# SigBü-API: Einheitliche kartenunabhängige API für die Zugriffe auf die Signaturkarten des Signaturbündnisses

Dr. Detlef Hillen

SRC Security Research & Consulting GmbH
Graurheindorfer Strasse 149a
53117 Bonn
detlef.hillen@src-gmbh.de

**Abstract:** Im Rahmen des Signaturbündnisses haben sich mehrere Herausgeber von Signaturkarten zusammengeschlossen, um eine einheitliche SigBü-API für die Zugriffe auf die verschiedenen Signaturkarten zu spezifizieren. Durch Verwendung dieser API für die Zugriffe auf die Signaturkarten kann eine Anwendung ohne Berücksichtigung der Unterschiede der verschiedenen Signaturkarten entwickelt werden, was zu einer wesentlichen Verringerung des Aufwandes bei der Integration von Signaturkarten in Anwendungen führt. Weitere Vorteile der SigBü-API sind ihre Unabhängigkeit von verschiedenen (Klassen von) Kartenlesern und das „Erzwingen" eines gewissen „Wohlverhaltens" zwischen verschiedenen Anwendungen auf einem System bei der Nutzung der Signaturkarten.

## 1 Einleitung

Auf Initiative der Bundesregierung haben Anfang 2003 Herausgeber von Signaturkarten und Anbieter/Entwickler von Anwendungen, die eine (qualifizierte) elektronische Signatur benötigen, gemeinsam das Signaturbündnis gegründet. Übergeordnetes Ziel dieser Partnerschaft von öffentlichen und privatwirtschaftlichen Institutionen ist die Förderung der Nutzung der (qualifizierten) elektronischen Signatur. Hierzu werden unter anderem einheitliche technische Standards erarbeitet, die die Nutzung einer gegebenen Signaturkarte in möglichst vielen (Vision: allen) vorhandenen Anwendungen von eGovernment und eCommerce ermöglichen.

Die Zugriffe auf eine Signaturkarte können innerhalb einer Anwendung heute entweder über die Nutzung bekannter APIs (PKCS#11, Microsoft CSP) erfolgen (vorausgesetzt, der Herausgeber der zu integrierenden Signaturkarte stellt entsprechende Softwaremodule zur Verfügung), oder die Zugriffe werden durch den Entwickler direkt durch Nutung der Kartenkommandos auf APDU-Ebene programmiert. Beide Wege führen zu erheblichen Problemen, falls eine Anwendung nicht nur eine Signaturkarte nutzen soll, sonder unterschiedliche Signaturkarten verschiedener Kartenherausgeber im Rahmen der Anwendung nutzbar sein müssen. Da die unterschiedlichen Signaturkarten sowohl verschiedene Betriebssysteme einsetzen als auch sich bezüglich der

karteninternen Realisierung der Signatur(karten-)anwendung wesentlich unterscheiden, führt die direkte Nutzung der Kartenkommandos auf APDU-Ebene für die Nutzung der Signaturkarten zu sehr hohen Aufwänden bei der Entwicklung einer Anwendung. Andererseits arbeiten PKCS#11-Module (bzw. CSP-Module) im Allgemeinen nur mit einer Signaturkarte zusammen. Für die Nutzung unterschiedlicher Signaturkarten muss eine Anwendung daher mit mehreren verschiedenen Modulen zusammenarbeiten, was ebenfalls nicht ohne weitere Probleme möglich ist.

(Mindestens) zwei weitere Probleme existieren bezüglich der Integration von Signaturkarten in Anwendungen. Das eine Problem bezieht sich auf das gegenseitige „Wohlverhalten“ zwischen verschiedenen Anwendungen auf einem PC, die für ihre Funktionen auf eine Signaturkarte zugreifen müssen. Nach der Installation einer neuen, eine Signaturkarte nutzenden Anwendung funktioniert häufig eine vorher installierte (und bis dahin fehlerfrei arbeitende) Anwendung bei ihren Zugriffen auf die Signaturkarte nicht mehr. Das zweite Problem bezieht sich auf die Integration verschiedener Kartenleser. Einheitliche Schnittstellen für Zugriffe auf einen Kartenleser existieren nur für Klasse-1 Leser ohne Display und Tastatur. Sollen höherwertige Leser (z.B. mit einer eigenen Tastatur für die PIN-Eingabe) genutzt werden, müssen diese explizit bei der Anwendungsentwicklung berücksichtigt werden, was zu einer weiteren Steigerung des Aufwandes für die Entwicklung einer Anwendung führt. Wünschenswert wäre hier, dass eine Anwendung ohne Berücksichtigung der verschiedenen (Klassen von) Kartenleser entwickelt werden kann.

Vor dem Hintergrund der genannten Probleme haben sich im Rahmen des Signaturbündnisses mehrere Herausgeber von Signaturkarten zusammengeschlossen, um die SigBü-API zu spezifizieren. Über die Funktionen der SigBü-API kann eine Anwendung die Funktionalität verschiedener Signaturkarte in gleicher Weise nutzen, ohne dass technische Detailkenntnisse über das Betriebssystem der Karte oder über die karteninterne Realisierung der Signaturanwendung (z.B. Dateistruktur) benötigt werden. Die Funktionen der SigBü-API sind ebenfalls unabhängig von einem konkreten Kartenleser. Durch Nutzung der SigBü-API muss sich daher eine Anwendung z.B. nicht mehr darum kümmern, ob eine PIN in einer konkreten Einsatzumgebung über die PC-Tastatur oder über eine eigene Tastatur eines Kartenlesers eingegeben wird.

Die SigBü-API soll keine neue eigenständige Krypto-API (wie z.B. PKCS#11) sein. Sie enthält nur Funktionen, die in einem direkten Zusammenhang mit einer karteninternen Berechnung einer Signaturkarte stehen. So bietet die SigBü-API (natürlich) eine Funktion für das Erzeugen einer elektronischen Signatur. Sie enthält aber keine Funktion für das Verifizieren einer Signatur, da dieses durch die beteiligten Signaturkarten nicht kartenintern durchgeführt wird. Die SigBü-API kann aber selbstverständlich für die Realisierung eines PKCS#11-Moduls (bzw. eines CSP-Moduls) verwendet werden. In diesem Fall kann dann ein einzelnes PKCS#11-Modul automatisch mit allen Signaturkarten zusammenarbeiten, die die SigBü-API unterstützen.

Bei der Spezifikation der SigBü-API musste als wesentliche Vorgabe berücksichtigt werden, dass die Signaturkarten der am Bündnis beteiligten Kartenherausgeber wie gegeben ohne weiterer Änderungen die API unterstützten können. Neue Anforderungen an den Aufbau der Signaturkarten (z.B. Vorhandensein einer PKCS#15-Anwendung) durften durch die Spezifikation der SigBü-API nicht entstehen.

## 2 Architektur der Kundenumgebung

Die folgende Abbildung zeigt die angestrebte (Software-) Architektur der Kundenumgebung bei Verwendung der SigBü-API.

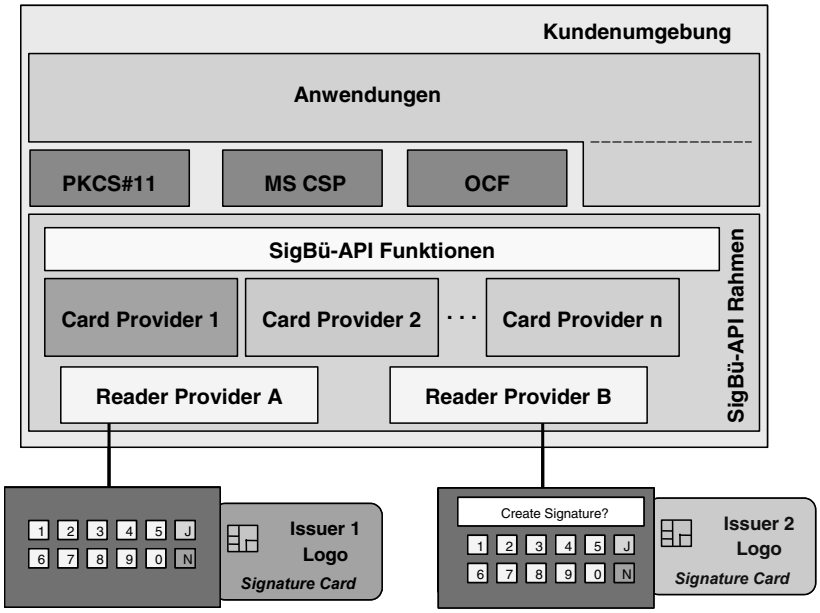


Abbildung 1: Architektur der Kundenumgebung bei Nutzung der SigBü-API

Eine Anwendung kann auf eine Signaturkarte zugreifen, indem sie entweder bekannte (Standard-) Krypto-Schnittstellen (PKCS#11, CSP, OCF) nutzt, oder die Funktionen der SigBü-API direkt nutzt.

Die Realisierungen der Krypto-Schnittstellen nutzen für ihre internen Zugriffe auf die Signaturkarten ebenfalls die Funktionen der SigBü-API. Dadurch kann eine einzelne Realisierung z.B. eines PKCS#11-Moduls für den Zugriff auf die verschiedenen Signaturkarten genutzt werden. Eine Anwendung muss daher unabhängig von der konkreten Signaturkarte immer nur mit einem Krypto-Modul zusammenarbeiten.

Bei der SigBü-API muss unterschieden werden zwischen der Spezifikation der eigentlichen API (Funktionen aus Sicht der Anwendungen) und der Spezifikation eines

Rahmenwerkes, durch das die Realisierung der SigBü-API Funktionen durch ein Zusammenspiel von "Card Providern" und "Reader Providern" geregelt wird. Im Rahmen des Signaturbündnisses werden beide Spezifikationen erstellt. Für die Entwicklung von Anwendungen ist dabei aber nur die Kenntnis der eigentlichen Funktionen der SigBü-API notwendig. Das Zusammenspiel verschiedener Komponenten bei der Realisierung der SigBü-API ist aus Sicht der Anwendungen nicht von Bedeutung.

Ein "Card Provider" wird durch den Herausgeber einer Signaturkarte zur Verfügung gestellt. Er bildet die Funktionen der SigBü-API auf die konkreten Abläufe für die Realisierung der Funktion durch die zugehörige Signaturkarte ab. Damit eine Anwendung in einer gegebenen Kundenumgebung mit einer Signaturkarte arbeiten kann, muss der zu der Karte gehörende "Card Provider" in der Kundenumgebung vorhanden sein. Wird eine Signaturkarte gesteckt und versucht die Anwendung auf die Karte zuzugreifen, wird der korrekte zugehörige "Card Provider" durch das System bestimmt, ohne dass die Anwendung eine entsprechende Auswahl unterstützen muss.

Für die Zugriffe auf einen Kartenleser nutzt ein "Card Provider" einen "Reader Provider". Der "Reader Provider" wird durch den Hersteller des Kartenlesers zur Verfügung gestellt. Für jeden angeschlossenen Kartenleser muss ein zugehöriger "Reader Provider" in dem System vorhanden sein.

Für die Schnittstelle zwischen "Card Provider" und "Reader Provider" wird durch das Signaturbündnis ebenfalls eine Spezifikation erarbeitet. Die Schnittstelle setzt dabei voraus, dass der Kartenleser eine eigene Tastatur und ein Display hat. Ist dies nicht der Fall, muss der zugehörige "Reader Provider" die fehlende Hardware simulieren. Dadurch können "Card Provider" unabhängig von den konkreten "Reader Providern" entwickelt werden.

Der "SigBü-API Rahmen" ist für das Verwalten der vorhandenen Provider zuständig. Er bietet insbesondere Funktionen für das Hinzufügen und Entfernen von "Card Providern" und "Reader Providern". In einer Kundenumgebung muss nur eine Realisierung des Rahmens vorhanden sein. Die Zuständigkeiten und das Vorgehen bei der Realisierung und Verteilung des "SigBü-Rahmens" muss noch innerhalb des Signaturbündnisses geklärt werden.

## 3 Leistungsumfang der SigBü-API

Die Funktionen der SigBü-API (aus Sicht einer Anwendung) können in die folgenden Klassen eingeteilt werden:

- Funktionen für die Verwaltung der API und der (logischen) Verbindungen zu Kartenlesern und Signaturkarten (hier nicht weiter betrachtet).

- Funktionen für die karteninterne Ausführung von kryptographischen Berechnungen.

- Funktionen, die Informationen über die Funktionalität einer Signaturkarte geben.
- Funktionen für das Auslesen von Zertifikaten aus einer Signaturkarte.
- Funktionen für das Verifizieren, Ändern und Rücksetzen von PINs.

Bei den kryptographischen Berechnungen wird weiter zwischen den "Diensten"

- Elektronische Signatur,
- Authentikation,
- Entschlüsseln und
- Hashwert-Berechnung

unterschieden. Die kryptographischen Berechnungen der ersten drei Dienste basieren dabei auf asymmetrischer Kryptographie (d.h. kryptographische Berechnungen einer Chipkarte mit symmetrischer Kryptographie können nicht über die SigBü-API angesprochen werden). Die bei den Berechnungen eingesetzten mathematischen Methoden können bei verschiedenen Signaturkarten unterschiedlich sein. Über eine Funktion der SigBü-API kann eine Anwendungen Informationen über die durch eine Signaturkarte unterstützten Methoden erhalten. Die gewünschte Methode wird durch Angabe der OID an der Schnittstelle durch die Anwendung ausgewählt. Die Beschreibung der möglichen Methoden ist nicht Bestandteil der Spezifikation der SigBü-API, stattdessen wird auf den entsprchenden Teil von ISIS-MTT [ISIS] verwiesen.

Bei der SigBü-API wird zwischen den Funktionen "Authentikation" und "Erzeugen einer elektronischen Signatur" unterschieden, obwohl beide kartenintern durch die gleichen mathematischen Methoden umgesetzt werden können. Elektronische Signatur wird bei der SigBü-API (und auch innerhalb des Signaturbündnisses) immer im Sinne von "non repudiation" bzw. "content commitment" interpretiert. Die Funktion Authentikation soll dagegen für die Authentizität von Nachrichten bzw. Komponenten eingesetzt werden (z.B. Client-Authentikation beim SSL Protokoll).

Eine Signaturkarte muss nicht alle Funktionen der SigBü-API unterstützten. Denkbar ist z.B. eine "reine Signaturkarte", die Anwendungen nur das Erzeugen einer elektronischen Signatur anbietet, die Funktionen Authentikation und Entschlüsseln jedoch nicht unterstützt. Andererseits kann eine Signaturkarte für eine Funktion auch mehrere Schlüsselpaare enthalten. Über eine Funktion der SigBü-API kann eine Anwendung Informationen erhalten, welche kryptographischen Funktionen eine Signaturkarte unterstützt und wie viel Schlüsselpaare für eine Funktion in der Karte gespeichert sind.

In einer Signaturkarte gespeicherte Zertifikate können mit einer Funktion der SigBü-API ausgelesen werden. Dabei wird zwischen User-Zertifikaten (die zu einem Schlüsselpaar gehören) und CA-Zertifikaten (die zu einem anderen Zertifikat gehören) unterschieden. Die SigBü-API macht dabei keine Vorgaben, wie viele Zertifikate in einer Karte

gespeichert sind. In einer Karte können für ein Schlüsselpaar mehrere User-Zertifikate (z.B. verschiedene Attribut-Zertifikate) gespeichert sein als auch für ein Zertifikat mehrere CA-Zertifikate (z.B. Cross-Zertifikate) gespeichert sein. Über spezielle Funktionen kann die Anwendung die Anzahl der für ein Schlüsselpaar gespeicherten User-Zertifikate bzw. die Anzahl der für ein Zertifikat gespeicherten CA-Zertifikate abfragen.

Bei dem Auslesen eines Zertifikats führt die SigBü-API keine internen Prüfungen des Zertifikats durch. Ob z.B. ein ausgelesenes User-Zertifikat ein qualifiziertes Zertifikat ist oder nicht muss durch die Anwendung anhand des Inhaltes des Zertifikats überprüft werden.

Im Allgemeinen enthält eine Signaturkarte ein oder mehrere PINs. Bezüglich des Umgangs mit PINs gibt es bei den verschiedenen Kartenherausgeber sehr unterschiedliche Anforderungen. Bei der SigBü-API wird daher im Allgemeinen das gesamte PIN-Handling durch den "Card Provider" durchgeführt, d.h. der "Card Provider" fragt bei Bedarf die PIN bei dem Benutzer ab und führt auch die Eingabe der PIN unter seiner Kontrolle durch. Eine Anwendung soll keine eigenen Aktionen für ein Abfragen/Einlesen einer PIN durchführen. Dieses Vorgehen hat den Vorteil, dass ein Benutzer immer unabhängig von der aktuellen Anwendung durch die gleichen (durch den "Card Provider" vorgegebenen) Dialoge zur PIN-Eingabe aufgefordert wird, und dass der "Card Provider" die interne Verarbeitung der PIN besser kontrollieren kann.

Das interne PIN-Handling muss bei der SigBü-API durch jeden "Card Provider" unterstützt werden. Zusätzlich kann ein "Card Provider" auch optional ein PIN-Handling durch die Anwendung akzeptieren, d.h. eine vorher durch die Anwendung eingelesene PIN wird als Parameter einer API-Funktion an den "Card Provider" übergeben. Da ein PIN-Handling durch die Anwendung jedoch nach jetzigem Stand nicht von allen Kartenherausgebern akzeptiert/unterstützt wird, sollte eine Anwendung, die mit allen Signaturkarten der Signaturbündnis-Mitglieder zusammenarbeiten möchte, auf ein eigenes PIN-Handling verzichten.

Die SigBü-API bietet keine Funktionen für die Administration einer Signaturkarte. Insbesondere gibt es keine Funktion der SigBü-API, mit der ein Zertifikat in eine Signaturkarte geladen werden kann. Das Vorgehen bei der Administration (z.B. welche Sicherheitsmechanismen werden benötigt) ist bei den verschiedenen Signaturkarten sehr unterschiedlich. Im Allgemeinen stellt ein Kartenherausgeber hierfür Spezialprogramme dem Karteninhaber zur Verfügung.

## 4 Abstrakte Schnittstelle zu den Kartenlesern

Neben der Programmierung der Zugriffe auf verschiedene Signaturkarten gibt es für die Entwicklung von Anwendungen (mindestens) ein weiteres Problem: Die Einbindung verschiedener Kartenleser.

Heute gibt es verschiedene Typen von Kartenlesers, die sich bezüglich der Bedienung durch den Karteninhaber wie folgt unterscheiden können:

- Kartenleser ohne Tastatur und ohne Display (auch Klasse-1 Leser genannt).
- Kartenleser mit Tastatur aber ohne Display (auch Klasse-2 Leser genannt).
- Kartenleser mit Tastatur und Display (auch Klasse-3 Leser genannt).

Weitere Unterscheidungsmerkmale wie die leserinterne Unterscheidung verschiedener Anwendungen, die Möglichkeit eines sicheren Software-Downloads in den Leser bzw. auch die Authentisierung eines Lesers durch ein internes Sicherheitsmodul sind bei einigen Kartenlesern vorhanden.

Zur Zeit gibt es nur für die Nutzung einfacher Klasse-1 Leser (ohne Display und ohne Tastatur) einheitliche Schnittstellen. Um höherwertige Leser nutzen zu können, muss eine Anwendung auf proprietäre Schnittstellen der Hersteller zurückgreifen.

Bei Nutzung der SigBü-API muss eine Anwendung die Zugriffe auf den Kartenleser nicht selber realisieren. Die Zugriffe werden vielmehr durch den "Card Provider" umgesetzt. Bei der Anwendungsentwicklung muss daher nicht berücksichtigt werden, was für einen Typ von Kartenleser der Karteninhaber später verwenden wird.

Die Nutzung eines Kartenlesers durch einen "Card Provider" erfolgt über einen "Reader Provider". Dieser wird durch den Hersteller des Kartenlesers zur Verfügung gestellt. Hierbei gilt (zur Zeit) die Forderung, dass jeder "Card Provider" mit jedem "Reader Provider" zusammenarbeiten können muss.

Damit "Card Provider" unabhängig von den einzelnen "Reader Providern" entwickelt werden können, wird durch das Signaturbündnis auch die Schnittstelle zwischen diesen beiden spezifiziert. Abstrakt wird dabei davon ausgegangen, dass der Kartenleser eine eigene Tastatur und ein Display hat. Für den Umgang mit diesen gelten die folgenden Regeln:

- Das Display des Kartenlesers kann nur intern durch den "Reader Provider" angesprochen werden.
- Über die Tastatur des Kartenlesers eingegebene Werte dürfen durch diesen nur an die Chipkarte gegeben werden, nicht jedoch an die Software im PC.

Weitere Anforderungen wie z.B. eine leserinterne Ablaufkontrolle werden zur Zeit im Rahmen der aktuellen Arbeiten nicht gefordert. Hier kann es evtl. jedoch in Zukunft zu Erweiterungen kommen.

Das Display des Kartenlesers soll nach den Vorgaben nur durch den "Reader Provider" ansprechbar sein. "Card Provider" bzw. andere Softwaremodule erhalten keinen direkten Zugriff auf das Display. Texte für die direkte Ansprache des Karteninhabers (z.B. die Benennung der einzelnen PINs) können aber durch den "Card Provider" vorgegeben werden. Dadurch wird erreicht, dass ein Karteninhaber immer über die gleichen Texte (unabhängig von der aktuellen Anwendung und unabhängig von dem konkreten Kartenleser) z.B. zur Eingabe einer PIN aufgefordert wird.

Bei einer korrekten Umsetzung der Anforderungen und der Nutzung eines Kartenlesers mit eigener Tastatur kann durch die SigBü-API die Sicherheit einer PIN gewährleistet werden. Weitere Sicherheitsanforderungen (z.B. "what you see is what you sign") müssen dagegen durch die Realisierung der Anwendungen sichergestellt werden.

Es können auch Kartenleser eingesetzt werden, die kein Display bzw. keine Tastatur besitzen. In diesem Fall muss aber der "Reader Provider" die fehlende Hardware simulieren. Die Schnittstelle des zugehörigen "Reader Providers" zu den "Card Providern" bleibt die gleiche wie bei einem Leser mit Tastatur und Display.

Für die Realisierung eines "Reader Providers" enthält die Spezifikation keine weiteren Vorgaben. Es bleibt dem Hersteller z.B. überlassen, ob er für die Zugriffe auf den Leser PC/SC, CT-API oder J-XFS nutzt. Durch seine Realisierung muss allerdings die Umsetzung der beiden folgenden Anforderungen ermöglicht werden:

- Zu einem gegebenen Zeitpunkt darf höchstens nur eine Anwendung (d.h. nur ein "Card Provider") einen Zugriff auf eine gegebene Signaturkarte haben.

- Eine Anwendung muss über das Stecken bzw. Ziehen einer Chipkarte informiert werden.

Anwendungen benötigen für einen Zugriff auf eine Signaturkarte einen exklusiven Zugriff auf den entsprechenden Kartenleser. Die Reservierung und Freigabe dieses exklusiven Zugriffs erfolgt jedoch bei der SigBü-API nicht durch die Anwendung selber, sondern durch die "Card Provider". Dadurch wird erreicht, dass einzelne Anwendungen die Zugriffe auf eine Signaturkarte nicht unnötig blockieren, d.h. es wird ein gewisses Wohlverhalten der einzelnen Anwendungen untereinander in Bezug auf die Nutzung der Signaturkarten erzwungen.

Die Schnittstelle zwischen einem "Card Provider" und einem "Reader Provider" wird ebenfalls im Rahmen der Arbeiten des Signaturbündnisses spezifiziert. Diese Spezifikation wird dabei mit den Herstellern von Kartenlesern abgestimmt und soll sich inhaltlich im Wesentlichen an den Vorgaben aus [MKT] orientieren.

## 5 Stand der Arbeiten, Aussicht

Die Spezifikationen der SigBü-API werden durch eine Unterarbeitsgruppe des AK Technik des Signaturbündnisses erarbeitet. Als Kartenherausgeber sind zur Zeit D-Trust, Datev, Deutsche Post Com, T-Systems sowie die vier kreditwirtschaftlichen Verlage (Bank-Verlag, DG-Verlag, Deutscher Sparkassen Verlag und VÖB-ZVD) an den Arbeiten beteiligt. Vertreter der Anwendungsseite sind ebenfalls direkt an den Arbeiten beteiligt bzw. erhalten die aktuellen Versionen der Spezifikationen zur Kommentierung.

Die Spezifikation der Funktionen des SigBü-API aus Sicht der Anwendungen liegt in der Version 1.0 vor. Für die Spezifikationen des „SigBü-API“ Rahmens und der Schnittstelle zwischen "Card Providern" und "Reader Providern" liegt ein Entwurf vor. Dieser wurde bereits mit Herstellern von Kartenlesern abgestimmt.

Es ist geplant, dass die Spezifikationen des SigBü-API bis zur nächsten Mitgliederversammlung des Signaturbündnisses (Frühjahr 2005) abgeschlossen werden.

In einer weiteren Unterarbeitsgruppe des AK Technik werden zur Zeit Konformitätskriterien für die Vergabe des "Signaturbündnis-Logos" erarbeitet. Ob bzw. wie weit die Nutzung der SigBü-API hierbei für bündniskonforme Anwendungen vorgeschrieben wird, ist noch nicht geklärt. Es kann aber davon ausgegangen werden, dass bündniskonforme Anwendungen mit allen Signaturkarten der am Signaturbündnis beteiligten Kartenherausgeber zusammenarbeiten können müssen. Aus Sicht der Anwendungsentwickler wird die Nutzung der SigBü-API in jedem Fall ein möglicher, einfacher Weg sein, diese Anforderung zu erfüllen.

## Literaturverzeichnis


[ISIS-MTT] ISIS-MTT Specifications, Part 6: Cryptographic Algorithms, T7 & TeleTrust, Version 1.1, 16.03.2004

[MKT] Multifunktionale Kartenterminals (MKT), TeleTrusT Deutschland e.V., Version 1.0, 15.04.1999

[SigBü] SigAll-API, Specification of the Application Programming Interface to the Signature Card, Signaturbündnis, Version 1.0, 14.01.2005