BSI

Bundesamt für Sicherheit in der Informationstechnik

SPEZIFIKATION ZUR ENTWICKLUNG INTEROPERABLER VERFAHREN UND KOMPONENTEN NACH SIGG/SIGV

SIGNATUR-INTEROPERABILITÄTSSPEZIFIKATION SIGI

ABSCHNITT A5 VERZEICHNISDIENST

STAND: 30.04.99
VERSION 3.0

Godesberger Allee 183, 53175 Bonn - Postfach 20 03 63, 53133 Bonn
Telefon: (0228) 9582 - 0, Telefax: (0228) 9582 - 400
Internet: www.bsi.bund.de

ABSCHNITT A5
VERZEICHNISDIENST

Andreas Berger, Alfred Giessler, Petra Glöckner, Wolfgang Schneider
GMD – Forschungszentrum Informationstechnik GmbH
Institut für Telekooperationstechnik
Dolivostr. 15, 64293 Darmstadt

INHALTSVERZEICHNIS

1 EINLEITUNG

VORZEITIGE BEENDIGUNG DER GÜLTIGKEITSDAUER EINES ZERTIFIKATES

Das grundlegende Prinzip der Vertrauensbildung in öffentlichen Sicherheitsinfrastrukturen beruht auf dem Vertrauen in die von Zertifizierungsstellen ausgestellten und von ihnen signierten Zertifikate, durch die letztlich eine Bindung der zugehörigen öffentlichen Schlüssel an die jeweiligen Schlüsselinhaber erreicht wird.

Bei der Erstellung eines Zertifikates wird u.a. eine Gültigkeitsdauer für dieses Zertifikat festgelegt und als Bestandteil in das Zertifikat integriert. Der Zeitpunkt der Zertifikatserstellung muß dabei nicht mit dem Beginn der Gültigkeitsdauer übereinstimmen. Er kann durch die private und optionale Zertifikatserweiterung *dateOfCertGen* [A1 98, Abschnitt 2.3.9.15.3] als Bestandteil des Zertifikates angegeben werden. Prinzipiell kann davon ausgegangen werden, daß ein Zertifikat während seiner gesamten Gültigkeitsdauer benutzt wird. Es gibt jedoch bestimmte Situationen, die eine vorzeitige Beendigung der Gültigkeitsdauer eines Zertifikates veranlassen bzw. erzwingen, d.h. das betreffende Zertifikat von der zuständigen Zertifizierungsstelle zu sperren ist. Die Ursachen und Gründe, die zu einer Sperrung eines Zertifikates führen, können unterschiedlicher Natur sein, wie z.B. bei den folgenden Situationen:

- Ein Teilnehmer benötigt sein Zertifikat während der restlich verbliebenen Gültigkeitsdauer nicht mehr und möchte seine Signaturkomponente unbrauchbar machen.
- Die Namensangaben eines Zertifikatsinhabers oder der zugehörigen Zertifizierungsstelle haben sich geändert und dadurch die Grundlage für die Bindung zwischen öffentlichen Schlüsseln und Personen entzogen.
- Es besteht der Verdacht oder der Nachweis einer Kompromittierung des privaten Schlüssels eines Zertifikatsinhabers oder der zugehörigen Zertifizierungsstelle und damit die Möglichkeit potentieller Angriffe.

MINIMIERUNG DES RESTRISIKOS FÜR ANGRIFFE

Sobald eine mißbräuchliche Nutzung eines Signaturschlüssels nicht mehr ausgeschlossen werden kann (z.B. aufgrund des Verlusts oder Diebstahls der Signaturkomponente), muß mit möglichen Angriffen und damit mit einer Verletzung und Gefährdung der Sicherheit während der restlich verbliebenen Gültigkeitsdauer eines Zertifikates gerechnet werden. Für diese kritischen Situationen müssen entsprechende Vorkehrungen und Sicherheitsmaßnahmen vorgesehen werden, deren Ziel es ist, das Restrisiko für Angriffe und dadurch verursachte Gefährdung der Sicherheitsinfrastruktur zu minimieren.

In diesem Zusammenhang spielen die Themenbereiche “Sperrlistenmanagement” und “Verzeichnisdienste” eine entscheidende Rolle, die in diesem Dokument eingehend dargestellt werden. Zu der geschilderten Problematik existieren bereits zahlreiche international etablierte Verfahren und Mechanismen wie z.B. CRL (certificate revocation list, Sperrliste von Zertifikaten) [ITU-T X.509 97, PKIX PRO 98], CDP (certificate distribution point, Verteilungspunkt für Sperrlisten), OpenCDP (open CRL distribution process, Mechanismen zur Verteilung von

Sperrlisten oder OCSP (online certificate status protocol, Protokoll zur Online-Abfrage des Zustandes von Zertifikaten) [PKIX OCSP 98].

Alle genannten Verfahren und Mechanismen für das Sperrlistenmanagement und Verzeichnisdienste lassen sich unabhängig von deren technischer Realisierung grob danach klassifizieren, welche Informationen sie enthalten, woher diese Informationen stammen, wie diese Informationen strukturiert sind und wem diese Informationen zugänglich gemacht werden, welchen Grad der Aktualität diese Informationen enthalten und wie sicher diese Informationen erstellt und verwaltet werden.

UNMITTELBARE KONSEQUENZEN FÜR TEILNEHMER UND ZERTIFIZIERUNGSSTELLEN

Als unmittelbare Folge der Forderung nach einer Minimierung des Restrisikos für Angriffe lassen sich für Teilnehmer und Zertifizierungsstellen folgende Konsequenzen ziehen:

Zertifizierungsstellen sollten möglichst frühzeitig

- Sperrsituationen erkennen,
- Meldungen von Teilnehmern erhalten,
- die Aktualisierung von Sperrlisten durchführen,
- Updates an ihren Verzeichnisdienst zur Verfügung stellen

und

- nur Online-Dienste mit bestimmten Sicherheitsrichtlinien zulassen.

Teilnehmer sollten möglichst frühzeitig

- Sperrsituationen erkennen,
- Meldung an die zugehörige Zertifizierungsstelle machen,
- sich aktuelle Sperrlisten beschaffen

und

- bei der Validierung von Signaturen in geeigneter Weise Online-Dienste zur Statusabfrage von Zertifikaten nutzen.

In den nächsten Kapiteln werden die Anfragen und Antworten des Verzeichnisdienstes sowie die einzelnen Sperrlistenfelder beschrieben. Jeder Abschnitt ist dabei durch die Punkte “Zweck”, “ ASN.1-Definitionen”, “ Allgemeine Konformitätsanforderungen” und “SigI-Konformitätsanforderungen” untergliedert, die folgende Bedeutung haben:

Zweck

Unter diesem Unterpunkt wird die Bedeutung des betreffenden Sperrlistenfeldes beschrieben.

ASN.1-Definitionen

Dieser Unterpunkt enthält die ASN.1-Definitionen der Verzeichnisdienstanfragen und -antworten sowie der Sperrlistenfelder gemäß der X.509-Norm. Für SigI-spezifische Sperrlistenfelder, die Objektbezeichner, private Erweiterungen oder Attribute betreffen, werden eigene ASN.1-Definitionen angegeben.

Allgemeine Konformitätsanforderungen

An dieser Stelle wird eine Zusammenstellung von wesentlichen internationalen Konformitätsanforderungen gegeben. Konformitätsanforderungen sind Aussagen in Normen oder Empfehlungen, die festlegen, was in einem bestimmten Kontext zu tun ist, was getan werden darf oder was nicht getan werden darf. Aus Gründen der Interoperabilität sind deshalb von der vorliegenden Signatur-Interoperabilitätsspezifikation auch internationale oder nationale Festlegungen, wie sie beispielsweise in [PKIX OCSP 98], [PKIX PRO 97], [DIN SigG/V 98] oder [MTRUST 96] getroffen wurden, zu beachten.

SigI-Konformitätsanforderungen

Dieser Unterabschnitt enhält Informationen über Einschränkungen und Anwendung der allgemeinen Konformitätsanforderungen hinsichtlich der durch [PKIX OCSP 98] bzw. X.509 möglichen Optionen. Im Rahmen der Signatur-Interoperabilitätsspezifikation SigI werden insbesondere weitere, durch die Normen und Empfehlungen zugelassene Strukturelemente wie spezielle Objektbezeichner, private Erweiterungen oder Attribute festgelegt oder die Benutzung bestimmter Elemente verboten.

Desweiteren enthält dieser Unterpunkt für Sperrlisten implementations-technische Informationen über einzelne Sperrlistenfelder und deren Unterstrukturen in einer taballelarischen Übersicht (Muster siehe folgende Tabelle). Die erste Spalte enthält den ASN.1-Bezeichner des betreffenden Sperrlistenfeldes. Falls ein Sperrlistenfeld aus einer zusammengesetzten Struktur besteht, so werden auch die Bezeichner der Teilfelder aufgeführt. In der zweiten Spalte werden der zugelassene Wertebereich bzw. die zugelassenen Einzelwerte der Sperrlistenfelder dargestellt. Die dritte Spalte zeigt den Hexadezimalcode des Sperrlistenfeldes. Die vierte Spalte enthält die aktuelle Länge der Beispielfelder. Aus diesen Werten wird eine maximale Länge abgeleitet, die als Empfehlung vorgegeben und in der Spalte durch Graudruck hervorgehoben wird. In den Spalten 5 bis 8 wird die Bedeutung eines Feldes entweder als obligatorisch, verboten oder optional gekennzeichnet. Die Spalten 9 bis 12 dienen zur Klassifikation von bestimmten Sperrlistenfeldern, die als Erweiterungen bezeichnet werden.

Sperrlistenformate sind nach der abstrakten ASN.1-Syntax definiert [ITU-T X.681 94] und konkrete Sperrlisten werden nach den ASN.1-Transfersyntaxregeln [ITU-T X.690 94] kodiert, deren Kenntnis vorausgesetzt wird. Datentypen und Datenwerte werden nach ASN.1 durch das rekursive Schema “Typ-Länge-Wert” kodiert. Die Typ-Komponente (auch als sog. Tag-Feld bezeichnet) spezifiziert hierbei den Typ einer Sperrlistenstruktur, die Längenkomponente enthält die Länge des folgenden Sperrlistenfeldes in Bytes und die Wert-Komponente enthält das eigentliche Nutzdatenfeld, das seinerseits aus Unterstrukturen gemäß des Schemas “Typ-Länge-Wert” aufgebaut sein kann. Der Wertebereich der Wert-Komponente ist durch das Tag-

Feld bestimmt. Prinzipiell besitzt nur die Typ-Komponente eine feste Kodierung und die beiden anderen Komponenten haben eine variable Länge. Aus diesem Grund haben die zweite und die dritte Spalte der beschrieben Tabelle überwiegend nur Beispielcharakter und dienen zur Illustration der Kodierung. Ebenso soll die in der vierten Spalte angegebene Längenangabe nur als minimale Länge verstanden werden, die ein System oder eine Anwendung unterstützen soll. In den angegebenen Beispielen sind die Tagfelder durch Fettschrift, die Längen durch Normalschrift und die Nutzdatenfelder durch Kursivschrift hervorgehoben.

Tabelle 1: Implementations-technische Informationen

BEZEICHNER	WERTEBEREICH EINZEILWERTE	HEXADEZIMALCODE	LÄNGE	ZERTIFIKATSTYP				RELEVANZ			KLASSIFIKATION			
	(BEISPIELE)	(BEISPIELE)	[BYTES]	Zertifizierungsstellen	Zeitstempeldienst	Verzeichnisdienst	Teilnehmer	obligatorisch	verboten	optional	Standarderweiterung	private Erweiterung	critical-Markierung	non-critical-Markierung
		Tag Länge *Wert*												

2 VERZEICHNISDIENST

2.1 Anforderungen aus dem Signaturgesetz und der Signaturverordnung

Anforderungen an das Sperren von Zertifikaten

Das Signaturgesetz [SigG 97] und die Signaturverordnung [SigV 97] regeln, wann Zertifikate zu sperren sind, wer Zertifikate sperren kann, wer eine Sperrung veranlassen kann, wie eine Sperrung vorgenommen werden muß und welche Angaben eine Sperrung enthalten muß.

Nach der Signaturverordnung [SigV 97, §4(1) 1.] hat ein Signaturschlüsselinhaber unverzüglich die Sperrung seines Signaturschlüssels zu veranlassen, wenn der private Signaturschlüssel verloren gegangen ist oder nicht mehr benötigt wird.

Gemäß Signaturgesetz [SigG 97, §8(1) Satz 1, §8(2), §8(3)] und Signaturverordnung [SigV 97, §9(1)] kann die Sperrung eines Zertifikates nicht nur vom Signaturschlüsselinhaber, sondern auch von seinem Vertreter oder von einer dritten Personen, von der Angaben in dieses Zertifikat aufgenommen wurden, veranlaßt werden. Desweiteren darf die Zertifizierungsstelle selbst ein Zertifikat sperren, wenn das Zertifikat auf Grund falscher Angaben erwirkt wurde oder sie ihre Tätigkeit einstellt und diese nicht von einer anderen Zertifizierungsstelle fortgeführt wird. Nach Signaturgesetz [SigG 97, §13(5) Satz 2] kann auch die zuständige Behörde (RegTP) eine Sperrung von Endanwender-Zertifikaten anordnen, falls sie einer Zertifizierungsstelle die Genehmigung entzogen hat und überzeugt ist, daß die von dieser Zertifizierungsstelle ausgestellten Zertifikate nicht hinreichend fälschungssicher sind oder daß zur Anwendung der Signaturschlüssel eingesetze technische Komponenten Sicherheitsmängel aufweisen.

Zur Sperrung eines Zertifikats muß nach Signaturverodnung [SigV 97, §9(2)] entweder ein schriftlicher oder ein mit einer digitalen Signatur versehener Antrag einer berechtigten Personen der Zertifizierungsstelle vorliegen oder ein vereinbartes Authentisierungsverfahren angewandt worden sein.

Gemäß Signaturgesetz [SigG 97, §8(1) Satz 2/3] und Signaturverordnung [SigV 97, §9(3)] ist die Sperrung von Zertifikaten mit Angabe des Datums und der Uhrzeit im Verzeichnis eindeutig kenntlich zu machen und darf nicht rückgängig gemacht werden. Dieser Zeitpunkt gibt an, ab wann die Sperrung gilt. Eine rückwirkende Sperrung ist unzulässig.

Stellt eine Zertifizierungsstelle ihre Tätigkeit ein und die von ihr ausgestellten noch gültigen Zertifikate werden von keiner anderen Zertifizierungsstelle übernommen, so muß sie nach Signaturgesetz [SigG 97, §11(1)] und Signaturverordnung [SigV 97, §14(2)] diese Zertifikate sperren.

Anforderungen an den Verzeichnisdienst

Das Signaturgesetz [SigG 97] und die Signaturverordnung [SigV 97] regeln welche Zertifikate abrufbar oder nachprüfbar gehalten werden müssen, wie lange Zertifikate in einem Verzeichnis gehalten werden müssen, wie Zertifikatsverzeichnisse geschützt werden müssen, und wie Auskünfte eines Verzeichnisdiensts beschaffen sein müssen.

Signaturschlüssel-Zertifikate und Attribut-Zertifikate müssen nach [SigG 97, §4(5)+§5(1)] des Signaturgesetzes nachprüfbar gehalten werden. Darüber hinaus müssen Zertifikate von Zertifizierungsstellen nicht nur nachprüfbar, sondern auch abrufbar sein; Endanwenderzertifikate hingegen dürfen nur mit Zustimmung des Signaturschlüssel-Inhabers abrufbar gehalten werden.

Die Signaturverordnung [SigV 97, §8] fordert, daß Zertifikate 35 Jahre nachprüfbar gehalten werden müssen. Das Nachprüfen selber ist im Einzelfall zu ermöglichen, d.h. es sind so lange Online-Mechanismen zur Verfügung zu stellen, wie der im Zertifikat aufgeführte Algorithmus mit den dazugehörigen Parametern nach [SigV 97, §17(2)] als geeignet beurteilt wird.

Die Zertifikatsverzeichnisse müssen nach Signaturgesetz [SigG 97, §14(3)] und Signaturverordnung [SigV 97, §16(4)] vor unbefugter Veränderung und unbefugtem Abruf geschützt werden, so daß nur befugte Personen Eintragungen und Veränderungen vornehmen können, die Sperrung von Zertifikaten nicht unbemerkt rückgängig gemacht werden kann und die Auskünfte auf ihre Echtheit überprüft werden können. Auskünfte vom Zertifikatsverzeichnis müssen beinhalten, ob die nachgeprüften Zertifikate zum angegeben Zeitpunkt vorhanden und nicht gesperrt waren.

Sperrlisten können nur Aufschluß über gesperrte Zertifikate geben, beinhalten aber keine Informationen über alle ausgestellten Zertifikate. Da nach Signaturverordnung [SigV 97, §16(4)] gefordert ist, daß der Verzeichnisdienst nicht nur über eine mögliche Sperrung Auskunft gibt, sondern auch darüber, ob ein Zertifikat jemals ausgestellt wurde, reichen für eine Signaturgesetz-Konformität keine Sperrlisten-Mechanismen aus, wie sie üblicherweise in diesem Zusammenhang verwendet werden.

Der Verzeichnisdienst im Sinne des Signaturgesetzes ist ein Dienst, der Auskunft über den Status aller jemals ausgestellten Zertifikate geben kann. Er verfügt über ein eigenes Signaturschlüsselpaar, so daß seine Auskünfte auf Echtheit überprüft werden können. Der Verzeichnisdienst muß von den Zertifizierungsstellen angeboten werden und sollte von der RegTP zertifiziert werden.

Der Verzeichnisdienst im Sinne des Signaturgesetzes ist von Diensten zu unterscheiden, die üblicherweise als Verzeichnisdienst (X.500 oder LDAP Directory Service) bezeichnet werden. Im folgenden Dokument wird mit Verzeichnisdienst der Verzeichnisdienst im Sinne des Signaturgesetzes bezeichnet. Andere Dienste werden zur Unterscheidung mit dem englischen Begriff Directory Service benannt.

Für den Verzeichnisdienst im Sinne des Signaturgesetzes soll eine definierte Untermenge (Profil) des Protokolls *Online Certificate Status Protocols* (OCSP) verwendet werden, welches innerhalb der Internet Engineering Task Force (IETF) entwickelt wird. Die Version 1 Draft 7 des OCSP ist Basis für dieses Dokument. Es wird erwartet, daß sich das Protokoll in der Definition noch ändern kann. Insbesondere die Identifikation der Zertifikate sowie die Ausarbeitung weiterer Ergebniswerte wurde bereits in der Arbeitsgruppe diskutiert und wird in die neueren Versionen Einfluß finden.

Der derzeitige Stand bietet aber bereits alle Möglichkeiten, um den Anforderungen des Signaturgesetzes an den Verzeichnisdienst gerecht zu werden. Da das OCSP im internationalen Kontext entwickelt wird, halten wir es für sinnvoll, kein eigenes Protokoll für die Zwecke des Signaturgesetzes zu entwickeln.

In diesem Dokument sind keine speziellen Datenaustauschformate für Sperrlisten bzw. OCSP-Anfragen und -Antworten beschrieben. Sperrlisten werden in der Regel über X.500 oder LDAP verteilt. Zum Transport von Verzeichnisdienstanfragen und -antworten über HTTP sind in Kapitel 2.3 Erläuterungen zu finden..

2.2 Online-Verzeichnisdienst OCSP

Regelmäßig verteilte Sperrlisten können Sperrinformationen nur in festen Intervallen der Anwenderinfrastruktur verfügbar machen. Der Zeitraum zwischen offizieller Sperrung eines Zertifikates und dem Bekanntwerden der Sperrung in der Anwenderinfrastruktur ist also von der Häufigkeit der Erstellung der Sperrlisten abhängig. Der vertretbare Zeitraum, der zwischen der Sperrung und dem Bekanntwerden der Sperrung vergeht, ist abhängig vom Verwendungszweck der geleisteten Signatur.

Da keine allgemein akzeptable Zeitverzögerung festgelegt werden kann, wird mit dem Verzeichnisdienst ein Auskunftsdienst definiert, der ein zeitnahes Sperren (innerhalb von ca. zehn Minuten) und eine direkte Abfrage der Sperrungen ermöglicht. Ein ebenso zeitnahes Sperren mit CRLs realisieren zu wollen würde bedeuten, daß alle zehn Minuten eine neue CRL erstellt und verteilt werden müßte. Dies ist nicht praktikabel. Die Verwendung eines Verzeichnisdienstes zur Prüfung des Sperrzustandes von Zertifikaten ersetzt die sonst übliche Methode der Prüfung mittels CRLs.

Es gibt bisher nur eine einzige Definition eines On-line Protokolls, das Statusinformationen zu Zertifikaten liefert, nämlich OCSP – online certificate status protocol [PKIX OCSP 98]. Der OCSP-Verzeichnisdienst wird durch die zwei ASN.1-Strukturtypen *OCSPRequest* und *OCSPResponse* definiert, mit denen Anfragen an den Verzeichnisdienst und zugehörige Antworten des Verzeichnisdienstes realisiert werden.

2.2.1 ANFRAGEN AN DEN VERZEICHNISDIENST

Die Anfrage zur Gültigkeit eines Zertifikates muß immer an die Zertifizierungsstelle gerichtet werden, die das betreffende Zertifikat ausgestellt hat. Dies würde bedeuten, daß der Anwenderinfrastruktur die Dienstadressen aller Zertifizierungsstellen bzw. deren Verzeichnisdienste bekannt sein müßten. Die Ableitung einer Dienstadresse aus dem Namen der Zertifizierungsstelle wurde in [A1 98, Anhang II] schon kurz beschrieben. Jeder Directory Service einer Zertifizierungsstelle sollte daher Informationen über die Dienstadressen der anderen Verzeichnisdienste halten. Dies kann auch durch einen Verweis auf ein signiertes Dokument erfolgen, welches alle Zertifizierungsstellen sowie deren Dienstadressen enthält.

Zweck

Die Anfrage des OCSP soll das zu prüfende Zertifikat festlegen. Die Anfrage besteht aus einem Anfrageblock mit einer Folge von Zertifikaten bzw. Zertifikatsbezeichnern. Jeder einzelnen Zertifikatsanfrage sowie der gesamten Anfrage können Erweiterungen angefügt werden. Zur Authentisierung des Anfragenden kann die Anfrage signiert werden. Eine Anfrage an den Verzeichnisdienst enthält folgende Daten:

- Protokollversion
- ein oder mehrere Zertifikatbezeichner
- zusätzliche Erweiterungen je Zertifikat und Erweiterungen je Anfrage
- optional den technischen Namen des Anfragenden
- eine optionale Signatur der Anfrage

Eine Anfrage bezieht sich auf ein oder mehrere Zertifikate. Üblicherweise wird die Anwenderinfrastruktur in einer Anfrage nur nach einem einzigen Zertifikat fragen, da für jedes Zertifikat einer Zertifikatskette der Verzeichnisdienst der jeweils ausstellenden Zertifizierungsstelle kontaktiert werden muß. Das Protokoll unterstützt die Anfrage mehrerer Zertifikate, um Anwenderinfrastrukturen eine Optimierung der Zahl der Anfragen zu ermöglichen.

Der Zeitpunkt einer Anfrage, auf den hin das Vorhandensein und der Zustand eines Zertifikats geprüft werden soll, muß nicht in der OCSP-Anfrage enthalten sein, es genügt, daß die OCSP-Antwort den Zeitpunkt enthält, seit dem das Zertifikat im Verzeichnis vorhanden ist, und daß darüberhinaus der Zeitpunkt der Antworterstellung und im Fall gesperrter Zertifikate der Sperrzeitpunkt enthalten sind (siehe Kapitel 2.2.2).

ASN.1-Definitionen

```
OCSPRequest          ::= SEQUENCE {
    tbsRequest               TBSRequest
    optionalSignature        [0]  Explicit Signature OPTIONAL }

TBSRequest           ::= SEQUENCE {
    version                  [0]  EXPLICIT Version DEFAULT v1,
    requestorName            [1]  EXPLICIT GeneralName OPTIONAL,
    requestList              SEQUENCE OF Request,
    requestExtensions        [2]  EXPLICIT Extensions OPTIONAL }
```

```
Version             ::= INTEGER { v1(0) }

Signature           ::= SEQUENCE {
    signatureAlgorithm    AlgorithmIdentifier,
    signature             BIT STRING,
    certs                 [0]  EXPLICIT SEQUENCE OF Certificate OPTIONAL}

Request             ::= SEQUENCE {
    reqCert               CertiD,
    singleRequestExtensions [0] EXPLICIT Extensions OPTIONAL }

CertID              ::= SEQUENCE {
    hashAlgorithm         AlgorithmIdentifier,
    issuerNameHash        OCTET STRING,
    issuerKeyHash         OCTET STRING,
    serialNumber          CertificateSerialNumber }

CertificateSerialNumber ::= INTEGER
```

Beschreibung der Einzelkomponenten:

- OCSPREQUEST

Anfragen an den OCSP-Verzeichnisdienst werden durch die Struktur *OCSPRequest* spezifiziert, die ihrerseits aus den Teilfeldern *tbsRequest* und *optionalSignature* besteht. Das Feld *tbsRequest* enthält die eigentliche Anfrage an den Verzeichnisdienst.

- TBSREQUEST

Das *tbsRequest*-Teilfeld enthält eine Versionsnummer *version*, den optionalen technischen Namen des Anfragenden *requestorName*, eine Folge von Einzelanfragen *requestList*, sowie optionale Anfrageerweiterungen *requestExtensions*.

- VERSION

Das *version*-Teilfeld enthält die Versionsnummer der OCSP-Anfrage. Die Voreinstellung für dieses Feld hat den Wert 0, der die Version 1 anzeigt.

Allgemeine Konformitätsanforderungen

Derzeit sind keine Konformitätsanforderungen in [PKIX OCSP 98] definiert.

SigI-Konformitätsanforderungen

Gegenwärtig ist nur die Version v1 mit dem Wert 0 definiert und über das DEFAULT Konstrukt auch vorbesetzt.

- REQUESTORNAME

Das optionale *requestorName*-Teilfeld enthält den Namen des Anfragenden, der seine Anfrage an den Verzeichnisdienst bei Bedarf signieren kann.

Allgemeine Konformitätsanforderungen

Falls eine OCSP-Anfrage signiert wird, so muß der Anfragende seinen Namen in dem *requestorName*- Teilfeld angeben.

SigI-Konformitätsanforderungen

Falls die Anfrage an den Verzeichnisdienst signiert ist, so muß der angegebene technische Name mit dem technischen Namen des Signierers übereinstimmen.

- REQUESTLIST

Das *requestList*-Teilfeld enthält die Anfragen selbst. Es können mehrere Anfragen zu verschiedenen Zertifikaten gleichzeitig an einen Verzeichnisdienst geschickt werden.

Allgemeine Konformitätsanforderungen

Derzeit sind keine Konformitätsanforderungen in [PKIX OCSP 98] definiert.

SigI-Konformitätsanforderungen

Die Anzahl der Anfragen im *requestList*-Feld muß mindestens eins betragen. Verzeichnisdienste müssen mehrere Anfragen verarbeiten können.

- REQUESTEXTENSIONS

Die Komponente *requestExtensions* dient zur Aufnahme von Erweiterungen, die für die gesamte OCSP-Anfrage gelten. Sie ist vom Typ *Extensions*, deren Syntax in der Teilspezifikation "A1 Zertifikate" im Abschnitt 2.3.9 beschrieben ist.

Allgemeine Konformitätsanforderungen

In [PKIX OCSP 98] wird die RequestExtension und ResponseExtension (siehe 2.2.2) *Nonce* definiert, die die Anfrage kryptografisch an die Antwort bindet, um Wiedereinspielungsangriffe (replay attacks) zu verhindern. Die Erweiterung wird durch den Objektbezeichner *id-pkix-ocsp-nonce* identifiziert.

```
id-pkix-ocsp-nonce   OBJECT IDENTIFIER ::= { id-pkix-ocsp 2 }
```

SigI-Konformitätsanforderungen

Im Rahmen des SigI Profils sind zur Zeit keine Anfrageerweiterungen für das *requestExtensions*-Teilfeld der *tbsRequest*-Struktur definiert.

- SIGNATURE

Das *optionalSignature*-Signaturfeld besteht aus der Beschreibung des verwendeten Signaturalgorithmus *signatureAlgorithm* und der Signatur *signature* selbst. Daneben kann vom Signierer noch eine Folge von Zertifikaten *certs* beigefügt werden, die dem Verzeichnisdienst das Auffinden von Zertifikaten für den Zertifizierungspfad ersparen.

Allgemeine Konformitätsanforderungen

Die Signatur wird über die gesamte *tbsRequest*-Struktur berechnet. Falls eine OCSP-Anfrage signiert wird, so muß der Anfragende seinen Namen im *requestorName*-Teilfeld der *tbsRequest*-Struktur angeben.

SigI-Konformitätsanforderungen

Bei der Erzeugung einer Verzeichnisdienstanfrage ist die Benutzung einer Signatur im *optionalSignature*-Teilfeld optional. Falls eine Anfrage von der Anwenderinfrastruktur signiert wird und eine optionale Folge von Zertifikaten *certs* enthält, so muß diese zumindest das Teilnehmerzertifikat und kann darüberhinaus optional das Zertifikat der Zertifizierungsstelle enthalten. Zusätzlich kann noch das Zertifikat der ausstellenden Behörde beigefügt werden.

- REQUEST

In der Datenstruktur *Request* werden die einzelnen Zertifikate bezeichnet, deren Status die Anwenderinfrastruktur abfragen möchte. Der Request besteht aus einer Bezeichnung des Zertifikates *certID* und einer optionalen Folge von Anfrageerweiterungen *singleRequestExtensions*, die sich auf dieses Zertifikat beziehen.

- CERTID

Ein Zertifikat wird mit der Datenstruktur *CertID* des *reqCert*-Teilfeldes eindeutig beschrieben. Die Kombination aus dem Namen des Ausstellers *issuerNameHash* und der Seriennummer *serialNumber* des betreffenden Zertifikates wird als Identifikator des Zertifikats verwendet. Das Feld *hashAlgorithm* enthält den Objektbezeichner eines geeigneten Hashalgorithmus. Das Feld *issuerNameHash* enthält das Ergebnis der Anwendung der Hashfunktion auf den nach DER-kodierten Namen des Ausstellers. Das Feld *issuerKeyHash* enthält das Ergebnis der Anwendung der Hasfunktion auf den Wert des Feldes *subjectPublicKey* (ohne den ASN.1-Tag und die Längenbeschreibung) aus dem Zertifikat des Ausstellers.

Allgemeine Konformitätsanforderungen

Der Hauptgrund für die Verwendung der beiden Hashwerte *issuerNameHash* und *issuerKeyHash*, um den Zertifikatsaussteller zu identifizieren, liegt darin, daß es möglich sein kann, daß zwei Zertifizierungsstellen den gleichen Namen wählen. Sie werden jedoch niemals den gleichen öffentlichen Schlüssel verwenden.

SigI-Konformitätsanforderungen

Um Anwenderinfrastrukturen auch Prüfungen von Zertifikaten zu ermöglichen, ohne daß das Zertifikat der entsprechenden Zertifizierungsstelle vorliegen muß, wird im SigI-Profil für Anfragen an den Verzeichnisdienst erlaubt, als *issuerKeyHash*-Teilfeld einen Octet-String der Länge 0 zu kodieren. Damit genügt das Wissen über den Namen des Ausstellers und die Seriennummer des Zertifikates. Die Antworten des Verzeichnisdienstes müssen aber immer den *issuerKeyHash* mit dem entsprechenden Wert belegen.

- SINGLEREQUESTEXTENSIONS

Die Komponente *singleRequestExtensions* dient zur Aufnahme von Erweiterungen, die für einzelne Anfragen gelten. Sie ist vom Typ *Extensions*, deren Syntax in der Teilspezifikation "A1 Zertifikate" im Abschnitt 2.3.9 beschrieben ist.

Allgemeine Konformitätsanforderungen

Die Unterstützung von Erweiterungen ist optional. Das *critical* flag sollte für keine Erweiterung gesetzt werden. Unbekannte Erweiterungen dürfen ignoriert werden, sofern das *critical* flag nicht gesetzt ist.

SigI-Konformitätsanforderungen

Im Rahmen des SigI Profils ist eine optionale Erweiterung *retrieveIfAllowed* für das *singleRequestExtensions*-Teilfeld der *Request*-Struktur vorgesehen, mit der die anfragende Anwenderinfrastruktur den Verzeichnisdienst anweisen kann, bei der Antwort das gesamte betroffene Zertifikat zu liefern, sofern dieses abrufbar gehalten wird. Die Erweiterung muß als *critical* markiert werden.

Außerdem wird eine obligatorische Erweiterung *certHash* für das *singleRequestExtensions*-Teilfeld der *Request*-Struktur vorgesehen, die den Hashwert des betreffenden Zertifikates enthält. Die Erweiterung muß als *non-critical* markiert werden. Diese Komponente etabliert eine kryptographische Bindung zwischen der Bytefolge des Zertifikates und der Antwort des Verzeichnisdienstes.

ASN.1 Definitionen

```
id-sigi OBJECT IDENTIFIER   ::=  { 1 3 36 8 }
id-sigi-at OBJECT IDENTIFIER ::= { 1 3 36 8 3 }
id-sigi-at-retrieveIfAllowed OBJECT IDENTIFIER ::=
                                          { 1 3 36 8 3 9 }
retrieveIfAllowed EXTENSION ::=  {
   SYNTAX                        BOOLEAN DEFAULT FALSE
   IDENTIFIED BY                 id-sigi-at-retrieveIfAllowed }
id-sigi-at-certHash OBJECT IDENTIFIER ::=  { 1 3 36 8 3 13 }
certHash EXTENSION          ::=  {
   SYNTAX                        CertHashSyntax
   IDENTIFIED BY                 id-sigi-at-certHash }
CertHashSyntax              ::=  SEQUENCE {
   hashAlgorithm                 AlgorithmIdentifier,
   certificateHash               OCTET STRING }
```

2.2.2 ANTWORTEN DES VERZEICHNISDIENSTES

Der Prüfzeitpunkt muß weder in der OCSP-Anfrage noch in der OCSP-Antwort enthalten sein, es genügt, daß die OCSP-Antwort den Erstellungszeitpunkt der Antwort und im Fall gesperrter Zertifikate den Sperrzeitpunkt enthält. Der Verzeichnisdienst liefert somit nicht direkt die Information, daß zu einem bestimmten Zeitpunkt ein Zertifikat gesperrt war, sondern er liefert den Zeitpunkt, ab dem ein Zertifikat gesperrt wurde. Die Anwenderinfrastruktur muß diesen Zeitpunkt in Bezug auf die zu prüfende Signatur auswerten, um festzustellen, ob das Zertifikat zum fraglichen Zeitpunkt bereits gesperrt war. Ist zur aktuellen Zeit ein Zertifikat nicht gesperrt, so war es auch zu keinem früheren Zeitpunkt gesperrt. Somit ist auch im Fall nicht gesperrter Zertifikate der Prüfzeitpunkt irrelevant. Auch bei der Frage nach unbekannten Zertifikaten, d.h. nach Zertifikaten, die nicht von der zugehörigen Zertifizierungsstelle ausgestellt wurden, ist der Prüfzeitpunkt nicht relevant. Ein Zertifikat, das nicht von der zugehörigen Zertifizierungsstelle ausgestellt wurde, war niemals und wird niemals in der Liste der ausgestellten Zertifikate enthalten sein. Es gibt jedoch auch die Möglichkeit, daß ein Zertifikat zwar schon ausgestellt, aber noch nicht in den Verzeichnisdienst eingestellt wurde. Hierfür wurde die SigI-spezifische Erweiterung *certInDirSince* eingeführt, die in einer OCSP-Antwort enthalten sein muß.

Die Antwort des Dienstes kann also für jedes angefragte Zertifikat grundsätzlich drei verschiedene Ergebnisse liefern: das Zertifikat ist vorhanden seit dem Zeitpunkt <*certInDirSince*> und nicht gesperrt, es ist vorhanden seit dem Zeitpunkt <*certInDirSince*> und gesperrt oder es ist nicht vorhanden (es befindet sich nicht in der Liste der ausgestellten Zertifikate dieser Zertifizierungsstelle). Die beiden ersten Ergebnisse können anstatt der Identifikation des Zertifikates auch das Zertifikat selbst enthalten. Jede dieser Antworten muß vom Verzeichnisdienst signiert werden. Die OCSP-Antwort enthält den Zeitpunkt, seit dem das Zertifikat in den Verzeichnisdienst gestellt wurde, und darüberhinaus im Fall nicht gesperrter Zertifikate die aktuelle Zeit *producedAt* und bei gesperrten Zertifikaten den Sperrzeitpunkt.

1. Das Zertifikat ist vorhanden seit dem Zeitpunkt <*certInDirSince*> und nicht gesperrt

Die Antwort enthält den Antwortzeitpunkt in der Teilkomponente *producedAt*. Damit ist sichergestellt, daß das Zertifikat auch zu keinem früheren Zeitpunkt gesperrt war. Sofern das Zertifikat abrufbar gehalten wird und der Abruf bei der Anfrage gewünscht wurde, liefert der Dienst das Zertifikat. Andernfalls wird nur die Statusinformation über das Zertifikat in der Antwort angegeben. Die Antwort “nicht gesperrt” wird mit der aktuellen Uhrzeit in der Teilkomponente *producedAt* versehen und vom Verzeichnisdienst signiert. Hierdurch werden Angriffe durch Wiedereinspielen alter Antworten verhindert.

2. Das Zertifikat ist vorhanden seit dem Zeitpunkt <*certInDirSince*> und gesperrt

Die Antwort enthält den Sperrzeitpunkt *revocationTime* und optional auch den Sperrgrund *revocationReason*. Sofern das Zertifikat abrufbar gehalten wird und der Abruf bei der Anfrage gewünscht wurde, liefert der Dienst das Zertifikat. Andernfalls wird nur die die Statusinformation *certStatus* über das Zertifikat in der Antwort angegeben. Ein Angriff durch Wiedereinspielen von Antworten mit der Information der Zertifikatssperrung ist nicht möglich. Aus-

künfte mit Sperrung können deshalb vorproduziert und signiert werden, sie enthalten den Zeitpunkt der Sperrung. Die Auswertung des Sperrzeitpunkts in Bezug auf die zu prüfende Signatur zu dem angegebenen Zeitpunkt muß von der Anwenderinfrastruktur durchgeführt werden.

3. Das Zertifikat war nicht ausgestellt

Diese Antwort enthält das entsprechende Ergebnis “nicht existent”, die aktuelle Uhrzeit sowie den Parameter zur Zertifikatsidentifizierung aus der Anfrage. Eine solche Antwort bedeutet, daß der befragte Verzeichnisdienst keine Auskunft über dieses Zertifikat geben kann, da es nicht von der zugehörigen Zertifizierungsstelle ausgestellt wurde oder es noch nicht in den Verzeichnisdienst eingestellt wurde.

ZUSAMMENSTELLUNG DER PRINZIPIELL IN DEN ANTWORTEN DES VERZEICHNISDIENSTES MÖGLICHEN ZEITPUNKTE:

producedAt

Die Komponente *producedAt* enthält den Zeitpunkt der Signatur dieser Antwort durch den Verzeichnisdienst. Dieser Zeitpunkt muß nicht mit der aktuellen Uhrzeit übereinstimmen, da Antworten bei bestimmten lokalen Sicherheitspolitiken wiederverwendet werden können. Der Zeitpunkt sollte nicht in der Zukunft liegen und die Zeitpunkte *thisUpdate* und *revocationTime* stimmen mit *producedAt* überein oder müssen vor diesem liegen. Der Zeitpunkt *nextUpdate* kann in der Zukunft liegen, sofern die Antwort auf einer Sperrliste mit einem festgelegtem Gültigkeitszeitraum basiert.

thisUpdate

Die Komponente *thisUpdate* enthält den Zeitpunkt, für den die hier gemachte Aussage gültig ist. Beim On-line Verzeichnisdienst stimmt dieser Zeitpunkt mit dem Zeitpunkt *producedAt* überein.

nextUpdate

Die Komponente *nextUpdate* enthält einen Hinweis, wann die Information, auf der die Antwort basiert, erneuert wird. Dieser Zeitpunkt ist nur für OCSP Antworten sinnvoll, die auf CRLs basieren. Beim On-line Verzeichnisdienst stimmt dieser Zeitpunkt immer mit dem Zeitpunkt *thisUpdate* überein.

revocationTime

Die Komponente *revocationTime* gibt bei gesperrtem Zertifikat den Zeitpunkt an, zu dem die Sperrung aktiv wurde, d.h. der Sperreintrag gemacht und über den Verzeichnisdienst für die Nutzer des Gesamtsystems verfügbar war.

certInDirSince

Der Zeitpunkt *certInDirSince* wurde eingeführt, um folgenden Sonderfall abzufangen: Ein Zertifikat wird in den Verzeichnisdienst eingestellt, wobei der Zeitpunkt der Einstellung in

den Verzeichnisdienst nach dem Beginn des Gültigkeitszeitraumes des Zertifikates liegt. Anfragen zu einem Zertifikat vor der Einstellung in den Verzeichnisdienst würden mit ungültig beantwortet werden (Antwort “Zertifikat nicht vorhanden”). Anfragen zu diesem Zertifikat nach der Einstellung würden mit “Zertifikat vorhanden seit und nicht gesperrt” beantwortet werden. Um diese widersprüchlichen Antworten klären zu können, wurde die SigI-spezifische Erweiterung *certInDirSince* eingeführt, die den Zeitpunkt enthält, seit dem das Zertifikat im Verzeichnis vorhanden ist.

ASN.1-Definitionen

```
OCSPResponse            ::=   SEQUENCE {
    responseStatus              OCSPResponseStatus,
    responseBytes               [0]   EXPLICIT ResponseBytes OPTIONAL }

OCSPResponseStatus      ::=   ENUMERATED {
    successful                  (0),
    malformedRequest            (1),
    internalError               (2),
    tryLater                    (3),
    -- not used                 (4),
    sigRequired                 (5),
    unauthorized                (6) }

ResponseBytes           ::=   SEQUENCE {
    responseType                OBJECT IDENTIFIER,
    response                    OCTET STRING }

id-ad OBJECT IDENTIFIER                 ::= { 1 3 6 1 5 5 7 48 }

id-pkix-ocsp OBJECT IDENTIFIER          ::= { 1 3 6 1 5 5 7 48 1 }

id-pkix-ocsp-basic OBJECT IDENTIFIER    ::= { 1 3 6 1 5 5 7 48 1 1 }

BasicOCSPResponse       ::=   SEQUENCE {
    tbsResponseData             ResponseData,
    signatureAlgorithm          AlgorithmIdentifier,
    signature                   BIT STRING,
    certs                       [0] EXPLICIT SEQUENCE OF Certificate OPTIONAL}

ResponseData            ::=   SEQUENCE {
    version                     [0] EXPLICIT Version DEFAULT v1,
    responderID                 ResponderID,
    producedAt                  GeneralizedTime,
    responses                   SEQUENCE OF SingleResponse,
    responseExtensions          [1] EXPLICIT Extensions OPTIONAL }

ResponderID             ::=   CHOICE {
    byName                      [1] Name,
    byKey                       [2] KeyHash }

KeyHash                 ::=   OCTET STRING

SingleResponse          ::=   SEQUENCE {
    certID                      CertID,
    certStatus                  CertStatus,
    thisUpdate                  GeneralizedTime,
```

```
    nextUpdate                  [0] EXPLICIT GeneralizedTime OPTIONAL,
    singleExtensions            [1] EXPLICIT Extensions OPTIONAL }
CertID                  ::=     SEQUENCE {
    hashAlgorithm               AlgorithmIdentifier,
    issuerNameHash              OCTET STRING,
    issuerKeyHash               OCTET STRING,
    serialNumber                CertificateSerialNumber }
CertificateSerialNumber     ::=  INTEGER


CertStatus              ::=     CHOICE {
    good                        [0] IMPLICIT NULL,
    revoked                     [1] IMPLICIT RevokedInfo,
    unknown                     [2] IMPLICIT UnknownInfo }
RevokedInfo             ::=     SEQUENCE {
    revocationTime              GeneralizedTime,
    revocationReason            [0] EXPLICIT CRLReason OPTIONAL }
UnknownInfo             ::=     NULL
CRLReason               ::=     ENUMERATED {
    unspecified                 (0),
    keyCompromise               (1),
    cACompromise                (2),
    affiliationChanged          (3),
    superseded                  (4),
    cessationOfOperation        (5),
    certificateHold             (6),
    removeFromCRL               (8) }
```

Beschreibung der Einzelkomponenten

- OCSPRESPONSE

Die Antwort des Verzeichnisdienstes *OCSPResponse* besteht aus dem allgemeinen Ergebnis der Anfrage *responseStatus* und dem optionalen Inhalt der Antwort *responseBytes*.

- OCSPRESPONSESTATUS

Das *responseStatus*-Feld der OCSPResponse-Struktur dient zur Anzeige des Ergebnisses einer Anfrage an den Verzeichnisdienst. Die möglichen Werte für dieses Feld sind in der folgenden Tabelle beschrieben.

Allgemeine Konformitätsanforderungen

Falls eine Anfrage vom Verzeichnisdienst erfolgreich bearbeitet werden konnte, so ist die explizite Antwort im *responseBytes*-Teilfeld von *OCSPResponse* enthalten.

Falls eine Anfrage ein Format enthält, das vom Verzeichnisdienst nicht bearbeitet werden kann, so erfolgt keine explizite Antwort im *responseBytes*-Teilfeld von *OCSPResponse*.

Falls ein interner Fehler im Verzeichnisdienst auftritt, so erfolgt keine explizite Antwort im *responseBytes*-Teilfeld von *OCSPResponse*.

Tabelle 2: Ergebnismeldungen des OCSP-Verzeichnisdienstes

ERGEBNIS DER ANFRAGE		BEDEUTUNG
NUMMER	NAME	
0	successful	Erfolgreiche Bearbeitung einer Anfrage
1	malformedRequest	nicht erfolgreiche Bearbeitung einer Anfrage wegen fehlerhaftem Anfrage-Format
2	internalError	Auftreten eines internen Fehlers beim Verzeichnisdienst
3	tryLater	Temporäre Nicht-Verfügbarkeit des Verzeichnisdienstes
4	certRequired	Forderung nach explizitem Vorhandensein des Zertifikates im *cert*-Teilfeld einer Anfrage, über das Auskünfte angefordert werden
5	sigRequired	Forderung nach signierten Anfragen

Falls der Verzeichnisdienst temporär nicht verfügbar ist, so erfolgt keine explizite Antwort im *responseBytes*-Teilfeld von *OCSPResponse.*

Falls der Verzeichnisdienst das explizite Vorhandensein des Zertifkates, über das Auskünfte eingeholt werden, im *cert*-Teilfeld einer Anfrage verlangt, so erfolgt keine explizite Antwort im *responseBytes*-Teilfeld von *OCSPResponse*.

Falls der Verzeichnisdienst signierte Anfragen verlangt und nicht-signierte Anfragen erhält, so erfolgt keine explizite Antwort im *responseBytes*-Teilfeld von *OCSPResponse*.

SigI-Konformitätsanforderungen

Als Ergebnisse von Verzeichnisdienstanfragen sind im Rahmen des SigI-Profiles die in der folgenden Tabelle dargestellten Ergebnisse erlaubt.

Tabelle 3: Ergebnismeldungen des SigI-OCSP-Verzeichnisdienstes

ERGEBNIS DER ANFRAGE			RELEVANZ		
NUMMER	NAME	BEDEUTUNG	obligatorisch	verboten	optional
0	successful	erfolgreiche Bearbeitung einer Anfrage	v		
1	malformedRequest	nicht erfolgreiche Bearbeitung einer Anfrage wegen fehlerhaftem Anfrage-Format	v		
2	internalError	Auftreten eines internen Fehlers beim Verzeichnisdienst	v		
3	tryLater	temporäre Nicht-Verfügbarkeit des Verzeichnisdienstes	v		
4	certRequired	Forderung nach explizitem Vorhandensein des Zertifikates im *cert*-Teilfeld einer Anfrage, über das Auskünfte angefordert werden		v	
5	sigRequired	Forderung nach signierten Anfragen		v	

- RESPONSEBYTES

Das *responseBytes*-Feld der *OCSPResponse*-Struktur enthält immer den Objektbezeichner *id-pkix-ocsp-basic* im *responseType*-Feld, sowie die Kodierung von *BasicOCSPResponse* im *response*-Feld der *ResponseBytes*-Struktur.

- BASICOCSPRESPONSE

Das *response*-Antwortfeld der *ResponsBytes*-Struktur vom Typ *BasicOCSPResponse* besteht aus den eigentlichen Daten der Antwort *tbsResponseData*, einer Beschreibung eines geeigneten Signaturalgorithmus *signatureAlgorithm*, dem Wert der Signatur *signature* selbst, sowie einer optionalen Folge von Zertifikaten *certs*, die den Zertifizierungspfad zum Verzeichnisdienst enthalten. Der Anwenderinfrastruktur wird bei der Verifikation der Signatur des Verzeichnisdienstes der Abruf weiterer Zertifkate erspart.

Allgemeine Konformitätsanforderungen

Falls die *BasicOCSPResponse*-Komponente eine Folge von Zertifikaten enthält, so darf deren Reihenfolge beliebig sein.

SigI-Konformitätsanforderungen

Falls die *BasicOCSPResponse*-Komponente eine Liste von Zertifikaten enthält, so sollte diese Liste das Zertifikat des Verzeichnisdienstes und das passende Zertifikat der ausstellenden Behörde enthalten. Optional kann noch das Zertifikat der Zertifizierungsstelle und das dazu passende Zertifikat der ausstellenden Behörde beigefügt werden.

- RESPONSEDATA

Die Nutzdaten der *BasicOCSPResponse*-Antwort sind in der Struktur *ResponseData* abgelegt, das eine Folge der Teilfelder *version, responderID, producedAt, responses* und optionalen *responseExtensions* ist.

- RESPONDERID

Die Identifizierung des Verzeichnisdienstes erfolgt über das *responderID*-Feld, welches entweder in der *byName*-Form den Namen des Verzeichnisdienstes enthält, so wie er in dem Zertifikat verzeichnet ist, welches zur Signatur der Antwort paßt oder in der *byKey*-Form den SHA-1-Hashwert (der nicht über das Tag- und Längenfeld gebildet wird) des öffentlichen Schlüssels des Verzeichnisdienstes enthält, wie es in [PKIX OCSP 98] festgelegt wurde.

Allgemeine Konformitätsanforderungen

Bei der *byKey*-Form zur Identifizierung des Verzeichnisdienstes erfolgt die SHA-1-Hashwertbildung nicht über das Tag- und Längenfeld des öffentlichen Schlüssels des Verzeichnisdienstes.

SigI-Konformitätsanforderungen

Im Rahmen der SigI-Spezifikation ist die Benutzung der *byName*-Form zur Identifikation des Verzeichnisdienstes obligatorisch und die der *byKey*-Form verboten.

- PRODUCEDAT

Der Zeitpunkt, zu dem diese Antwort signiert wurde, wird in dem Feld *producedAt* angezeigt, das vom Typ *GeneralizedTime* ist.

Allgemeine Konformitätsanforderungen

Sofern ein Verzeichnisdienst Antworten für gesperrte Zertifikate erstellt und für späteren Gebrauch signiert, kann dieser Zeitpunkt auch vom Zeitpunkt der Anfrage abweichen. In diesem Fall muß der Zeitpunkt von *producedAt* zwischen dem Sperrzeitpunkt und dem Zeitpunkt der Anfrage liegen.

SigI-Konformitätsanforderungen

Bei der Kodierung des *producedAt*-Feldes muß für GeneralizedTime das Format YYYY MMDDHHSSZ (siehe Abschnitt 3.2.3.4) genommen werden.

- SINGLERESPONSES

Das Feld *responses* enthält eine Folge von Einzelantworten *SingleResponses*, die den entsprechenden Anfragen zugeordnet werden.

Allgemeine Konformitätsanforderungen

Die Reihenfolge der Einzelantworten *SingleResponses* im *responses*-Feld ist beliebig, und die Anwenderinfrastruktur muß die einzelnen Felder der Antworten den jeweiligen Anfragen zuordnen. Der Verzeichnisdienst muß zu jeder einzelnen Anfrage aus dem *requestList*-Feld eine Antwort im dem Feld *responses* bereitstellen. Sofern der Verzeichnisdienst dies in einem konkreten Fall nicht kann, muß er mit der Fehlermeldung *internalError* antworten.

SigI-Konformitätsanforderungen

Es gelten die allgemeinen Konformitätsanforderungen.

- RESPONSEEXTENSIONS

Die Komponente *responseExtensions* dient zur Aufnahme von Erweiterungen, die für die gesamte OCSP-Antwort gelten. Sie ist vom Typ *Extensions*, der in der Teilspezifikation “A1 Zertifikate” im Abschnitt 2.3.9 beschrieben ist.

Allgemeine Konformitätsanforderungen

In [PKIX OCSP 98] wird die RequestExtension (siehe 2.2.1) und ResponseExtension *Nonce* definiert, die die Anfrage kryptografisch an die Antwort bindet, um Wiedereinspielungsangriffe (replay attacks) zu verhindern. Die Erweiterung wird durch den Objektbezeichner *id-pkix-ocsp-nonce* identifiziert.

```
id-pkix-ocsp-nonce   OBJECT IDENTIFIER ::= { id-pkix-ocsp 2 }
```

SigI-Konformitätsanforderungen

Im Rahmen des SigI Profils sind zur Zeit keine neuen Erweiterungen für das *responseExtensions*-Teilfeld der *ResponseData*-Struktur definiert.

- SINGLERESPONSE

Antworten zu den einzelnen Zertifikaten werden in der Struktur *SingleResponse* kodiert, das seinerseits eine Folge der Teilfelder *certID*, *certStatus* und *thisUpdate*, sowie der optionalen Teilfelder *nextUpdate* und *singleExtensions* ist.

- CERTID

Ein Zertifikat wird mit der Datenstruktur *CertID* des *responses*-Teilfeldes eindeutig beschrieben. Die Kombination aus dem Namen des Ausstellers *issuerNameHash* und der Seriennummer *serialNumber* des betreffenden Zertifikates wird als Identifikator des Zertifikates verwendet. Das Feld *hashAlgorithm* enthält den Objektbezeichner eines geeigneten Hashalgorithmus. Das Feld *issuerNameHash* enthält das Ergebnis der Anwendung der Hashfunktion auf den nach DER-kodierten Namen des Ausstellers. Das Feld *issuerKeyHash* enthält das Ergebnis der Anwendung der Hasfunktion auf den Wert des Feldes *subjectPublicKey* (ohne den ASN.1-Tag und die Längenbeschreibung) aus dem Zertifikat des Ausstellers.

Allgemeine Konformitätsanforderungen

Der Verzeichnisdienst liefert üblicherweise im *certID*-Teilfeld nur einen Verweis auf das entsprechende Zertifikat.

SigI-Konformitätsanforderungen

Im SigI-Profil für Anfragen an den Verzeichnisdienst ist es erlaubt, als *issuerKeyHash*-Teilfeld einen Octet-String der Länge 0 zu kodieren. Die Antworten des Verzeichnisdienstes müssen aber immer den *issuerKeyHash* mit dem entsprechenden Wert belegen.

- CERTSTATUS

Das *certStatus*-Teilfeld enthält Informationen über den Sperrzustand des in *certID* bezeichneten oder enthaltenen Zertifikates.

Der Zustand *good* sagt aus, daß das Zertifikat von der zugehörigen Zertifizierungsstelle ausgestellt wurde, dem Verzeichnisdienst bekannt ist und zum Zeitpunkt *thisUpdate* nicht gesperrt ist.

Der Zustand *revoked* sagt aus, daß das Zertifikat von der zugehörigen Zertifizierungsstelle ausgestellt wurde, dem Verzeichnisdienst bekannt ist und gesperrt ist.

Der Zustand *unknown* sagt aus, daß das Zertifikat von der zugehörigen Zertifizierungsstelle nicht ausgestellt wurde und dieser nicht bekannt ist.

Allgemeine Konformitätsanforderungen

Falls ein *revoked* Zustand vorliegt, so wird der Sperrzeitpunkt im Teilfeld *revocationTime* und optional der Sperrgrund im Teilfeld *revocationReason* der Struktur *RevokedInfo* abgelegt.

Für die Sperrgründe gelten die gleichen Regelungen wie bei den Sperrgründen für Sperrlisten (siehe 3.2.3.6.1).

Falls ein *unknown* Zustand vorliegt, so muß im Feld *certID* der Struktur *SingleResponse* der Inhalt des *certID*-Feld in der Struktur *Request* der Anfrage wiederholt werden.

SigI-Konformitätsanforderungen

Im SigI-Profil für den Verzeichnisdienst muß in Verbindung mit den Zuständen *good* und *revoked* die SigI-spezifische Erweiterung *CertInDirSince* für das *singleExtensions*-Teilfeld der *SingleResponse*-Struktur gesetzt sein, die den Zeitpunkt anzeigt, seit wann das in *certID* bezeichnete oder enthaltene Zertifikat im Verzeichnis vorhanden ist. Im Falle des Zustands *unknown* muß die SigI-spezifische Erweiterung *CertInDirSince* nicht in der OCSP-Antwort enthalten sein.

- THISUPDATE

Im *thisUpdate*-Teilfeld wird der Zeitpunkt der Gültigkeit der Antwort des Verzeichnisdienstes abgelegt.

Allgemeine Konformitätsanforderungen

OCSP-Antworten, deren *thisUpdate* Zeitpunkt nach der lokalen Systemzeit liegt, sollten als ungültig erachtet werden.

SigI-Konformitätsanforderungen

Die Komponente *thisUpdate* enthält den Zeitpunkt, für den die hier gemachte Aussage gültig ist. Beim on-line Verzeichnisdienst stimmt dieser Zeitpunkt mit dem Zeitpunkt *producedAt* überein.

- NEXTUPDATE

Im optionalen *nextUpdate*-Teilfeld kann der Ablaufzeitpunkt der Gültigkeit der Antwort eingetragen werden. Bevor oder ab diesem Zeitpunkt gibt es neuere Informationen über den Status von Zertifikaten.

Allgemeine Konformitätsanforderungen

OCSP-Antworten, die keinen *nextUpdate* Zeitpunkt enthalten, zeigen an, daß jederzeit neuere Statusinformation zu Zertifikaten vorhanden sein kann.

SigI-Konformitätsanforderungen

Im SigI-Profil ist die Benutzung des *nextUpdate*-Teilfeld verboten, da Antworten des Verzeichnisdienstes im Kontext des Signaturgesetzes nicht für Zeiträume gültig sind.

- SINGLEEXTENSIONS

Die Komponente *singleExtensions* dient zur Aufnahme von Erweiterungen, die für einzelne Antworten gelten. Sie ist vom Typ *Extensions*, der in der Teilspezifikation "A1 Zertifikate" im Abschnitt 2.3.9 beschrieben ist.

Allgemeine Konformitätsanforderungen

Die Anwenderinfrastruktur muß Antworten mit vorhandenem *singleExtensions*-Teilfeld verarbeiten können. Sie muß aber nicht auf deren Inhalt reagieren. Insbesondere können Erweiterung die Aussagen im *certStatus*-Teilfeld weder modifizieren oder einschränken.

SigI-Konformitätsanforderungen

Im Rahmen der BSI-Spezifikation soll der Zeitpunkt, zu dem das nachgefragte Zertifikat in den Verzeichnisdienst gestellt wurde, in der Antwort des OCSP-Dienstes enthalten sein. Hierfür wurde die obligatorische SigI-spezifische Erweiterung *CertInDirSince* für das *singleExtensions*-Teilfeld der *SingleResponse*-Struktur vorgesehen, mit der der Zeitpunkt angegeben wird, seit wann das Zertifikat im Verzeichnis vorhanden ist. Die Erweiterung muß als *critical* markiert werden.

Sofern bei der Anfrage für ein betroffenes Zertifikat die SigI-Erweiterung *retrieveIfAllowed* gesetzt war und der Inhaber des Zertifikates dem öffentlichen Abruf zugestimmt hat, muß der Verzeichnisdienst das Zertifikat zurückliefern.

Zu diesem Zweck wurde im Rahmen des SigI Profils die optionale Erweiterung *requestedCertificate* für das *singleExtensions*-Teilfeld der *SingleResponse*-Struktur vorgesehen. Diese Erweiterung muß als *non-critical* markiert werden.

ASN.1 Definitionen

```
id-sigi OBJECT IDENTIFIER  ::=  { 1 3 36 8 }
```

```
id-sigi-at OBJECT IDENTIFIER ::= { 1 3 36 8 3 }
id-sigi-at-CertInDirSince OBJECT IDENTIFIER    ::=
                                { 1 3 36 8 3 12 }

CertInDirSince EXTENSION    ::=  {
   SYNTAX                        CertInDirSinceSyntax
   IDENTIFIED BY                 id-sigi-at-CertInDirSince }
CertInDirSinceSyntax        ::=  GeneralizedTime

id-sigi-at-requestedCertificate OBJECT IDENTIFIER   ::=
                                { 1 3 36 8 3 10 }
requestedCertificate EXTENSION   ::=  {
   SYNTAX                        RequestedCertificateSyntax
   IDENTIFIED BY                 id-sigi-at-requestedCertificate }
RequestedCertificateSyntax  ::=  Certificate

id-sigi-at-certHash OBJECT IDENTIFIER ::=  { 1 3 36 8 3 13 }
certHash EXTENSION          ::=  {
   SYNTAX                        CertHashSyntax
   IDENTIFIED BY                 id-sigi-at-certHash }
CertHashSyntax              ::=  SEQUENCE {
   hashAlgorithm                 AlgorithmIdentifier,
   certificateHash               OCTET STRING }
```

Wenn die SigI-Erweiterung *retrieveIfAllowed* in der Anfrage gesetzt war, das Zertifikat aber nicht abrufbar ist, muß der Verzeichnisdienst nur den Verweis auf das Zertifikat im *certID*-Teilfeld liefern. Die Anwenderinfrastruktur kann so feststellen, daß ein Zertifikat nicht abrufbar ist.

Die SigI-Erweiterungen *certHash* muß sowohl in der Anfrage an den Verzeichnisdienst als auch in in der zugehörigen Antwort gesetzt sein, wobei *certHash* als *non-critical* zu markieren ist. Die SigI-Erweiterung *certHash* etabliert eine kryptographische Bindung zwischen der Bytefolge des Zertifikates und der Antwort des Verzeichnisdienstes.

2.3 Transport von Verzeichnisdienstanfragen über HTTP

Die Kommunikation erfolgt über die beschriebenen Datenformate, die als Nachrichten über das Hypertext Transfer Protokoll (HTTP) an den Verzeichnisdienst übermittelt werden. Die Verwendung von HTTP [RFC 2068 97] erlaubt eine einfache Erstellung von Software für den Zugriff auf Verzeichnisdienste unter Verwendung erprobter Programmbibliotheken. Weiterhin

sind für HTTP Übergangsmöglichten von firmeninternen Netzen in öffentliche Netze, sogenannten Firewalls, definiert und in der Form von HTTP Proxies bereits erprobt.

Sofern in einer speziellen Anwendung die Geheimhaltung der Anfragen gewünscht wird, kann die Übertragung verschlüsselt über Transport Layer Security (TLS) bzw. Secure Socket Layer (SSL) erfolgen. Da diese Sicherungsmechanismen nur einen gesicherten Kanal für HTTP bereitstellen, sind diese transparent zu den hier beschriebenen Definitionen.

2.3.1 DEFINITIONEN FÜR DIE ANFRAGEN

HTTP basierte OCSP Anfragen können entweder die HTTP Zugriffsmethode GET oder POST verwenden, wobei Anfragen, die eine Länge von mehr als 254 Bytes haben, ausschließlich mit der Methode POST an den Verzeichnisdienst übermittelt werden müssen. Der Verzeichnisdienst muß Anfragen beider Methoden verstehen und verarbeiten können.

Anfragen an den Verzeichnisdientst mit der Zugriffsmethode GET verwenden folgendes Format, wobei der base-64 DER kodierten Anfrage die ASN1. Struktur *OCSPRequest* aus Kapitel 2.2.1 zugrunde liegt:

```
GET {url}/{url-kodiertes base-64 DER kodierte Anfrage}
```

Die zu verwendende URL wird aus der lokalen Konfiguration des Clients oder mit einer der in [A1 98, Anhang II] beschriebenen Methoden zur Ermittlung von Dienstadressen abgeleitet.

Eine Anfrage mit der Methode POST verwendet den HTTP Headerfeld “Content-Type” mit dem Wert “application/ocsp-request”. Die Länge der Anfrage muß im HTTP Headerfeld “Content-Length” mit der Länge des Inhaltes der HTTP Anfrage belegt werden. Als Inhalt der HTTP Anfrage wird die DER kodierte Anfrage an den Verzeichnisdienst verwendet.

Anfragen mit der Methode POST können entweder binär oder base 64 kodiert erfolgen. Die Anfrage sollte binär übermittelt werden, sofern das Transportmedium dies zuläßt. Verzeichnisdienste müssen beide Kodierungen verarbeiten können.

Anfragen an den Verzeichnisdientst mit der Zugriffsmethode POST verwenden folgendes Format, wobei der base-64 DER kodierten Anfrage die ASN1. Struktur *OCSPRequest* aus Kapitel 2.2.1 zugrunde liegt:

```
POST {url} ...
Content-Type: application/ocsp-request
Content-Length: ...
{binaer oder base-64 DER kodierte Anfrage}
```

2.3.2 DEFINITIONEN FÜR DIE ANTWORTEN

Eine HTTP Antwort besteht aus den üblichen HTTP Headern, der Inhalt des Dokumentes ist die DER kodierte Antwort, wobei der DER kodierten Antwort die ASN1. Struktur *OCSPResponse* aus Kapitel 2.2.2 zugrunde liegt. Der Transport kann entweder binär oder base 64 kodiert erfolgen. Die Antwort sollte binär übermittelt werden, sofern das Transportmedium dies zuläßt.

Im HTTP Header "Content-Type" muß der Wert "application/ocsp-response" angegeben werden, im Header "Content-Length" sollte die die Länge des folgenden Dokumentes verzeichnet sein. Andere HTTP Header können vorhanden sein und können von der Anwenderinfrastrutkur ignoriert werden.

2.3.2.1 Abspeichern von Antworten

Antworten vom Verzeichnisdienst können von der Anwenderinfrastruktur für spätere Nachweise abgespeichert werden. In diesem Fall sollte die Datei in ihrem Format der Antwort des HTTP Protokolls entsprechen. Dabei kann die erste Zeile des HTTP Proktolls ("HTTP.. ") weggelassen werden. Minimal vorhanden sein muß die Headerzeile "Content-Type". Die Dateinamenserweiterung für Antworten des Verzeichnsidienstes ist ".ors".

2.3.3 VERWENDUNG VON PROXIES

2.3.3.1 Verwendung von HTTP Proxy Servern

Um Anwenderinfrastrukturen in Unternehmensnetzwerken eine Kommunikation mit Verzeichnisdiensten über Firewallsysteme zu erlauben, können die Anfragen über einen sogenannten HTTP Proxy Server übermittelt werden. Der Proxy Server dient dabei als Mittler zwischen der Anwenderinfrastruktur und dem Verzeichnisdienst.

Üblicherweise leitet ein Proxy Server eine HTTP Anfrage an einen Rechner weiter, der vom Anwender in der HTTP Anfrage spezifiziert wurde. Diese Methode erlaubt einer Anwenderinfrastruktur den transparenten Zugriff auf Verzeichnisdienste.

2.3.3.2 Verwendung von SLL Proxy Servern

Sofern eine Verschlüsselung der Anfragen und Antworten des Verzeichnisdienstes gewünscht wird, kann die HTTP Anfrage über SSL gesichert werden. Sofern auch hier ein Firewallsystem zur Trennung von Netzen eingesetzt wird, kann über die SSL Proxy Spezifikation eine SSL Verbindung zum Verzeichnisdienst aufgebaut werden.

2.3.3.3 Verwendung eines Verzeichnisdienst Proxies

Alternativ zum normalen HTTP Proxy Server könnte ein HTTP Proxy Server so konfiguriert werden, daß die Anwenderinfrastruktur alle Anfragen an Verzeichnisdienste an einen – möglicherweise lokal konfigurierten – Verzeichnisdienst sendet. Dort würde der HTTP Proxy die in der HTTP Anfrage kodierte Verzeichnisdienstanfrage dekodieren und so den korrekten Verzeichnisdienst ermitteln. An diesen wird die Anfrage dann unverändert weitergeleitet. Die Integrität, Authentizität und Aktualität der Antwort ist durch die Zeitangaben und die digitale Signatur sichergestellt.

Eine solche Konfiguration hat den Vorteil, daß die einzelne Anwenderinfrastruktur keinen Zugriff auf ein öffentliches Netz haben muß und auch keine Informationen über die verschiedenen Dienstadressen halten muß.

Zusätzlich könnten Anfragen dann vom Verzeichnisdienst Proxy per SSL verschlüsselt an den jeweiligen Verzeichnisdienst weitergeleitet werden, so daß – etwa bei Großanwendern – die Daten vertraulich behandelt werden und dieses Verhalten auch unternehmensweit eingeführt werden kann.

2.3.3.4 Dienstäquivalenz von Verzeichnisdiensten

Prinzipiell wäre es möglich, daß jeder der Verzeichnisdienste einen äquivalenten Dienst anbietet, so daß die Anwenderinfrastruktur eine beliebigen Verzeichnisdienst für alle Auskünfte kontaktieren kann. Dies erfodert natürlich, daß die äquivalenten Dienste synchron arbeiten müsssen.

Neben einer solchen Implementation wäre es auch möglich, daß Verzeichnisdienst bemerken, wenn Anfragen zu einem falschen Dienst gesendet werden und in der Antwort mittels eines HTTP Redirects die korrekte Dienstadresse angeben.

2.4 Transport von Verzeichnisdienstanfragen über E-Mail

Im Rahmen der SigI-Spezifikation wird die Benutzung von E-Mail für die Kommunikation mit dem Verzeichnisdienst nicht empfohlen, weil die zugehörigen Prozesse asynchron ablaufen und insbesondere den Verifikationsprozeß des Anfragenden unterbrechen. Die Antworten des Verzeichnisdienstes gelangen in die Mailbox des Anfragenden und müssen über manuelle Interaktionen dem Verifikationsprozeß zugänglich gemacht werden.

3 SPERRLISTENMANAGEMENT

3.1 Sperrlistenformate

Zweck

Sperrlistenformate und Erweiterungen für Sperrlisten spielen eine wichtige Rolle bei Realisierung von öffentlichen Sicherheitsinfrastrukturen. Gegenwärtig dient das von der PKIX-Arbeitsgruppe, auf dem internationalen Standard X.509 v2 basierende, Sperrlistenprofil [ITU-T X.509 97, PKIK PRO 98] als Grundlage für die Entwicklung zahlreicher Anwendungen und Systemumgebungen.

Der ASN.1-Sperrlistentyp *CertificateList* besteht syntaktisch aus einer Folge von jeweils drei Feldern, die zur Trennung der zu signierenden Daten *tbsCertList*, des zu benutzenden Signaturalgorithmus *signatureAlgorithm* und der eigentlichen Signatur *signature* dienen.

ASN.1-Definitionen

```
CertificateList   ::= SEQUENCE {
    tbsCertList           TBSCertList,
    signatureAlgorithm    AlgorithmIdentifier,
    signature             BIT STRING }
```

Allgemeine Konformitätsanforderungen

In PKIX wird von konformen Zertifizierungsstellen nicht generell die Erstellung von Sperrlisten (CRL, certificate revocation list) gefordert, falls andere Mechanismen für die Zurückziehung oder die Zustandsanzeige von Zertifikaten bereitgestellt werden. Konforme Zertifizierungsstellen, die CRLs erstellen, sollen hierbei die Version 2 (siehe Abschnitt 3.1.3.1) benutzen und müssen das Datum und den Zeitpunkt (siehe Abschnitt 3.1.3.5) der nächsten CRL-Erstellung anzeigen.

Von konformen Systemen und Anwendungen wird erwartet, daß sie CRLs der Version 1 und 2 verarbeiten können.

SigI- Konformitätsanforderungen

Signatur-gesetzeskonforme (SigI-) Zertifizierungsstellen müssen CRLs mit der Version 2 erstellen und müssen hierbei das nächste CRL-Erstellungsdatum im *nextUpdate*-Feld der CRL eintragen.

3.1.1 SIGNATURALGORITHMUS

Zweck

Das Signaturfeld *signatureAlgorithm* vom Typ *AlgorithmIdentifier* enthält den Bezeichner des kryptographischen Algorithmus, der von der Zertifizierungsstelle zum Signieren der Sperrliste benutzt wird. Hierbei ist zu beachten, daß Signaturalgorithmen immer in Kombination mit Einweg-Hash-Funktionen und digitalen Signaturformaten (message formatting,

padding) benutzt werden. Das Signaturfeld besteht syntaktisch aus einer Folge von Teilfeldern *algorithm* und *parameters*. Das Teilfeld *algorithm* ist ein Objektbezeichner, der zur Identifikation des Algorithmus dient. Der Inhalt des optionalen *parameters*-Teilfeldes ist abhängig vom angegebenen Algorithmus und dem Algorithmusbezeichner.

ASN.1-Definitionen

```
Certificate       ::= SEQUENCE {
    ...,
    signatureAlgorithm    AlgorithmIdentifier,
    ... }
Algorithmidentifier   ::=   SEQUENCE {
    algorithm             OBJECT IDENTIFIER,
    parameters            ANY DEFINED BY algorithm OPTIONAL }
```

Allgemeine Konformitätsanforderungen

Das Signaturfeld *signatureAlgorithm* muß denselben Algorithmusbezeichner wie das *signature*-Teilfeld der *tbsCertList*-Struktur enthalten.

SigI-Konformitätsanforderungen

Das optionale *parameters*-Teilfeld darf nicht zur Übergabe von Parametern an den Algorithmus benutzt werden, da dieses Feld nicht durch die Signatur der Zertifizierungsstelle geschützt ist. Diese Einschränkung gilt nur für den "äußeren" Algorithmusbezeichner der *CertificateList*-Struktur und nicht für den "inneren" Algorithmusbezeichner der *tbsCertList*-Struktur, da letzterer durch die Signatur der Zertifizierungsstelle beglaubigt ist. Trotzdem darf auch der innere Algorithmusbezeichner nicht mit Parametern versehen werden und dessen Komponente parameters ist mit dem Wert NULL zu belegen. Zum Signieren geeignete und zugelassene Algorithmen werden von der Regulierungsbehörde für Telekommunikation und Post im Bundesanzeiger veröffentlicht. Die in der Ausgabe von Februar 1998 aufgeführten und geeigneten Kryptoalgorithmen gelten für die kommenden sechs Jahre (14. Februar 1998 – 14. Februar 2004).

Tabelle 4: Implementations-technische Informationen über signatureAlgorithm

BEZEICHNER	WERTEBEREICH EINZELWERTE	HEXADEZIMALCODE	LÄNGE [BYTES]	RELEVANZ		
	(BEISPIELE)	(BEISPIELE)	20	obligatorisch	verboten	optional
signatureAlgorithm Algorithm algorithm parameters	rsaSignatureWithripemd160 SEQUENCE { {1 3 36 3 3 1 2 }, NULL }	**30** 0A **06** 06 *2B 24 03 03 01 02* **05** 00	11	v		

Die Algorithmen und Parameter, mit denen eine Zertifizierungsstelle eine Sperrliste signiert, müssen mindestens für die Gültigkeitsdauer der Sperrliste als geeignet beurteilt sein. Weitere Einzelheiten zu dem Thema “Signaturalgorithmen” sind in der Teilspezifikation “A2 Signatur” angegeben. Die maximale Länge des *signatureAlgorithm*-Feldes beträgt 20 Bytes.

3.1.2 SIGNATUR EINER SPERRLISTE

Zweck

Das Signaturfeld *signature* enthält eine digitale Signatur, die für das in ASN.1-DER kodierte Sperrlistenfeld *tbsCerList* berechnet wird. Bei der Berechnung der Signatur wird das Sperrlistenfeld *tbsCerList* als Eingabe in eine Einweg-Hash-Funktion benutzt. Auf den Ergebniswert der Hashfunktion wird der private Schlüssel der Zertifizierungsstelle angewandt und als ASN.1-Bitstring kodiert. Er liefert die konkrete digitale Signatur der Sperrliste im Signaturfeld *signature*. Durch den Signaturvorgang beglaubigt eine Zertifizierungsstelle die Gültigkeit der im Sperrlistenfeld *tbsCerList* enthaltenen Informationen und gewährleistet insbesondere die Ungültigkeit der betreffenden Zertifikate.

ASN.1-Definition

```
Certificate        ::= SEQUENCE {
   ...,
   signature               BIT STRING }
```

Allgemeine Konformitätsanforderungen

Geeignete Signaturformate finden sich in den Spezifikationen [PKCS1 93, Abschnitt 8.1] und [DIN SigG/V 98, Anhang A]. Üblicherweise wird das Ergebnis der Einweg-Hash-Funktion an die Signaturkomponente übergeben. Die Komponente ergänzt gegebenenfalls den ihr übergebenen Hashwert um zusätzliche Komponenten, bevor die eigentliche mathematische Signaturfunktion angewendet wird (siehe Teilspezifikation “A2 Signatur”).

SigI-Konformitätsanforderungen

Bei der Erzeugung von Sperrlisten ist die Benutzung des Sperrlisten feldes *signature* obligatorisch. Es dürfen nur die in der Teilspezifikation “A2 Signatur” aufgeführten Signaturalgorithmen und Signaturformate benutzt werden.

Tabelle 5: Implementations-technische Informationen über *signature*

BEZEICHNER	WERTEBEREICH EINZEILWERTE	HEXADEZIMALCODE	LÄNGE [BYTES]	RELE-VANZ		
	(BEISPIELE)	(BEISPIELE)	261	obligatorisch	verboten	optional
Signature	BITSTRING --256 Byte-Schlüssellänge	**03** 82 01 01 ...	261	v		

3.1.3 ZU SIGNIERENDE SPERRLISTENINFORMATIONEN

Zweck

Das Sperrlistenfeld *tbsCertList* besteht aus einer Folge von optionalen und vorgeschriebenen Teilfeldern, die Informationen enthalten, die in unmittelbarem Zusammenhang mit dem Sperren von Zertifikaten stehen. Die in dieser Struktur vorgeschriebenen Bestandteile *signature*, *issuer* und *thisUpdate* dienen zur Kennung des benutzten Signaturalgorithmus, zur Identifikation des Erstellers der Sperrliste, sowie zur Angabe des Erstellungsdatums der Sperrliste. Die optionalen Teilfelder *version*, *nextUpdate*, *revokedCertificates* und *crlExtensions* der Sperrlistenstruktur enthalten die Version der aktuellen CRL, das Erstellungsdatums der nächsten Sperrliste, Listen von gesperrten Zertifikaten und mögliche CRL-Erweiterungen. Zertifikate werden innerhalb der Folge *revokedCertificates* durch ihre Seriennummer *userCertificate*, das Datum der Sperrung *revocationDate* und durch mögliche zertifikatsspezifische CRL-Eintragserweiterungen *crlEntryExtensions* als gesperrte Zertifikate gekennzeichnet.

ASN.1-Definitionen

```
CertificateList   ::= SEQUENCE {
   tbsCertList          TBSCertList,
   ... }
TBSCertList       ::= SEQUENCE {
   version              Version OPTIONAL,
   signature            AlgorithmIdentifier,
   issuer               Name,
   thisUpdate           Time,
   nextUpdate           Time OPTIONAL,
   revokedCertificates  SEQUENCE OF SEQUENCE {
     userCertificate    CertificateSerialNumber,
     revocationDate     Time,
     crlEntryExtensions Extensions OPTIONAL } OPTIONAL,
   crlExtensions        [0]  EXPLICIT Extensions OPTIONAL }
```

Tabelle 6: Implementations-technische Informationen über *tbsCertList*

FELD	ZERTIFIKATSTYP				RELEVANZ			FELD	ZERTIFIKATSTYP				RELEVANZ		
	Zertifizierungsstellen	Zeitstempeldienst	Verzeichnisdienst	Teilnehmer	obligatorisch	verboten	optional		Zertifizierungsstellen	Zeitstempeldienst	Verzeichnisdienst	Teilnehmer	obligatorisch	verboten	optional
Version	v	v	v	v	v			nextUpdate	v	v	v	v	v		
Signature	v	v	v	v	v			revokedCertificates	v	v	v	v	v		
Issuer	v	v	v	v	v			crlExtensions	v	v	v	v	v		
ThisUpdate	v	v	v	v	v										

SigI-Konformitätsanforderungen

Bei der Erzeugung von Sperrlisten ist die Verwendung des Feldes *crlExtensions* obligatorisch, da die beiden Sperrlistenerweiterungen *authorityKeyIdentifier* und *cRLNumber* verwendet werden müssen. Außerdem ist auch die Benutzung der Teilfelder *version* und *nextUpdate* obligatorisch.

3.1.3.1 Versionsnummer

Zweck

Das optionale Versionsfeld *version* gibt die Version des Sperrlistenformates an.

ASN.1-Definitionen

```
TBSCertList         ::= SEQUENCE {
   version                  Version OPTIONAL,
   ... }

Version             ::= INTEGER { v1(0), v2(1) }
```

Allgemeine Konformitätsanforderungen

Sperrlisten, die optionale CRL-Erweiterungen *crlExtensions* und/oder optionale CRL-Eintragserweiterungen *crlEntryExtensions* enthalten, sollen das Versionsfeld mit der Version v2 verwenden. Sperrlisten, die keine optionale Erweiterungen enthalten, sollen die Version v1 verwenden und bei der Kodierung das Versionsfeld weglassen.

SigI-Konformitätsanforderungen

Bei der Erzeugung von Sperrlisten muß die Version v2 benutzt werden.

Tabelle 7: Implementations-technische Informationen über *version*

BEZEICHNER	WERTEBEREICH EINZEILWERTE	HEXADEZIMALCODE	LÄNGE [BYTES]	RELEVANZ		
	(BEISPIELE)	(BEISPIELE)	3	obligatorisch	verboten	optional
version	1	**02** 01 *01*	3	v		

3.1.3.2 Signatur

Zweck

Das Signaturfeld enthält den Bezeichner des Algorithmus, der von der Zertifizierungsstelle zum Signieren der Sperrliste benutzt wird.

ASN.1-Definitionen

```
TBSCertList        ::= SEQUENCE {
    ...,
    signature              AlgorithmIdentifier,
    ... }

Algorithmidentifier    ::=   SEQUENCE {
    algorithm              OBJECT IDENTIFIER,
    parameters             ANY DEFINED BY algorithm OPTIONAL }
```

Allgemeine Konformitätsanforderungen

Das Signaturfeld *signature* der *tbsCertList*-Struktur muß denselben Algorithmusbezeichner wie im *signatureAlgorithm*-Feld der *CertificateListe*-Struktur enthalten.

SigI-Konformitätsanforderungen

Bei der Erstellung von Sperrlisten ist die Benutzung des Sperrlistenfeldes *signature* obligatorisch. Zum Signieren geeignete Algorithmen werden von der Regulierungsbehörde für Telekommunikation und Post im Bundesanzeiger veröffentlicht. Die für die kommenden sechs Jahre (14. Februar 1998 – 14. Februar 2004) als geeignet beurteilten Kryptoalgorithmen sind in der Teilspezifikation "A2 Signatur" aufgelistet.

Tabelle 8: Implementations-technische Informationen über *signature*

BEZEICHNER	WERTEBEREICH EINZELWERTE	HEXADEZIMALCODE	LÄNGE [BYTES]	RELE-VANZ		
	(BEISPIELE)	(BEISPIELE)	1120	obligatorisch	verboten	optional
signatureAlgorithm	rsaSignatureWithripemd160		11	v		
Algorithm	SEQUENCE {	**30** 0A				
Algorithm	{1 3 36 3 3 1 2 },	**06** 06 *2B 24 03 03 01 02*				
Parameters	NULL }	**05** 00				

3.1.3.3 Namen von Sperrlistenerstellern

Zweck

Das *issuer*-Namensfeld dient zur technischen Identifikation der Instanz bzw. Zertifizierungsstelle, die die betreffende Sperrliste erstellt und signiert hat.

Es sind bei der technischen Namensgebung von Zertifizierungsstellen nur Namen gemäß der X.500-Syntax [ITU-T X.500 97] für *distinguished name*-Typen zugelassen. *Der distinguished name* ist vom Typ RDNSequence und somit aus einer Folge von *AttributeType*- und *AttributeValue*-Paaren zusammengesetzt. *AttributeType* wird i.a. durch X.500 festgelegt, und für *AttributeValue* wird der Typ *DirectoryString* (für den unspezifischen Typ *ANY*) verwendet, der seinerseits ein Auswahltyp von *PrintableString*, *TeletexString*, *UniversalString* und *BMPString* ist. Eine Übersicht der möglichen Objektbezeichner für *AttributeType* ist in der folgenden Tabelle gegeben.

ASN.1-Definitionen

```
TBSCertList         ::= SEQUENCE {
    ...,
    issuer                  Name,
    ... }

Name                ::= CHOICE { RDNSequence }

RDNSequence         ::= SEQUENCE OF RelativeDistinguishedName

RelativeDistinguishedName   ::=  SET OF AttributeTypeAndValue

AttributeTypeAndValue ::=   SEQUENCE {
    type                    AttributeType,
    value                   AttributeValue }

AttributeType       ::= OBJECT IDENTIFIER

AttributeValue      ::= ANY DEFINED BY AttributeType

DirectoryString     ::= CHOICE {
    printableString         PrintableString (SIZE (1..maxSize))
    teletexString           TeletexString (SIZE (1..maxSize))
    bmpString               BMPString (SIZE (1..maxSize))
    universalString         UniversalString (SIZE (1..maxSize)) }
```

Allgemeine Konformitätsanforderungen

Bei der Konstruktion von *DirectoryString* ist stets die restriktivste Auswahl zu treffen und deshalb der minimalste Zeichensatz zur Repräsentation von *AttributeValue* wählen. Die Reihenfolge, in der die Zeichensätze auf ihre konkrete Anwendbarkeit hin geprüft werden sollen, lautet somit: *PrintableString, TeletexString, BMPString* und *UniversalString*.

Der Name einer Zertifizierungsstelle kann nach [ITU-T X.509 97] auch alternativ oder zusätzlich zum *issuer*-Feld im optionalen *extensions*-Feld unter *issuerAltName* [A1 98, Abschnitt 2.3.9.6] angegeben werden. Im ersten Fall kann das *issuer*-Feld als leere Folge kodiert werden und die *issuerAltName*-Erweiterung muß als *critical*,

d.h. als “wichtige und zu berücksichtigende” Erweiterung gekennzeichnet werden. Weitere Informationen über Namenskonventionen sind in [A1 98, Anhang I] zu finden.

SigI-Konformitätsanforderungen

Bei der Erstellung von Sperrlisten ist die Benutzung des *issuer*-Feldes obligatorisch. Diese Anforderung ergibt sich aus der Notwendigkeit einer eindeutigen technischen Benennung der Zertifizierungsstelle. Das *issuer*-Feld soll mit dem technischen Namen der Zertifizierungsstelle belegt werden, damit die Konformität zu vielen Anwendungen im internationalen Kontext gewährleistet bleibt. Namen von Zertifizierungsstellen enthalten zumindest die obligatorischen Attribute *organization* und *countryName*. Alle anderen Attribute sind optional.

Beispiele für technische Namen von Zertifizierungsstellen

(1) Technischer Name der RegTP
`OU=Wurzelzertifizierungsstelle, O=RegTP, C=DE`

(2) Technischer Name der RegTP mit Acronym
`CN=DEPCA, OU=Wurzelzertifizierungsstelle, O=RegTP, C=DE`

(3) Technischer Name einer untergeordneten Zertifizierungsstelle
`CN=Name-der-ZS, O=Organisation-der-ZS, C=DE`

Die Länge der *AttributeValue*-Stringtypen ist durch den Systemparameter *maxSize* festgelegt, dessen Wert für die einzelnen Attribute gemäß der folgenden Tabelle begrenzt ist. Hieraus ergeben sich die in der Längenspalte angegebenen maximalen Längen der Attribute inklusive der ASN.1-Kontrollinformationen, die eine Länge von 11 Bytes haben.

Tabelle 9: Implementations-technische Informationen über Längen von Attributtypen

OBJEKTBEZEICHNER		MAXSIZE	LÄNGE	OBJEKTBEZEICHNER		MAXSIZE	LÄNGE
NAME	NUMMER	[BYTES]	[BYTES]	NAME	NUMMER	[BYTES]	[BYTES]
commonName	{ 2 5 4 3 }	64	75	organizationName	{ 2 5 4 10 }	64	75
surName	{ 2 5 4 4 }	32	43	organizationalUnit	{ 2 5 4 11 }	64	75
serialNumber	{ 2 5 4 5 }	3	14	title	{ 2 5 4 12 }	10	21
countryName	{ 2 5 4 6 }	2	13	businessCategory	{ 2 5 4 15 }	32	43
localityName	{ 2 5 4 7 }	32	43	postalCode	{ 2 5 4 17 }	10	21
stateOrProvince	{ 2 5 4 8 }	32	43	givenName	{ 2 5 4 47 }	32	43

Für das *issuer*-Feld bestehend aus nur obligatorischen Attributen ergibt sich aus dem Längenfeld der Tabelle 9 die Maximallänge von 75+13+2 (ASN.1-Kontrollinformation) = 90 Bytes und für das *issuer*-Feld bestehend aus allen Attributen der Tabelle 9 von 75+43+14+ 13+43+43+75+75+21+43+21+43+4 (ASN.1-Kontrollinformation) = 513 Bytes.

Tabelle 10: Implementations-technische Informationen über *issuer*

BEZEICHNER	WERTEBEREICH EINZEILWERTE	HEXADEZIMALCODE	LÄNGE [BYTES]	RELE-VANZ		
	(BEISPIELE)	(BEISPIELE)	513	obligatorisch	verboten	optional
issuer	SEQUENCE OF {	**30** 53	85	v		
countryName value	SET OF SEQUENCE { { { 2 5 4 6 }, "DE" } }	**31** 0B **30** 09 **06** 03 *55 04 06* **13** 02 *44 45*	13	v		
organization Name value	SET OF SEQUENCE { { { 2 5 4 10 }, "regtp" } }	**31** 0E **30** 0C **06** 03 *55 04 0A* **13** 05 *72 65 67 74 70*	16	v		

organizational Unit
value
	SET OF SEQUENCE { {
	{ 2 5 4 11 },
	"Wurzelzertifizierungsstelle" }
	}
		31 24 **30** 22
		06 03 *55 04 0B*
		13 1B *57 75 72 7A 65 6C 7A 65 72 74 69 66 69 7A 69 65 72 75 6E 67 73 73 74 65 6C 6C 65*
			38			v

commonName value	SET OF SEQUENCE { { { 2 5 4 3 }, "DEPCA" } } }	**31** 0E **30** 0C **06** 03 *55 04 03* **13** 05 *44 45 50 43 41*	16			v

3.1.3.4 Datum und Zeitpunkt der Erstellung von Sperrlisten

Zweck

Das *thisUpdate*-Datums- und Zeitfeld gibt das Datum und den Zeitpunkt der Erstellung einer Sperrliste an und kann dabei entweder im *UTCTime*- oder im *GeneralizedTime*-Datums- und Zeitformat kodiert werden. Dieser Zeitpunkt kann durch die Standard-ASN.1-Zeittypen *UTCTime* (coordinated universal time, Weltzeit) oder *GeneralizedTime* (allgemeines Datums- und Zeitformat) repräsentiert werden, die Datums- und Zeitangaben bis auf Sekundengenauigkeit sowie die Angabe von Zeitverschiebungen der lokalen gegenüber der Weltzeit gestatten. Die Hauptunterschiede zwischen beiden Formaten bestehen darin, daß mit dem verallgemeinerten Zeittyp kleinere Zeiteinheiten und vollständige Jahreszahlen angegeben werden können.

ASN.1-Definitionen

```
TBSCertList         ::= SEQUENCE {
    ...,
    thisUpdate              Time,
    ... }

Time                ::= CHOICE {
    utcTime                 UTCTime,
    generalizedTime         GeneralizedTime }
```

Allgemeine Konformitätsanforderungen

Zur Kodierung des Zeitpunktes der Erstellung einer Sperrliste ist bis zum Jahr 2049 als Zeittyp stets der Typ *UTCTime* und ab dem Jahr 2050 der Typ *GeneralizedTime* zu benutzen. Zertifizierungsstellen sollen bei der Verwendung eines dieser Typen die Werte von Zeitpunken in Greenwich-Zeit (GMT, Greenwich Mean Time) bis auf Sekundengenauigkeit ausdrücken, wobei auch die Null-Sekunde zu kodieren ist. Bei der Kodierung der Datums- und Zeitangaben sind für *GeneralizedTime* das Format YYYYMMDDHHMMSSZ und für *UTCTime* das Format YYMMDDHHMMSSZ zu beachten. Die Bedeutung der einzelnen Felder der Datums- und Zeitformate ist in der folgenden Tabelle zusammengefaßt.

Tabelle 11: Bedeutung der Felder in Datums- und Zeitformaten

DATUMSANGABEN		ZEITANGABEN	
FELD	BEDEUTUNG	FELD	BEDEUTUNG
YYYY	vollständige Jahreszahl, nur bei *GeneralizedTime*	HH	Stunde 00, 01, ..., 23
YY	letzte zwei Ziffern der Jahreszahl, nur bei *UTCTime*	MM	Minute 00, 01, ..., 59
MM	Monat 01, 02, ..., 12	SS	Sekunde 00, 01, ..., 59
DD	Tag 01, 02, ..., 31	Z	GMT

Bei der Benutzung des *UTCTime* Typs istdas 2-stellige Jahresfeld YY gemäß der folgenden Konvention (links) zu interpretieren.

Für das 2-stellige Jahresfeld YY gilt nach [MTRUST 96] die folgende Konvention (rechts), die jedoch nicht kompatibel zu [ITU-T X.509 97], [PKIX PRO 97]und [MISPC 97] ist.

$$Jahr(YY) = \begin{cases} 19YY & |YY \in [50,99] \\ 20YY & |YY \in [0,49] \end{cases}$$

$$Jahr(YY) = \begin{cases} 19YY & |YY \in [65,99] \\ 20YY & |YY \in [0,64] \end{cases}$$

Die Inkompatibilität betrifft die Zeiträume zwichen 1950 und 1964, sowie zwischen 2050 und 2064. Der erste Zeitraum (1950 bis 1964) bereitet keine Probleme, da es hierfür noch keine Sperrlisten gibt. Sperrlisten, deren Gültigkeitsdauern in den zweiten Jahreszeitraum (2050 bis 2064) fallen, sollten zur Kodierung ebenfalls im *GeneralizedTime*-Format erstellt werden.

SigI-Konformitätsanforderungen

Bei der Erzeugung von Sperrlisten muß das allgemeinen Datums- und Zeitformat *GeneralizedTime* verwendet werden, bei dessen Kodierung außerdem das Format YYYYMMDDHH MMSSZ genommen werden muß.

Tabelle 12: Implementations-technische Informationen über *thisUpdate*

BEZEICHNER	WERTEBEREICH EINZEILWERTE	HEXADEZIMALCODE	LÄNGE [BYTES]	RELE-VANZ		
	(BEISPIELE)	(BEISPIELE)	17	obligatorisch	verboten	optional
thisUpdate	GeneralizedTime "19980101000000Z"	**18** 0F *31 39 39 38 30 31 30 31 30 30 30 30 30 30 5A*	17	v		

3.1.3.5 Datum und Zeitpunkt der Erstellung der nächsten Sperrliste

Zweck

Das optionale *nextUpdate*-Datums- und Zeitfeld gibt das Datum und den Zeitpunkt der Erstellung der nächsten Sperrliste an und kann dabei entweder im *UTCTime*- oder im *GeneralizedTime*-Datums- und Zeitformat kodiert werden.

ASN.1-Definitionen

```
TBSCertList        ::= SEQUENCE {
    ...,
    nextUpdate             Time OPTIONAL,
    ... }
Time               ::= CHOICE {
    utcTime                UTCTime,
    generalizedTime        GeneralizedTime }
```

Allgemeine Konformitätsanforderungen

Aus praktischen Gründen darf eine neue Sperrliste schon vor dem spezifizierten Zeitpunkt erzeugt werden, sie muß aber spätestens bis zu dem angegebenen Zeitpunkt verfügbar sein. Zur Kodierung des Zeitpunktes der Erstellung einer nächsten Sperrliste ist bis zum Jahr 2049 als Zeittyp stets der Typ *UTCTime* und ab dem Jahr 2050 der Typ *GeneralizedTime* zu benutzen. Zertifizierungsstellen sollen bei der Verwendung eines dieser Typen die Werte von Zeitpunken in Greenwich-Zeit (GMT, Greenwich Mean Time) bis auf Sekundengenauigkeit ausdrücken, wobei auch die Null-Sekunde zu kodieren ist. Bei der Kodierung der Datums- und Zeitangaben sind

für *GeneralizedTime* das Format YYYYMMDDHHMMSSZ und für *UTCTime* das Format YYMMDDHHMMSSZ zu beachten. Die Bedeutung der einzelnen Felder der Datums- und Zeitformate ist in der Tabelle 11 zusammengefaßt. Konforme Zertifizierungsstellen sollen bei der Erstellung von Sperrlisten mit der Version v2 das *nextUpdate*-Datums- und Zeitfeld benutzen.

SigI-Konformitätsanforderungen

Bei der Erzeugung von Sperrlisten muß stets der Zeitpunkt der Erstellung einer neuen Sperrliste angegeben werden und hierfür das allgemeine Datums- und Zeitformat *GeneralizedTime* verwendet werden, bei dessen Kodierung außerdem das Format YYYYMMDDHHMMSSZ genommen werden muß.

Es ist verboten, zeitlich überlappende CRLs auszustellen. Somit entsprich *nextUpdate* auch dem Ende der Gültigkeit einer Sperrliste. Der Zeitraum zwischen den durch *thisUpdate* und *nextUpdate* definierten Zeitpunkten muß hinreichend kurz sein, beispielsweise eine Stunde oder ein Tag.

Tabelle 13: Implementations-technische Informationen über *nextUpdate*

BEZEICHNER	WERTEBEREICH EINZEILWERTE	HEXADEZIMALCODE	LÄNGE [BYTES]	RELEVANZ		
	(BEISPIELE)	(BEISPIELE)	17	obligatorisch	verboten	optional
nextUpdate	GeneralizedTime "19980102010000Z"	**18** 0F *31 39 39 38 30 31 30 32 30 31 30 30 30 30 5A*	17	v		

3.1.3.6 Sperrlisteneinträge

Zweck

Das optionale *revokedCertificates*-Feld repräsentiert die Liste von widerrufenen bzw. gesperrten Zertifikaten einer Zertifizierungsstelle. Jedes einzelne gesperrte Zertifikat wird in dieser Liste über seine Seriennummer *userCertificate* vom Typ *CertificateSerialNumber* aufgeführt und ist durch die Kombination aus der Seriennummer und dem Namen der Zertifizierungsstelle *issuer* global eindeutig identifiziert. Desweiteren enthält das *revokedCertificates*-Feld den Zeitpunkt der Sperrung *revocationDate* vom Typ *Time*, der – wie in 3.2.3.4 beschrieben – zu behandeln ist. Weitere Informationen über gesperrte Zertifikate können in dem optionalen Teilfeld *crlEntryExtensions* als CRL-Eintragserweiterungen spezifiziert werden. Diese zertifikatsspezifischen CRL-Erweiterungen [ITU-T X.509 97, ANSI X9.55 95], für die auch private Erweiterungen definiert werden können, gestatten das Hinzufügen von zusätzlichen Attributen zu CRL-Einträgen.

Derzeit sind für das Internet die als *non-critical* eingestuften optionalen CRL-Eintragserweiterungen *reason code, hold instruction code, invalidity date* und *certificateIssuer* definiert, die in den folgenden Unterpunkten beschrieben werden.

ASN.1-Definitionen

```
TBSCertList             ::=  SEQUENCE {
    ...,
    revokedCertificates    .     ..,
    ... }
revokedCertificates             SEQUENCE OF SEQUENCE {
    userCertificate             CertificateSerialNumber,
    revocationDate              Time,
    crlEntryExtensions          Extensions OPTIONAL } OPTIONAL,
CertificateSerialNumber         INTEGER
Time                    ::=     CHOICE {
    utcTime                     UTCTime,
    generalizedTime             GeneralizedTime }
Extensions              ::=     SEQUENCE (1..MAX) OF Extension
Extension               ::=     SEQUENCE {
    extnId                      OBJECT IDENTIFIER,
    critical                    BOOLEAN DEFAULT FALSE,
    extnValue                   OCTET STRING }
EXTENSION               ::=     CLASS {
    &id                         OBJECT IDENTIFIER UNIQUE,
    &ExtType }
WITH SYNTAX {
    SYNTAX                      &ExtnType
    IDENTIFIED BY               &id }
certificateExtension OBJECT IDENTIFIER    ::=  { 2 5 29 }
id-ce OBJECT IDENTIFIER     ::=  certificateExtension
```

Allgemeine Konformitätsanforderungen

Erweiterungen in Sperrlisteneinträgen können als *critical* oder als *non-critical* gekennzeichnet werden. Der CRL-Verifikationsprozeß soll zu einem *fail*-Ergebnis führen, falls er eine als *critical* markierte CRL-Erweiterung nicht verarbeiten kann. Unbekannte und als *non-critical* spezifizierte Erweiterungen dürfen bei der Validierung ignoriert werden.

Die Unterstützung der optionalen und als *non-critical* festgelegten Erweiterungen für das Internet ist für konforme Zertifizierungsstellen und Anwendungen freiwillig. Zertifizierungsstellen sollten jedoch bei der Erstellung von Sperrlisten die *reason code*-Erweiterung benutzen, sofern ihnen diese Information vorliegt.

SigI-Konformitätsanforderungen

Bei der Erstellung von Sperrlisten ist die Benutzung der *holdInstructionCode*-, und der *invalidityDate*-CRL-Eintragserweiterung verboten. Die Benutzung der *certificateIssuer- und der reasonCode*-CRL-Eintragserweiterung ist optional.

Tabelle 14: Implementations-technische Informationen über CRL-Eintragserweiterung

ERWEITERUNG	RELE-VANZ			KLASSIFI-KATION			
	obligatorisch	verboten	optional	Standarderweiterung	private Erweiterung	critical-Markierung	non-critical-Markierung
reasonCode			v	v			v
holdInstructionCode		v		v			v
InvalidityDate		v		v			v
Certificate issuer			v	v		v	

3.1.3.6.1 ERWEITERUNG DER SPERRLISTEN-EINTRÄGE

SPERRGRÜNDE

Zweck

Das *reasonCode*-Feld ist eine *non-critical* CRL-Eintragserweiterung, die den Grund für die Sperrung eines Zertifikates anzeigt. Sperrgründe werden syntaktisch durch einen ASN.1-Aufzähltyp beschrieben, der aus einer Menge von benamten Integerwerten besteht. Sperrgründe können von Anwendungen dazu benutzt werden, um zu entscheiden, wie sie auf die Anzeige eines gesperrten Zertifikates in Abhängigkeit von ihren lokalen Sicherheitsrichtlinien reagieren sollen.

ASN.1-Definitionen

```
Extension               ::=   SEQUENCE {
    extnId                        OBJECT IDENTIFIER,
    critical                      BOOLEAN DEFAULT FALSE,
    extnValue                     OCTET STRING }
id-ce OBJECT IDENTIFIER     ::=  { 2 5 29 }
id-ce-cRLReason OBJECT IDENTIFIER ::= { 2 5 29 21 }
cRLReason EXTENSION     ::=   {
    SYNTAX                        CRLReason
    IDENTIFIED BY                 id-ce-cRLReason }
CRLReason               ::=   ENUMERATED {
    unspecified                   (0),
    keyCompromise                 (1),
    cACompromise                  (2),
    affiliationChanged            (3),
    superseded                    (4),
    cessationOfOperation          (5),
    certificateHold               (6),
    removeFromCRL                 (7) }
```

Allgemeine Konformitätsanforderungen

Die Verwendung der *reason codes*-Erweiterung im *crlEntryExtension*-Feld durch Zertifizierungsstellen wird dringenst empfohlen. Sie sollte nur dann unterbleiben, falls der Grund für die Sperrung eines Zertifikates nicht bekannt ist, d.h. der Sperrgrund *unspecified* (nicht spezifiziert) sollte nicht benutzt werden. Die Bedeutung der einzelnen Spergründe ist in der folgenden Tabelle zusammengefaßt.

Tabelle 15: Bedeutung von Sperrgründen

BEZEICHNER	WERT	SPERRGRÜNDE
unspecified	0	nicht-spezifizierter Sperrgrund
keyCompromise	1	Kompromittierung eines privaten Teilnehmerschlüssels
CACompromise	2	Kompromittierung eines privaten Schlüssels einer Zertifizierungsstelle
affiliationChanged	3	Änderung der Namensinformationen eines Zertifikates, ohne daß eine Kompromittierung des privaten Schlüssels vorliegt
superseded	4	Ablauf der Gültigkeit eines Zertifikates, ohne daß eine Kompromittierung des privaten Schlüssels vorliegt
cessationOfOperation	5	Zertifikat wird vor Ablauf seiner Gültigkeit nicht mehr benötigt, ohne daß eine Kompromittierung des privaten Schlüssels vorliegt
certificateHold	6	vorübergehende Sperrung eines Zertifikates
removeFromCRL	7	wird in Verbindung mit delta-CRLs benutzt

SigI-Konformitätsanforderungen

Die Benutzung des *reasons*-Teilfeld ist bei der Generierung von Sperrlisten optional. Hierbei sind nur die Sperrgründe *keyComromise*, *CACompromise*, *affiliationChanged* und *cessationOfOperation* zulässig. Die Benutzung aller anderen Sperrgründe ist verboten.

Tabelle 16: Implementations-technische Informationen über *cRLReason*

BEZEICHNER	WERTEBEREICH EINZELWERTE	HEXADEZIMALCODE	LÄNGE	RELE-VANZ			KLASSIFI-KATION			
	(BEISPIELE)	(BEISPIELE)	[BYTES] 12	obligatorisch	verboten	optional	Standarderweiterung	private Erweiterung	critical-Markierung	non-critical-Markierung
cRLReason	SEQUENCE {	**30** 0A	12			v	v			v
extnId	{ 2 5 29 21 },	**06** 03 *55 1D 15*								
critical	FALSE,									
extnValue	OCTET STRING	**04** 03								
CACompromise	ENUMERATED }	**0A** 01 *02*								

SPERRINSTRUKTIONEN

Zweck

Das *holdInstructionCode*-Feld ist eine *non-critical* CRL-Erweiterung, die einen registrierten Instruktionsbezeichner enthält, der die auszuführenden Aktionen festlegt, die für temporär gesperrte Zertifikate (*certificate hold*) gelten. Gegenwärtig sind für das Internet die drei Instruktionscodes *holdinstruction-none*, *holdinstruction-callissuer* und *holdinstruction-reject* festgelegt.

ASN.1-Definitionen

```
Extension                       ::=  SEQUENCE {
    extnId                           OBJECT IDENTIFIER,
    critical                         BOOLEAN DEFAULT FALSE,
    extnValue                        OCTET STRING }

id-ce OBJECT IDENTIFIER         ::=  { 2 5 29 }

id-ce-holdInstructionCode OBJECT IDENTIFIER ::= { 2 5 29 23 }

holdInstructionCode   EXTENSION ::=  {
    SYNTAX                           HoldInstructionCode
    IDENTIFIED BY                    id-ce-holdInstructionCode }

HoldInstructionCode             ::=  CHOICE {
    id-holdinstruction-none          OBJECT IDENTIFIER,
    id-holdinstruction-callissuer    OBJECT IDENTIFIER,
    id-holdinstruction-reject        OBJECT IDENTIFIER }

id-holdInstruction                   OBJECT IDENTIFIER   ::=
    { 1 2 840 10040 2 }

id-holdinstruction-none              OBJECT IDENTIFIER   ::=
    {1 2 840 10040 2 1}

id-holdinstruction-callissuer        OBJECT IDENTIFIER   ::=
    {1 2 840 10040 2 2}

id-holdinstruction-reject            OBJECT IDENTIFIER   ::=
    {1 2 840 10040 2 3}
```

Allgemeine Konformitätsanforderungen

Konforme Anwendungen, die diese Erweiterung verarbeiten, sollen beim Empfang einer *callissuer*-Instruktion entweder den Zertifikatsersteller kontaktieren oder das Zertifikat zurückweisen. Desweiteren sollen sie beim Empfang einer *reject*-Instruktion das Zertifikat zurückweisen. Die *none*-Instruktion ist semantisch äquivalent zu einem fehlenden *holdInstructionCode*-Feld und wird für die Internet-PKI verworfen.

SigI-Konformitätsanforderungen

Die Benutzung der *holdInstructionCode*-Eintragserweiterung ist bei der Generierung von Sperrlisten verboten, da auch die Benutzung des *certificateHold*-Sperrgrundes und damit eine zeitweilige Sperrung von Zertifikaten verboten ist. Zum einen darf nach [SigG 97 §8(1)] und [SigV 97 §9(3)] eine Sperrung nicht rückgängig gemacht werden, d.h. ein Zertifikat darf auch nicht vorübergehend gesperrt und später wieder reaktiviert werden. Zum anderen werden zur

Suspendierung zeitlich überlappende CRLs benötigt, um eine kurzfristige Suspendierung zu ermöglichen. Zeitlich überlappende CRLs auszustellen ist jedoch verboten (siehe 3.1.3.5). Wenn es zu einem Zeitpunkt mehrere gültige Sperrlisten gibt, so kann nicht sichergestellt werden, daß der Verifizierer einer digitalen Signatur die neueste CRL verwendet

Tabelle 17: Implementations-technische Informationen über *holdInstructionCode*

BEZEICHNER	WERTEBEREICH EINZELWERTE	HEXADEZIMALCODE	LÄNGE	RELE-VANZ			KLASSIFI-KATION			
	(BEISPIELE)	(BEISPIELE)	[BYTES]	obligatorisch	verboten	optional	Standarderweiterung	private Erweiterung	critical-Markierung	non-critical-Markierung
holdInstructionCode					v		v			v

SPERRDATUM

Zweck

Das *invalidityDate*-Feld ist eine *non-critical* CRL-Eintragserweiterung, die den Zeitpunkt und das Datum des Bekanntwerdens eines kompromittierten privaten Schlüssels oder eines anderweitigen Ungültigwerdens eines Zertifikates anzeigt. Dieses Feld ist nicht zu verwechseln mit dem CRL-Feld *revocationDate* (siehe Abschnitt 3.1.3.6), das für jedes widerrufene Zertifikat den Sperrzeitpunkt angibt.

ASN.1-Definitionen

```
Extension                    ::=  SEQUENCE {
   extnId                         OBJECT IDENTIFIER,
   critical                       BOOLEAN DEFAULT FALSE,
   extnValue                      OCTET STRING }

id-ce OBJECT IDENTIFIER      ::=  { 2 5 29 }

id-ce-invalidityDate OBJECT IDENTIFIER ::= { 2 5 29 24 }

invalidityDate EXTENSION     ::=  {
   SYNTAX                         GeneralizedTime
   IDENTIFIED BY                  id-ce-invalidityDate }
```

Allgemeine Konformitätsanforderungen

Zeitlich betrachtet kann der Wert *invalidityDate*-Feldes vor dem in *revocationDate* (siehe Abschnitt 3.1.3.6) angegebenem Zeitpunkt liegen. Bei der Kodierung des Feldes im *GeneralizedTime* Zeit- und Datumsformat sind die im Abschnitt 3.1.3.4 angegebenen Konformitätsanforderungen zu beachten.

SigI-Konformitätsanforderungen

Bei der Erzeugung von Sperrlisten ist die Benutzung des *invalidityDate*-Feldes verboten.

Tabelle 18: Implementations-technische Informationen über *invalidityDate*

BEZEICHNER	WERTEBEREICH EINZELWERTE	HEXADEZIMALCODE	LÄNGE	RELE-VANZ			KLASSIFI-KATION			
	(BEISPIELE)	(BEISPIELE)	[BYTES]	obligatorisch	verboten	optional	Standarderweiterung	private Erweiterung	critical-Markierung	non-critical-Markierung
invalidityDate					v		v			v

IDENTIFIZIERUNG DES ZERTIFIKATERSTELLERS

Zweck

Das *certificateIssuer*-Feld ist eine CRL-Eintragserweiterung, die den Ersteller eines gesperrten Zertifikates in einer indirekten CRL identifiziert. *IndirectCRLs* sind Sperrlisten, die nicht nur gesperrte Zertifikate einer Zertifizierungsstelle enthalten, sondern Sperrinformationen verschiedener Zertifizierungsstellen zusammenfassen. Zur Kenntlichmachung einer *indirectCRL* muß das Teilfeld *indirectCRL* in der *issuingDistributionPoint* CRL-Erweiterung gesetzt werden. Falls das certificateIssuer-Feld nicht im ersten Eintrag einer indirekten CRL gesetzt ist, so nimmt es per Voreinstellung den durch das issuer-Feld gegebenen Wert für den CRL-Ersteller an. Fehlt dieses Feld in den nachfolgenden Einträgen einer indirekten CRL, so wird der Wert für den CRL-Ersteller durch den Wert des vorangegangenen Eintrages bestimmt.

ASN.1-Definitionen

```
Extension               ::=  SEQUENCE {
    extnId                       OBJECT IDENTIFIER,
    critical                     BOOLEAN DEFAULT FALSE,
    extnValue                    OCTET STRING }

certificateIssuer EXTENSION ::= {
    SYNTAX                       CertificateIssuer
    IDENTIFIED BY                id-ce-certificateIssuer }
id-ce-certificateIssuer OBJECT IDENTIFIER ::=  { 2 5 29 29 }
CertificateIssuer       ::=  GeneralNames
```

Allgemeine Konformitätsanforderungen

Konforme Zertifizierungsstellen, die diese Erweiterung benutzen, sollten sie stets als *critical* kennzeichnen.

Konforme Anwendungen sollten diese Erweiterung nicht ignorieren, weil sie ansonsten die CRL-Einträge nicht eindeutig an bestimmte Zertifikate zuordnen können.

SigI-Konformitätsanforderungen

Die *certificateIssuer*-Erweiterung ist optional. Das *certificateIssuer*-Feld wird im Falle der Übernahme von Zertifikaten durch eine andere Zertifizierungsstelle benötigt [SigG 97, §13 (4)], da ansonsten die Gefahr besteht, daß Seriennummern doppelt vorkommen.

Tabelle 19: Implementations-technische Informationen über *certificateIssuer*

BEZEICHNER	WERTEBEREICH EINZELWERTE	HEXADEZIMALCODE	LÄNGE	RELE-VANZ			KLASSIFI-KATION			
	(BEISPIELE)	(BEISPIELE)	[BYTES]	obligatorisch	verboten	optional	Standarderweiterung	private Erweiterung	critical-Markierung	non-critical-Markierung
CertificateIssuer						v	v		v	

3.1.3.7 Sperrlistenerweiterungen

Zweck

Das CRL-Erweiterungsfeld *crlExtensions* besteht aus einer Folge von CRL-Erweiterungen [ITU-T X.509 97, ANSI X9.55 95], die auch als private Erweiterungen definiert werden können und die das Hinzufügen von zusätzlichen Attributen zu Sperrlisten gestatten.

In [ITU-T X.509 97, PKIX PRO 98] sind die als non-critical eingestuften optionalen CRL-Erweiterungen *authority key identifier, issuer alternative name, CRL number, issuing distribution point* und die als critical eingestufte CRL-Erweiterung *delta CRL indicator* definiert, die in den folgenden Unterpunkten beschrieben werden.

OpenCDP definiert darüber hinaus die CRL-Erweiterung *cRLScope*, die im Abschnitt 3.3.2.1 beschrieben ist.

Allgemeine Konformitätsanforderungen

Das *crlExtensions*-Teilfeld darf nur in Verbindung mit Sperrlisten der Version v2 benutzt werden.

Erweiterungen in Sperrlisten können als *critical* oder als *non-critical* gekennzeichnet werden. Der CRL-Verifikationsprozeß soll zu einem *fail*-Ergebnis führen, falls er eine als *critical* markierte CRL-Erweiterung nicht verarbeiten kann. Unbekannte und als *non-critical* spezifizierte Erweiterungen dürfen bei der Validierung ignoriert werden.

Die Unterstützung der optionalen und als *non-critical* festgelegten Erweiterungen für das Internet ist für konforme Zertifizierungsstellen und Anwendungen freiwillig. Konforme Zertifizierungsstellen sollten jedoch bei der Erstellung von Sperrlisten die *crlNumber*-Erweiterung benutzen und in allen erstellten Sperrlisten einsetzen. Konforme Anwendungen sollten die als *critical* gekennzeichneten Erweiterungen unterstützen können.

SigI-Konformitätsanforderungen

Bei der Erstellung von Sperrlisten müssen die als *non-critical* eingestuften optionalen CRL-Erweiterungen *authority key identifier* und *CRL number* verwendet werden. Die Benutzung von *issuingDistributionPoint* ist verboten, während *deltaCRLIndicator* und *issuerAlternativeName* optional sind.

Tabelle 20: Implementations-technische Informationen über Erweiterungen

ERWEITERUNG	RELEVANZ			KLASSIFIKATION			
	obligatorisch	verboten	optional	Standarderweiterung	private Erweiterung	critical-Markierung	non-critical-Markierung
authority key identifier	v			v			v
issuer alternative name			v	v			v
CRL number	v			v			v
issuing distribution point		v		v			v
delta CRL indicator			v	v		v	

3.1.3.7.1 IDENTIFIZIERUNG VON SIGNATURSCHLÜSSELN VON ZERTIFIZIERUNGSSTELLEN

Zweck

Das *authorityKeyIdentifier*-Erweiterungsfeld dient zur Identifizierung eines bestimmten öffentlichen Schlüssels einer Zertifizierungsstelle, der zum Signieren einer Sperrliste verwendet wurde. Die Erweiterung wird dann verwendet, wenn eine Zertifizierungsstelle mehrere Signaturschlüssel – sei es als gleichzeitig aktive Schlüssel oder zum Schlüsselwechsel – besitzt. Die Identifizierung kann entweder durch den Schlüsselnamen im *keyIdentifier*-Teilfeld oder durch den Namen der Zertifizierungsstelle im *authorityCertIssuer*-Teilfeld und die Seriennummer im *authorityCertSerialNumber*-Teilfeld erfolgen.

Die Kombination *authorityCertIssuer* und *authorityCertSerialNumber* identifiziert eindeutig ein bestimmtes Zertifikat einer Zertifizierungsstelle. Der *keyIdentifier* kennzeichnet den öffentlichen Schlüssel der Zertifizierungsstelle und somit nicht nur eine, sondern gegebenfalls eine Reihe von Zertifikaten. Damit erlaubt die Verwendung des *keyIdentifiers* eine größere Flexibilität zum Auffinden eines Zertifizierungspfades. Außerdem müssen die Zertifikate, die den *keyIdentifier* im *authorityKeyIdentifier*-Erweiterungsfeld benutzen, nicht zurückgezogen werden, wenn die Zertifizierungsstelle sich bei gleichbleibendem Schlüssel ein neues Zertifikat ausstellen läßt.

Andererseits ist die Flexibilität zum Auffinden eines Zertifizierungspfades nicht immer gewünscht. Verfügt eine Zertifizierungsstelle über mehrere Zertifikate für den gleichen Schlüssel, die aber beispielsweise verschiedene Haftungsgrenzen beinhalten, so ist es erforderlich, nicht nur den öffentlichen Schlüssel sondern genau dasjenige Zertifikat der Zertifizierungsstelle zu referenzieren, das für den jeweiligen Teilnehmer gültig ist.

ASN.1-Definitionen

```
Extension                ::=  SEQUENCE {
    extnId                        OBJECT IDENTIFIER,
    critical                      BOOLEAN DEFAULT FALSE,
    extnValue                     OCTET STRING }

authorityKeyIdentifier EXTENSION ::= {
    SYNTAX                        AuthorityKeyIdentifier
    IDENTIFIED BY                 id-ce-authorityKeyIdentifier }

id-ce-authorityKeyIdentifier OBJECT IDENTIFIER ::= { 2 5 29 35}

AuthorityKeyIdentifier ::=  SEQUENCE {
    keyIdentifier                 [0] KeyIdentifier OPTIONAL,
    authorityCertIssuer           [1] GeneralNames  OPTIONAL,
    authorityCertSerialNumber     [2] CertificateSerialNumber OPTIONAL }

KeyIdentifier          ::=    OCTET STRING
```

Allgemeine Konformitätsanforderungen

Entweder sind die beiden *authorityCertIssuer*- und *authorityCertSerialNumber*-Teilfelder in Zertifikate zu integrieren oder beide wegzulassen. Im zweiten Fall muß stattdessen das *keyIdentifier*-Teilfeld eingebaut werden.

Falls beide Identifizierungsmethoden benutzt werden, sollte die Zertifizierungsstelle deren Konsistenz sicherstellen. Ein Schlüsselbezeichner soll bezüglich aller Schlüsselbezeichner, die eine Zertifizierungsstelle für einen Zertifikatsinhaber benutzt, eindeutig sein.

Systeme sollten die Fähigkeit besitzen, Zertifizierungspfade finden und validieren zu können, wenn die ausstellende Zertifizierungsstelle mehrere Signaturschlüssel besitzt. Sie sollten eine der beiden Identifikationsmethoden zum Auffinden von Zertifzierungspfaden unterstützen.

SigI-Konformitätsanforderungen

Die Benutzung dieser Erweiterung ist in allen Sperrlisten obligatorisch und sie muß als *non-critical* gekennzeichnet werden. Außerdem soll als Schlüsselidentifizierungsmethode die Verwendung der beiden *authorityCertIssuer*- und *authorityCertSerialNumber*-Teilfelder unterstützt werden, die ein bestimmtes Zertifikat der Zertifizierungsstelle eindeutig identifiziert. Dabei muß im *authorityCertIssuer*-Teilfeld zumindest der *issuer*-Name des Zertifikaterstellers vom Typ *directoryName* angegeben werden.

Tabelle 21: Implementations-technische Informationen über authorityKeyIdentifier

BEZEICHNER	WERTEBEREICH EINZEILWERTE	HEXADEZIMALCODE	LÄNGE	RELE-VANZ			KLASSIFI-KATION			
	(BEISPIELE)	(BEISPIELE)	[BYTES] 280	obligatorisch	verboten	optional	Standarderweiterung	private Erweiterung	critical-Markierung	non-critical-Markierung
authorityKey Identifier	SEQUENCE {	**30** 68	106	v			v			v
ExtnId	{ 2 5 29 35 },	**06** 03 *55 1D 23*								
Critical	FALSE,									
ExtnValue	OCTET STRING	**04** 60								
	SEQUENCE {	**30** 5E								
AuthCertIssuer	[1] SEQUENCE OF {	**80** 57								
DirectoryName	[4] SEQUENCE OF {	**84** 55 **30** 53								
	SET OF SEQUENCE {	**31** 0B **30** 09								
CountryName	{ 2 5 4 6 },	**06** 03 *55 04 06*								
Value	"DE" }	**13** 02 *44 45*								
Organization	SET OF SEQUENCE {	**31** 0E **30** 0C								
Name value	{ 2 5 4 10 },"REGTP" }	**06** 03 *55 04 0A* **13** 05 *52 45 47*								
Organizational		*54 50*								
Unit	SET OF SEQUENCE {	**31** 24 **30** 22								
Value	{ 2 5 4 11 },	**06** 03 *55 04 0B*								
	"Wurzelzertifizierungs stelle" } }	**13** 1B *57 75 72 7A 65 6C 7A 65 72 74 69 66 69 7A 69 65 72 75 6E 67 73 73 74 65 6C 6C 65*								
CommonName		**31** 0E **30** 0C								
	SET OF SEQUENCE {	**06** 03 *55 04 03*								
Value	{ 2 5 4 3 },	**13** 05 *44 45 50 43 41*								
AuthCertSerNum	"DEPCA" } } }, [2] 1 }	**82** 03 **02** 01 *01*								

3.1.3.7.2 ALTERNATIVE NAMEN VON SPERRLISTENERSTELLERN

Zweck

Das *issuerAltName*-Erweiterungsfeld enthält einen oder mehrere alternative Namen für den Ersteller einer Sperrliste, durch die zusätzliche Identitäten an den Sperrlistenersteller gebunden werden.

Neben dem *distinguished name* des Sperrlistenerstellers können im alternativen Namensfeld des Ausstellers zusätzliche Adreßinformationen zur Erreichbarkeit im Internet abgelegt werden. Dazu gehören insbesondere die Angabe einer Internetadresse für elektronische Post, sowie Angaben über den DNS-Namen des Sperrlistenerstellers (DNS, domain name system).

Die Adresse für elektronische Post (rfc822) sollte eine symbolische Mailadresse sein, die es einem Teilnehmer ermöglicht, Kontakt zur Zertifizierungsstelle aufzunehmen. Es sollen an dieser Stelle keine persönlichen Mailadressen von Mitarbeitern verwendet werden.

Der DNS-Name der Zertifzierungsstelle sollte der registrierte Domain-Name der Zertifizierungsstelle sein. Über diesen Namen können Anwendungen die Adressen zusätzlicher Dienste und Protokolle der Zertifizierungsstelle ermitteln, wie beispielsweise die Adresse eines X.500-Verzeichnisdienstes. Diese Vorgehensweise ist eine Alternative zur Verwendung des globalen X.500 Verzeichnisdienst. Mit etablierten Verfahren (beispielsweise [RFC 2052 96]) können die Adressen der gewünschten Dienste aus dem angegebenen DNS-Namen abgeleitet werden. Zu beachten ist, daß Informationen über den DNS-Namen einer Zertifizierungsstelle auch im *distinguished name* der Zertifizierungsstelle angegeben sein können. Dies geschieht durch die Definition des sog. *DC*-Bezeichners (DC, domain component, Teilname eines Domänennamens) für *distingushed names* [RFC 2247 98]).

Der im Rahmen der SigI-Spezifikation definierte Datentyp *PersonalData* kann auch als *issuerAltName* verwendet werden, um den gesetzlichen Namen bzw. das Pseudonym der Zertifizierungsstelle in den von ihr ausgestellten Sperrlisten einzutragen.

ASN.1-Definitionen

```
Extension           ::= SEQUENCE {
   extnId                   OBJECT IDENTIFIER,
   critical                 BOOLEAN DEFAULT FALSE,
   extnValue                OCTET STRING }

issuerAltName EXTENSION     ::= {
   SYNTAX                   IssuerAltName
   IDENTIFIED BY            id-ce-issuerAltName }
id-ce-issuerAltName OBJECT IDENTIFIER ::=  { 2 5 29 18 }
IssuerAltName       ::= GeneralNames
GeneralNames        ::= SEQUENCE SIZE (1..MAX) OF GeneralName
GeneralName         ::= CHOICE {
   otherName                [0]  OTHER-NAME,
```

```
    rfc822Name              [1]  IA5String,
    dNSName                 [2]  IA5String,
    x400Address             [3]  ORAddress,
    directoryName           [4]  Name,
    ediPartyName            [5]  EDIPartyName,
    uniformResourceId       [6]  IA5String,
    iPAddress               [7]  OCTET STRING,
    registeredID            [8]  OBJECT IDENTIFIER }

OTHER-NAME          ::= SEQUENCE {
    type-id                 OBJECT IDENTIFIER,
    value                   [0]  EXPLICIT ANY DEFINED BY type-id }

EDIPartyName        ::= SEQUENCE {
    nameAssigner            [0]  DirectoryString OPTIONAL,
    partyName               [1]  DirectoryString }

id-sigi-on OBJECT IDENTIFIER     ::=  { 1 3 36 8 4 }

id-sigi-on-personalData     OBJECT IDENTIFIER
                                 ::=   { 1 3 36 8 4 1 }

PersonalData        ::= SEQUENCE {
    nameOrPseudonym         NameOrPseudonym,
    nameDistinguisher       [0]  INTEGER              OPTIONAL,
    dateOfBirth             [1]  GeneralizedTime      OPTIONAL,
    placeOfBirth            [2]  DirectoryString      OPTIONAL,
    gender                  [3]  PrintableString      OPTIONAL,
    postalAddress           [4]  DirectoryString      OPTIONAL }

NameOrPseudonym     ::= CHOICE {
    surAndGivenName         SEQUENCE {
      surName               DirectoryString,
      givenName             SEQUENCE OF DirectoryString },
      pseudoNym             DirectoryString }
```

Statische Semantik

Bei Namensgleichheit ist die Benutzung der *nameDistinguisher*-Komponente obligatorisch. Das Geburtsdatum *dateOfBirth* darf nur Datumsangaben gemäß der Syntax YYYYMM.DD (Jahreszahl Monat Tag) enthalten. Für die Komponente *gender* sind die zwei Werte "F" für female (weiblich) und "M" für male (männlich) vordefiniert.

Allgemeine Konformitätsanforderungen

Die Benutzung des String-Platzhaltersymbols “*” in Namenstypen des *issuerAltName*-Erweiterungsfeldes ist verboten.

SigI-Konformitätsanforderungen

Die optionale *issuerAltName*-Erweiterung kann bei der Erstellung von Sperrlisten benutzt werden und ist dabei als *non-critical* zu kennzeichnen. Diese Erweiterung besitzt jedoch keine Bedeutung für die technische Identifikation des Erstellers der Sperrliste, sondern sie bindet lediglich weitere Merkmale (z.B. E-Mail-Adressen) an ihn.

BEISPIELE FÜR DIE BENUTZUNG DER *ISSUERALTNAME*-ERWEITERUNG:

(1) Mailadresse: *rfc822Name*: rootca@regtp.de

(2) X.500-Verzeichnisdienstname mit Angabe der E-Mail Adresse der Zertifizierungsstelle: *directoryName*: CN=Verzeichnisdienst, EMAIL=ca@cert.de, O=ZS1, C=DE

Tabelle 22: Implementations-technische Informationen über *issuerAltName*

BEZEICHNER	WERTEBEREICH EINZEILWERTE	HEXADEZIMALCODE	LÄNGE	RELE-VANZ			KLASSIFI-KATION			
	(BEISPIELE)	(BEISPIELE)	[BYTES] 500	obligatorisch	verboten	optional	Standarderweiterung	private Erweiterung	critical-Markierung	non-critical-Markierung
issuerAltName extnId critical extnValue rfc822Name	SEQUENCE { { 2 5 29 18 }, FALSE, OCTET STRING SEQUENCE OF{[1] "ca@cert.de" } }	**30** 15 **06** 03 *55 1D 12* **04** 0E **30** 0C **81** 0A *63 61 40 63 65 72 74 2E 64 65*	23			v	v			v

3.1.3.7.3 SPERRLISTENNUMMERN

Zweck

Das *cRLNumber*-Feld ist eine *non-critical* CRL-Erweiterung, die eine um den Wert 1 anwachsende Folgenummer für jede von einer bestimmtem Zertifizierungsstelle erzeugte Sperrliste enthält. Sperrlisten können entweder über X.500-Dienste, mit Hilfe der *crlDistributionPoints*-Erweiterung oder über den OpenCDP-Mechanismus (siehe Abschnitt 3.2.2) den Teilnehmern zur Verfügung gestellt werden. Diese Erweiterung gestattet den Teilnehmern eine einfache Unterscheidungsmöglichkeit darüber, ob eine bestimmte CRL eine andere CRL ersetzt.

ASN.1-Definitionen

```
Extension               ::=   SEQUENCE {
    extnId                        OBJECT IDENTIFIER,
    critical                      BOOLEAN DEFAULT FALSE,
    extnValue                     OCTET STRING }

cRLNumber EXTENSION     ::=   {
    SYNTAX                        CRLNumber
    IDENTIFIED BY                 id-ce-cRLNumber }

id-ce-cRLNumber OBJECT IDENTIFIER    ::=  { 2 5 29 20 }

CRLNumber          ::= INTEGER (0..MAX)
```

Allgemeine Konformitätsanforderungen

Konforme Zertifizierungsstellen sollen diese Erweiterung in allen Sperrlisten verwenden.

SigI-Konformitätsanforderungen

Bei der Erstellung von Sperrlisten muß die *cRLNumber*-Erweiterung benutzt und dabei als *non-critical* gekennzeichnet werden. Der Systemparameter MAX ist auf den Wert 2^{8*15-1}-1 begrenzt.

Tabelle 23: Implementations-technische Informationen über *cRLNumber*

BEZEICHNER	WERTEBEREICH EINZEILWERTE	HEXADEZIMALCODE	LÄNGE	RELE-VANZ			KLASSIFI-KATION			
	(BEISPIELE)	(BEISPIELE)	[BYTES] 26	obligatorisch	verboten	optional	Standarderweiterung	private Erweiterung	critical-Markierung	non-critical-Markierung
cRLNumber extnId critical extnValue	SEQUENCE { { 2 5 29 20 }, FALSE, OCTET STRING 1}	**30** 0A **06** 03 *55 1D 14* **04** 03 **02** 01 *01*	12	v			v			v

3.1.3.7.4 IDENTIFIKATION DER QUELLEN VON SPERRLISTEN

Zweck

Das *issuingDistributionPoint*-Feld ist eine *critical* CRL-Erweiterung, die den Verteilungspunkt (CRL distribution point) einer bestimmten Sperrliste identifiziert und anzeigt, ob die Sperrliste nur Endanwenderzertifikate, oder nur CA-Zertifikate oder nur Zertifikate für eine begrenzte Menge von Sperrgründen enthält.

Sperrlisten werden mit dem privaten Signaturschlüssel der betreffenden Zertifizierungsstelle signiert. *CRL-distribution points* besitzen keine eigenen Schlüsselpaare. Falls CRLs in einem X.500-Verzeichnisdienst gespeichert werden, so beinhaltet das *distributionPoint*-Feld im *issuingDistributionPoint* die entsprechende Adresse, die nicht mit dem X.500-Eintrag der Zertifizierungsstelle übereinstimmen muß. Zertifizierungsstellen können das *issuingDistributionPoint*-Feld zur Partitionierung von Sperrlisten einsetzen.

ASN.1-Definitionen

```
Extension               ::=  SEQUENCE {
    extnId                       OBJECT IDENTIFIER,
    critical                     BOOLEAN DEFAULT FALSE,
    extnValue                    OCTET STRING }
```

```
issuingDistributionPoint EXTENSION    ::= {
   SYNTAX                             IssuingDistributionPoint
   IDENTIFIED BY                      id-ce-issuingDistributionPoint }

id-ce-issuingDistributionPoint OBJECT IDENTIFIER    ::=
   { 2 5 29 28 }

IssuingDistributionPoint    ::=  SEQUENCE {
   distributionPoint           [0]   DistributionPointName OPTIONAL,
   onlyContainsUserCerts       [1]   BOOLEAN DEFAULT FALSE,
   onlyContainsCACerts         [2]   BOOLEAN DEFAULT FALSE,
   onlySomeReasons             [3]   ReasonFlags OPTIONAL,
   indirectCRL                 [4]   BOOLEAN DEFAULT FALSE }

DistributionPointName       ::=  CHOICE {
   fullName                         [0]   GeneralNames,
   nameRelativeToCRLIssuer          [1]   RelativeDistinguishedName }

ReasonFlags                 ::=  Bit String {
   unused                           (0),
   keyCompromise                    (1),
   cACompromise                     (2),
   affiliationChanged               (3),
   superseded                       (4),
   cessationOfOperation             (5),
   certificateHold                  (6) }
```

Allgemeine Konformitätsanforderungen

Obwohl diese Erweiterung als *critical* gekennzeichnet ist, muß sie nicht von Anwenderinfrastrukturen unterstützt werden.

Zertifizierungsstellen können das *issuingDistributionPoint*-Feld zur Partitionierung von Sperrlisten einsetzen. In diesem Fall müssen z.B. alle mit dem Sperrgrund *keyCompromise* gesperrten Zertifikate in einem bestimmten *distributionPoint* und alle mit einem anderen Sperrgrund gesperrten Zertifikate in einem anderen *distributionPoint* abgelegt sein.

Die mit *distributionPoints* verknüpften Sperrgründe müssen bei segmentierten Sperrlisten im *onlySomeReasons*-, *onlyContainUserCerts*- oder *onlyContainCACerts*-Teilfeld spezifiziert werden. Ansonsten, d.h. falls eine vollständige Sperrliste für alle Sperrgründe vorliegt, können diese Teilfelder weggelassen werden.

Falls das *issuingDistributionPoint*-Feld eine URL enthält, so soll diese als ein Verweis auf die aktuelle CRL der ausstellenden Zertifizierungsstelle betrachtet werden. Als Namensformate dienen hierbei die URI-Formate ftp, http, mailto und ldap, die über einen absoluten Pfadnamen den betreffenden Rechner angeben müssen.

SigI-Konformitätsanforderungen

Bei der Erstellung von Sperrlisten ist die Benutzung der *issuingDistributionPoint*-Erweiterung verboten.

Tabelle 24: Implementations-technische Informationen über *issuingDistributionPoint*

BEZEICHNER	WERTEBEREICH EINZELWERTE	HEXADEZIMALCODE	LÄNGE	RELE-VANZ			KLASSIFI-KATION			
	(BEISPIELE)	(BEISPIELE)	[BYTES]	obligatorisch	verboten	optional	Standarderweiterung	private Erweiterung	critical-Markierung	non-critical-Markierung
IssuingDistributionPoint					v		v		v	

3.1.3.7.5 INDIKATOR VON SPERRLISTENÄNDERUNGEN

Zweck

Das *deltaCRLIndicator*-Feld ist eine *critical* CRL-Erweiterung, die eine Sperrlistenänderung (delta-CRL) identifiziert. Durch die Benutzung dieser Erweiterung kann die Effektivität von Anwendungen erhöht werden, die Sperrinformationen in einem von der CRL-Struktur abweichendem Format speichern, weil hierbei nur die tatsächlichen Änderungen in der lokalen Datenbank durchgeführt und die unveränderten und lokal bereits gespeicherten Informationen ignoriert werden können.

Durch das Feld *BaseCRLNumber* vom Typ *CRLNumber* wird die Folgenummer der CRL identifiziert, die als Ausgangspunkt für die Erzeugung dieser *delta-CRL* genommen wurde.

Die *delta-CRL* enthält die Änderungen zwischen der Ausgangs-CRL und der aktuellen CRL, die zusammen mit der *delta-CRL* erstellt wird.

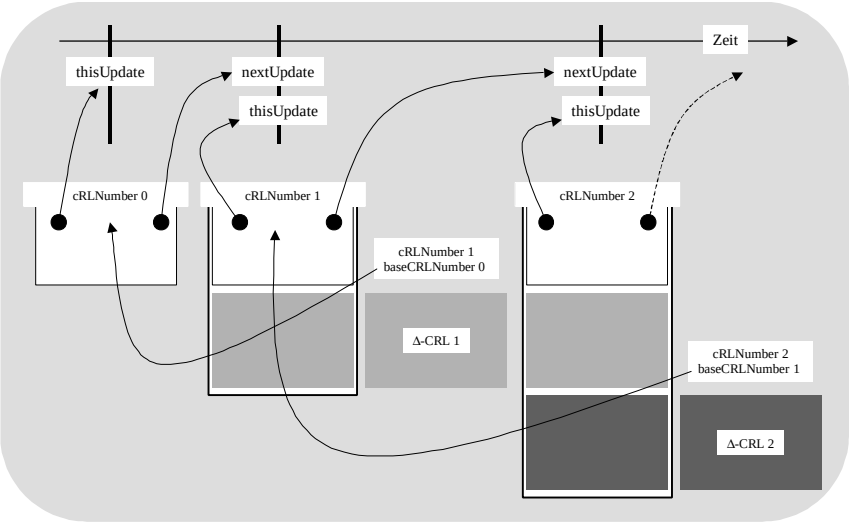
Der delta-CRL-Mechanismus ist in der Abbildung 1 veranschaulicht.

ASN.1-Definitionen

```
Extension               ::=  SEQUENCE {
    extnId                       OBJECT IDENTIFIER,
    critical                     BOOLEAN DEFAULT FALSE,
    extnValue                    OCTET STRING }

deltaCRLIndicator EXTENSION ::= {
    SYNTAX                       BaseCRLNumber
    IDENTIFIED BY                id-ce-deltaCRLIndicator }
id-ce-deltaCRLIndicator OBJECT IDENTIFIER ::=
    { 2 5 29 27 }
BaseCRLNumber           ::=  CRLNumber
CRLNumber               ::=  INTEGER (0..MAX)
```

Abbildung 1: *deltaCRL*-Mechanismus



Allgemeine Konformitätsanforderungen

Zertifizierungsstellen dürfen selbst darüber entscheiden, ob sie *delta-CRLs* erstellen oder nicht. Wenn eine *delta-CRL* ausgestellt wird, so muß die Zertifizierungsstelle auch eine vollständige CRL erstellen. Der Wert der CRL-Folgenummern *cRLNumber* (siehe Abschnitt 3.1.3.7.3) in CRLs und *delta-CRLs* muß identisch sein.

CRL-Benutzer, die eine lokal gespeicherte CRL mit Hilfe von *delta-CRLs* aktualisieren, sollen eine konstruierte CRL als unvollständig betrachten, falls sie eine *delta-CRL* empfangen, deren Folgenummer um mehr als 1 größer ist als die der zuletzt empfangenen *delta-CRL*.

SigI-Konformitätsanforderungen

Die Benutzung der *deltaCRLIndicator*-CRL-Erweiterung bei der Erstellung von Sperrlisten ist optional. SigI-konforme Systeme und Anwendungen müssen diese Erweiterung erkennen und verarbeiten können, sofern sie den deltaCRL-Mechanismus benutzen.

Tabelle 25: Implementations-technische Informationen über *deltaCRLIndicator*

BEZEICHNER	WERTEBEREICH EINZELWERTE	HEXADEZIMALCODE	LÄNGE	RELE-VANZ			KLASSIFI-KATION			
	(BEISPIELE)	(BEISPIELE)	[BYTES] 29	obligatorisch	verboten	optional	Standarderweiterung	private Erweiterung	critical-Markierung	non-critical-Markierung
deltaCRLIndicator extnId critical cRLNumber	SEQUENCE { { 2 5 29 27 }, TRUE, OCTET STRING 1 }	**30** 0D **06** 03 *55 1D 1B* **01** 01 *FF* **04** 03 **02** 01 *01*	15			v	v		v	

3.2 Verwaltung und Bereitstellung von Sperrlisten

Zertifizierungsstellen müssen Sperrlisten on-line zur Verfügung stellen und diese in regelmäßigen Abständen aktualisieren. Endanwender können diese Sperrlisten entweder lokal in ihrer Umgebung speichern und periodisch aktualisieren oder bei jeder Verifikation on-line auf die aktuelle Sperrliste zugreifen.

Entscheidet sich ein Anwender, Sperrlisten lokal zu speichern, so ist er selber dafür verantwortlich, stets im Besitz der für ihn hinreichend aktuellen Sperrlisten zu sein. In diesem Fall sollten lokal Sperrlisten für alle oder jedenfalls einige der Zertifizierungsstellen gespeichert werden. Auf diese Weise ist es möglich, auch bei Rechnern, die nicht permanent mit dem öffentlichen Netz verbunden sind, Signaturen mit hinreichend hoher Sicherheit zu verifizieren.

Um bei der Verifikation einer digitalen Signatur eine aktuelle Sperrliste abrufen zu können, gibt es folgende Varianten, die derzeit in der PKIX Arbeitsgruppe für die Internet “X.509 Public Key Infrastructure” vorgeschlagen sind:

- CDP (Certificate Distribution Point)
- OpenCDP (Open CRL Distribution Process)
- OCSP (Online Certificate Status Protocol) unter Verwendung von CRLs

Alle drei Varianten basieren auf CRLs, d.h. sie geben nur Informationen über zurückgezogene Zertifikate. Damit können sie nicht den nach dem Signaturgesetz geforderten Verzeichnisdienst ersetzten, der über alle ausgestellten Zertifikate Auskunft geben können muß.

Mit der Verwendung von Sperrlisten geht der Anwender stets das Risiko ein, daß bei einer Verifikation eine Unterschrift irrtümmlich als gültig anerkannt wird, sofern der Sperrzeitpunkt innerhalb des Gütligkeitszeitraumes der Sperrliste liegt. Diese Unsicherheit ist durch die Anwenderinfrastruktur dem Benutzer entsprechend kenntlich zu machen. Geeignete Maßnahmen sind in diesem Zusammenhang die Unterrichtung des Teilnehmers, sowie die Fixierung des Problems in den zugehörigen Sicherheitsrichtlinien.

3.2.1 CDP (CERTIFICATE DISTRIBUTION POINT)

Der Certificate Distribution Point (CDP) wurde schon in [A1 98, Abschnitt 2.3.9.9] erläutert. Hierbei handelt es sich um die X.509v3 Zertifikatserweiterung *cRLDistributionPoints*, die zur Beschaffung von Sperrlisten dient. Im Zertifikat ist bei dieser Variante direkt der Verweis auf die zugehörige Sperrliste enthalten, in der das Zertifikat eingetragen wird, wenn es zurückgezogen werden sollte. Diese Lösung ist insofern inflexibel, als die Information statisch in das Zertifikat eingebunden ist und somit das Zertifikat zurückgezogen werden muß, wenn sich die Adresse ändert, von der die Sperrliste verfügbar ist.

Gedacht war diese Zertifikatserweiterung außerdem für eine Segmentierung von CRLs, jedoch beinhaltet CDP kein Segmentierkriterium und ist somit zur Segmentierung ungeeignet.

An dieser Stelle sei darauf hingewiesen, daß zu dem Thema “cRLDistributionPoints” das US-Patent 5,699,431 von Entrust Technologies Inc. existiert, das aber weltweit und gebührenfrei benutzt werden darf.

ASN.1-Definitionen

```
Extension               ::=   SEQUENCE {
    extnId                    OBJECT IDENTIFIER,
    critical                  BOOLEAN DEFAULT FALSE,
    extnValue                 OCTET STRING }

cRLDistributionPoints EXTENSION  ::= {
    SYNTAX                    CRLDistPointsSyntax
    IDENTIFIED BY             id-ce-cRLDistributionPoints }

id-ce-cRLDistributionPoints OBJECT IDENTIFIER ::= { 2 5 29 31}

CRLDistPointsSyntax     ::=  SEQUENCE SIZE (1..MAX) OF
                             DistributionPoint

DistributionPoint       ::=  SEQUENCE {
    distributionPoint         [0]   DistributionPointName OPTIONAL,
    reasons                   [1]   ReasonFlags OPTIONAL,
    cRLIssuer                 [2]   GeneralNames OPTIONAL }

DistributionPointName ::=    CHOICE {
    fullName                  [0]   GeneralNames,
    nameRelativeToCRLIssuer   [1]   RelativeDistinguishedName }

ReasonFlags             ::=  BIT STRING {
    unused                    (0),
    keyCompromise             (1),
    cACompromise              (2),
    affiliationChanged        (3),
    superseded                (4),
    cessationOfOperation      (5),
    certificateHold           (6) }
```

Allgemeine Konformitätsanforderungen

Falls der *DistributionPointName*-Name einer *CRL*-Verteilungsstelle im *URI*-Format angegeben wird, so ist die *URI* als ein Pointer auf die aktuelle Sperrliste anzusehen, deren zugehörigen Sperrgründe durch das Feld *reasons* und deren Ersteller durch das Feld *cRLIssuer* gekennzeichnet werden können. Die Werte im *URI*-Format (http, ldap, ftp) unterliegen denselben Einschränkungen wie für *subjectAltName*-Erweiterungen. Falls das optionale *reasons*-Teilfeld in der Erweiterung nicht verwendet wird, so soll die Sperrliste gesperrte Zertifikate für alle Sperrgründe enthalten. Falls das optionale *cRLIssuer*-Teilfeld nicht benutzt wird, so soll die Sperrliste von derjenigen Zertifizierungsstelle erstellt werden, die das Zertifikat erzeugt hat.

SigI-Konformitätsanforderungen

Die optionale *cRLDistribututionPoints*-Erweiterung muß als *non-critical* markiert werden, so daß statt der Benutzung von Sperrlisten auch andere Mechanismen wie z.B. On-line Prüfdienste zur Verifikation herangezogen werden können. Diese Methode zur Beschaffung von Sperrlisten kann durch Zertifizierungsstellen und Anwendungen unterstützt werden. Die Benutzung des *reasons*-Teilfeld ist bei der Generierung von Zertifikaten verboten. Eine Segmen-

tierung der Sperrliste über die *cRLDistribututionPoints*-Erweiterung ist verboten, sofern die Sperrliste nicht selbst das Segmentierkriterium enthält (siehe Abschnitt 3.2.2.1). Verwendet eine Zertifizierungsstelle mehrere verschiedene URIs, so müssen diese alle auf die gleiche Information zeigen. Wenn der CRL-Herausgeber nicht die Zertifizierungsstelle ist, die dieses Zertifikat ausgestellt hat, so muß der Name des CRL-Herausgebers im *cRLIssuer*-Teilfeld angegeben werden.

3.2.2 OPENCDP (OPEN CRL DISTRIBUTION PROCESS)

Eine Alternative zu CDP bietet der Open CRL Distribution Process (OpenCDP), der eine Segmentierung von CRLs nach verschiedenen Kriterien ermöglicht. Die Aufteilung einer CRL in mehrere Teil-CRLs erfordert zwei Funktionen:

- eine Funktion zur Lokalisierung der entsprechenden CRL
- eine Funktion, die die Verbindung zwischen Zertifikat und Teil-CRL herstellt

Zur Lokalisierung der entsprechenden CRL wurde in OpenCDP das X.500 Attribut *Revocation Information Attribute* eingeführt. Es beinhaltet keine sicherheits-kritischen Informationen und muß deshalb nicht im Zertifikat integriert und von der Zertifizierungsstelle signiert werden. Darüber hinaus bietet das *Revocation Information Attribute* die Möglichkeit, weitere Informationen zum Status eines Zertifikats abzulegen, wie beispielsweise die Adresse eine OCSP Dienstes.

Eine weitere Möglichkeit, CRLs zu lokalisieren, bietet das *CRL List Attribute*. Dieses X.500 Attribut einer Zertifizierungsstelle beinhaltet eine Liste aller (Teil)-CRLs.

Die Verbindung zwischen Zertifikat und CRL wird im Gegensatz zu CDP nicht statisch, sondern dynamisch hergestellt, indem eine Zertifizierungsstelle ein Segmentierkriterium festlegt und als X.509v2 CRL-Erweiterung *cRLScope* in die CRLs einträgt. Dieses Kriterium kann sie aber auch jederzeit wieder ändern ohne daß das Einfluß auf bereits ausgestellte Zertifikate oder CRLs hat.

Darüber hinaus definiert OpenCDP den *Revocation Issuer* als eine weitere X.509v3 Zertifikatserweiterung, mit der eine Zertifizierungsstelle im Fall von *indirectCRLs* anzeigen kann, daß sie das Ausstellen und Pflegen von Sperrlisten oder die Aufgabe des Auskunftdienstes (OCSP) an einen oder mehrere Dritte delegiert hat.

3.2.2.1 CRL-Erweiterung cRLScope

Zweck

Die optionale X.509v2 CRL-Erweiterung *cRLScope* ermöglicht eine Segmentierung von CRLs, da hiermit jede CRL das Segmentierungs-Kriterium beinhaltet. Mögliche Segmentierkriterien sind beispielsweise die Seriennummern oder der Gültigkeitszeitraum, mit denen eine CRL in mehrere Teil-CRLs zerlegt werden kann.

ASN.1-Definitionen

```
Extension           ::= SEQUENCE {
    extnId                  OBJECT IDENTIFIER,
    critical                BOOLEAN DEFAULT FALSE,
    extnValue               OCTET STRING }

cRLScope            EXTENSION ::=  {
    SYNTAX                  CRLScopeSyntax
    IDENTIFIED BY           { <oid tbd> } }


CRLScopeSyntax      ::= SEQUENCE   {
    serialNumberRange       [0]  NumberRange OPTIONAL,
    subjectKeyIdRange       [1]  NumberRange OPTIONAL,
    nameSubtrees            [2]  GeneralNames OPTIONAL,
    notBeforeRange          [3]  NotBeforeRange OPTIONAL,
    onlyContainsUserCerts   [4]  BOOLEAN DEFAULT FALSE,
    onlyContainsCACerts     [5]  BOOLEAN DEFAULT FALSE,
    onlySomeReasons         [6]  ReasonFlags OPTIONAL,
    indirectCRL             [7]  BOOLEAN DEFAULT FALSE }

NumberRange         ::= SEQUENCE   {
    startingNumber          INTEGER,
    endingNumber            INTEGER,
    modulus                 INTEGER OPTIONAL }

NotBeforeRange      ::= SEQUENCE   {
    startingNotBeforeTime   GeneralizedTime,
    endingNotBeforeTime     GeneralizedTime }
```

Allgemeine Konformitätsanforderungen:

Wenn das Feld *serialNumberRange* oder *subjectKeyIdRange* verwendet wird und ein Modulus angegeben ist, so muß die Seriennummer bzw. der Identifier zuerst modulo dieses Wertes *modulus* genommen werden, bevor überprüft wird, ob die Zahl im Intervall zwischen *startingNumber* (inklusive) und *endingNumber* liegt.

Für das Feld *nameSubtrees* gelten die gleichen Konventionen wie für die X.509v3 Zertifikatserweiterung *NameConstraints* in [A1 98, 2.3.9.13]

Wird das Feld *notBeforeRange* als Segmentierkriterium verwendet, so gehören alle Zertifikate zu dieser CRL, deren Gültigkeitsbeginn ab *startingNotBeforeTime* (inklusive) und deren Gültigkeitsende vor *endingNotBeforeTime* liegt.

Für die Felder *onlyContainsUserCerts*, *onlyContainsCACerts*, *onlySomeReasons* und *indirectCRL* gelten die gleichen Konventionen wie bei der X.509v2 CRL Erweiterung *issuingDistributionPoint* (siehe Abschnitt 3.1.3.7.4).

Die CRL Erweiterung *cRLScope* steht im Widerspruch zur X.509v2 CRL Erweiterung *issuingDistributionPoint* und darf nicht gleichzeitig mit ihr verwendet werden.

SigI-Konformitätsanforderungen:

Die Benutzung der *cRLScope*-CRL-Erweiterung bei der Erstellung von Sperrlisten ist optional und als *critical* zu markieren. SigI-konforme Systeme und Anwendungen müssen

diese Erweiterung erkennen und verarbeiten können, sofern sie den OpenCDP-Mechanismus benutzen.

3.2.2.2 X.500-Attribut revocation information attribute

Um die Inflexibilität von CDP zu vermeiden, ist bei OpenCDP der Verweis auf die entsprechende CRL nicht im Zertifikat integriert, sondern OpenCDP definiert ein *Revocation Information Attribute,* das den Verweis auf die CRL und weitere Informationen zum Status eines Zertifikats enthält. Dieses Attribut *revocInfo* kann entweder in einem X.500 Verzeichnis abgelegt werden oder zusätzlich zu dem Zertifikat mitgeschickt werden, z.B. bietet [PKCS7 93] die Möglichkeit den Daten noch weitere Attribute hinzuzufügen, die nicht signiert werden.

ASN.1-Definitionen

```
revocInfo           ATTRIBUTE ::=  {
    WITH SYNTAX             RevocInfo
    ID                      <oid tbd> }

RevocInfo           ::= SEQUENCE {
    certIssuer              GeneralNames,
    certSerialNumber        INTEGER,
    infoLocations           [0]  SEQUENCE SIZE (1..MAX) OF InfoLocation
                                 OPTIONAL,
    extensions              [1]  SEQUENCE OF INSTANCE OF
                                 TYPE-IDENTIFIER OPTIONAL }
InfoLocation        ::= SEQUENCE   {
    locator                 GeneralNames,
    infoType                InfoType DEFAULT crl,
    reasons                 ReasonFlags OPTIONAL }
InfoType            ::= ENUMERATED {
    crl                     (0),
    oCSPServer              (1) }
ReasonFlags         ::= BIT STRING {
    unused                  (0),
    keyCompromise           (1),
    cACompromise            (2),
    affiliationChanged      (3),
    superseded              (4),
    cessationOfOperation    (5),
    certificateHold         (6) }
```

SigI-Konformitätsanforderungen

Die Bedeutung der einzelnen Sperrgründe ist in der Tabelle 15 angegeben. Zertifizierungsstellen, die zuOpenCDP konform sind, dürfen das *revocation information* Attribut nicht in Zertifikate einbauen. Die Teilkomponente *locator* der *InfoLocation*-Struktur enthält für CRLs entweder den zugehörigen Verzeichnisdienstnamen oder die zugehörige URL und für den OCSP-Dienst den zugehörigen DNS Namen. Zur Zeit werden keine Einschränkungen hinsichtlich der einzelnen Teilkomponenten von *revocInfo* gemacht.

3.2.2.3 X.500-Attribut CRL list attribute

Außerdem definiert OpenCDP mit *CRL List Attribute* ein Attribut für X.500 konforme Directory Services, welches in der Objektklasse *certificationAuthority* verwendet werden kann. Es enthält eine signierte Liste von CRLs, wobei zu jeder CRL der Zeitpunkt des letzten Updates angeben ist. Dieses Attribut kann ebenfalls im X.500-Verzeichniseintrag der Zertifizierungsstelle gehalten werden, um eine Liste aller ausgestellten Teil-CRLs zur Verfügung zu stellen. Der Vorteil dieses Attributs *cRLList* ist, daß nur diese Liste häufig neu herausgegeben werden muß, nicht aber alle Teil-CRLs, sofern sie sich nicht geändert haben. Der Anwender muß nur diese Liste überprüfen, ob seine lokal gespeicherten CRLs noch aktuell sind, anstatt alle CRLs laden zu müssen. Darüberhinaus kann der Anwender über diesen Mechanismus erkennen, wenn vor dem Ablaufdatum einer CRL eine neue CRL herausgegeben wurde.

ASN.1-Definitionen

```
CRLList             ATTRIBUTE ::=  {
   WITH SYNTAX           CRLList
   ID                    <oid tbd> }

CRLList             ::= SIGNED { SEQUENCE {
   signature             AlgorithmIdentifier,
   issuer                GeneralNames,
   thisUpdate            GeneralizedTime,
   nextUpdate            GeneralizedTime  OPTIONAL,
   cRLLocators           CRLLocators } }
CRLLocators         ::= SEQUENCE SIZE (1..MAX) OF SEQUENCE {
   locator               GeneralName,
   cRLScope              CRLScopeSyntax,
   lastUpdate            GeneralizedTime OPTIONAL }
```

SigI-Konformitätsanforderungen

Die Teilkomponente *locator* der *CRLLocators*-Struktur enthält für CRLs entweder den zugehörigen Verzeichnisdienstnamen oder die zugehörige URL und für den OCSP-Dienst den zugehörigen DNS Namen. Zur Zeit werden keine Einschränkungen hinsichtlich der einzelnen Teilkomponenten von *CRLList* gemacht.

3.2.2.4 Erweiterung revocation issuer

Wenn *indirectCRLs* verwendet werden, kann eine Zertifizierungsstelle über die X.509v3-Zertifikatserweiterung *revocation issuer* in von ihr ausgestellten Zertifikat kenntlich machen, daß sie das Ausstellen und Pflegen von Sperrlisten oder die Aufgabe des Auskunftdienstes (OCSP) an einen oder mehrere Dritte übertragen hat. Diese Zertifikatserweiterung ersetzt das Feld *cRLIssuer* der *cRLDistributionPoint*-Erweiterung, die bei Verwendung von OpenCDP nicht verwendet werden darf.

ASN.1-Definitionen

```
RevocationIssuer  EXTENSION ::=  {
   SYNTAX                GeneralNames
   IDENTIFIED BY         { <oid tbd> } }
```

Allgemeine Konformitätsanforderungen

OpenCDP konforme Zertifizierungsstellen dürfen keine Zertifikate ausstellen, die die Sperrlisteninformationen *cRLDistributionPoint* beeinhalten.

SigI-Konformitätsanforderungen

Zertifizierungsstellen können OpenCDP unterstützen. Hierfür müssen sie zumindest das *Revocation Information Attribute* erzeugen und zur Vefügung stellen. Darüber hinaus kann eine Segmentierung der Sperrliste vorgenommen werden. Für diesem Fall müssen Zertifizierungsstellen die X.509v2 CRL-Erweiterung *cRLScope* in ihren Sperrlisten verwenden.

Eine Zertifizierungstelle darf entweder nur CDP oder nur OpenCDP unterstützen, auf gar keinen Fall dürfen beide Mechanismen parallel verwendet werden.

3.2.3 OCSP (ONLINE CERTIFICATE STATUS PROTOCOL) AUF DER BASIS VON CRLS

Das Online Certificate Status Protocol (OCSP) ermöglicht aktuellere Sperrinformationen als dies mit periodisch herausgegebenen Sperrlisten möglich ist. Dieses Protokoll basiert von der Grundidee her nicht auf CRLs, kann aber auch CRLs als Informationsquelle verwenden, d.h. die OCSP-Antworten werden auf der Basis von CRLs gegeben. Sind die Informationen nicht auf Auskünfte aus Sperrlisten beschränkt, sondern basieren auf einer Positivliste aller jemals ausgestellten Zertifikate, so erfüllt OCSP die Anforderungen eines Verzeichnisdiensts gemäß des Signaturgesetzes. Dieser Fall wurde in Kapitel 2 ausführlich behandelt.

Das Protokoll OCSP kann auch auf der Basis von CRLs arbeiten. Damit geht zwar der Vorteil der Aktualität der Auskünfte verloren, aber es bietet für Anwenderinfrastrukturen eine zusätzliche Möglichkeit zur Verwaltung und Bereitstellung von Sperrlisten. Es können mittels OCSP allerdings keine CRLs an den Anwender zurückgeliefert werden.

Die Verifikation einer Signatur verläuft on-line, d.h. es wird nicht gegen eine lokale CRL geprüft, sondern es wird ein Verzeichnisdienst befragt, der auf der Basis einer ihm vorliegenden CRL Auskünfte erteilt. Mögliche Antworten sind in diesem Fall entweder *notRevoked*, *revoked* oder *unknown.*

Dieser OCSP Dienst muß nicht von der Zertifizierungsstelle betrieben werden, sondern kann von anderen Organisationen übernommen werden, die dafür verantwortlich sind, sich stets die aktuelle Sperrliste von der Zertifizierungsstelle zu laden.

Beispielsweise könnte ein Kaufhaus diese Variante des Sperrlisten-Managements nutzen, die eine Prüfung über den signaturgesetz-konformen Verzeichnisdienst nur ab einer bestimmten Höhe des Rechnungsbetrags durchführen will. In den übrigen Fällen genügt es ihnen, ihren internen auf CRLs basierenden OCSP Dienst zu befragen, der die Sperrlisteninformationen in regelmäßigen Abständen oder unregelmäßig beim Auftreten von Sperrereignissen von der Zertifizierungsstelle erhält.

Die Verwendung eines solchen internen OCSP-Dienstes bei der Verifikation von Signaturen ist nicht gesetzeskonform, die erzielten Verifikationsergebnisse können von dem Ergebnis abweichen, welches eine gesetzteskonforme Verifikation liefert. Die Einschätzung des Risikos sowie die Abwägung dieses Risikos gegen die niedrigeren Kommunikationskosten und Wartezeiten obliegt dem Anwender.

3.2.4 ABFRAGE VON SPERRLISTEN

Für den Abruf von Sperrlisten werden keine gesonderten Protokolle definiert. Für diesen Zweck sei auf die Dokumente der IETF PKIX Working Group verwiesen [PKIX OP-LDAP 98, PKIX OP-FTPHTTP 98], die diese Funktionen beschreiben. In diesen Dokumenten werden die Protokolle definiert, über die Sperrlisten von Servern abgerufen werden können. Diese Dokumente adressieren das Problem des Abrufens von Zertifikaten und Sperrlisten.

Als Protokolle werden LDAPv2 sowie FTP und HTTP definiert. Nicht definiert wird, wie die Anwenderinfrastruktur die Adresse des für eine Zertifizierungsstelle zuständigen Servers erhält. Üblicherweise wird die Anwenderinfrastruktur den technischen Namen der Zertifizierungsstelle verwenden, um im globalen X.500 Directory den entsprechenden Eintrag abzurufen. Hier können dann Attribute abgegelegt werden, die eine oder mehrere URLs für den Abruf beschreiben (siehe Abschnitt 2.3.2). Möglich ist auch, daß im alternativen Namen der Zertifizierungsstelle eine Internet-Adresse direkt angegeben wird. In diesem Fall ist kein globales X.500 notwendig.

Beispiele für URLs aus [PKIX OP-FTPHTTP]:

```
ftp://ftp.netcom.com/sp/spyrus/housley.cer
ftp://ftp.your.org/pki/id48.cer
ftp://ftp.your.org/pki/id48.no42.crl
http://www.netcom.com/sp/spyrus/housley.cer
http://www.your.org/pki/id48.cer
http://www.your.org/pki/id48.no42.crl
```

In [PKIX OP-LDAP 98] werden die minimalen Protokollelement beschrieben, die ein LDAP-Server zum Abruf von Daten im Rahmen der PKIX unterstützen muß. Daneben wird ebenfalls definiert, welche Protokollelemente ein LDAP-Server unterstützen muß, wenn die in ihm gespeicherten Daten verändert werden sollen. Die in diesen Dokumenten spezifizierten Protokolle decken diesen Themenbereich hinreichend ab.

ANHANG ? OBJEKTBEZEICHNER

Die folgende Tabelle enthält eine Übersicht über alle Objektbezeichner, die in diesem Dokument benutzt wurden.

Tabelle 26: Objektbezeichner

OBJEKTBEZEICHNERNUMMERN										OBJEKTBEZEICHNERNAMEN	REFERENZ
1										**iso**	
	2									member-body	
		840								data country code, USA	
			10040							X9-57	
				2						holdInstruction	
					1					none	3.2.3.6.1
					2					callissuer	3.2.3.6.1
					3					reject	3.2.3.6.1
	3									identified-organization	
		6								dod	
			1							internet	
				5						security	
					5					mechanisms	
						7				pkix	
							48			id-ad, access description	
								1		ocsp	
									1	ocsp-basic	3.2.2
								2		caIssuers	
		36								teletrust	
			8							id-sigi	
				3						id-sigi-at	
					9					id-sigi-at-retrieveIfAllowed	2.2.1
					10					id-sigi-at-requestedCertificate	2.2.2
					12					id-sigi-at-CertInDirSince	2.2.2
					13					id-sigi-at-certHash	2.2.1
				4						id-sigi-on	
					1					id-sigi-on-personalData	3.2.3.7.2

Fortsetzung von Tabelle 26

OBJEKTBEZEICHNERNUMMERN										OBJEKTBEZEICHNERNAMEN	REFERENZ
2										**joint-iso-ccitt**	
	5									ds	
		4								attributeType	
			3							commonName	3.2.3.3
			4							surName	3.2.3.3
			5							serialNumber	3.2.3.3
			6							countryName	3.2.3.3
			7							localityName	3.2.3.3
			8							stateOrProvinceName	3.2.3.3
			10							organizationName	3.2.3.3
			11							organizationalUnit	3.2.3.3
			12							title	3.2.3.3
			15							businessCategory	3.2.3.3
			17							postalCode	2.2.3.3
			47							givenName	3.2.3.3
		29								id-ce, certificate extensions	
			18							issuerAltName	3.2.3.7.2
			20							cRLNumber	3.2.3.7.3
			21							cRLReason	3.2.3.6.1
			23							holdInstructionCode	3.2.3.6.1
			24							invalidityDate	3.2.3.6.1
			27							deltaCRLIndicator	3.2.3.7.5
			28							issuingDistributionPoint	3.2.3.7.4
			29							certificateIssuer	3.2.3.6.1
			31							cRLDistributionPoints	3.3.1
			35							authorityKeyIdentifier	3.2.3.7.1

ANHANG ?? ASN.1 DEFINITIONEN

Dieser Abschnitt enthält eine Zusammenfassung aller ASN.1-Definitionen in alphabetischer Reihenfolge, die in diesem Dokument benutzt werden.

```
Algorithmidentifier   ::=   SEQUENCE {
    Algorithm                OBJECT IDENTIFIER,
    parameters               ANY DEFINED BY algorithm OPTIONAL }

AttributeType         ::=   OBJECT IDENTIFIER

AttributeTypeAndValue ::=   SEQUENCE {
    type                     AttributeType,
    value                    AttributeValue }

AttributeValue        ::=   ANY DEFINED BY AttributeType

authorityKeyIdentifier EXTENSION ::= {
    SYNTAX                   AuthorityKeyIdentifier
    IDENTIFIED BY            id-ce-authorityKeyIdentifier }

AuthorityKeyIdentifier ::=  SEQUENCE {
    keyIdentifier            [0] KeyIdentifier OPTIONAL,
    authorityCertIssuer      [1] GeneralNames  OPTIONAL,
    authorityCertSerialNumber    [2] CertificateSerialNumber OPTIONAL }

BaseCRLNumber          ::=  CRLNumber

BasicOCSPResponse      ::=  SEQUENCE {
    tbsResponseData          ResponseData,
    signatureAlgorithm       AlgorithmIdentifier,
    signature                BIT STRING,
    certs                [1]  EXPLICIT SEQUENCE OF Certificate OPTIONAL}

certHash EXTENSION     ::=   {
    SYNTAX                   CertHashSyntax
    IDENTIFIED BY            id-sigi-at-certHash }

CertHashSyntax         ::=   SEQUENCE {
    hashAlgorithm            AlgorithmIdentifier,
    certificateHash          OCTET STRING }

CertID                 ::=   SEQUENCE {
    hashAlgorithm            AlgorithmIdentifier,
    issuerNameHash           OCTET STRING,
    issuerKeyHash            OCTET STRING,
    serialNumber             CertificateSerialNumber }

certificateIssuer EXTENSION ::= {
    SYNTAX                   CertificateIssuer
    IDENTIFIED BY            id-ce-certificateIssuer }

CertificateIssuer      ::=   GeneralNames

CertificateList        ::=   SEQUENCE {
    tbsCertList              TBSCertList,
    signatureAlgorithm       AlgorithmIdentifier,
    signature                BIT STRING }
```

```
CertificateSerialNumber     INTEGER

CertInDirSince EXTENSION    ::= {
    SYNTAX                      CertInDirSinceSyntax
    IDENTIFIED BY               id-ce-authorityKeyIdentifier }

CertInDirSinceSyntax        GeneralizedTime

CertStatus              ::= CHOICE {
    good                        [0]   IMPLICIT NULL,
    revoked                     [1]   IMPLICIT RevokedInfo,
    unknown                     [2]   IMPLICIT UnknownInfo }

CRLDistPointsSyntax     ::= SEQUENCE SIZE(1..MAX) OF
                            DistributionPoint

cRLDistributionPoints EXTENSION  ::=  {
    SYNTAX                      CRLDistPointsSyntax
    IDENTIFIED BY               id-ce- cRLDistributionPoints }

CRLList ATTRIBUTE       ::= {
    WITH SYNTAX                 CRLList
    ID                          <oid tbd> }

CRLList                 ::= SIGNED { SEQUENCE {
    signature                   AlgorithmIdentifier,
    issuer                      GeneralNames,
    thisUpdate                  GeneralizedTime,
    nextUpdate                  GeneralizedTime  OPTIONAL,
    cRLLocators                 CRLLocators } }

CRLLocators             ::= SEQUENCE SIZE (1..MAX) OF SEQUENCE {
    locator                     GeneralName,
    cRLScope                    CRLScopeSyntax,
    lastUpdate                  GeneralizedTime OPTIONAL )

cRLNumber EXTENSION     ::= {
    SYNTAX                      CRLNumber
    IDENTIFIED BY               id-ce-cRLNumber }

CRLNumber               ::= INTEGER (0..MAX)

CRLReason EXTENSION     ::= {
    SYNTAX                      CRLReason
    IDENTIFIED BY               id-ce-cRLReason }

CRLReason               ::= ENUMERATED {
    unspecified                 (0),
    keyCompromise               (1),
    cACompromise                (2),
    affiliationChanged          (3),
    superseded                  (4),
    cessationOfOperation        (5),
    certificateHold             (6),
    removeFromCRL               (7) }

CRLScope EXTENSION      ::= {
    SYNTAX                      CRLScopeSyntax
    IDENTIFIED BY               { <oid tbd > } }
```

```
CRLScopeSyntax          ::=   SEQUENCE  {
    serialNumberRange           [0] NumberRange OPTIONAL,
    subjectKeyIdRange           [1] NumberRange OPTIONAL,
    nameSubtrees                [2] GeneralNames OPTIONAL,
    notBeforeRange              [3] NotBeforeRange OPTIONAL,
    onlyContainsUserCerts       [4] BOOLEAN DEFAULT FALSE,
    onlyContainsCACerts         [5] BOOLEAN DEFAULT FALSE,
    onlySomeReasons             [6] ReasonFlags OPTIONAL,
    indirectCRL                 [7] BOOLEAN DEFAULT FALSE }

deltaCRLIndicator EXTENSION ::= {
    SYNTAX                      BaseCRLNumber
    IDENTIFIED BY               id-ce-deltaCRLIndicator }

DigestInfo              ::=   SEQUENCE {
    digestAlgorithm             AlgorithmIdentifier,
    digest                      OCTET STRING }

DirectoryString         ::=   CHOICE {
    printableString             PrintableString (SIZE (1..maxSize))
    teletexString               TeletexString (SIZE (1..maxSize))
    bmpString                   BMPString (SIZE (1..maxSize))
    universalString             UniversalString (SIZE (1..maxSize)) }

DistributionPoint       ::=   SEQUENCE {
    distributionPoint           [0]   DistributionPointName OPTIONAL,
    reasons                     [1]   ReasonFlags OPTIONAL,
    cRLIssuer                   [2]   GeneralNames OPTIONAL }

DistributionPointName   ::=   CHOICE {
    fullName                    [0]   GeneralNames,
    nameRelativeToCRLIssuer     [1]   RelativeDistinguishedName }

EDIPartyName            ::=   SEQUENCE {
    nameAssigner                [0]   DirectoryString OPTIONAL,
    partyName                   [1]   DirectoryString }

EXTENSION               ::=   CLASS {
    &id                         OBJECT IDENTIFIER UNIQUE,
    &ExtType }
WITH SYNTAX {
    SYNTAX                      &ExtnType
    IDENTIFIED BY               &id }

Extensions              ::=   SEQUENCE (1..MAX) OF Extension

GeneralName             ::=   CHOICE {
    otherName                   [0]   OTHER-NAME,
    rfc822Name                  [1]   IA5String,
    dNSName                     [2]   IA5String,
    x400Address                 [3]   ORAddress,
    directoryName               [4]   Name,
    ediPartyName                [5]   EDIPartyName,
    uniformResourceIdentifier   [6]   IA5String,
    iPAddress                   [7]   OCTET STRING,
    registeredID                [8]   OBJECT IDENTIFIER }

GeneralNames        ::= SEQUENCE SIZE (1..MAX) OF GeneralName
```

```
HoldInstructionCode EXTENSION    ::=  {
    SYNTAX                                HoldInstructionCode
    IDENTIFIED BY                         id-ce-holdInstructionCode }

HoldInstructionCode          ::=  CHOICE {
    id-holdinstruction-none           OBJECT IDENTIFIER,
    id-holdinstruction-callissuer     OBJECT IDENTIFIER,
    id-holdinstruction-reject         OBJECT IDENTIFIER }

InfoLocation                 ::=  SEQUENCE  {
    locator                           GeneralNames,
    infoType                          InfoType DEFAULT crl,
    reasons                           ReasonFlags OPTIONAL }

InfoType                     ::=  ENUMERATED     {
    crl                               (0),
    oCSPServer                        (1) }

invalidityDate EXTENSION     ::=  {
    SYNTAX                            GeneralizedTime
    IDENTIFIED BY                     id-ce-invalidityDate }

issuerAltName EXTENSION      ::=  {
    SYNTAX                            IssuerAltName
    IDENTIFIED BY                     id-ce-issuerAltName }

IssuerAltName                ::=  GeneralNames

issuingDistributionPoint EXTENSION    ::= {
    SYNTAX                            IssuingDistributionPoint
    IDENTIFIED BY                     id-ce-issuingDistributionPoint }

IssuingDistributionPoint     ::=  SEQUENCE {
    distributionPoint           [0]   DistributionPointName OPTIONAL,
    onlyContainsUserCerts       [1]   BOOLEAN DEFAULT FALSE,
    onlyContainsCACerts         [2]   BOOLEAN DEFAULT FALSE,
    onlySomeReasons             [3]   ReasonFlags OPTIONAL,
    indirectCRL                 [4]   BOOLEAN DEFAULT FALSE }

KeyHash                      ::=  OCTET STRING

KeyIdentifier                ::=  OCTET STRING

Name                         ::=  CHOICE { RDNSequence }

NameOrPseudonym         ::=  CHOICE {
    surAndGivenName               SEQUENCE {
      surName                     DirectoryString,
      givenName                   SEQUENCE OF DirectoryString },
      pseudoNym                   DirectoryString }

NotBeforeRange          ::=  SEQUENCE {
    startingNotBeforeTime         GeneralizedTime,
    endingNotBeforeTime           GeneralizedTime }

NumberRange             ::=  SEQUENCE {
    startingNumber                INTEGER,
    endingNumber                  INTEGER,
    modulus                       INTEGER OPTIONAL }
```

```
OCSPRequest              ::=  SEQUENCE {
    tbsRequest                    TBSRequest,
    optionalSignature             [0] EXPLICIT Signature OPTIONAL }

OCSPResponse             ::=  SEQUENCE {
    responseStatus                OCSPResponseStatus,
    responseBytes                 [0] EXPLICIT ResponseBytes OPTIONAL }

OCSPResponseStatus       ::=  ENUMERATED {
    successful                    (0),
    malformedRequest              (1),
    internalError                 (2),
    tryLater                      (3),
    certRequired                  (4), -- not used
    sigRequired                   (5),
    unauthorized                  (6) }

OTHER-NAME               ::=  SEQUENCE {
    type-id                       OBJECT IDENTIFIER,
    value                         [0] EXPLICIT ANY DEFINED BY type-id }

PersonalData       ::= SEQUENCE {
    nameOrPseudonym       NameOrPseudonym,
    nameDistinguisher     [0]  INTEGER             OPTIONAL,
    dateOfBirth           [1]  GeneralizedTime     OPTIONAL,
    placeOfBirth          [2]  DirectoryString     OPTIONAL,
    gender                [3]  PrintableString     OPTIONAL,
    postalAddress         [4]  DirectoryString     OPTIONAL }

RDNSequence        ::= SEQUENCE OF RelativeDistinguishedName

ReasonFlags              ::=  Bit String {
    unused                        (0),
    keyCompromise                 (1),
    cACompromise                  (2),
    affiliationChanged            (3),
    superseded                    (4),
    cessationOfOperation          (5),
    certificateHold               (6) }

RelativeDistinguishedName  ::=  SET OF AttributeTypeAndValue

Request                  ::=  SEQUENCE {
    reqCert                       CertID,
    singleRequestExtensions       [0] EXPLICIT Extensions OPTIONAL}

requestedCertificate EXTENSION   ::=  {
    SYNTAX                        RequestedCertificateSyntax
    IDENTIFIED BY                 id-sigi-at-requestedCertificate }

RequestedCertificateSyntax  ::=  Certificate

ResponderID              ::=  CHOICE {
    byName                        [0] Name,
    byKey                         [1] KeyHash }
```

```
ResponseBytes          ::=   SEQUENCE {
    responseType               OBJECT IDENTIFIER,
    response                   OCTET STRING }

ResponseData           ::=   SEQUENCE {
    version                    [0] EXPLICIT Version DEFAULT v1,
    responderID                ResponderID,
    producedAt                 GeneralizedTime,
    responses                  SEQUENCE OF SingleResponse,
    responseExtensions         [1] EXPLICIT Extensions OPTIONAL }

retrieveIfAllowed EXTENSION ::=  {
    SYNTAX                     BOOLEAN DEFAULT FALSE
    IDENTIFIED BY              id-atRetrieveIfAllowed }

RevocationIssuer EXTENSION  ::=  {
    SYNTAX                     GeneralNames
    IDENTIFIED BY              <oid tbd> }

revocInfo ATTRIBUT     ::=   {
    WITH SYNTAX                RevocInfo
    ID                         <oid tbd> }

RevocInfo              ::=   SEQUENCE {
    certIssuer                 GeneralNames,
    certSerialNumber           INTEGER,
    infoLocations          [0] SEQUENCE SIZE (1..MAX) OF InfoLocation
                               OPTIONAL,
    extensions             [1] SEQUENCE OF INSTANCE OF
                               TYPE-IDENTIFIER OPTIONAL }

revokedCertificates            SEQUENCE OF SEQUENCE {
    userCertificate            CertificateSerialNumber,
    revocationDate             Time,
    crlEntryExtensions         Extensions OPTIONAL } OPTIONAL

RevokedInfo            ::=   SEQUENCE {
    revocationTime             GeneralizedTime,
    revocationReason           [0] EXPLICIT CRLReason OPTIONAL }

Signature              ::=   SEQUENCE {
    signatureAlgorith          AlgorithmIdentifier,
    signature                  BIT STRING,
    certs                      [0] EXPLICIT SEQUENCE OF Certificate
                               OPTIONAL }

SingleResponse         ::=   SEQUENCE {
    certID                     CertID,
    certStatus                 CertStatus,
    thisUpdate                 GeneralizedTime,
    nextUpdate                 [0] EXPLICIT GeneralizedTime OPTIONAL,
    singleExtensions           [1] EXPLICIT Extensions OPTIONAL }
```

```
TBSCertList             ::=   SEQUENCE {
    version                       Version OPTIONAL,
    signature                     AlgorithmIdentifier,
    issuer                        Name,
    thisUpdate                    Time,
    nextUpdate                    Time OPTIONAL,
    revokedCertificates           SEQUENCE OF SEQUENCE {
      userCertificate             CertificateSerialNumber,
      revocationDate              Time,
      crlEntryExtensions          Extensions OPTIONAL } OPTIONAL,
    crlExtensions                 [0] EXPLICIT Extensions OPTIONAL }

TBSRequest              ::=   SEQUENCE {
    version                       [0] EXPLICIT Version OPTIONAL,
    requestorName                 [1] EXPLICIT GeneralName OPTIONAL,
    requestList                   SEQUENCE OF Request,
    requestExtensions             [2] EXPLICIT Extensions OPTIONAL }

Time                    ::=   CHOICE {
    utcTime                       UTCTime,
    generalizedTime               GeneralizedTime }

UnknownInfo             ::=   NULL

CertInDirSince EXTENSION    ::=  {
    SYNTAX                        CertInDirSinceSyntax
    IDENTIFIED BY                 id-sigi-at-CertInDirSince }

VerifySyntax            ::=   GeneralizedTime

Version                 ::=   INTEGER { v1(0), v2(1) }
```

Objektbezeichner

```
id-ad OBJECT IDENTIFIER ::=                    { 1 3 6 1 5 5 7 48 }

id-ce OBJECT IDENTIFIER ::=                       { 2 5 29 }

id-ce-authorityKeyIdentifier OBJECT IDENTIFIER ::=
                                                  { 2 5 29 35}

id-ce-certificateIssuer OBJECT IDENTIFIER ::=   { 2 5 29 29 }

id-ce-cRLDistributionPoints OBJECT IDENTIFIER ::= {2 5 29 31}

id-ce-cRLNumber OBJECT IDENTIFIER ::=           { 2 5 29 20 }

id-ce-cRLReason OBJECT IDENTIFIER ::=           { 2 5 29 21 }

id-ce-deltaCRLIndicator OBJECT IDENTIFIER ::=   { 2 5 29 27 }

id-ce-holdInstructionCode OBJECT IDENTIFIER ::=
                                                { 2 5 29 23 }

id-ce-invalidityDate OBJECT IDENTIFIER ::=      { 2 5 29 24 }
```

```
id-ce-issuerAltName OBJECT IDENTIFIER ::=       { 2 5 29 18 }

id-ce-issuingDistributionPoint OBJECT IDENTIFIER ::=
                                                { 2 5 29 28 }

id-holdInstruction OBJECT IDENTIFIER ::=  {1 2 840 10042 2 }

id-holdinstruction-callissuer OBJECT IDENTIFIER ::=
                                          { 1 2 840 10042 2 2}

id-holdinstruction-none OBJECT IDENTIFIER ::=
                                          { 1 2 840 10042 2 1}

id-holdinstruction-reject OBJECT IDENTIFIER ::=
                                          { 1 2 840 10042 2 3}

id-pkix-ocsp OBJECT IDENTIFIER ::=     { 1 3 6 1 5 5 7 48 1 }

id-pkix-ocsp-basic OBJECT IDENTIFIER ::=
                                     { 1 3 6 1 5 5 7 48 1 1 }

id-sigi OBJECT IDENTIFIER ::=                    { 1 3 36 8  }

id-sigi-at OBJECT IDENTIFIER ::=                { 1 3 36 8 3 }

id-sigi-at-certHash OBJECT IDENTIFIER ::=
                                               { 1 3 36 8 3 13 }

id-sigi-at-requestedCertificate OBJECT IDENTIFIER ::=
                                               { 1 3 36 8 3 10 }

id-sigi-at-retrieveIfAllowed OBJECT IDENTIFIER ::=
                                                { 1 3 36 8 3 9 }

id-sigi-at-CertInDirSince OBJECT IDENTIFIER ::=
                                               { 1 3 36 8 3 12 }

id-sigi-on OBJECT IDENTIFIER ::=                  { 1 3 36 8 4 }

id-sigi-on-personalData OBJECT IDENTIFIER ::=
                                                { 1 3 36 8 4 1 }
```

ANHANG ?II ABKÜRZUNGEN UND BEGRIFFE

Bemerkung

Begriffe und Definitionen aus dem Bereich der Sichterheitstechnik werden international in englischer Sprache formuliert. Die folgendene Tabelle enthält deutsche Übersetzungen zu den wichtigsten englischen Fachausdrücken und eine Erläuterung von häufig benutzten Abkürzungen und Symbolen.

Tabelle 27: Abkürzungen und Begriffe

ABKÜRZUNG	ENGLISCH	DEUTSCH
*	wild card	Platzhaltersymbol für Teilstrings
||		Konkatenierung von Daten
ANS	american national standard	US-Normen
ANSI	american national standards institute	US-Normungsgremium
ASN.1	abstract syntax notation one	abstrakte Notation zur Beschreibung von Datentypen und Datenwerten
BER	basic encoding rules	Variante von ASN.1-Kodierungsvorschriften, mehrdeutig
BSI		Bundesamt für Sicherheit in der Informationstechnik
C	country	Länderbezeichnung nach ISO 3166, Attribut in *distinguished name*-Typen
CA	certification authority	Zertifizierungsstelle
CDP	certificate distribution point	Verteilungspunkt für Zertifikate
CN	common name	Personenname, Attribut in *distinguished name*-Typen
CRL	certificate revocation list	Liste zurückgezogener Zertifikate, Sperrliste
DAP	directory access protocol	Protokoll für den Zugriff auf ein X.500-Verzeichnis
DC	domain component	Teilname eines Domänennamens, Attribut in *distinguished name*-Typen
DD	day	zweistellige Tageszahl
delta-CRL	deltacertificate revocation list	Änderungen einer Sperrliste
DER	distinguished encoding rules	Variante von ASN.1-Kodierungsvorschriften
DIN		Deutsches Institut für Normung e.V.
DN	distinguished name	eindeutiger Name gemäß X.500
DNS	domain name system	Methode zur Konvertierung zwischen Namen und Adressen im Internet
DSA	digital signature algorithm	asymmetricher Verschlüsselungsalgorithmus zum Generieren digitaler Signaturen
DSI	digital signature input	Signatur-Verschlüsselungsformate

DSS	digital signature standard	von Nist entwickelter Standard für digitale Signaturen bestehend aus DSA und SHA-1
ECDSA	elliptic curve digital signature algorithm	Signaturalgorithmus basierend auf elliptischen Kurven
FTP	file transfer protocol	Filetransferprotokoll
GET		HTTP-Zugriffsmethode
GMT	Greenwich Mean Time	Greenwich-Zeit
HH	hour	zweistellige Stundenzahl
HTML	hyper text markup language	auf SGML basierende Sprache zur Beschreibung von Hyper-text-Dokumenten
HTTP	hypertext transfer protocol	Protokoll zum Laden von Dokumenten und/oder beschreiben-den Kopfinformationen des WWW
IEC	international electrotechnical commission	internationales Normungsgremium auf dem Gebiet der Elektrik und Elektronik
IETF	internet engineering task force	verantwortliches Gremium zur Entwicklung von Internet-Standards
IP	internet protocol	Übertragungsprotokoll der Netzwerkebene
IPSEC	internet protocol security	internet protocol, das Authentizität, Vertraulichkeit und Integrität gewährleistet
ISO	international organization for standardization	internationales Standardisierungsgremium
ITU	international telecommuni-cation union	Standardisierungsbehörde der UN
ITU-T	telecommunication standardization sector of ITU	Teilbereich der ITU (früher als CCITT bezeichnet), der für die Standardisierung im Telekommunikationsbereich zuständig ist.
L	locality	Angabe eine geographishen Ortes, Attribut in *distinguished name*-Typen
LDAP	lightweight directory acces protocol	Zugriffsprotokoll für Klienten auf Verzeichnisse, Alternative zu X.500
MIME	multipurpose internet mail extensions	Internet-Standardformat für erweiterte elektronische Post
MM	month, minute	zweistellige Monats- oder Minutenzahl
NIST	national institue of standards and technology	nationales US-Institut (früher als NBS, national bureau of standards bezeichnet), das für Normen und deren technische Anwendungen zuständig ist
O	organization	Bezeichnung einer Organisation, Attribut in *distinguished name*-Typen
OCSP	online certificate status protocol	Anwendungsprotokoll zur Bestimmung des aktuellen Zustan-des eines digitalen Zertifikates ohne die Benutzng von Sperr-listen
OID	object identifier	Objektbezeichner

OIW	open systems environment implementors workshop	Objektbezeichnerbaum unter dem Objektbezeichnerzweig iso(1)-identified-organization(3)-OIW(14)
OpenCDP	open CRL distribution process	Mechanismen zur Verteilung von Sperrlisten
OU	organizational unit	Bezeichnung einer untergeordneten Organisation, Atrtribut in *distinguished name*-Typen
PCA	policy certification authority	Wurzelzertifizierungsstelle
PEM	privacy enhaunced mail	Internetstandard für sichere elektronische Post
PK	public key	öffentlicher Schlüssel
PKCS	public key crypto systems public key cryptographic standard	Kryptosysteme basierend auf öffentlichen Schlüsseln RSA-Standards im Sicherheitsbereich
PKI	public key infrastructure	Sicherheitsinfrastruktur basierend auf öffentlichen Schlüsseln
PKIX	internet public key infrastructure	Internetprotokolle für die Sicherheitsinfrastruktur im Internet basierend auf öffentlichen Schlüsseln
PN	*pseudoNym*	BSI-spezifisches Attribut für Pseudonyme in *distinguished* name-Typen
POST		HTTP-Zugriffsmethode
RA	registration authority	Registrierungsstelle, an die eine Zertifizierungsstelle bestimmte Verwaltungsaufgaben delegieren kann
RCA	root CA	Wurzelzertifizierungsinstanz, zuständige Behörde
RDN	relative distinguished name	Komponente eines nach X.500 eindeutigen Namen
RegTP		Regulierungsbehörde für Telekommunikation und Post
RFC	request for comment	Internet-Report
RIPEMD-160		Von H.Dobbertin, A. Bosselaers und B. Preneel entwickelte Hashfunktion, die einen 160 Bit langen Hashwert ergibt. Der RIPEMD-160 ist eine Verbesserung des RIPEMD.
RSA	Rivest Shamir Adleman Algorithm	asymmetrischer Verschlüsselungsalgorithmus zum Verschlüsseln und Signieren von Daten US-Firma, die Kryptoprotokolle und -software entwickelt und die das Patent an dem Algorithmus besitzt
S	surname	Nachnahme einer Person, Attribut in *distinguished name*-Typen
S/MIME	secure/multipurpose internet mail extensions	Protokoll, das zusätzlich zu MIME digitale Signaturen und Verschlüsselung enthält
SECSIG	security special interest group of OIW	spezielle Arbeitsgruppe innerhalb von OIW, sie sich mit Sicherheitsfragen beschäftigt
SET	secure electronic transaction	Protokoll entwickelt von Visa und MasterCard für sichere elektronische Transaktionen
SGML	standard generalized markup language	Standard zur allgemeinen Beschreibung von Dokumenten, dient als Grundlage für HTML

SHA-1	secure hash algorithm 1	Hashfunktion, Weiterentwicklung von SHA
SigG		Gesetz zur digitalen Signatur
SigV		Verordnung zur digitalen Signatur
SN	serial number *serialNumber*	Zertifikatsfeld, das die Seriennummer des Zertifikates enthält, die innerhalb der Zertifizierungsstelle eindeutig sein muß
SP	state or province	Bezeichnung eines Bundeslandes, Attribut in *distinguished name*-Typen
SS	second	zweistellige Sekundenzahl
SSL	secure socket layer	Sicherung der Kommunikation auf der Transport/Sessionebene
ST	street address	Straße als Teil einer postalischen Adresse, Attribut in *distinguished name*-Typen
T	title	Angabe eines Titels, Attribut in *distinguished name*-Typen
TLS	transport layer security	Aufbau einer sicheren Transportverbindung über einen unsicheren Kanal
TSS	time stamp service	Zeitstempeldienst
URI	universal resource identifier	weltweit eindeutiger Bezeichner für Betriebsmittel
URL	universal resource location	weltweit eindeutiger Name für den Ort eines Betriebsmittels
UTCTime	coordinated universal time	ASN.1-Typ für Datums- und Zeitformate, Weltzeit
WWW	world wide web	Dienst zum Laden von graphischen Informationen über das Internet
YY	year	zweistellige Jahreszahl
YYYY	year	vierstellige Jahreszahl
ZS		Zertifizierungsstelle

LITERATUR

[A1 99] BSI: Signatur-Interoperabilitätsspezifikation SigI, Abschnitt A1 Zertifikate, Januar 1999

[A2 99] BSI: *Signatur-Interoperabilitätsspezifikation SigI, Abschnitt A2 Signatur*, Januar 1999

[A6 99] BSI: *Signatur-Interoperabilitätsspezifikation SigI, Abschnitt A6 Gültigkeitsmodell für digitale Signaturen*, Januar 1999

[ANS X9.30] ANSI X9.30-199x:.*Public Key Cryptography Using Irreversible Algorithms for the Financial Services Industry, Part 1: The Digital Signature Algorithm (DSA)*, 1992

[ANS X9.31] ANSI X9.31-199x: *Public Key Cryptography Using Reversible Algorithms for the Financial Services Industry, Part 1: The RSA Signature Algorithm*, 199?

[ANS X9.55] ANSI X9.55-1995: *Public Key Cryptography for the Financial Services Industry, Extensions to Public Key Certificates and Revocation Lists*, Dec. 1995

[ANS X9.62] ANSI X9.62-199x: *Public Key Cryptography for the Financial Services Industry: The Elliptic Curve Digital Signature Algorithm (ECDSA)*, 1997

[CCITT X.208 88] CCITT X.208:*Specification of Abstract Syntax Notation One (ASN.1)*, 1988

[DIN SigG/V 98] DIN NI-17.4: *Spezifikation der Schnittstelle zu Chipkarten mit Digitaler Signatur-Anwendung/Funktion nach SigG und SigV, Version 0.98*, Juli 1998

[ISO CD 15782] ISO CD 15782: *Security Management and General Banking Operations* , 1998

[ISO/IEC 14888] ISO/IEC 14888: *IT-Security Techniques - Digital Signatures With Appendix- Part 3: Certificate-Based Mechanisms* , 1997

[ISO/IEC 9796-2] ISO/IEC 9796-2: *IT-Security Techniques - Digital Signaturs Schemes Giving Message Recovery- Part 2: Mechanisms using a hash-function* , 1996

[ITU-T X.411] ITU-T X.411: *Information Technology - Message Handling Systems - Message Transfer SystemAbstract service definition and procedures*, 19??

[ITU-T X.500 97] ITU-T X.500: *Information Technology - Open Systems Interconnection -*

The Directory: Overview of concepts, models and services, 1997

[ITU-T X.501 97] ITU-T X.501: *Information Technology - Open Systems Interconnection - The Directory:Models*, 1997

[ITU-T X.509 97] ITU-T X.509: *Information Technology - Open Systems Interconnection - The Directory: Authentication Framework*, 1997

[ITU-T X.660 92] ITU-T X.660: *Information Technology - Open Systems Interconnection - Procedures for the operation of OSI Registration Authorities: General procedures*, 1992

[ITU-T X.681 94] ITU-T X.681: *Information Technology - Abstract Syntax Notation One (ASN.1): Information object specification*, 1994

[ITU-T X.690 94] ITU-T X.690: *Information Technology - ASN.1 Encoding Rules - Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)*, 1994

[MISPC 97] William Burr, Donna Dodson, Noel Nazario, W. Timothy Polk*: Minimum Interoperability Specification for PKI Components*, Version 1, June 1997

[MKAT 97] *Regulierungsbehörde für Telekommunikation und Post: Maßnahmenkatalog für digitale Signaturen, Version 1.0*, November 1997

[MTRUST 96] Fritz Bauspieß, *TelTrusT: MailTrusT Spezifikation, Version 1.1*, Dezember 1996

[OIW 95] OIW, *Stable Implementation Agreements for Open Systems Interconnection Protocols: Part 12 - OS Security*, June 1995

[PKCS1 93] RSA Laboratories, Technical Note, *PKCS #1: RSA Encryption Standard*, Version 1.5, November 1993

[PKCS7 93] RSA Laboratories, Redwood City, California*: The Public-Key Cryptography Standards (PKCS)*, November 1993

[PKIX ECDSA 97] L. Bassham, D. Johnson*: Internet Public Key Infrastructure -Representation of Elliptic Curve Digital Signature Algorithml (ECDSA) Keys and Signatures in Internet X.509 Public Key Infrastructure Certificates*, November 1997

[PKIX OCDP 98] P. Hallam-Baker, and W. Ford*: Internet Public Key Infrastructure - Open CRL Distribution Process (OpenCDP)*, April 1998

[PKIX OCSP 98] Michael Myers*: Internet Public Key Infrastructure -Online Certificate Status Protocol (OCSP)*, August 1998

[PKIX OP-FTPHTTP 98], Russ Housley, Paul Hoffman *Internet Public Key Infrastructure* - Operational Protocols - FTP and HTTP, July 1998

[PKIX OP-LDAP 98] Tim Howes, S. Boeyen, P. Richard*: Internet Public Key Infrastructure* - Operational Protocols - LDAPv2, September 1998

[PKIX PRO 98] R. Housley, W. Ford, W. Polk, and D. Solo*: Internet Public Key Infrastructure - X.509 Certificate and CRL Profile*, July 1998

[RFC 1035 87] P. Mockapetris: *Domain Names - Implementation and Specification*, 1987

[RFC 1422 93] S. Kent: *Privacy Enhancement for Internet Electronic Mail - Part II: Certificate-Based Key Management*, February 1993

[RFC 1630 94] T. Berners-Lee: *Universal Resource Identifiers in WWW*, 1994

[RFC 1959 96] T. Howes, and M. Smith: *An LDAP URL Format*, June 1996

[RFC 2052 96] A. Gulbrandsen, and M. Smith: *A DNS RR for specifying the location of services (DNS SRV)*, October 1996

[RFC 2068 97] R. Fielding, J. Gettys, J.Mogul, H. Frystyk, T. Berners-Lee: *Hypertext Transfer Protocol - HTTP/1.1*, January 1997

[RFC 2247 98] S. Kille, M. Wahl, A. Grimstad, R. Huber and S. Sataluri: *Using Domains in LDAP/X.500 Distinguished Name*, January 1998

[RFC 791 81] J. B. Postel: *Internet Protocol*, 1981

[RFC 822 82] David H. Crocker: *Standard for the Format of ARPA Internet Text Messages: Message Encryption and Authentication*, August 1982

[RIPEMD-160 96] H. Dobbertin, A. Bosselaers, and B. Preneel: *A strengthened version of RIPEMD*, April 1996

[SEC 98] GMD, SECUDE-Tool: *http://www.darmstadt.gmd.de/secude/*, 1998

[SigG 97] BRD: *Gesetz zur Regelung der Rahmenbedingungen für Informations- und Kommunikationsdienste (Informations- und Kommunikationsdienste-Gesetz - IuKDG), Artikel 3, Gesetz zur digitalen Signatur (Signaturgesetz - SigG)*, Juli 1997

[SigV 97] BRD: *Verordnung zur digitalen Signatur (Signaturverordnung - SigV*), Juli 1997

[TS ZF 98] TeleSec, Deutsche Telekom AG: *Zertifikatsformate im Zertifizierungsbereich Signaturgesetz*, Mai 1998