur ausgeführt werden, wenn sie mit Secure Messaging durchgeführt werden und die Kommandonachricht mit einem MAC abgesichert ist. Der Returncode wird MAC-gesichert. Die MAC-Bildung erfolgt für Kommando- und Antwortnachricht mit dem K_{HBCI_Admin} (Zugriffsregel im Record 5 des EF_RULE).

- **Daten**

  Das lokale EF_PWDD enthält in Record '01' einen 21 Byte langen Record, der die Zusatzinformationen zu der HBCI-PIN enthält.

Tag	Länge	Wert	Beschreibung
'93'	'02'	'01 01'	Passwortreferenz: Passwort '01' im Record '01' des EF_PWD
'89'	'02'	'11 50'	Speicherformat des Passwortes (minimal 5 Ziffern)
'7B'	'0B'		SE-DO, Tag und Länge
'80'	'01'	'00'	SE Referenz-DO: Für alle SEs
'A1'	'03'	'8B 01 08'	Zugriffsregel-Referenz
'89'	'01'	'12'	Übertragungsformat der Authentikationsdaten: PIN Format 2 Block

C.2.1.7 EF_FBZ

- **Beschreibung**

  EF_FBZ bezeichnet das lineare EF, das in Record '01' den Fehlbedienungszähler und den zugehörigen Initialwert für die im DF-spezifischen EF_PWD abgelegte HBCI-PIN enthält.

- **Format**

  Für das EF_FBZ im DF_BANKING_20 sind die folgenden FCP festzulegen:

Tag	Länge	Wert	Erläuterung
'62'	'18'		Tag und Länge für FCP
'82'	'05'	'12 41 00 02 01'	Datei-Deskriptor für lineares EF fester Recordlänge
'83'	'02'	'00 16'	Datei-ID des EF_FBZ
'88'	'01'	'28'	SFI '05' für das EF_FBZ
'A1'	'08'	'8B 06 00 30 01 02 02 05'	Zugriffsregel-Referenzen

  Im SE # 1 dürfen nur die Kommandos READ / SEARCH RECORD mit ungesicherter Kommando und Antwortnachricht ausgeführt werden (Zugriffsregel im Record 2 des EF_RULE).

  Im SE #2 dürfen die Kommandos APPEND RECORD und UPDATE RECORD nur ausgeführt werden, wenn sie mit Secure Messaging durchgeführt werden und die Kommandonachricht mit einem MAC abgesichert ist. Der Returncode wird MAC-gesichert. Die MAC-Bildung erfolgt für Kommando- und Antwortnachricht mit dem K_{HBCI_Admin} (Zugriffsregel im Record 5 des EF_RULE).

- **Daten**

  Das EF_FBZ enthält in Record '01' einen 2 Byte langen Record, der den Fehlbedienungszähler und den zugehörigen Initialwert '03' für die HBCI-PIN enthält.

Initialwert des FBZ	FBZ
'03'	'03'

C.2.1.8 EF_BNK

- **Beschreibung**

  Bei dem EF_BNK handelt es sich um ein lineares EF mit 5 Records in dem Bankverbindungen abgelegt sind.

- **Format**

  Für das EF_BNK in einer HBCI-Chipkarte sind die folgenden FCP festzulegen:

Tag	Länge	Wert	Erläuterung
'62'	'18'		Tag und Länge für FCP
'82'	'05'	'12 41 00 58 05'	Datei-Deskriptor für lineares EF mit fester Recordlänge 88 Byte und 5 Records
'83'	'02'	'03 01'	Datei-ID des EF_BNK
'88'	'01'	'D0'	SFI '1A' für das EF_BNK
'A1'	'08'	'8B 06 00 30 01 06 02 03'	Zugriffsregel-Referenzen

  Im SE #1 dürfen READ / SEARCH RECORD immer ausgeführt werden, die Antwortnachricht wird nicht abgesichert. UPDATE RECORD darf nur ausgeführt werden, wenn zuvor eine Karteninhaber-Authentikation mit dem lokalen Passwort 1 (HBCI-PIN) erfolgt ist. Der Returncode wird nicht MAC-gesichert (Zugriffsregeln im Record 6 des EF_RULE).

  Im SE #2 darf das Kommando APPEND RECORD nur ausgeführt werden, wenn es mit Secure Messaging durchgeführt wird. Die MAC-Bildung erfolgt für Kommando- und Antwortnachricht mit dem K_{HBCI_Admin} (Zugriffsregel im Record 3 des EF_RULE).

- **Daten**

  Die Records setzen sich aus einer Bankkurzbezeichnung, der Bankleitzahl, dem Kommunikationsdienst, der Adresse und dem Adresszusatz für den Kommunikationszugang, dem Länderkennzeichen und der Benutzerkennung zusammen.

Byte	Länge	Wert	Erläuterung
1-20	20	'aa .. aa'	Kurzbezeichner des Kreditinstituts
21-24	4	'nn nn nn nn'	Kreditinstitutscode des kontoführenden Instituts
25-25	1	'n'	Kommunikationsdienst
26-53	28	'aa .. aa'	Kommunikationsadresse
54-55	2	'aa aa'	Kommunikationsadressenzusatz
56-58	3	'aa aa aa'	Länderkennzeichen des kontoführenden Instituts
59-88	30	'aa .. aa'	Benutzerkennung

  Alphanumerische Feldinhalte ('a') werden ASCII-kodiert, linksbündig eingestellt und mit Leerzeichen (X'20') auf die vorgegebene Länge aufgefüllt. Numerische Feldinhalte ('n') werden BCD-kodiert.

C.2.1.9 EF_MAC

- **Beschreibung**

  Das EF_MAC wird für die MAC-Bildung über den Hashwert einer Nachricht benötigt. Es besteht aus einem 12 Byte langem Record deren Zugriffsregeln so gesetzt werden müssen, dass beim Lesen des Records der MAC produziert wird.

- **Format**

  Für das EF_MAC sind die folgenden FCP festzulegen:

Tag	Länge	Wert	Erläuterung
'62'	'18'		Tag und Länge für FCP
'82'	'05'	'12 41 00 0C 01'	Datei-Deskriptor für lineares EF mit einem Record der Länge 12 Byte
'83'	'02'	'03 02'	Datei-ID des EF_MAC
'88'	'01'	'D8'	SFI '1B' für das EF_MAC
'A1'	'08'	'8B 06 00 30 01 07 02 03'	Zugriffsregel-Referenzen

  Im SE #1 dürfen READ / SEARCH RECORD nach Karteninhaber-Authentikation ausgeführt werden, die Antwortnachricht wird mit einem K_{DAK}-MAC versehen. UPDATE RECORD darf nur ausgeführt werden, wenn zuvor eine Karteninhaber-Authentikation mit dem lokalen Passwort 1 (HBCI-PIN) erfolgt ist. Der Returncode eines UPDATE RECORD wird nicht MAC-gesichert (Zugriffsregeln im Record 7 des EF_RULE).

  Im SE #2 darf das Kommando APPEND RECORD nur ausgeführt werden, wenn es mit Secure Messaging durchgeführt wird. Die MAC-Bildung erfolgt für Kommando- und Antwortnachricht mit dem K_{HBCI_Admin} (Zugriffsregel im Record 3 des EF_RULE).

- **Daten**

  Das EF_MAC enthält einen Record, der den folgenden Aufbau hat:

Byte	Wert	Erläuterung
1-12	'XX..XX'	Hashwert

C.2.1.10 EF_SEQ

- **Beschreibung**

  Bei dem EF_SEQ handelt es sich um ein lineares EF, dessen Record ein 2 Byte langes binär definiertes Element enthält. Dieser binäre aufsteigende Zähler fließt als Sicherheitsreferenznummer (Signatur-ID) zur Absicherung der Daten gegen Doppeleinreichung ein. Der Startwert des Zählers ist 1. Ein Rücksetzen bei Überlauf findet nicht statt.

- **Format**

  Für das EF_SEQ sind die folgenden FCP festzulegen:

Tag	Länge	Wert	Erläuterung
'62'	'18'		Tag und Länge für FCP
'82'	'05'	'12 41 00 02 01'	Datei-Deskriptor für lineares EF mit 1 Record der Länge 2 Byte
'83'	'02'	'03 03'	Datei-ID des EF_SEQ
'88'	'01'	'E0'	SFI '1C' für das EF_SEQ
'A1'	'08'	'8B 06 00 30 01 06 02 05'	Zugriffsregel-Referenzen

  Im SE #1 dürfen READ / SEARCH RECORD immer ausgeführt werden, die Antwortnachricht wird nicht abgesichert. UPDATE RECORD darf nur ausgeführt werden, wenn zuvor eine Karteninhaber-Authentikation mit dem lokalen Passwort 1 (HBCI-PIN) erfolgt ist. Der Returncode wird nicht MAC-gesichert (Zugriffsregeln im Record 6 des EF_RULE).

  Im SE #2 dürfen die Kommandos APPEND RECORD und UPDATE RECORD nur ausgeführt werden, wenn sie mit Secure Messaging durchgeführt werden. Die MAC-Bildung erfolgt für Kommando- und Antwortnachrichten jeweils mit dem K_{HBCI_Admin} (Zugriffsregel im Record 5 des EF_RULE).

- **Daten**

  Das EF_SEQ enthält 1 Record, der den folgenden Aufbau hat:

Byte	Wert	Erläuterung
1-2	'XX XX'	Sequenznummer

C.2.2 Daten der Applikation HBCI-Banking für SECCOS 6

Die folgende Grafik gibt eine Übersicht über die Dateien einer HBCI-Karte mit der Applikation HBCI-Banking für Seccos 6.

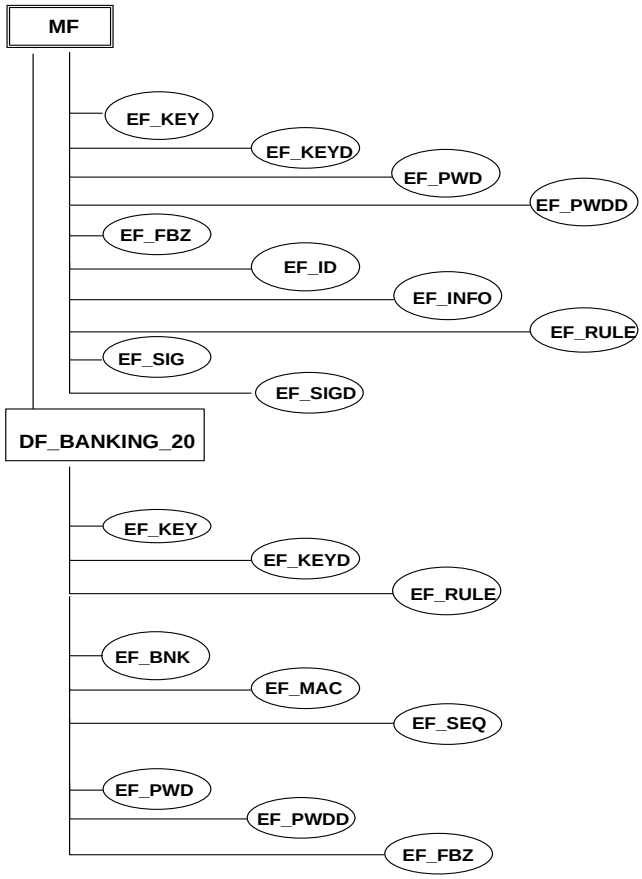


Abbildung 21: Datenelemente der Applikation "HBCI", Bankensignaturkarte mit Zertifikat

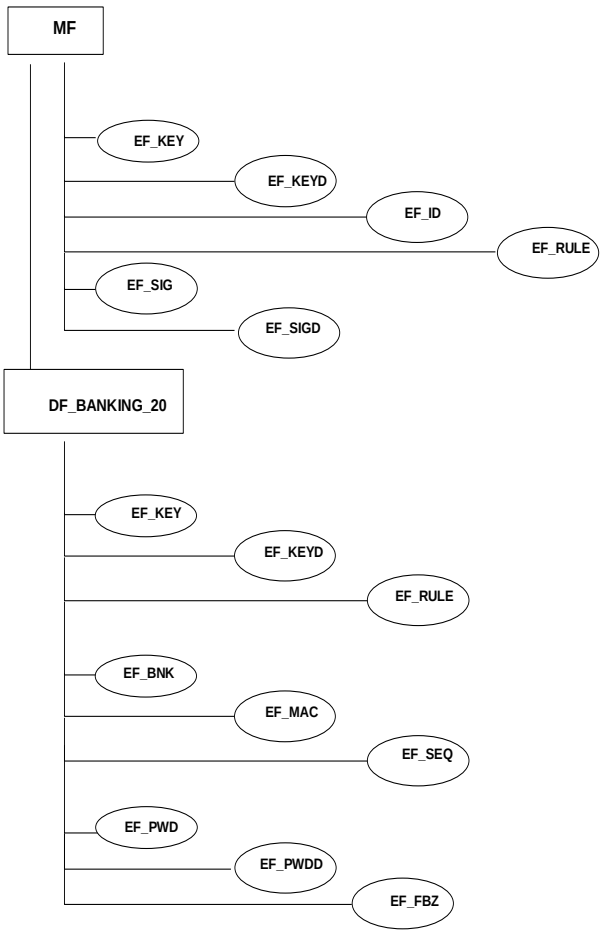


Abbildung 22: Datenelemente der Applikation "HBCI", Bankensignaturkarte ohne Zertifikat

C.2.2.1 ADF der Applikation HBCI-Banking

Für das ADF der Applikation HBCI-Banking (DF_BANKING_20) sind beim Anlegen die folgenden FCP festzulegen:

Tag	Länge	Wert	Erläuterung
'62'	'1A'		Tag und Länge für FCP
'82'	'01'	'38'	Datei-Deskriptor für DF
'83'	'02'	'A6 00'	Datei-ID des DF_BANKING_20
'84'	'09'	'D2 76 00 00 25 48 42 02 00'	DF-Name (AID) des DF_BANKING_20
'A1'	'06'	'8B 04 00 30 02 01'	Zugriffsregel-Referenzen
'A2'	'09'	'88 01 C8 51 04 3F 00 00 03'	SFI-Template: SFI '19' für das EF_ID im MF (Datei-ID: 00 03')

Der DF-Name (die AID) des DF_BANKING_20 bestehend aus der nationalen RID des ZKA ('D2 76 00 00 25'), der ASCII-kodierten Kennung "HB" ('48 42') sowie der Version der Applikation 2.0 ('02 00').

Die Zugriffsregeln für das DF_BANKING_20 stehen in der zugeordneten Regeldatei EF_RULE. Durch die Zugriffsregeln werden für die DF-spezifischen Kommandos die folgenden Festlegungen getroffen:

Wenn das DF_BANKING_20 selektiert ist, darf ein CREATE FILE (EF) oder ein DELETE FILE (self) nur ausgeführt werden, wenn die Kommandonachricht mit Secure Messaging ausgeführt wird und mit einem korrekten MAC versehen ist, der unter Verwendung des Schlüssels K_{HBCI_Admin} aus dem EF_KEY des DF_BANKING_20 gebildet ist. Der Returncode wird für jedes dieser Kommandos durch die Karte mit einem MAC mit dem Schlüssel K_{HBCI_Admin} versehen. Die Kommandos CREATE FILE (DF), DELETE FILE (child DF), ACTIVATE, DEACTIVATE und TERMINATE dürfen nie ausgeführt werden. Alle zulässigen Administrationskommandos dürfen nur im SE #2 ausgeführt werden (Zugriffsregeln im Record 1 des EF_RULE).

Der Applikation HBCI-Banking sind 10 Dateien als AEF zuzuordnen:

SFI '01': EF_RULE im DF_BANKING_20

SFI '02': EF_KEY im DF_BANKING_20,

SFI '03': EF_PWD im DF_BANKING_20,

SFI '04': EF_PWDD im DF_BANKING_20,

SFI '05': EF_FBZ im DF_BANKING_20,

SFI '19': EF_ID im MF,

SFI '1A': EF_BNK im DF_BANKING_20,

SFI '1B': EF_MAC im DF_BANKING_20,

SFI '1C': EF_SEQ im DF_BANKING_20,

SFI '1E': EF_KEYD im DF_BANKING_20.

Wenn das DF_BANKING_20 mittels SELECT FILE selektiert wird und die entsprechende Option im Parameterbyte P2 des Kommandos gesetzt ist, wird die folgende FCI ausgegeben:

Tag	Länge	Wert	Erläuterung
'6F'	'0D'		Tag und Länge für FCI
'84'	'09'	'D2 76 00 00 25 48 42 02 00'	DF-Name (AID) des DF_BANKING_20
'A5'	'00'		keine proprietären Informationen

Wird das DF_BANKING_20 mittels SELECT FILE selektiert und die entsprechende Option im Parameterbyte P2 des Kommandos gesetzt, werden die folgenden FMD mit den Pfaden der AEFs ausgegeben (vorausgesetzt, das DF_BANKING_20 befindet sich direkt im MF):

Tag	Länge	Wert	Erläuterung
'64'	'6E'		Tag und Länge für FMD
'A2'	'6C'		Tag und Länge SFI-Template
'88'	'01'	'08'	SFI '01' (EF_RULE im DF_BANKING_20)
'51'	'06'	'3F 00 A6 00 00 30'	Pfad für AEF mit SFI '01'
'88'	'01'	'10'	SFI '02' (EF_KEY im DF_BANKING_20)
'51'	'06'	'3F 00 A6 00 00 10'	Pfad für AEF mit SFI '02'
'85'	'01'	'18'	SFI '03' (EF_PWD im DF_BANKING_20)
'51'	'06'	'3F 00 A6 00 00 12'	Pfad für AEF mit SFI '03'
'85'	'01'	'20'	SFI '04' (EF_PWDD im DF_BANKING_20)
'51'	'06'	'3F 00 A6 00 00 15'	Pfad für AEF mit SFI '04'
'85'	'01'	'28'	SFI '05' (EF_FBZ im DF_BANKING_20)
'51'	'06'	'3F 00 A6 00 00 16'	Pfad für AEF mit SFI '05'
'88'	'01'	'C8'	SFI '19' (EF_ID im MF)
'51'	'04'	'3F 00 00 03'	Pfad für AEF mit SFI '19'
'85'	'01'	'D0'	SFI '1A' (EF_BNK im DF_BANKING_20)
'51'	'06'	'3F 00 A6 00 03 01'	Pfad für AEF mit SFI '1A'
'85'	'01'	'D8'	SFI '1B' (EF_MAC im DF_BANKING_20)
'51'	'06'	'3F 00 A6 00 03 02'	Pfad für AEF mit SFI '1B'
'85'	'01'	'E0'	SFI '1C' (EF_SEQ im DF_BANKING_20)
'51'	'06'	'3F 00 A6 00 03 03'	Pfad für AEF mit SFI '1C'
'85'	'01'	'F0'	SFI '1E' (EF_KEYD im DF_BANKING_20)
'51'	'06'	'3F 00 A6 00 00 13'	Pfad für AEF mit SFI '1E'

C.2.2.2 EF_RULE

- **Beschreibung**

  Die Datei EF_RULE enthält die Zugriffsregeln für die Applikation DF_BANKING_20. In den FCP von Dateien und Verzeichnissen wird auf diese Zugriffsregeln referenziert.

- **Format**

  Für das EF_RULE des DF_BANKING_20 sind die folgenden FCP festzulegen:

Tag	Länge	Wert	Erläuterung
'62'	'1C'		Tag und Länge für FCP
'82'	'05'	'14 41 00 24 08'	Datei-Deskriptor für lineares EF mit variabler Recordlänge (max. 36 Byte), 8 Records
'83'	'02'	'00 30'	Datei-ID des EF_RULE
'85'	'02'	'00 7D'	für Nutzdaten allokierter Speicherplatz in Byte
'88'	'01'	'08'	SFI '01' für das EF_RULE
'A1'	'08'	'8B 06 00 30 01 02 02 03'	Zugriffsregel-Referenzen

  Im SE #1 dürfen nur die Kommandos READ / SEARCH RECORD mit ungesicherter Kommando und Antwortnachricht ausgeführt werden (Zugriffsregel im Record 2 des EF_RULE).

  Im SE #2 darf APPEND RECORD nur ausgeführt werden, wenn es mit Secure Messaging ausgeführt wird. Die MAC-Bildung erfolgt für Kommando- und Antwortnachricht mit dem K_{HBCI_Admin}. UPDATE RECORD darf nie ausgeführt werden (Zugriffsregel im Record 3 des EF_RULE).

- **Daten**

  Das EF_RULE im DF_BANKING_20 enthält 8 Records mit den Zugriffsregeln für das Verzeichnis und die Datenfelder des Verzeichnisses.

  Die folgende Tabelle zeigt die Belegung dieser Records für eine HBCI-Chipkarte:

Rec.Nr.	Record-Inhalt	Byte
1	'80 01 42 B4 05 83 03 80 01 FF'	10
2	'80 01 01 90 00'	5
3	'80 01 04 B4 05 83 03 80 01 FF'	10
4	'80 01 06 AF 11 B4 05 83 03 80 01 FF B8 08 95 01 10 83 03 80 01 FF'	22
5	'80 01 06 B4 05 83 03 80 01 FF'	10
6	'80 01 02 A4 07 95 01 08 83 02 80 01 80 01 01 90 00'	17
7	'80 01 02 A4 07 95 01 08 83 02 80 01 80 01 01 AF 13 B4 08 95 01 20 83 03 80 02 FF A4 07 95 01 08 83 02 80 01'	36
8	'80 01 03 90 00 80 01 84 B4 05 83 03 80 01 FF'	15

  Die Records 1 bis 5 enthalten jeweils eine, die Records 6 bis 8 jeweils zwei Zugriffsregeln.

  Im folgenden werden die einzelnen Records des EF_RULE näher erläutert.

  Record 1 wird referenziert als Zugriffsregel von DF_BANKING_20 in SE #2.

CREATE FILE (EF), DELETE FILE (self): MAC-SM-AC für Kommando- und Antwortnachricht mit K_{HBCI_Admin}:

Tag	Länge	Wert	Erläuterung
'80'	'01'	'42'	Zugriffsart für CREATE FILE (EF), DELETE FILE (self)
'B4'	'05'		CCT - Tag und Länge
'83'	'03'	'80 01 FF'	Schlüsselreferenz für K_{HBCI_Admin}

Record 2 wird referenziert als Zugriffsregel von EF_RULE, EF_KEYD, EF_PWDD und EF_FBZ in SE #1.

READ / SEARCH RECORD: ALW

Tag	Länge	Wert	Erläuterung
'80'	'01'	'01'	Zugriffsart für READ / SEARCH RECORD
'90'	'00'		Zugriffsbedingung ALW

Record 3 wird referenziert als Zugriffsregel von EF_RULE, EF_BNK und EF_MAC in SE #2.

APPEND RECORD: MAC-SM-AC für Kommando- und Antwortnachricht mit dem Schlüssel K_{HBCI_Admin}.

Tag	Länge	Wert	Erläuterung
'80'	'01'	'04'	Zugriffsart für APPEND RECORD
'B4'	'05'		CCT - Tag und Länge
'83'	'03'	'80 01 FF'	Schlüsselreferenz für K_{HBCI_Admin}

Record 4 wird referenziert als Zugriffsregel von EF_KEY und EF_PWD in SE #2.

APPEND RECORD, UPDATE RECORD: MAC-ENC-SM-AC für Kommandonachricht und MAC-SM-AC für Antwortnachricht mit K_{HBCI_Admin}.

Tag	Länge	Wert	Erläuterung
'80'	'01'	'06'	Zugriffsart für APPEND RECORD, UPDATE RECORD
'AF'	'11'		AND- Template, Tag und Länge
'B4'	'05'		CCT - Tag und Länge
'83'	'03'	'80 01 FF'	Schlüsselreferenz für K_{HBCI_Admin}
'B8'	'08'		CT - Tag und Länge
'95'	'01'	'10'	Usage Qualifier: Nur für Kommandonachricht
'83'	'03'	'80 01 FF'	Schlüsselreferenz für K_{HBCI_Admin}

Record 5 wird referenziert als Zugriffsregel von EF_KEYD, EF_SEQ, EF_PWDD und EF_FBZ in SE #2.

APPEND RECORD, UPDATE RECORD: MAC-SM-AC für Kommando- und Antwortnachricht mit dem Schlüssel K_{HBCI_Admin}.

Tag	Länge	Wert	Erläuterung
'80'	'01'	'06'	Zugriffsart für APPEND RECORD, UPDATE RECORD
'B4'	'05'		CCT - Tag und Länge
'83'	'03'	'80 01 FF'	Schlüsselreferenz für K_{HBCI_Admin}

Record 6 wird referenziert als Zugriffsregel von EF_BNK und EF_SEQ in SE #1.

UPDATE RECORD: Karteninhaber-Authentikation (PWD) mit lokalem Passwort 1.

READ / SEARCH RECORD: ALW

Tag	Länge	Wert	Erläuterung
'80'	'01'	'02'	Zugriffsart für UPDATE RECORD
'A4'	'07'		AT - Tag und Länge
'95'	'01'	'08'	Usage Qualifier für Karteninhaber-Authentikation
'83'	'02'	'80 01'	Passwort-Referenz, lokales Passwort mit der Nummer 1
'80'	'01'	'01'	Zugriffsart für READ / SEARCH RECORD
'90'	'00'		ALW

Record 7 wird referenziert als Zugriffsregel von EF_MAC in SE #1.

UPDATE RECORD: Karteninhaber-Authentikation (PWD) mit lokalem Passwort 1.

READ / SEARCH RECORD: Karteninhaber-Authentikation (PWD) mit lokalem Passwort 1 und MAC-SM-AC für die Antwortnachricht mit dem Schlüssel K_{DAK}.

Tag	Länge	Wert	Erläuterung
'80'	'01'	'02'	Zugriffsart für UPDATE RECORD
'A4'	'07'		AT - Tag und Länge
'95'	'01'	'08'	Usage Qualifier für Karteninhaber-Authentikation
'83'	'02'	'80 01'	Passwort-Referenz, lokales Passwort mit der Nummer 1
'80'	'01'	'01'	Zugriffsart für READ / SEARCH RECORD
'AF'	'13'		AND - Template, Tag und Länge
'B4'	'08'		CCT - Tag und Länge
'95'	'01'	'20'	Usage Qualifier: Nur Antwortnachricht
'83'	'03'	'80 02 FF'	Schlüsselreferenz für K_{DAK}
'A4'	'07'		AT - Tag und Länge
'95'	'01'	'08'	Usage Qualifier für Karteninhaber-Authentikation
'83'	'02'	'80 01'	Passwort-Referenz, lokales Passwort mit der Nummer 1

Record 8 wird referenziert als Zugriffsregel im EF_PWDD.

VERIFY, CHANGE REFERENCE DATA: ALW

RESET RETRY COUNTER: MAC-SM-AC für Kommando- und Antwortnachricht mit K_{HBCI_Admin}

Tag	Länge	Wert	Beschreibung
'80'	'01'	'03'	Zugriffsart für VERIFY, CHANGE REFERENCE DATA
'90'	'00'		ALW
'80'	'01'	'04'	Zugriffsart für Kommando: RESET RETRY COUNTER
'B4'	'05'		CCT - Tag und Länge
'83'	'03'	'80 01 FF'	Schlüsselreferenz für K_{HBCI_Admin}

C.2.2.3 EF_KEY

♦ **Beschreibung**

Die applikationsspezifischen Schlüssel der Applikation HBCI-Banking sind im EF_KEY des Applikationsverzeichnisses DF_BANKING_20 gespeichert. Dies sind

- ein 16 Byte langer kartenindividueller Schlüssel K_{HBCI_Admin} mit der Schlüsselnummer '01' zur Administration der Applikation DF_BANKING_20,
- ein 16 Byte langer kartenindividueller Schlüssel K_{DAK} mit der Schlüsselnummer '02' als kundenindividueller Daten-Authentikationsschlüssel (DAK = Data Authentication Key)[16], sowie
- ein 16 Byte langer kartenindividueller Schlüssel K_{ENC} mit der Schlüsselnummer '03' als kundenindividueller Chiffrierschlüssel.

Die Schlüssel KHBCI_Admin, KDAK und KENC sind nur der HBCI-Chipkarte und dem für sie zuständigen Hintergrundsystem bekannt. Sie werden jeweils aus einem KGK (Key Generating Key) unter Verwendung der Kartenidentifikationsdaten im EF_ID des MF abgeleitet (vgl. Kapitel 8.4.1 von [DATKOM]). Das zuständige Hintergrundsystem kennt die jeweiligen KGK und leitet die kartenindividuellen Schlüssel bei Bedarf ab.

Es können pro logischer Schlüsselnummer verschiedene KGK verwendet werden. Ein KGK wird wie alle daraus abgeleiteten Schlüssel anhand der Schlüsselversion identifiziert. Die Schlüsselversion zur jeweiligen logischen Schlüsselnummer im zugehörigen EF_KEYD zeigt an, aus welchem KGK der jeweilige kartenindividuelle Schlüssel abgeleitet ist.

♦ **Format**

Für das EF_KEY des DF_BANKING_20 sind die folgenden FCP festzulegen:

Tag	Länge	Wert	Erläuterung
'62'	'16'		Tag und Länge für FCP
'82'	'05'	'32 41 00 12 03'	Datei-Deskriptor für sicherheitsrelevantes lineares EF mit fester Recordlänge (18 Byte), 3 Records
'83'	'02'	'00 10'	Datei-ID des EF_KEY
'88'	'01'	'10'	SFI '02' für das EF_KEY
'A1'	'06'	'8B 04 00 30 02 04'	Zugriffsregel-Referenzen

Auf das EF_KEY darf nur im SE #2 zugegriffen werden.

Die Kommandos APPEND RECORD und UPDATE RECORD dürfen nur ausgeführt werden, wenn sie mit Secure Messaging durchgeführt werden, der Record-Inhalt verschlüsselt (ENC) ist und die Kommandonachricht mit einem MAC abgesichert ist. Verschlüsselung und MAC-Bildung erfolgen mit dem K_{HBCI_Admin}. Der Returncode eines APPEND RECORD oder UPDATE RECORD wird mit dem K_{HBCI_Admin} MAC-gesichert. Das Kommando READ RECORD darf nie ausgeführt werden. (Zugriffsregel im Record 4 des EF_RULE)

---

[16] Um den Begriff „Signierschlüssel“ für Anwendungen nach SigG bzw. EU-Richtlinie freizuhalten, wurde hier der Begriff „Daten-Authentikationsschlüssel“ gewählt. Im weiteren Text wird jedoch zur besseren Lesbarkeit weiterhin davon gesprochen, dass eine Nachricht mit diesem Schlüssel signiert wird.

♦ **Daten**

Das EF_KEY im DF_BANKING_20 enthält 3 Records mit den DF-spezifischen Schlüsseln des DF_BANKING_20.

Logische Schlüsselnummer	Schlüssel-Version	Schlüssel
'01'	'XX'	16 Byte langer K_{HBCI_Admin}
'02'	'XX'	16 Byte langer K_{DAK}
'03'	'XX'	16 Byte langer K_{ENC}

Es werden die Schlüsselversionen 1 bis 127 verwendet.

C.2.2.4 EF_KEYD

- **Beschreibung**

  Das EF_KEYD im DF_BANKING_20 enthält die Zusatzinformationen zu den DF-spezifischen Schlüsseln des DF_BANKING_20.

- **Format**

  Für das EF_KEYD sind die folgenden FCP festzulegen:

Tag	Länge	Wert	Erläuterung
'62'	'1C'		Tag und Länge für FCP
'82'	'05'	'14 41 00 1C 03'	Datei-Deskriptor für lineares EF mit variabler Recordlänge (max. 28 Byte) und 3 Records
'83'	'02'	'00 13'	Datei-ID des EF_KEYD
'85'	'02'	'00 48'	für Nutzdaten allokierter Speicherplatz in Byte
'88'	'01'	'F0'	SFI '1E' für das EF_KEYD
'A1'	'08'	'8B 06 00 30 01 02 02 05'	Zugriffsregel-Referenzen

  Im SE #1 dürfen nur die Kommandos READ / SEARCH RECORD mit ungesicherter Kommando und Antwortnachricht ausgeführt werden (Zugriffsregel im Record 2 des EF_RULE).

  Im SE #2 dürfen die Kommandos APPEND RECORD und UPDATE RECORD nur ausgeführt werden, wenn sie mit Secure Messaging durchgeführt werden. Die MAC-Bildung erfolgt für Kommando- und Antwortnachricht mit dem K_{HBCI_Admin} (Zugriffsregel im Record 5 des EF_RULE).

- **Daten**

  Das EF_KEYD enthält 3 Records, die die Zusatzinformation zu den DF-spezifischen Schlüsseln des DF_BANKING_20 enthalten.

  Das Datenobjekt mit Tag '93' enthält im Wertfeld als zweites Byte die Version des entsprechenden Schlüssels.

  Im folgenden wird der Aufbau der Schlüsselzusatzinformation dargestellt:

  Eintrag 1 (K_{HBCI_Admin}):

Tag	Länge	Wert	Erläuterung
'93'	'02'	'01 XX'	Schlüsselnummer und Schlüssel-Version
'C0'	'02'	'81 10'	Symmetrischer Schlüssel der Länge 16 Byte
'90'	'01'	'FF'	Fehlbedienungszähler
'7B'	'0F'		SE-Datenobjekt
'80'	'01'	'02'	Festlegung für SE #2
'B4'	'04'		CCT - Tag und Länge (Usage Qualifier '30' ist Defaultwert)
'89'	'02'	'12 22'	Algorithmus-ID: Schlüssel darf zur Bildung eines Retail-MAC im CFB-Mode verwendet werden
'B8'	'04'		CT - Tag und Länge (Usage Qualifier '10' ist Defaultwert)
'89'	'02'	'11 23'	Algorithmus-ID: Schlüssel darf zur Verschlüsselung als Triple-DES Schlüssel im CBC-Mode mit ICV ≠ 0 und ICV-Variante verwendet werden

Eintrag 2 (K_{DAK}):

Tag	Länge	Wert	Erläuterung
'93'	'02'	'02 XX'	Schlüsselnummer und Schlüssel-Version
'C0'	'02'	'81 10'	Symmetrischer Schlüssel der Länge 16 Byte
'7B'	'0C'		SE-Datenobjekt
'80'	'01'	'01'	Festlegung für SE #1
'B4'	'07'		CCT - Tag und Länge
'95'	'01'	'20'	Usage Qualifier: Nur SM-Antwortnachricht
'89'	'02'	'12 22'	Algorithmus-ID: Schlüssel darf zur Bildung eines Retail-MAC im CFB-Mode verwendet werden

Eintrag 3 (KENC):

Tag	Länge	Wert	Erläuterung
'93'	'02'	'03 XX'	Schlüsselnummer und Schlüssel-Version
'C0'	'02'	'81 10'	Symmetrischer Schlüssel der Länge 16 Byte
'7B'	'0C'		SE-Datenobjekt
'80'	'01'	'01'	Festlegung für SE #1
'A4'	'07'		AT - Tag und Länge
'95'	'01'	'40'	Usage Qualifier: Nur interne Authentikation
'89'	'02'	'21 12'	Algorithmus-ID: Schlüssel darf zur Authentikation der Chipkarte mit Triple-DES verwendet werden

C.2.2.5 EF_PWD

- **Beschreibung**

  Das lokale EF_PWD im DF_BANKING_20 enthält in dem 9 Byte langen Record '01' die Länge der HBCI-PIN und einen Referenzwert der HBCI-PIN der ZKA-Chipkarte. Die HBCI-PIN hat eine Mindestlänge von 5 Ziffern und darf maximal 12 Ziffern lang sein.

- **Format**

  Für das EF_PWD des DF_BANKING_20 sind die folgenden FCP festzulegen:

Tag	Länge	Wert	Erläuterung
'62'	'16'		
'82'	'05'	'32 41 00 09 01'	Datei-Deskriptor für sicherheitsrelvantes lineares EF mit fester Recordlänge von 9 Byte
'83'	'02'	'00 12'	Datei-ID des EF_PWD
'88'	'01'	'18'	SFI '03' für das EF_PWD
'A1'	'06'	'8B 04 00 30 02 04'	Zugriffsregel-Referenz

  Auf das EF_PWD darf nur im SE #2 zugegriffen werden: Die Kommandos APPEND RECORD und UPDATE RECORD dürfen nur ausgeführt werden, wenn sie mit Secure Messaging durchgeführt werden, der Recordinhalt verschlüsselt (ENC) ist und die Kommandonachricht mit einem MAC abgesichert ist. Verschlüsselung und MAC-Bildung erfolgen dabei mit dem K_{HBCI_Admin}. Der Returncode eines APPEND RECORD oder UPDATE RECORD wird MAC-gesichert. Die MAC-Bildung erfolgt für die Antwortnachricht mit dem K_{HBCI_Admin}. Das Kommando READ RECORD darf nie ausgeführt werden (Zugriffsregel im Record 4 des EF_RULE).

- **Daten**

  Der Record '01' des EF_PWD enthält einen Referenzwert der HBCI-PIN.

Byte	Inhalt	Beschreibung
1	'05'	Länge der PIN
2 - 9	'XX..XX'	Referenzwert der PIN

  Zur Erzeugung des Referenzwertes wird aus der HBCI-PIN zunächst der 8 Byte lange 'Format 2 PIN Block' gemäß [ISO PIN1] wie folgt gebildet:

C	L	P	P	P	P	P	P/F	P/F	P/F	P/F	P/F	P/F	P/F	F	F

  Erläuterung:

  Jedes Feld repräsentiert ein Halbbyte.

C:	Kontroll-Feld, binär kodiert	hat immer den Wert '2'
L:	PIN-Länge, binär kodiert	mögliche Werte von '5' bis 'C'
P:	PIN-Ziffer, BCD-kodiert	
F:	Filler, binär kodiert	hat immer den Wert 'F'
P/F:	PIN-Ziffer/Filler	Belegung abhängig von der PIN-Länge

Der erzeugte Format 2 PIN Block wird mit PB bezeichnet. Aus diesem PIN Block wird der zu speichernde Referenzwert durch DES-Verschlüsselung mit sich selbst erzeugt:

PIN-Referenzwert: ePB(PB)

Falls erforderlich, wird vor der Verwendung von PB als DES-Schlüssel ein Parity Adjustment vorgenommen. PB wird als Klartext unverändert verwendet.

C.2.2.6 EF_PWDD

- **Beschreibung**

  Das EF_PWDD im DF_BANKING_20 enthält in Record '01' die Zusatzinformationen zu der im EF_PWD des DF_BANKING_20 abgelegten HBCI-PIN.

- **Format**

  Für das EF_PWDD sind die folgenden FCP festzulegen:

Tag	Länge	Wert	Erläuterung
'62'	'1C'		
'82'	'05'	'14 41 00 15 01'	Datei-Deskriptor für lineares EF mit variabler Recordlänge (max. 21 Byte) und 1 Record
'83'	'02'	'00 15'	Datei-ID des EF_PWDD
'85'	'02'	'00 15'	Für Nutzdaten allokierter Speicherplatz in Byte
'88'	'01'	'20'	SFI '04' für das EF_PWDD
'A1'	'08'	'8B 06 00 30 01 02 02 05'	Zugriffsregel-Referenzen

  Im SE #1 dürfen nur die Kommandos READ / SEARCH RECORD mit ungesicherter Kommando und Antwortnachricht ausgeführt werden (Zugriffsregel im Record 2 des EF_RULE).

  Im SE #2 dürfen APPEND RECORD und UPDATE RECORD nur ausgeführt werden, wenn sie mit Secure Messaging durchgeführt werden und die Kommandonachricht mit einem MAC abgesichert ist. Der Returncode wird MAC-gesichert. Die MAC-Bildung erfolgt für Kommando- und Antwortnachricht mit dem K_{HBCI_Admin} (Zugriffsregel im Record 5 des EF_RULE).

- **Daten**

  Das lokale EF_PWDD enthält in Record '01' einen 21 Byte langen Record, der die Zusatzinformationen zu der HBCI-PIN enthält.

Tag	Länge	Wert	Beschreibung
'93'	'02'	'01 01'	Passwortreferenz: Passwort '01' im Record '01' des EF_PWD
'89'	'02'	'11 50'	Speicherformat des Passwortes (minimal 5 Ziffern)
'A1'	'03'	'8B 01 08'	Zugriffsregel-Referenz
'7B'	'06'		SE-DO, Tag und Länge
'80'	'01'	'00'	SE Referenz-DO: Für alle SEs
'89'	'01'	'12'	Übertragungsformat der Authentikationsdaten: PIN Format 2 Block

C.2.2.7 EF_FBZ

- **Beschreibung**

  EF_FBZ bezeichnet das lineare EF, das in Record '01' den Fehlbedienungszähler und den zugehörigen Initialwert für die im DF-spezifischen EF_PWD abgelegte HBCI-PIN enthält.

- **Format**

  Für das EF_FBZ im DF_BANKING_20 sind die folgenden FCP festzulegen:

Tag	Länge	Wert	Erläuterung
'62'	'18'		Tag und Länge für FCP
'82'	'05'	'12 41 00 02 01'	Datei-Deskriptor für lineares EF fester Recordlänge
'83'	'02'	'00 16'	Datei-ID des EF_FBZ
'88'	'01'	'28'	SFI '05' für das EF_FBZ
'A1'	'08'	'8B 06 00 30 01 02 02 05'	Zugriffsregel-Referenzen

  Im SE # 1 dürfen nur die Kommandos READ / SEARCH RECORD mit ungesicherter Kommando und Antwortnachricht ausgeführt werden (Zugriffsregel im Record 2 des EF_RULE).

  Im SE #2 dürfen die Kommandos APPEND RECORD und UPDATE RECORD nur ausgeführt werden, wenn sie mit Secure Messaging durchgeführt werden und die Kommandonachricht mit einem MAC abgesichert ist. Der Returncode wird MAC-gesichert. Die MAC-Bildung erfolgt für Kommando- und Antwortnachricht mit dem K_{HBCI_Admin} (Zugriffsregel im Record 5 des EF_RULE).

- **Daten**

  Das EF_FBZ enthält in Record '01' einen 2 Byte langen Record, der den Fehlbedienungszähler und den zugehörigen Initialwert '03' für die HBCI-PIN enthält.

Initialwert des FBZ	FBZ
'03'	'03'

C.2.2.8 EF_BNK

- **Beschreibung**

  Bei dem EF_BNK handelt es sich um ein lineares EF mit 5 Records in dem Bankverbindungen abgelegt sind.

- **Format**

  Für das EF_BNK in einer HBCI-Chipkarte sind die folgenden FCP festzulegen:

Tag	Länge	Wert	Erläuterung
'62'	'18'		Tag und Länge für FCP
'82'	'05'	'12 41 00 58 05'	Datei-Deskriptor für lineares EF mit fester Recordlänge 88 Byte und 5 Records
'83'	'02'	'03 01'	Datei-ID des EF_BNK
'88'	'01'	'D0'	SFI '1A' für das EF_BNK
'A1'	'08'	'8B 06 00 30 01 06 02 03'	Zugriffsregel-Referenzen

  Im SE #1 dürfen READ / SEARCH RECORD immer ausgeführt werden, die Antwortnachricht wird nicht abgesichert. UPDATE RECORD darf nur ausgeführt werden, wenn zuvor eine Karteninhaber-Authentikation mit dem lokalen Passwort 1 (HBCI-PIN) erfolgt ist. Der Returncode wird nicht MAC-gesichert (Zugriffsregeln im Record 6 des EF_RULE).

  Im SE #2 darf das Kommando APPEND RECORD nur ausgeführt werden, wenn es mit Secure Messaging durchgeführt wird. Die MAC-Bildung erfolgt für Kommando- und Antwortnachricht mit dem K_{HBCI_Admin} (Zugriffsregel im Record 3 des EF_RULE).

- **Daten**

  Die Records setzen sich aus einer Bankkurzbezeichnung, der Bankleitzahl, dem Kommunikationsdienst, der Adresse und dem Adresszusatz für den Kommunikationszugang, dem Länderkennzeichen und der Benutzerkennung zusammen.

Byte	Länge	Wert	Erläuterung
1-20	20	'aa .. aa'	Kurzbezeichner des Kreditinstituts
21-24	4	'nn nn nn nn'	Kreditinstitutscode des kontoführenden Instituts
25-25	1	'n'	Kommunikationsdienst
26-53	28	'aa .. aa'	Kommunikationsadresse
54-55	2	'aa aa'	Kommunikationsadressenzusatz
56-58	3	'aa aa aa'	Länderkennzeichen des kontoführenden Instituts
59-88	30	'aa .. aa'	Benutzerkennung

  Alphanumerische Feldinhalte ('a') werden ASCII-kodiert, linksbündig eingestellt und mit Leerzeichen (X'20') auf die vorgegebene Länge aufgefüllt. Numerische Feldinhalte ('n') werden BCD-kodiert.

C.2.2.9 EF_MAC

- **Beschreibung**

  Das EF_MAC wird für die MAC-Bildung über den Hashwert einer Nachricht benötigt. Es besteht aus einem 12 Byte langem Record deren Zugriffsregeln so gesetzt werden müssen, dass beim Lesen des Records der MAC produziert wird.

- **Format**

  Für das EF_MAC sind die folgenden FCP festzulegen:

Tag	Länge	Wert	Erläuterung
'62'	'18'		Tag und Länge für FCP
'82'	'05'	'12 41 00 0C 01'	Datei-Deskriptor für lineares EF mit einem Record der Länge 12 Byte
'83'	'02'	'03 02'	Datei-ID des EF_MAC
'88'	'01'	'D8'	SFI '1B' für das EF_MAC
'A1'	'08'	'8B 06 00 30 01 07 02 03'	Zugriffsregel-Referenzen

  Im SE #1 dürfen READ / SEARCH RECORD nach Karteninhaber-Authentikation ausgeführt werden, die Antwortnachricht wird mit einem K_{DAK}-MAC versehen. UPDATE RECORD darf nur ausgeführt werden, wenn zuvor eine Karteninhaber-Authentikation mit dem lokalen Passwort 1 (HBCI-PIN) erfolgt ist. Der Returncode eines UPDATE RECORD wird nicht MAC-gesichert (Zugriffsregeln im Record 7 des EF_RULE).

  Im SE #2 darf das Kommando APPEND RECORD nur ausgeführt werden, wenn es mit Secure Messaging durchgeführt wird. Die MAC-Bildung erfolgt für Kommando- und Antwortnachricht mit dem K_{HBCI_Admin} (Zugriffsregel im Record 3 des EF_RULE).

- **Daten**

  Das EF_MAC enthält einen Record, der den folgenden Aufbau hat:

Byte	Wert	Erläuterung
1-12	'XX..XX'	Hashwert

C.2.2.10 EF_SEQ

- **Beschreibung**

  Bei dem EF_SEQ handelt es sich um ein lineares EF, dessen Record ein 2 Byte langes binär definiertes Element enthält. Dieser binäre aufsteigende Zähler fließt als Sicherheitsreferenznummer (Signatur-ID) zur Absicherung der Daten gegen Doppeleinreichung ein. Der Startwert des Zählers ist 1. Ein Rücksetzen bei Überlauf findet nicht statt.

- **Format**

  Für das EF_SEQ sind die folgenden FCP festzulegen:

Tag	Länge	Wert	Erläuterung
'62'	'18'		Tag und Länge für FCP
'82'	'05'	'12 41 00 02 01'	Datei-Deskriptor für lineares EF mit 1 Record der Länge 2 Byte
'83'	'02'	'03 03'	Datei-ID des EF_SEQ
'88'	'01'	'E0'	SFI '1C' für das EF_SEQ
'A1'	'08'	'8B 06 00 30 01 06 02 05'	Zugriffsregel-Referenzen

  Im SE #1 dürfen READ / SEARCH RECORD immer ausgeführt werden, die Antwortnachricht wird nicht abgesichert. UPDATE RECORD darf nur ausgeführt werden, wenn zuvor eine Karteninhaber-Authentikation mit dem lokalen Passwort 1 (HBCI-PIN) erfolgt ist. Der Returncode wird nicht MAC-gesichert (Zugriffsregeln im Record 6 des EF_RULE).

  Im SE #2 dürfen die Kommandos APPEND RECORD und UPDATE RECORD nur ausgeführt werden, wenn sie mit Secure Messaging durchgeführt werden. Die MAC-Bildung erfolgt für Kommando- und Antwortnachrichten jeweils mit dem K_{HBCI_Admin} (Zugriffsregel im Record 5 des EF_RULE).

- **Daten**

  Das EF_SEQ enthält 1 Record, der den folgenden Aufbau hat:

Byte	Wert	Erläuterung
1-2	'XX XX'	Sequenznummer

C.2.3 Platzbedarf der Applikation im Chip

Die Platzbedarfsberechnung ist sehr stark von der Stärke der ROM-Maske abhängig. Der notwendige Platz für die EF-Verwaltung z.B. Recordnummern- bzw. Adressverwaltung steht im direkten Zusammenhang mit der Verwaltung des E^2-PROM. Diese Verwaltung ist Bestandteil der ROM-Maske. Der tatsächliche exakte Platzbedarf kann nur von den ROM-Maskenentwicklern ermittelt werden. Er ist von Chip zu Chip und ROM-Maske zu ROM-Maske unterschiedlich.

- **Typ 0**

Die nachfolgende Tabelle enthält daher nur die Nettodatengröße der "Banking"-Applikation.

Dateiname	Headergröße[17]	Datengröße
DF_Banking	28	26
EF_KEY	23	17
EF_KEYD	23	5
EF_AUT	23	17
EF_AUTD	23	4
EF_PWD1	25	8
EF_PWDD1	23	5
EF_BNK	23	440
EF_MAC	23	12
EF_SEQ	23	2
	237	**536**

Demnach hat die Applikation "Banking" einen Mindestplatzbedarf von **773 Byte**.

- **Typ 1**

Die nachfolgende Tabelle enthält daher nur eine grobe Abschätzung der Nettodatengrößen (in Byte) der Applikation. Dabei wurde als Overhead die Größe des jeweiligen FCP zugrundegelegt. Zusätzlich wurde das FMD des DF_BANKING_20 (enthält die vergebenen SFIs sowie deren Pfade) als ”Nutzdaten” des DF interpretiert.

Dateiname	Overhead	Nutzdaten
DF_BANKING_20	28	68
EF_KEY	24	54
EF_KEYD	30	72
EF_PWD	24	9
EF_PWDD	30	21
EF_FBZ	26	2
EF_RULE	30	125
EF_BNK	26	440
EF_MAC	26	12
EF_SEQ	26	2
	270	**805**

Demnach hat die HBCI-Applikation einen Platzbedarf von ca. **1075 Byte**.

[17] Größenangaben in Byte

C.2.4 Terminalabläufe (Typ 1 und SECCOS 6)

Nachfolgend werden die Anwendungsabläufe aus Endgerätesicht spezifiziert. Hierbei werden ausschließlich die chipkartenbezogenen Aspekte berücksichtigt. Anwendungsbezogene Details sind nicht Bestandteil dieser Spezifikation.

Falls bei der Ausführung der Kommandos ein Fehler auftritt, bricht das Terminal den Vorgang ab, es sei denn, es ist ein abweichendes Verhalten spezifiziert.

C.2.4.1 Startdialog

HBCI-Chipkarte			Endgerät/Gateway	
			A1	Anzeige: 'Bitte Karte einstecken'
		←	C1	Reset HBCI-Chipkarte
R1	ATR der HBCI-Chipkarte	→		
		←	C2	SELECT FILE DF_BANKING(_20)
R2	OK	→		
		←	C3	READ RECORD EF_ID
R3	Kartenidentifikationsdaten (CID)	→		
			A3	CID prüfen und speichern
			A4	HBCI-PIN-Eingabe und Formatierung
		←	C4	VERIFY HBCI-PIN
R4	OK	→		
		←	C5	READ RECORD EF_SEQ
R4	Sequenznummer (SEQ)	→		
			A5	SEQ speichern
		←	C6	READ RECORD EF_BNK
R5	Bankverbindung	→		
			A6	Daten prüfen und speichern

♦ **Erläuterung**

1. Nachdem die HBCI-Chipkarte eingesteckt ist, wird ein Reset der Karte durchgeführt (Kommunikationsprotokoll T = 1). Der korrekte ATR und seine Behandlung sind z.B. in [LT] spezifiziert.
2. Die Applikation HBCI-Banking wird geöffnet, indem das ADF der Applikation, DF_BANKING_20 für HBCI-Karten von Typ 1 oder DF_BANKING für HBCI-Karten von Typ 0, durch das Terminal mittels des Kommandos SELECT FILE ausgewählt wird. Dabei wird zunächst versucht, die neue Applikation DF_BANKING_20 zu selektieren. Bei einem Returncode '6A 82' ist die Applikation nicht vorhanden. Es wird dann die "alte" Applikation DF_BANKING selektiert.

   Command APDU:

Byte	Wert	Erläuterung
1-2	'00 A4'	CLA, INS
3	'04'	P1, Selektion mit DF-Name
4	'0C'	P2, Keine Antwortdaten
5	'09'	L_C
6-14	'D2 76 00 00 25 48 42 0X 00'	AID der HBCI-Applikation (X=1,2)

Nachdem der Applikationskontext geöffnet ist, können die AEFs der Applikation mittels SFI referenziert werden. Das Terminal hält die Information vor, um welchen Kartentyp es sich handelt

3. Das Terminal liest mittels READ RECORD die Kartenidentifikationsdaten im Record '01' des EF_ID im MF der HBCI-Karte (SFI '19').

   Command APDU:

Byte	Wert	Erläuterung
1-2	'00 B2'	CLA, INS
3	'01'	P1, Recordnummer
4	'CC'	P2, Reference Control Byte
5	'00'	L_e

   Wenn das READ RECORD erfolgreich ausgeführt wird, gibt die HBCI-Karte eine Antwortnachricht mit der folgenden Struktur zurück.

Byte	Wert	Erläuterung
1	'67'	Branchenhauptschlüssel
2-4	'2n nn nn'	Kurz-BLZ kartenausgebendes Institut
5-9	'nn..nn'	individuelle Kartennummer
10	'nD'	Prüfziffer für Byte 1 - 9
11-12	'JJ MM'	Verfalldatum der Karte
13-15	'JJ MM TT'	Aktivierungsdatum der Karte
16-17	'0280'	Ländercode
18-20	'44 45 4D' oder '45 55 52'	Währungskennzeichen "DEM" oder "EUR"
21	'01'	Wertigkeit der Währung
22	'XX'	Chiptyp
23	'00'	Filler
24	'XX'	Betriebssystem-Version
23-24 oder 25-26	'XX XX'	Positiver Returncode SW1 SW2

   Die Antwortdaten sind mindestens 22 Byte lang und können für Karten von Typ 1 länger als 24 Byte sein.

   Die Kodierung der empfangenen Daten wird geprüft:

   Wenn eine Karte von Typ 0 mehr als 22 Byte Antwortdaten ausgibt, oder wenn eine Karte von Typ 1 weniger als 24 Byte Antwortdaten ausgibt, oder wenn Währungskennzeichen in Byte 18-20 oder Wertigkeit der Währung in Byte 21 nicht korrekt kodiert sind, oder wenn eine Karte von Typ 0 das Währungskennzeichen "EUR" oder eine Karte von Typ 1 das Währungskennzeichen "DEM" ausgibt, oder wenn Byte 24 einer Karte von Typ 1 den Wert '00' hat sowie bei jedem anderen Fehlerfall wird mit einer Fehlermeldung abgebrochen.

4. Das Terminal fordert den Karteninhaber auf, die PIN einzugeben und formatiert dann die eingegebene PIN zum Format 2 PIN-Block FPIN2. Das Terminal baut eine Kommandonachricht für das Kommando VERIFY auf.

Command APDU:

Byte	Wert	Erläuterung
1	'00 20'	CLA, INS
3	'00'	P1, fester Wert
4	'81'	P2, PIN im EF_PWD1 des DF suchen (bzw. hat PWDID ´01´)
5	'08'	L_c
6-13	'XX..XX'	FPIN2

Die Chipkarte führt die PIN-Prüfung durch und setzt das Flag des entsprechenden Sicherheitszustands, wenn die PIN-Prüfung erfolgreich war. Andernfalls wird der PIN-Fehlbedienungszähler dekrementiert.

Durch den Returncode des Kommandos VERIFY teilt die Chipkarte dem Terminal mit, ob die Prüfung erfolgreich war, bzw. wie viele Versuche noch möglich sind.

5. Das Terminal liest mittels READ RECORD die Sequenznummer im Record '01' des EF_SEQ (SFI '1C').

   Command APDU:

Byte	Wert	Erläuterung
1-2	'00 B2'	CLA, INS
3	'01'	P1, Recordnummer
4	'E4'	P2, Reference Control Byte
5	'00'	L_e

   Wenn das READ RECORD erfolgreich ausgeführt wird, gibt die HBCI-Karte eine Antwortnachricht mit der folgenden Struktur zurück.

Byte	Wert	Erläuterung
1-2	'XX XX'	Sequenzzähler
3-4	'XX XX'	Positiver Returncode SW1 SW2

   Das Terminal speichert den Wert des Sequenzzählers.

6. Das Terminal liest mittels READ RECORD sukzessive die Bankverbindungsdaten in den Records des EF_BNK (SFI '1A'), bis der ”passende” Eintrag gefunden wird.

   Command APDU:

Byte	Wert	Erläuterung
1-2	'00 B2'	CLA, INS
3	'0X'	P1, Recordnummer X
4	'D4'	P2, Reference Control Byte
5	'00'	L_e

   Wenn das READ RECORD erfolgreich ausgeführt wird, gibt die HBCI-Karte eine Antwortnachricht mit der folgenden Struktur zurück:

Byte	Länge	Wert	Erläuterung
1-20	20	'aa .. aa'	Kurzbezeichner des Kreditinstituts
21-24	4	'nn nn nn nn'	Bankleitzahl des kontoführenden Instituts
25-25	1	'n'	Kommunikationsdienst
26-53	28	'aa .. aa'	Kommunikationsadresse
54-55	2	'aa aa'	Kommunikationsadressenzusatz
56-58	3	'aa aa aa'	Länderkennzeichen des kontoführenden Instituts
59-88	30	'aa .. aa'	Benutzerkennung
89-90	2	'XX XX'	Positiver Returncode SW1 SW2

Alternativ kann für Chipkarten vom Typ 1 das Kommando SEARCH RECORD verwendet werden, um mittels eines mit übergebenen Suchmusters den ”passenden” Eintrag in einem Schritt zu finden.

Beispiel: Es soll der erste Eintrag zu einer vorgegebenen Bankleitzahl des kontoführenden Instituts (an Byteposition 21-24) gefunden werden:

Command APDU:

Byte	Wert	Erläuterung
1-2	'00 A2'	CLA, INS
3	'01'	P1, Recordnummer an der die Suche startet
4	'D7'	P2, Reference Control Byte (SFI + spezifische Suche)
5	'07'	L_C
6	'04'	Control Byte
7	'14'	Offset 20 = Byte 21
8	'0E'	Konfigurationsbyte: Suche an dieser Position bis zum ersten erfolgreichen Record mit Rückgabe des Inhalts
9-12	'nn nn nn nn'	Bankleitzahl-Suchmuster
13	'00'	L_e

Das Kommando SEARCH RECORD gibt bei erfolgreicher Kommandoausführung die folgende Antwortnachricht aus:

Byte	Wert	Erläuterung
1	'XX'	Recordnummer
2-89	'XX..XX'	Recordinhalt
90-91	'XX XX'	Statusbytes

Es sind auch weitere, umfangreichere Suchoptionen möglich (z.B. alle passenden Einträge ermitteln oder Intervallsuche), siehe hierzu [LIT 1´].

C.2.4.2 Nachricht generieren

Dieser Teil des Gesamtablaufs ist nur insofern chipkartenrelevant, als Bankverbindungsdaten, die für die Auftragsgenerierung benötigt werden, aus der Chipkarte entnommen werden. Für die folgende Ablaufbeschreibung wird angenommen, dass die Anwendung bereits HBCI-Nachrichten generiert hat. Diese Nachrichten müssen jetzt ggf. noch kryptographisch gesichert werden, d.h. es werden Segmente für die elektronische(n) Signatur(en) und für die Verschlüsselung entsprechend den HBCI-Spezifikationen eingefügt.

C.2.4.3 Nachricht signieren

Die folgenden Abläufe können offline, d.h. außerhalb des Übertragungsdialogs vollzogen werden. Dies gilt für alle Nachrichten mit Ausnahme der Dialoginitialisierung. Der Grund besteht darin, dass für die Absicherung aller Kreditinstitutsnachrichten der Schlüssel des Senders der Dialoginitialisierungsnachricht erforderlich ist. Daher muss auch die Chipkarte des Senders während des gesamten Dialogs im Endgerät stecken.

Die Abläufe für die Signatur der Dialoginitialisierungsnachricht sind grundsätzlich identisch mit den im Folgenden beschriebenen Abläufen für die Signatur von Auftragsnachrichten. Da aber für die Dialoginitialisierung anwendungsseitig noch weitere Chipkartendaten (Benutzerkennung, Dialog-ID, Kommunikationszugang etc.) benötigt werden, wird der komplette Ablauf einschließlich der Signatur der Dialoginitialisierung im Kapitel C.2.4.5 "Übertragungsdialog" noch einmal beschrieben.

HBCI-Chipkarte			Endgerät/Gateway	
		←	C1a	GET KEYINFO **(nur Typ 1)**
R1a	KV	→		
			A1a	Schlüsselversion KV speichern
		←	C1b	SELECT EF_KEYD **(nur Typ 0)**
R1b	OK	→		
		←	C1c	READ RECORD EF_KEYD **(nur Typ 0)**
R1c	Datensatz	→		
			A1c	Schlüsselversion KV speichern
			A2	Sequenzzähler (Signatur-ID) SEQ inkrementieren
			A3	Signaturkopf aufbauen und in HBCI-Nachricht einfügen
			A4	Daten (Signaturkopf, HBCI-Nutzdaten) für MAC-Berechnung bereitstellen
		←	M5	MAC über Daten berechnen (siehe Kap. C.2.5.1)
		→		
		←	C6	UPDATE RECORD EF_SEQ mit SEQ
R6	OK	→		
			A7	Signaturabschluss aufbauen und in HBCI-Nachricht einfügen
			A8	ggf. A2 bis A7 für weitere Nachrichten wiederholen
			A9	signierte HBCI-Nachrichten zur Weiterverarbeitung speichern
			A10	ggf. Startdialog und A1 bis A9 für Mehrfachsignaturen wiederholen

♦ **Erläuterung**

1. In diesem Schritt stellt das Terminal fest, welcher Daten-Authentikationsschlüssel KGK_{DAK} bzw. K_{DAK} zur Signatur der Nachricht verwendet werden muss. Dabei wird Schritt 1a *nur* für Karten vom Typ 1, Schritt 1b und 1c *nur* für Karten vom Typ 0 durchgeführt.

1a. Falls es sich um eine **HBCI-Karte von Typ 1** handelt, wird hierzu das Kommando GET KEYINFO verwendet.

Command APDU:

Byte	Wert	Erläuterung
1-2	‘B0 EE’	CLA,INS
3	'80'	P1 für "DF-spezifisch"
4	‘02’	P2, Schlüsselnummer
5	‘00’	L_e

Bei der erfolgreichen Ausführung des GET KEYINFO gibt die HBCI-Karte eine Antwortnachricht mit der folgenden Struktur zurück:

Byte	Wert	Erläuterung
1	'XX'	1 vorhandene Schlüssel-Version KV
2-3	‘XX XX’	Positiver Returncode SW1 SW2

Die Schlüsselversion wird gespeichert.

1b. Falls es sich um eine HBCI-Karte von Typ 0 handelt, wird hierzu das EF_KEYD im DF_BANKING mittels SELECT FILE EF_KEYD ausgewählt.

Command APDU:

Byte	Wert	Erläuterung
1-2	'00 A4'	CLA, INS
3	'02'	P1, Selektion eines EF im aktuellen DF
4	'0C'	P2, Keine Antwortdaten
5	'02'	L_c
6-7	'00 13'	Datei-ID von EF_KEYD

1c. Mittels READ RECORD liest das Terminal aus Record ‘02’ die Zusatzinformationen für den Schlüssel K_{DAK}.

Command APDU:

Byte	Wert	Erläuterung
1-2	'00 B2'	CLA, INS
3	'02'	P1, Recordnummer für logische Schlüsselnr. ‘02’
4	‘04'	P2, Reference Control Byte
5	'00'	L_e

Wenn das READ RECORD erfolgreich ausgeführt wurde, gibt die HBCI-Karte die folgende Antwortnachricht zurück:

Byte	Wert	Erläuterung
1	'02'	Logische Schlüsselnummer
2	'10'	Schlüssellänge
3	'07'	Algorithmus-ID
4	'XX'	Fehlbedienungszähler
5	'XX'	Schlüssel-Version
6-7	‘XX XX’	Positiver Returncode SW1 SW2

Die Schlüsselversion wird gespeichert.

2. Der zuvor gelesene und gespeicherte Sequenzzähler SEQ wird inkrementiert.
3. Der Signaturkopf wird aufgebaut und in die HBCI-Nachricht eingefügt.
4. Die Daten (Signaturkopf, HBCI-Nutzdaten) für die MAC-Berechnung werden bereitgestellt.
5. Der MAC über die Daten wird berechnet (siehe hierzu Kap. C.2.5.1).
6. Das Terminal überschreibt den Sequenzzähler in EF_SEQ mit dem inkrementierten Wert. Dies geschieht durch ein UPDATE RECORD EF_SEQ ohne Secure Messaging. Aufgrund der Zugriffsbedingungen für das EF_SEQ kann das Kommando nur ausgeführt werden, wenn zuvor die HBCI-PIN erfolgreich verifiziert wurde.

   Command APDU:

Byte	Wert	Erläuterung
1-2	'00 DC'	CLA, INS
3	'01'	P1, Recordnummer
4	'E4'	P2, Reference Control Byte (SFI ‘1C‘)
5	'02'	L_C
6-7	'XX XX'	neuer Sequenzzähler SEQ

7. Der Signaturabschluß wird aufgebaut und in die HBCI-Nachricht eingefügt.
8. Ggf. können die Schritte 2 bis 7 für weitere Nachrichten wiederholt werden. Schritt 1 braucht nicht erneut durchgeführt zu werden, da die zu verwendende Schlüsselversion bereits gespeichert ist..
9. Die signierten HBCI-Nachrichten können zur Weiterverarbeitung gespeichert werden.
10. Ggf. werden Startdialog und die Schritte 1 bis 9 für Mehrfachsignaturen wiederholt.

C.2.4.4 Nachricht verschlüsseln

HBCI-Chipkarte			Endgerät/Gateway	
		←	C1a	GET KEYINFO **(nur Typ 1)**
R1a	KV	→		
			A1a	Schlüsselversion KV speichern
		←	C1b	SELECT EF_AUTD **(nur Typ 0)**
R1b	OK	→		
		←	C1c	READ RECORD EF_AUTD **(nur Typ 0)**
R1c	Datensatz	→		
			A1c	Schlüsselversion KV speichern
			A2	Daten (HBCI-Nutzdaten und ggf. Signaturkopf/-abschluss) für die Verschlüsselung bereitstellen
		←	C3	GET CHALLENGE
R3	RND	→		
			A3	RND als Nachrichtenschlüssel-Hälfte KS_L speichern
		←	C4	INTERNAL AUTHENTICATE mit KS_L
R4	e* $K_{ENC}(KS_L)$	→		
			A4	e* $K_{ENC}(KS_L)$ speichern
		←	C5	GET CHALLENGE
R5	RND	→		
			A5	RND als Nachrichtenschlüssel-Hälfte KS_R speichern
		←	C6	INTERNAL AUTHENTICATE mit KS_R
R6	e* $K_{ENC}(KS_R)$	→		
			A6	e* $K_{ENC}(KS_R)$ speichern
			A7	e* $K_{ENC}(KS_L)$ mit e* $K_{ENC}(KS_R)$ zu e* $K_{ENC}(KS)$ konkatenieren und speichern
			A8	KS_L mit KS_R zu KS konkatenieren und Daten mit KS verschlüsseln (Triple-DES CBC-Mode, IV=0, X9.23 Padding)
			A9	Verschlüsselungskopf aufbauen und in HBCI-Nachricht einfügen
			A10	Verschlüsselte Daten als Binärdaten in HBCI-Nachricht einfügen
			A11	ggf. A2 bis A10 für weitere Nachrichten wiederholen
			A12	Verschlüsselte und signierte HBCI-Meldungen zur weiteren Bearbeitung speichern

♦ **Erläuterung**

1. In diesem Schritt stellt das Terminal fest, welche Version des Chiffrierschlüssels KGK_{ENC} bzw. K_{ENC} zur Verschlüsselung der Nachricht verwendet werden muß. Dabei wird Schritt 1a *nur* für Karten vom Typ 1, Schritt 1b und 1c *nur* für Karten vom Typ 0 durchgeführt.

1a. Falls es sich um eine HBCI-Karte von Typ 1 handelt, wird hierzu das Kommando GET KEYINFO verwendet.

Command APDU:

Byte	Wert	Erläuterung
1-2	'B0 EE'	CLA,INS
3	'80'	P1 für "DF-spezifisch"
4	'03'	P2, Schlüsselnummer
5	'00'	L_e

Bei der erfolgreichen Ausführung des GET KEYINFO gibt die HBCI-Karte eine Antwortnachricht mit der folgenden Struktur zurück:

Byte	Wert	Erläuterung
1	'XX'	1 vorhandene Schlüssel-Version KV
2-3	'XX XX'	Positiver Returncode SW1 SW2

Die Schlüsselversion wird gespeichert.

1b. Falls es sich um eine HBCI-Karte von Typ 0 handelt, wird hierzu das EF_AUTD im DF_BANKING mittels SELECT FILE EF_AUTD ausgewählt.

Command APDU:

Byte	Wert	Erläuterung
1-2	'00 A4'	CLA, INS
3	'02'	P1, Selektion eines EF im aktuellen DF
4	'0C'	P2, Keine Antwortdaten
5	'02'	L_c
6-7	'00 14'	Datei-ID von EF_AUTD

1c. Mittels READ RECORD liest das Terminal die Zusatzinformationen für den Schlüssel K_{ENC}. Diese sind im Record '01' des selektierten EF_AUTD zu finden.

Command APDU:

Byte	Wert	Erläuterung
1-2	'00 B2'	CLA, INS
3	'01'	P1, Recordnummer für logische Schlüsselnr. '00'
4	'04'	P2, Reference Control Byte
5	'00'	L_e

Wenn das READ RECORD erfolgreich ausgeführt wurde, gibt die HBCI-Karte die folgende Antwortnachricht zurück:

Byte	Wert	Erläuterung
1	'03'	Logische Schlüsselnummer
2	'10'	Schlüssellänge
3	'07'	Algorithmus-ID
4	'XX'	Schlüssel-Version
5-6	'XX XX'	Positiver Returncode SW1 SW2

Die Schlüsselversion wird gespeichert.

2. Die Daten (HBCI-Nutzdaten und ggf. Signaturkopf/-abschluss) für die Verschlüsselung werden bereitgestellt.

3. Mit dem Kommando GET CHALLENGE lässt sich das Terminal eine Zufallszahl von der HBCI-Karte geben.

   Command APDU:

Byte	Wert	Erläuterung
1-2	'00 84'	CLA, INS
3	'00'	P1
4	'00'	P2
5	'00'	L_e

   Wenn das Kommando erfolgreich ausgeführt wurde, gibt die HBCI-Karte eine 8 Byte lange Zufallszahl als Antwortdatum aus, die als Nachrichtenschlüssel-Hälfte KS_L gespeichert wird.

4. Mit dem Kommando INTERNAL AUTHENTICATE wird der Wert KS_L von der HBCI-Karte mit dem Schlüssel K_{ENC} verschlüsselt und in der Antwortnachricht als e* $K_{ENC}(KS_L)$ übergeben.

   Command APDU:

Byte	Wert	Erläuterung
1-2	'00 88'	CLA, INS
3	'00'	P1
4	'80' oder '83'	P2, Typ 0: '80' (log. Schlüsselnummer '00'), Typ 1: '83' (log. Schlüsselnummer '03')
5	'08'	L_c
6-13	'XX .. XX'	Zufallszahl KS_L
14	'00'	L_e

   Das Kommando INTERNAL AUTHENTICATE gibt folgende Antwortnachricht zurück:

Byte	Wert	Erläuterung
1-8	'XX .. XX'	Verschlüsselter Wert e* $K_{ENC}(KS_L)$
9-10	'XX XX'	Positiver Returncode SW1 SW2

5. Mit dem Kommando GET CHALLENGE lässt sich das Terminal eine weitere Zufallszahl von der HBCI-Karte geben, die als Nachrichtenschlüssel-Hälfte KS_R gespeichert wird.
6. Analog zu Schritt 4 wird ein INTERNAL AUTHENTICATE mit KS_R durchgeführt.
7. e* $K_{ENC}(KS_L)$ wird mit e* $K_{ENC}(KS_R)$ zu e* $K_{ENC}(KS)$ konkateniert und gespeichert.
8. KS_L wird mit KS_R zu KS konkateniert und die Daten werden mit KS verschlüsselt (Triple-DES CBC-Mode, IV=0, X9.23 Padding).
9. Der Verschlüsselungskopf wird aufgebaut und in die HBCI-Nachricht eingefügt.
10. Die verschlüsselten Daten als Binärdaten in die HBCI-Nachricht eingefügt.
11. Ggf. werden die Schritte 2 bis 10 für weitere Nachrichten wiederholt (eine Wiederholung von Schritt 1 ist nicht nötig).
12. Die verschlüsselten und signierten HBCI-Meldungen werden zur weiteren Bearbeitung gespeichert.

C.2.4.5 Übertragungsdialog

HBCI-Chipkarte			Endgerät/Gateway			Kreditinstitut	
			A1	Sequenzzähler (Signatur-ID) SEQ inkrementieren			
			A2	Benutzerkennung aus der bereits gelesenen Bankverbindung (EF_BNK) ermitteln			
			A3	Dialoginitialisierungsnachricht aufbauen			
			A4	Signaturkopf aufbauen und in HBCI-Nachricht einfügen			
			A5	Daten (Signaturkopf, HBCI-Nutzdaten) für MAC-Berechnung bereitstellen			
		← →	M6	MAC über Daten berechnen (siehe Kap. C.2.5.1)			
R7	OK	← →	C7	UPDATE RECORD EF_SEQ mit SEQ			
			A8	Signaturabschluss aufbauen und in HBCI-Nachricht einfügen			
			A9	Kommunikationszugang aus Bankverbindung herstellen			
			C10	Nachricht (beginnend mit Dialoginitialisierungsnachricht) senden	→ ←	R10	Antwortnachricht senden
			A11	falls Antwortnachricht verschlüsselt: Daten (Binärdaten nach dem Signaturkopf) und $d*K_{ENC}(KS)$ aus dem Signaturkopf für die Entschlüsselung bereitstellen			
		← →	M12	Daten entschlüsseln (siehe Kap. C.2.5.2)			
			A13	falls Kreditinstitutsnachricht signiert: Daten (Signaturkopf, Nutzdaten) und Referenz-MAC für MAC-Prüfung bereitstellen			
		← →	M14	MAC über Daten und Referenz-MAC prüfen (siehe Kap. C.2.5.2)			
			A15	C10 bis M14 für alle weiteren HBCI-Nachrichten wiederholen			

C.2.5 Makros

C.2.5.1 MAC-Berechnung / Prüfung

HBCI-Chipkarte			Endgerät/Gateway	
			A1	Hashwert HASH über Daten berechnen (RIPEMD160)
			A2	HASH zerlegen in $HASH_L$ (die linken 8 Byte von HASH) und $HASH_R$ (die restlichen 12 Byte)
R3	OK	← C3 →		UPDATE RECORD EF_MAC mit $HASH_R$
R4	OK	← C4 →		PUT DATA mit $HASH_L$ **(nur Typ 0)**
R5	Daten aus EF_MAC mit CFB-64 MAC über $HASH_R$ (identisch mit CBC-MAC über HASH)	← C5 → A5		READ RECORD EF_MAC mit Secure Messaging **(für Typ 1 wird hier $HASH_L$ mit übergeben)** Bei MAC-Berechnung: MAC zwischenspeichern Bei MAC-Prüfung: MAC aus Kreditinstitutsnachricht mit MAC der Chipkarte vergleichen

♦ **Erläuterung**

1. Der Hashwert HASH wird über die Daten berechnet (RIPEMD160).
2. Der Hashwert HASH wird zerlegt in $HASH_L$ (die linken 8 Byte von HASH) und $HASH_R$ (die restlichen 12 Byte).
3. $HASH_R$ wird in den ersten Record des EF_MAC eingetragen. Die Zugriffsbedingung für das EF_MAC stellt sicher, daß das UPDATE-Kommando nur ausgeführt werden kann, wenn zuvor die HBCI-PIN verifiziert wurde.

   Command APDU:

Byte	Wert	Erläuterung
1-2	'00 DC'	CLA, INS
3	'01'	P1, Recordnummer
4	'DC'	P2, Reference Control Byte (SFI '1B')
5	'0C'	L_C
6-17	'XX .. XX'	Recordinhalt $HASH_R$

4. Das Terminal übergibt $HASH_L$ mittels PUT DATA an die HBCI-Karte. Dieser Schritt wird *nur* für Karten vom Typ 0 durchgeführt, da für Karten vom Typ 1 der Zufallswert als Bestandteil des Kommandos im nächsten Schritt übergeben wird.

Command APDU:

Byte	Wert	Erläuterung
1-2	'00 DA'	CLA, INS
3-4	'01 00'	P1, P2
5	'08'	L_C
6-13	'XX..XX'	$HASH_L$

5. Das Terminal liest mittels READ RECORD den soeben in EF_MAC eingetragenen Hash-Wert mit Secure Messaging.

Command APDU für Chipkarten vom Typ 0:

Byte	Wert	Erläuterung
1-2	'04 B2'	CLA, INS
3	'01'	P1, Recordnummer
4	'DC'	P2, Reference Control Byte
5	'00'	L_e

Wenn das READ RECORD erfolgreich ausgeführt wird, gibt die HBCI-Karte eine Antwortnachricht mit der folgenden Struktur zurück:

Byte	Wert	Erläuterung
1-12	'XX ... XX'	Recordinhalt $HASH_R$
13-20	'XX ... XX'	CFB-MAC mit K_{ENC} über die 16 Byte 1-12|'00 00 00 00' mit ICV= $HASH_L$
21-22	'XX XX'	Positiver Returncode SW1 SW2

Command APDU für Chipkarten vom Typ 1:[18]

Byte	Wert	Erläuterung
1-2	'08 B2'	CLA, INS mit Secure Messaging
3	'01'	P1, Recordnummer
4	'DC'	P2, Reference Control Byte
5	'11'	L_C
6-7	'BA 0C'	Tag und Länge für Response Descriptor
8-9	'B4 0A'	Tag und Länge für CCT
10-11	'87 08'	Tag und Länge für Zufallszahl
12-19	'XX..XX'	Zufallszahl $HASH_L$
20-22	'96 01 00'	Tag, Länge und Wert des L_e-Datenobjekts
23	'00'	L_e

Wenn das READ RECORD erfolgreich ausgeführt wird, gibt die HBCI-Karte eine Antwortnachricht mit der folgenden Struktur zurück:

[18] Bezüglich der Übergabe von ICVs über Response Descriptors siehe Kapitel 8.6.1.1 von [DATKOM].

Byte	Wert	Erläuterung
1-2	'81 0C'	Tag und Länge des Klartext-Datenobjekts
3-14	'XX ... XX'	Recordinhalt $HASH_R$
15-16	'8E 08'	Tag und Länge des MAC-Datenobjekts
17-24	'XX ... XX'	CFB-MAC mit K_{DAK} über die 16 Byte 1-14|'80 00' mit ICV = $HASH_L$
25-26	'XX XX'	Positiver Returncode SW1 SW2

Das Terminal speichert den Wert des MAC.

C.2.5.2 Entschlüsselung

HBCI-Chipkarte			Endgerät/Gateway	
			A1	d* $K_{ENC}(KS)$ in die zwei Hälften d* $K_{ENC}(KS_L)$ und d* $K_{ENC}(KS_R)$ zerlegen
R2	KS_L	←	C2	INTERNAL AUTHENTICATE mit d* $K_{ENC}(KS_L)$
		→	A2	KS_L zwischenspeichern
R3	KS_R	←	C3	INTERNAL AUTHENTICATE mit d* $K_{ENC}(KS_R)$
		→	A3	KS_R zwischenspeichern
			A4	KS_L mit KS_R zu KS konkatenieren und Daten mit KS entschlüsseln (Triple-DES CBC-Mode, IV=0, X9.23 Padding)

♦ **Erläuterung**

1. d* $K_{ENC}(KS)$ wird in die zwei Hälften d* $K_{ENC}(KS_L)$ und d* $K_{ENC}(KS_R)$ zerlegt.
2. Mit dem Kommando INTERNAL AUTHENTICATE wird der Wert d* $K_{ENC}(KS_L)$ von der HBCI-Karte mit dem Schlüssel K_{ENC} entschlüsselt und in der Antwortnachricht als KS_L übergeben.

   Command APDU:

Byte	Wert	Erläuterung
1-2	'00 88'	CLA, INS
3	'00'	P1
4	'80' oder '83'	P2, Typ 0: '80' (log. Schlüsselnummer '00'), Typ 1: '83' (log. Schlüsselnummer '03')
5	'08'	L_C
6-13	'XX .. XX'	Parameterwert d* $K_{ENC}(KS_L)$
14	'08'	L_e

   Das Kommando INTERNAL AUTHENTICATE gibt folgende Antwortnachricht zurück:

Byte	Wert	Erläuterung
1-8	'XX .. XX'	Entschlüsselter Wert KS_L
9-10	'XX XX'	Positiver Returncode SW1 SW2

   KS_L wird gespeichert.
3. Analog zu Schritt 2 wird ein INTERNAL AUTHENTICATE mit d* $K_{ENC}(KS_R)$ durchgeführt. Das Ergebnis wird als KS_R gespeichert.
4. KS_L wird mit KS_R zu KS konkateniert und die Daten werden mit KS entschlüsselt (Triple-DES CBC-Mode, IV=0, X9.23 Padding).

C.2.6 Übersicht der Chip-Applikations-Parameter

- **Dateistruktur**

Lage	Datei-ID	Name	SFI	Zugriffsregel SE #1 (Standard)	Zugriffsregel SE #2 (Admin)
MF	'00 03'	EF_ID	'19'		
	'A6 00'	DF_BANKING_20			1
DF_BANKING_20	'00 30'	EF_RULE	'01'	2	3
	'00 10'	EF_KEY	'02'	--	4
	'00 12'	EF_PWD	'03'	--	4
	'00 13'	EF_KEYD	'1E'	2	5
	'00 15'	EF_PWDD	'04'	2	5
	'00 16'	EF_FBZ	'05'	2	5
	'03 01'	EF_BNK	'1A'	6	3
	'03 02'	EF_MAC	'1B'	7	3
	'03 03'	EF_SEQ	'1C'	6	5

- **Zugriffsregeln**

#	READ / SEARCH RECORD	APPEND RECORD	UPDATE RECORD	IN-/EXCLUDE CREATE EF DELETE self	VERIFY CHANGE REF DATA	RESET RETRY COUNTER
1				K_{HBCI_Admin}-MAC		
2	ALW					
3		K_{HBCI_Admin}-MAC	NEV			
4		K_{HBCI_Admin}-ENC-MAC (K) K_{HBCI_Admin}-MAC (A)				
5		K_{HBCI_Admin}-MAC				
6	ALW		HBCI-PIN			
7	HBCI-PIN K_{DAK}-MAC (A)		HBCI-PIN			
8					ALW	K_{HBCI_Admin}-MAC

Die angegebenen Access Conditions gelten sowohl für Kommando- (K) als auch Antwortnachrichten (A), sonst in Klammern eingeschränkt.

- **Schlüssel der Applikation**

Logische Schlüsselnr.	Erlaubte SE #	Schlüssel	Wer kennt den Masterschlüssel
'01'	2	K_{HBCI_Admin}	zuständiges Hintergrundsystem
'02'	1	K_{DAK}	zuständiges Hintergrundsystem
'03'	1	K_{ENC}	zuständiges Hintergrundsystem

D. DATA DICTIONARY

A

Austauschkontrollreferenz

Dialog-ID der korrespondierenden Nachricht des Kunden (vgl. [HBCI], Kapitel II.6.2).

Typ:	DE
Format:	id
Länge:	#
Version:	1

B

Benutzerdefinierte Signatur

Bei nicht-schlüsselbasierten Sicherheitsverfahren kann der Benutzer hier Angaben zur Authentisierung machen. Ob das Feld verpflichtend ist, ist vom jeweiligen Sicherheitsverfahren abhängig.

Format: s. Spezifikation „Sicherheitsverfahren PIN/TAN“

Typ:	DEG
Format:	
Länge:	
Version:	1

Benutzerkennung

Eindeutig vergebene Kennung, anhand deren die Identifizierung des Benutzers erfolgt. Die Vergabe obliegt dem Kreditinstitut. Das Kreditinstitut hat zu gewährleisten, dass die Benutzerkennung institutsweit eindeutig ist. Sie kann beliebige Informationen enthalten, darf aber bei Verwendung des RAH- bzw. RDH-Verfahrens aus Sicherheitsgründen nicht aus benutzer- oder kreditinstitutsspezifischen Merkmalen hergeleitet werden.

Typ:	DE
Format:	id
Länge:	#
Version:	1

Bereich der Sicherheitsapplikation, kodiert

Information darüber, welche Daten vom kryptographischen Prozess verarbeitet werden. Diese Information wird benötigt um z.B. zwischen relevanter und belangloser Reihenfolge von Signaturen zu unterscheiden (vgl. [HBCI], Kapitel VI.4).

Wenn SHM gewählt wird, so bedeutet dies, dass nur über den eigenen Signaturkopf sowie die HBCI-Nutzdaten ein Hashwert gebildet wird, der in die Signatur eingeht. Dies entspricht bei Mehrfachsignaturen einer bedeutungslosen Reihenfolge.

Wenn SHT gewählt wird, dann werden auch alle schon vorhandenen Signaturköpfe und -abschlüsse mitsigniert. Das heißt, dass die Reihenfolge der Signaturen relevant ist.

Codierung:

1: Signaturkopf und HBCI-Nutzdaten (SHM)

2: Von Signaturkopf bis Signaturabschluss (SHT)

Typ: DE
Format: code
Länge: ..3
Version: 2

Bezeichner für Algorithmusparameter, IV

Eigenschaft betreffend den Initialisierungswert für die Verfahren DDV, RAH und RDH (Die Steuerung erfolgt in den BPD, vgl. [HBCI], Kapitel IV.4).

Codierung:

1: Initialization value, clear text (IVC)

Typ: DE
Format: code
Länge: ..3
Version: 2

Bezeichner für Algorithmusparameter, Schlüssel

Eigenschaft des Schlüssels für die Verfahren DDV, RAH und RDH (Die Steuerung erfolgt in den BPD, vgl. [HBCI], Kapitel IV.4).

Codierung:

5: Symmetrischer Schlüssel, ver- bzw. entschlüsselt mit einem symmetrischen Schlüssel bei DDV (KYE) (vgl. [HBCI], Kapitel VI.2.2.1).

6: Symmetrischer Schlüssel, verschlüsselt mit einem öffentlichen Schlüssel bei RAH und RDH (KYP).

Typ: DE
Format: code
Länge: ..3
Version: 2

Bezeichner für Exponent

Enthält den Bezeichner für den Exponent des öffentlichen Schlüssels.

Codierung:

13: Exponent (EXP)

Typ: DE
Format: code
Länge: ..3
Version: 2

Bezeichner für Funktionstyp

Enthält den Bezeichner für den Funktionstyp des Key-Management.

Codierung:

112: 'Certificate Replacement' (Ersatz des Zertifikats) im Zusammenhang mit der Schlüsseländerung

124: 'Certificate Status Request' im Zusammenhang mit der Anfrage für einen öffentlichen Schlüssel

224: 'Certificate Status Notice' im Zusammenhang mit der Übermittlung eines öffentlichen Schlüssels

130 : 'Certificate Revocation' (Zertifikatswiderruf) im Zusammenhang mit der Schlüsselsperrung

231: 'Revocation Confirmation' (Bestätigung des Zertifikatswiderrufs) im Zusammenhang mit der Bestätigung der Schlüsselsperrung

Typ:	DE
Format:	code
Länge:	..3
Version:	2

Bezeichner für Hashalgorithmusparameter

Bezeichner für den Hashalgorithmusparameter.

Codierung:

1: IVC (Initialization value, clear text)

Typ:	DE
Format:	code
Länge:	..3
Version:	2

Bezeichner für Modulus

Bezeichner für den Modulus des öffentlichen Schlüssels.

Codierung:

12: Modulus (MOD)

Typ:	DE
Format:	code
Länge:	..3
Version:	2

Bezeichner für Sicherheitspartei

Identifikation der Funktion der beschriebenen Partei, in diesem Falle des Kunden.

Codierung:

1: Message Sender (MS), wenn ein Kunde etwas an sein Kreditinstitut sendet.

2: Message Receiver (MR), wenn das Kreditinstitut etwas an seinen Kunden sendet.

Typ: DE
Format: code
Länge: ..3
Version: 2

Bezugssegment

Sofern sich ein Kreditinstitutssegment auf ein bestimmtes Kundensegment bezieht (z.B. Antwortrückmeldung auf einen Kundenauftrag) hat das Kreditinstitut die Segmentnummer des Segments der Kundennachricht einzustellen, auf das sich das aktuelle Segment bezieht (s. DE „Segmentnummer“). In Zusammenhang mit den Angaben zur Bezugsnachricht aus dem Nachrichtenkopf ist hierdurch eine eindeutige Referenz auf das Segment einer Kundennachricht möglich.

Falls die Angabe eines Bezugssegments erforderlich ist, ist dieses bei der Formatbeschreibung eines Kreditinstitutsegments angegeben.

Typ: DE
Format: num
Länge: ..3
Version: 1

C

CID

(Cardholder Identification) Identifikation der verwendeten Chipkarte. Die CID steht sowohl bei DDV-Chipkarten als auch bei Signaturkarten im EF_ID der Karte. Im DDV-Verfahren dient die CID dem Kreditinstitut zur Herleitung des kartenindividuellen Schlüssels.

Typ: DE
Format: bin
Länge: ..256
Version: 1

D

Daten, verschlüsselt

Enthält die verschlüsselten (und komprimierten) Daten.

Typ: DE
Format: bin
Länge: ..
Version: 1

Datum

Datumsangabe, zur Bestimmung eines Zeitpunktes.

Typ: DE
Format: dat
Länge: #
Version: 1

Datum- und Zeitbezeichner, kodiert

Enthält die Bedeutung des Zeitstempels.

Codierung:

1: Sicherheitszeitstempel (STS)

6: Certificate Revocation Time (CRT)

Typ:	DE
Format:	code
Länge:	..3
Version:	2

F

Filterfunktion

Falls das Übertragungsverfahren eine Umwandlung der Nachricht in eine 7 Bit-Zeichendarstellung erfordert (z.B. Internet), so ist hier das anzuwendende Filterverfahren anzugeben. Die Nachricht ist stets komplett zu filtern, auch wenn eine Filterung nicht notwendig wäre, da bspw. keine binären Daten enthalten sind. Ein Kreditinstitut darf jeweils nur eine Filterfunktion unterstützen.

Codierung:

MIM: MIME Base 64

UUE: Uuencode/Uudecode

Typ:	DE
Format:	an
Länge:	3
Version:	1

H

Hashalgorithmus

Angaben zu einem kryptographischen Algorithmus, seinen Operationsmodus, sowie dessen Einsatz.

Nr.	Name	Typ	Format	Länge	Status	Anzahl	Restriktionen
1	Verwendung des Hashalgorithmus, kodiert	DE	code	..3	M	1	1
2	Hashalgorithmus, kodiert	DE	code	..3	M	1	1, 3, 4, 5, 6, 999
3	Bezeichner für Hashalgorithmusparameter	DE	code	..3	M	1	1
4	Wert des Hashalgorithmusparameters	DE	bin	..512	O	1	

Typ: DEG
Format:
Länge:
Version: 2

Hashalgorithmus, kodiert

Code des verwendeten Hash-Algorithmus.

Codierung:

1: SHA-1

2: belegt

3: SHA-256

4: SHA-384

5: SHA-512

6: SHA-256 / SHA-256

999: Gegenseitig vereinbart (ZZZ); hier: RIPEMD-160

Typ: DE
Format: code
Länge: ..3
Version: 2

Wird als „Hashalgorithmus, kodiert“ die Option „6: SHA-256 / SHA256“ gewählt, so findet ein Hashing sowohl in Software als auch in der Bankensignaturkarte statt.
Die Anwendung muss dafür Sorge tragen, dass in der Karte das gewünschte Hashverfahren – hier SHA-256 – selektiert wird; ansonsten würde in dort das Default-Hashverfahren angewendet, was nicht zulässig ist.

I

Identifizierung der Partei

Code, welcher die (Kommunikations-)Partei identifiziert. Bei Verwendung des RDH-Verfahrens ist die Kundensystem-ID einzustellen.

Typ: DE
Format: id
Länge: #
Version: 1

K

Kommunikationsadresse

Beim Zugang über TCP/IP ist die IP-Adresse als alphanumerischer Wert (z.B. '123.123.123.123') einzustellen.

Beim Zugang über https ist die Adresse des Servlets als alphanumerischer Wert (z.B. „https://www.xyz.de:7000/Servlet") einzustellen.

Typ: DE
Format: an
Länge: ..512
Version: 1

Kommunikationsadressenzusatz

Beim Zugang über TCP/IP und https wird das Feld nicht belegt.

Typ: DE
Format: an
Länge: ..512
Version: 1

Kommunikationsdienst

Unterstütztes Kommunikationsverfahren (Protokollstack).

Zur Zeit unterstützte Kommunikationsverfahren:

1: nicht belegt

2: TCP/IP (Protokollstack SLIP/PPP)

3: https

Typ: DE
Format: code
Länge: ..2
Version: 3

Kommunikationsparameter

Die Kommunikationsparameter enthalten Informationen für den Aufbau der Transportverbindung.

Nr.	Name	Typ	Format	Länge	Status	Anzahl	Restriktionen
1	Kommunikationsdienst	DE	num	..2	M	1	1,2,3
2	Kommunikationsadresse	DE	an	..512	M	1	
3	Kommunikationsadressenzusatz	DE	an	..512	C	1	M: ‚Kommunikationsdienst' = 1 N: sonst
4	Filterfunktion	DE	an	3	C	1	MIM, UUE M: ‚Kommunikationsdienst' = 2 N: sonst
5	Version der Filterfunktion	DE	num	..3	C	1	O: ‚Filterfunktion' belegt N: sonst

Typ: DEG
Format:
Länge:
Version: 2

Komprimierungsfunktion

Code der unterstützten Komprimierungsfunktion.

Codierung:

0: Keine Kompression (NULL)

1: Lempel, Ziv, Welch (LZW)

2: Optimized LZW (COM)

3: Lempel, Ziv (LZSS)

4: LZ + Huffman Coding (LZHuf)

5: PKZIP (ZIP)

6: deflate (GZIP) (http://www.gzip.org/zlib)

7: bzip2 (http://sourceware.cygnus.com/bzip2/)

999: Gegenseitig vereinbart (ZZZ)

Typ: DE
Format: code
Länge: ..3
Version: 2

Komprimierungsversion

Version der unterstützten Komprimierungsfunktion.

Momentan werden alle zulässigen Komprimierungsfunktionen mit Version 1 verwendet. Falls keine Komprimierung verwendet wird (Komprimierungsfunktion 0), wird Version 0 angegeben.

Typ: DE
Format: num
Länge: ..3
Version: 1

Kreditinstitutscode

Landesspezifische Kennung, die das Kreditinstitut eindeutig identifiziert. In Deutschland wird die Bankleitzahl eingestellt. Bei Kreditinstituten, die in Ländern ohne Institutskennungssystem beheimatet sind, kann die Belegung entfallen.

Typ: DE
Format: an
Länge: ..30
Version: 1

Kreditinstitutskennung

Kennung eines Kreditinstituts.

Typ:	DEG
Formatkennung	kik
Länge:	#
Version:	1

Kunden-ID

Institutsweit eindeutige Identifikation des Kunden. Die Vergabe obliegt dem Kreditinstitut. Die Kunden-ID kann beliebige Informationen enthalten. Es steht dem Kreditinstitut frei, ob es jedem Kunden genau eine Kunden-ID zuordnet oder dem Kunden in Abhängigkeit vom Benutzer jeweils eine unterschiedliche Kunden-ID zuordnet.

Typ:	DE
Format:	id
Länge:	#
Version:	1

L

Länderkennzeichen

Länderkennzeichen gemäß ISO 3166-1 (numerischer Code) (s. [Formals], Kap. „Anlagen“). Für Deutschland wird der Code 280 verwendet da dieser im Kreditgewerbe gebräuchlicher als der neue Code 276 ist.

Typ:	DE
Format:	ctr
Länge:	#
Version:	1

N

Nachrichtenbeziehung, kodiert

Code der Nachrichtenbeziehung. Im Zusammenhang mit der Übermittlung eines öffentlichen Schlüssels oder mit der Bestätigung der Schlüsselsperrung ist der Wert „1“ vorgesehen. Im Zusammenhang mit der Schlüsseländerung, mit der Anfrage nach einem öffentlichen Schlüssel oder mit der Schlüsselsperrung ist der Wert „2“ vorgesehen.

Codierung:

1: Key-Management-Nachricht ist Antwort

2: Key-Management-Nachricht erwartet Antwort

Typ:	DE
Format:	code
Länge:	1
Version:	2

Nachrichtenreferenznummer

Nachrichtennummer der korrespondierenden Nachricht des Kunden.

Typ: DE
Format: num
Länge: ..4
Version: 1

O

Öffentlicher Schlüssel

Information, die beim RAH-/RDH-Key-Management zum Transport des öffentlichen Schlüssels zwischen Kunde und Kreditinstitut bzw. umgekehrt dient.

Nr.	Name	Typ	Format	Länge	Status	Anzahl	Restriktionen
1	Verwendungszweck für öffentlichen Schlüssel	DE	code	..3	M	1	5, 6
2	Operationsmodus, kodiert	DE	code	..3	M	1	2, 16, 17, 18, 19
3	Verfahren Benutzer	DE	code	..3	M	1	10
4	Wert für Modulus	DE	bin	..512	M	1	
5	Bezeichner für Modulus	DE	code	..3	M	1	12
6	Wert für Exponent	DE	bin	..512	M	1	65537
7	Bezeichner für Exponent	DE	code	..3	M	1	13

Typ: DEG
Format:
Länge:
Version: 2

Operationsmodus, kodiert

Information über den Operationsmodus für den jeweils verwendeten Kryptoalgorithmus (zur Signaturbildung oder zur Verschlüsselung).

Codierung:

Code	Operationsmodus	Verwendung
2:	Cipher Block Chaining (CBC)	Nur für Verschlüsselung erlaubt (vgl. [HBCI], Kapitel VI.2.2)
16:	ISO 9796-1 (bei RDH),	Nur für Signatur erlaubt
17:	ISO 9796-2 mit Zufallszahl (bei RDH)	Nur für Signatur erlaubt
18:	RSASSA-PKCS#1 V1.5 (bei RDH) bzw. RSAES-PKCS#1 V1.5 (bei RAH, RDH)	Nur für Signatur erlaubt Nur für Verschlüsselung erlaubt
19:	RSASSA-PSS (bei RAH, RDH)	Nur für Signatur erlaubt
999:	Gegenseitig vereinbart (ZZZ); bei DDV bedeutet dies die Bildung eines Retail-MAC für die Berechnung der Signatur	Nur für Signatur erlaubt (vgl. [HBCI], Kap. VI.2.1)

Typ: DE
Format: code
Länge: ..3
Version: 2

R

Rolle des Sicherheitslieferanten, kodiert

Kodierte Information über das Verhältnis desjenigen, der bezüglich der zu sichernden Nachricht die Sicherheit gewährleistet.

Die Wahl ist von der bankfachlichen Auslegung der Signatur, respektive vom vertraglichen Zustand zwischen Kunde und Kreditinstitut abhängig.

Codierung:

1: Der Unterzeichner ist Herausgeber der signierten Nachricht, z.B. Erfasser oder Erstsignatur (ISS)

3: Der Unterzeichner unterstützt den Inhalt der Nachricht, z.B. bei Zweitsignatur (CON)

4: Der Unterzeichner ist Zeuge, aber für den Inhalt der Nachricht nicht verantwortlich, z.B. Übermittler, welcher nicht Erfasser ist (WIT)

Typ: DE
Format: code
Länge: ..3
Version: 2

S

Schlüsselart

Information über die Art des Schlüssels.

Bei den Sicherheitsverfahren RAH und RDH steht die Schlüsselart in engem Zusammenhang mit dem Datenelement "Verwendungszweck für öffentlichen Schlüssel". Die Inhalte beider Datenelemente sind konsistent zu halten.

Codierung:

D: Schlüssel zur Erzeugung digitaler Signaturen (DS-Schlüssel)

S: Signierschlüssel

V: Chiffrierschlüssel

Der DS-Schlüssel steht nur im Zusammenhang mit einer Bankensignaturkarte zur Verfügung.

Im Falle der Bankensignaturkarte ergibt sich folgende Zuordnung zu den Kartenschlüsseln:

- DS-Schlüssel: SK.CH.DS
- Signierschlüssel: SK.CH.AUT
- Chiffrierschlüssel: SK.CH.KE

Typ: DE
Format: code
Länge: 1
Version: 2

Schlüsselname

Verwendeter Schlüsselnamen beim RAH- und RDH-Verfahren respektive die Referenz auf den Chiffrierschlüssel beim DDV-Verfahren in strukturierter Form. Mit dieser Information kann die Referenz auf einen Schlüssel hergestellt werden.

Dabei enthält das DE „Benutzerkennung“ bei Schlüsseln des Kunden die Benutzerkennung, mit der der Kunde eindeutig identifiziert wird. Bei Schlüsseln des Kreditinstituts ist dagegen eine beliebige Kennung einzustellen, die dazu dient, den Kreditinstitutsschlüssel eindeutig zu identifizieren. Diese Kennung darf weder einer anderen gültigen Benutzerkennung des Kreditinstituts noch der Benutzerkennung für den anonymen Zugang entsprechen.

Nr.	Name	Typ	Format	Länge	Status	Anzahl	Restriktionen
1	Kreditinstitutskennung	DEG	kik	#	M	1	
2	Benutzerkennung	DE	id	#	M	1	
3	Schlüsselart	DE	code	1	M	1	D, S, V
4	Schlüsselnummer	DE	num	..3	M	1	
5	Schlüsselversion	DE	num	..3	M	1	

Typ: DEG
Format:
Länge:
Version: 3

Schlüsselnummer

Schlüsselnummer des entsprechenden Schlüssels.

Typ: DE
Format: num
Länge: ..3
Version: 1

Schlüsselversion

Versionsnummer des entsprechenden Schlüssels.

Typ: DE
Format: num
Länge: ..3
Version: 1

Segmentkennung

Segmentspezifische Kennung, die jedem Segment bzw. Auftrag zugeordnet ist (z.B. "HKUEB" für "Einzelüberweisung"). Die Angabe hat in Großschreibung zu erfolgen.

Typ: DE
Format: an
Länge: ..6
Version: 1

Falls der Kunde ein Segment mit einer veralteten Versionsnummer einreicht, sollte ihm in einer entsprechenden Warnung rückgemeldet werden, dass sein Kundenprodukt aktualisiert werden sollte.

Segmentkopf

Informationen, die jedem Segment als Kopfteil vorangestellt sind. Im Unterschied zu Nachrichten enthalten Segmente jedoch keinen Abschlussteil, da das Segmentende durch das Segmentende-Zeichen markiert ist.

Im Segmentkopf stehen die Segmentkennung und Segmentversion unabhängig von der HBCI-Version (s. DE HBCI-Version) immer an derselben Stelle, damit ein Segment auch in späteren HBCI-Versionen immer eindeutig als solches identifiziert werden kann.

Nr.	Name	Typ	Format	Länge	Status	Anzahl	Restriktionen
1	Segmentkennung	DE	an	..6	M	1	
2	Segmentnummer	DE	num	..3	M	1	>=1
3	Segmentversion	DE	num	..3	M	1	
4	Bezugssegment	DE	num	..3	C	1	>=1 O: Verwendung in Kreditinstitutsnachricht N: Verwendung in Kundennachricht

Typ: DEG
Format:
Länge:
Version: 1

Segmentnummer

Information zur eindeutigen Identifizierung eines Segments innerhalb einer Nachricht. Die Segmente einer Nachricht werden in Einerschritten streng monoton aufsteigend nummeriert. Die Nummerierung beginnt mit 1 im ersten Segment der Nachricht (Nachrichtenkopf).

Typ: DE
Format: num
Länge: ..3
Version: 1

Segmentversion

Versionsnummer zur Dokumentation von Änderungen eines Segmentformats.

Die Segmentversion von administrativen Segmenten (die Segmentart 'Administration' bzw. 'Geschäftsvorfall' ist bei jeder Segmentbeschreibung angegeben) wird bei jeder Änderung des Segmentformats inkrementiert.

Bei Geschäftsvorfallssegmenten wird die Segmentversion auf logischer Ebene verwaltet, d.h. sie ist für das Auftrags-, das Antwort- und das Parametersegment des Geschäftsvorfalls stets identisch und wird inkrementiert, wenn sich das Format von mindestens einem der drei Segmente ändert.

Dieses Verfahren gilt bei Standardsegmenten einheitlich für alle Kreditinstitute. Bei verbandsindividuellen Segmenten obliegt die Versionssteuerung dem jeweiligen Verband. Der Zeitpunkt der Unterstützung einer neuen Segmentversion kann jedoch zwischen den Verbänden variieren.

Die für die jeweilige HBCI-Version gültige Segmentversion ist bei der jeweiligen Segmentbeschreibung vermerkt.

Falls der Kunde ein Segment mit einer veralteten Versionsnummer einreicht, sollte ihm in einer entsprechenden Warnung rückgemeldet werden, dass sein Kundenprodukt aktualisiert werden sollte.

Typ: DE
Format: num
Länge: ..3
Version: 1

Sicherheitsdatum und -uhrzeit

Zeitstempel, beispielsweise Datum und Uhrzeit des lokalen Rechners, an dem die elektronische Unterschrift geleistet wurde, sowie die Bedeutung des Zeitstempels.

Nr.	Name	Typ	Format	Länge	Status	Anzahl	Restriktionen
1	Datum- und Zeitbezeichner, kodiert	DE	code	..3	M	1	1, 6
2	Datum	DE	dat	#	O	1	
3	Uhrzeit	DE	tim	#	C	1	O: ‚Datum' belegt N: sonst

Typ: DEG
Format:
Länge:
Version: 2

Sicherheitsfunktion, kodiert

Bis HBCI 2.2 war die Sicherheitsfunktion die Unterscheidung zwischen DDV und RDH, wobei die 1 nur das RDH-Verfahren kennzeichnete und 2 das DDV-Verfahren. Ab FinTS 3.0 existieren beim RDH-Verfahren drei Schlüssel (DS-Schlüssel für Non-Repudiation, Signierschlüssel für Authentication und Chiffrierschlüssel für Verschlüsselung) und somit auch drei Sicherheitsfunktionen (Sicherheitsfunktion 1 bei Verwendung des DS-Schlüssels, Sicherheitsfunktion 2 bei Verwendung des Signierschlüssel und Sicherheitsfunktion 4 bei Verwendung des Chiffrierschlüssels) beim RAH- und RDH-Verfahren.

Die Sicherheitsfunktion hat ab FinTS 3.0 lediglich informatorischen Wert, da die eigentliche Steuerung über die Sicherheitsprofile und –Klassen erfolgt.

Kodierte Information über die Sicherheitsfunktion, die auf die Nachricht angewendet wird.

Codierung:

1: Non-Repudiation of Origin, für RAH, RDH (NRO)

2: Message Origin Authentication, für RAH, RDH und DDV (AUT)

4: Encryption, Verschlüsselung und evtl. Komprimierung (ENC)

Typ: DE
Format: code
Länge: ..3
Version: 2

Sicherheitsidentifikation, Details

Identifikation der im Sicherheitsprozess involvierten Parteien. Dient zur Übermittlung der CID bei kartenbasierten Sicherheitsverfahren bzw. der Kundensystem-ID bei softwarebasierten Verfahren (z.B. Speicherung der Schlüssel in einer Schlüsseldatei).

Nr.	Name	Typ	Format	Länge	Status	Anzahl	Restriktionen
1	Bezeichner für Sicherheitspartei	DE	code	..3	M	1	1, 2
2	CID	DE	bin	..256	C	1	M: Sicherheitsmedium = Chipkarte N: sonst
3	Identifizierung der Partei	DE	id	#	C	1	M: Sicherheitsmedium = Software N: sonst

Typ: DEG
Format:
Länge:
Version: 2

Sicherheitskontrollreferenz

Referenzinformation, mit der die Verbindung zwischen Signaturkopf und dazu gehörigem Signaturabschluss hergestellt werden kann. Die Sicherheitskontrollreferenz im Signaturkopf muss mit der entsprechenden Information im Signaturabschluss übereinstimmen.

Typ: DE
Format: an
Länge: ..14
Version: 1

Sicherheitsprofil

Verfahren zur Absicherung der Transaktionen, das zwischen Kunde und Kreditinstitut vereinbar wurde. Das Sicherheitsprofil wird anhand der Kombination der beiden Elemente „Sicherheitsverfahren“ und „Version“ bestimmt (z. B. RDH-3, DDV-1). Für das Sicherheitsverfahren PINTAN ist als Code der Wert PIN und als Version der Wert 1 einzustellen.

Nr.	Name	Typ	Format	Länge	Status	Anzahl	Restriktionen
1	Sicherheitsverfahren, Code	DE	code	3	M	1	DDV,RAH, RDH, PIN
2	Version des Sicherheitsverfahrens	DE	num	..3	M	1	1, 2, 3, 4, 5, 6, 7, 8, 9, 10

Typ: DEG
Format:
Länge:
Version: 1

Sicherheitsreferenznummer

Sicherheitsrelevante Nachrichtenidentifikation (Signatur-ID), welche zur Verhinderung der Doppeleinreichung, respektive Garantie der Nachrichtensequenzintegrität eingesetzt werden kann.

Bei chipkartenbasierten Verfahren ist der Sequenzzähler der Chipkarte einzustellen. Dies ist bei Typ-1 Karten der Wert „EF_SEQ“ in der Application DF_BANKING und bei SECCOS Bankensignaturkarten der Wert „usage counter“ der beiden Signierschlüssel SK.CH.DS und SK.CH.AUT.

Bei softwarebasierten Verfahren wird die Sicherheitsreferenznummer auf Basis des DE Kundensystem-ID und des DE Benutzerkennung der DEG Schlüsselnamen verwaltet.

Typ: DE
Format: num
Länge: ..16
Version: 1

Sicherheitsverfahren, Code

Code des unterstützten Signatur- bzw. Verschlüsselungsalgorithmus.

Weitere Informationen zu den Verfahren sind Kapitel B.1 zu entnehmen.

Codierung:

DDV: DES-DES-Verfahren

RAH: RSA-AES-Hybridverfahren

RDH: RSA-DES-Hybridverfahren

PIN: PIN/TAN-Verfahren

EMV: EMV-AC-Variante (S-Fkt=820, 821) bei AZS-Verfahren

Typ: DE
Format: code
Länge: 3
Version: 3

Signaturalgorithmus

Angaben zum kryptographischen Algorithmus, zu seinem Operationsmodus, so wie zu dessen Einsatz, in diesem Fall für die Signaturbildung über DDV bzw. RAH / RDH.

Nr.	Name	Typ	For-mat	Län-ge	Sta-tus	An-zahl	Restriktionen
1	Verwendung des Signaturalgorithmus, kodiert	DE	code	..3	M	1	6
2	Signaturalgorithmus, kodiert	DE	code	..3	M	1	1, 10
3	Operationsmodus, kodiert	DE	code	..3	M	1	16, 17, 18, 19, 999

Typ: DEG
Format:
Länge:
Version: 2

Signaturalgorithmus, kodiert

Kodierte Information über den Signaturalgorithmus.

Codierung:

1: DES-Algorithmus (bei DDV)

10: RSA-Algorithmus (bei RAH und RDH)

Typ: DE
Format: code
Länge: ..3
Version: 2

Sperrenkennzeichen

Information zur Begründung der Sperrung.

Codierung:

1: Schlüssel des Zertifikatseigentümers kompromittiert

501: Zertifikat ungültig wegen Verdacht auf Kompromittierung

999: gesperrt aus sonstigen Gründen

Typ: DE
Format: code
Länge: ..3
Version: 2

U

Uhrzeit

Uhrzeit eines Ereignisses (meist zusammen mit „Datum“ verwendet).

Typ: DE
Format: tim
Länge: #
Version: 1

V

Validierungsresultat

Elektronische Signatur, die zur Validierung berechnet wurde.

Typ: DE
Format: bin
Länge: ..512
Version: 1

Verfahren Benutzer

Information über das Benutzer-Verfahren, die beim öffentlichen Schlüssel angegeben wird.

Es ist nur der folgende Wert zugelassen:

10: RSA-Verfahren

Typ: DE
Format: code
Länge: ..3
Version: 2

Verschlüsselungsalgorithmus

Angaben zum kryptographischen Algorithmus, zu seinem Operationsmodus, so wie zu dessen Einsatz, in diesem Fall für die Nachrichtenverschlüsselung.

Nr.	Name	Typ	Format	Länge	Status	Anzahl	Restriktionen
1	Verwendung des Verschlüsselungs-algorithmus, kodiert	DE	code	..3	M	1	2
2	Operationsmodus, kodiert	DE	code	..3	M	1	2, 16, 17, 18, 19
3	Verschlüsselungs-algorithmus, kodiert	DE	code	..3	M	1	13
4	Wert des Algorith-musparameters, Schlüssel	DE	bin	..512	M	1	
5	Bezeichner für Algo-rithmusparameter, Schlüssel	DE	code	..3	M	1	5, 6
6	Bezeichner für Algo-rithmusparameter, IV	DE	code	..3	M	1	1
7	Wert des Algorith-musparameters, IV	DE	bin	..512	O	1	

Typ: DEG
Format:
Länge:
Version: 2

Verschlüsselungsalgorithmus, kodiert

Kodierte Information über den verwendeten Verschlüsselungsalgorithmus.

Codierung:

13: 2-Key-Triple-DES

14: AES-256 [AES]

Typ: DE
Format: code
Länge: ..3
Version: 3

Version der Filterfunktion

Version der Filterfunktion.

Typ: DE
Format: num
Länge: ..3
Version: 1

Version des Sicherheitsverfahrens

Version des unterstützten Sicherheitsverfahrens (s. „Sicherheitsverfahren, Code“).

In Kombination mit dem Sicherheitsverfahren RAH sind die folgenden Versionen gültig:

Version	Signaturverfahren	Schlüssellänge (bit)	Hashverfahren	Schlüsselart*
7	PKCS#1 PSS	gemäß [DK Krypto]	SHA-256	D, S, V
9	PKCS#1 PSS	gemäß [DK Krypto]	SHA-256	S, V
10	PKCS#1 PSS	gemäß [DK Krypto]	SHA-256	S, V

In Kombination mit dem Sicherheitsverfahren RDH sind die folgenden Versionen gültig:

Version	Signaturverfahren	Schlüssellänge (bit)	Hashverfahren	Schlüsselart*
1	ISO 9796-1	708-768	RIPEMD-160	S, V
2	DIN, ISO 9796-2	1024-2048	RIPEMD-160	S, V
3	DIN, ISO 9796-2 PKCS#1 V1.5	1024-2048	RIPEMD-160 SHA-1	D, S, V
4	PKCS#1 V1.5	1024-2048	SHA-1	D, S, V
5	PKCS#1 V1.5	1024-2048	SHA-1	S,V
6	PKCS#1 V1.5	gemäß [DK Krypto]	SHA-256	D, S, V
7	PKCS#1 PSS	gemäß [DK Krypto]	SHA-256	D, S, V
8	PKCS#1 V1.5	gemäß [DK Krypto]	SHA-256	S, V
9	PKCS#1 PSS	gemäß [DK Krypto]	SHA-256	S, V
10	PKCS#1 PSS	gemäß [DK Krypto]	SHA-256	S, V

In Kombination mit dem Sicherheitsverfahren DDV sind die folgenden Versionen gültig:

Version	Signaturverfahren	Schlüssellänge (bit)	Hashverfahren	Schlüsselart*
1	MAC	128	RIPEMD-160	S, V
2	MAC	128	SHA-256	S, V

* s. Element „Schlüsselart“

Andere als die genannten Profile sind nicht zulässig.

Um Multibankfähigkeit zu gewährleisten, ist die Unterstützung eines der Verfahren RAH-9 bzw. übergangsweise RDH-9 kunden- und kreditinstitutsseitig verpflichtend.

Typ: DE
Format: num
Länge: ..3
Version: 2

Verwendung des Hashalgorithmus, kodiert

Kodierte Information über die Verwendung des Hashalgorithmus.

Im Zusammenhang mit Hash-Funktionen ist derzeit nur folgender Wert möglich:

Codierung:

1: Owner Hashing (OHA)

Typ:	DE
Format:	code
Länge:	..3
Version:	2

Verwendung des Signaturalgorithmus, kodiert

Kodierte Information über die Verwendung des Signaturalgorithmus.

Im Zusammenhang mit Signaturbildung ist derzeit nur folgender Wert möglich:

Codierung:

6: Owner Signing (OSG)

Typ:	DE
Format:	code
Länge:	..3
Version:	2

Verwendung des Verschlüsselungsalgorithmus, kodiert

Kodierte Information über die Verwendung des Verschlüsselungsalgorithmus.

Im Zusammenhang mit der Verschlüsselung sind derzeit folgende Werte möglich:

Codierung:

2: Owner Symmetric (OSY)

Typ:	DE
Format:	code
Länge:	..3
Version:	2

Verwendungszweck für öffentlichen Schlüssel

Kodierte Information über die Verwendung des öffentlichen Schlüssels. Diese Information muss konsistent zur Schlüsselart gehalten werden.

Codierung:

5: Owner Ciphering (Chiffrierschlüssel)

6: Owner Signing (Signierschlüssel)

Typ: DE
Format: code
Länge: ..3
Version: 2

W

Wert des Algorithmusparameters, IV

Initialisierungswert für den kryptographischen Algorithmusparameter. Zur Zeit ist die Angabe eines Wertes nicht zulässig; es wird dafür folgender Initialisierungswert als Default verwendet: X'00 00 00 00 00 00 00 00'

In einer zukünftigen Version kann ein abweichender Initialisierungswert definiert werden.

Typ: DE
Format: bin
Länge: ..512
Version: 1

Wert des Algorithmusparameters, Schlüssel

Verschlüsselter Nachrichtenschlüssel für den kryptographischen Algorithmusparameter.

Typ: DE
Format: bin
Länge: ..512
Version: 1

Wert des Hashalgorithmusparameters

Initialisierungswert für den Hashalgorithmusparameter. Zur Zeit ist die Angabe eines Wertes nicht zulässig; es wird für RIPEMD-160 folgender Initialisierungswert als Default verwendet:

X'01 23 45 67 89 AB CD EF FE DC BA 98 76 54 32 10 F0 E1 D2 C3' (Little-Endian-Notation)

In einer zukünftigen Version kann ein abweichender Initialisierungswert definiert werden.

Typ: DE
Format: bin
Länge: ..512
Version: 1

Wert für Exponent

Exponent des öffentlichen Schlüssels (z.Zt. 65537). Die Kürzung um führende 0-Bytes ist empfehlenswert, aber nicht verbindlich.

Typ: DE
Format: bin
Länge: ..512
Version: 1

Wert für Modulus

Modulus des öffentlichen Schlüssels. Die Kürzung um führende 0-Bytes ist empfehlenswert, aber nicht verbindlich.

Typ: DE
Format: bin
Länge: ..512
Version: 1

Z

Zertifikat

Zertifikat eines öffentlichen Schlüssels.

Da Zertifikate Informationen beinhalten, die auch in den HBCI-Formaten enthalten sind (z.B. Zertifikatsreferenz respektive Schlüsselnamen), können Daten redundant vorkommen. Diese müssen dann auf Konsistenz überprüft werden. Bei Unstimmigkeiten hat das Zertifikat Vorrang.

Nr.	Name	Typ	Format	Länge	Status	Anzahl	Restriktionen
1	Zertifikatstyp	DE	code	1	M	1	1, 2, 3
2	Zertifikatsinhalt	DE	bin	.. 4096	M	1	

Typ: DEG
Format:
Länge:
Version: 2

Zertifikatsinhalt

Transparenter Inhalt eines Zertifikats.

Bei der Bankensignaturkarte handelt es sich hier um

- das Signaturzertifikat C_X509.CH.DS,

- das CSA-(KE-)Zertifikat C_X509.CH.AUTC/S[&KE]

- und das KE-Zertifikat C_X509.CH.KE

Typ: DE
Format: bin
Länge: ..4096
Version: 1

Zertifikatstyp

Information über Aufbau und Inhalt eines Zertifikats.

Codierung:

1: ZKA

2: UN/EDIFACT

3: X.509 v3 (gemäß [ISIS/MTT])

Typ:	DE
Format:	code
Länge:	1
Version:	2

E. ANLAGEN

E.1 Übersicht der Segmente

Nr.	Segmentname	Kennung	Sender[1]	Version
1	Anforderung eines öffentlichen Schlüssels	HKISA	K	3
2	Bestätigung der Schlüsselsperrung	HISSP	I	3
3	Schlüsseländerung	HKSAK	K	3
4	Schlüsselsperrung	HKSSP	K	3
5	Signaturkopf	HNSHK	K/I	4
6	Übermittlung eines öffentlichen Schlüssels	HIISA	I	3
7	Verschlüsselte Daten	HNVSD	K/I	1
8	Verschlüsselungskopf	HNVSK	K/I	3

[1] K: Kunde, I: Kreditinstitut