



# ***TELETRUST***

Verein zur Förderung der Vertrauenswürdigkeit  
von Informations- und Kommunikationstechnik

## **MailTrusT Version 2**

### **PKI-Management**

Stand: 16. März 1999

Jobst Biester, Fritz Bauspieß, Dirk Fox  
Secorvo Security Consulting GmbH



## Inhaltsübersicht

<b>1 Zusammenfassung .....</b>	<b>3</b>
<b>2 Definitionen .....</b>	<b>4</b>
<b>3 Management von Zertifikaten und Schlüsseln .....</b>	<b>5</b>
3.1 Schlüsselgenerierung und -zertifizierung .....	5
3.1.1 Varianten .....	5
3.1.2 Protokolle .....	8
3.2 Sperrung von Zertifikaten .....	9
<b>4 Datenstrukturen .....</b>	<b>11</b>
4.1 Übergeordnete Datenstrukturen .....	11
4.1.1 Der Kopf einer PKI-Nachricht .....	11
4.1.2 Der Rumpf einer PKI-Nachricht .....	14
4.1.3 Der Schutz einer PKI-Nachricht .....	14
4.1.4 Beigefügte Zertifikate in einer PKI-Nachricht .....	15
4.2 Allgemeine Datenstrukturen .....	15
4.2.1 Datenstrukturen zur Spezifikation eines Zertifikates .....	15
4.2.2 Verschlüsselte Werte .....	15
4.2.3 Status und Fehlerinformationen .....	16
4.2.4 Identifikatoren für Zertifikate .....	17
4.3 Datenstrukturen für Zertifizierungen .....	18
4.3.1 Datenstrukturen für Zertifizierungsanfragen .....	18
4.3.2 Datenstrukturen für Zertifizierungsantworten .....	21
4.3.3 Datenstrukturen für Bestätigungsnachrichten .....	23
4.4 Datenstrukturen für die Sperrung von Zertifikaten .....	23
4.4.1 Sperrantrag .....	24
4.4.2 Antwort auf Sperrantrag .....	24
4.4.3 Sperrnachricht .....	24
4.5 Datenstrukturen für die Initialisierung einer Root-CA .....	24
<b>5 Profile für PKI-Nachrichten .....</b>	<b>26</b>
5.1 Profile für Zertifizierungen .....	26
5.1.1 Profile für das dezentrale Schema .....	28
5.1.2 Profile für das zentrale Schema .....	35
5.2 Profile für Sperrungen .....	41
5.2.1 Profil für Sperrantrag .....	41
5.2.2 Profil für die Antwort auf einen Sperrantrag .....	41
<b>6 Literaturverzeichnis .....</b>	<b>43</b>

## Abkürzungen

ASN.1	Abstract Syntax Notation 1
BER	Basic Encoding Rules
BSI	Bundesamt für Sicherheit in der Informationstechnik
CA	Certification Authority (Zertifizierungsinstanz)
CRL	Certificate Revocation List (Sperrliste)
DER	Distinguished Encoding Rules
DES	Data Encryption Standard
DIR	Directory Service (Verzeichnisdienst)
DIT	Directory Information Tree
DN	(X.500-) Distinguished Name
FTP	File Transfer Protocol
IAK	Initial Authentication Key
HTTP	Hyper Text Transport Protocol
ID	Identification
IETF	Internet Engineering Task Force
ISO	International Organization for Standardization
ITU	International Telecommunication Union
LDAP	Lightweight Directory Access Protocol
MAC	Message Authentication Code
MISPC	Minimum Interoperability Specification for PKI Components
MTT	MailTrust
NIST	National Institute of Standards and Technology
OID	Object Identifier
PDU	Protocol Data Unit
PKCS	Public-Key Cryptography Standard
PKI	Public Key Infrastructure
PKIX	Public Key Infrastructure (X.509)
RA	Registration Authority (Registrierungsstelle)
RFC	Request for Comments
Sigl	Schnittstellenspezifikation zur Entwicklung interoperabler Verfahren und Komponenten nach SigG/SigV (Signatur-Interoperabilitätspezifikation)
TTT	TeleTrust
URI	Universal Resource Identifier

# 1 Zusammenfassung

Im Dokument „Gesamtkonzeption, Aufbau und Komponenten einer PKI“ wird der Aufbau einer MailTrusT-PKI beschrieben. Im vorliegenden Dokument werden die Management-Protokolle für online-Interaktionen zwischen den PKI-Komponenten beschrieben. Die Protokolle umfassen alle relevanten Aspekte der Generierung von Zertifikaten und Schlüsseln und des Managements während ihres Lebenszyklus.

Der Aufbau des vorliegende Dokuments entspricht im wesentlichen dem des PKIX-Dokumentes " Certificate Management Protocols " [PKIX-CMP 98], das ein Profil für Management-Protokolle einer Internet-PKI spezifiziert. Das MailTrusT-Profil für das PKI-Management orientiert sich auch inhaltlich an dieser Spezifikation.

Gegenstand der Spezifikation sind die folgenden Funktionen:

- **Generierung** von Schlüsseln:  
Lokale und zentrale Generierung einschließlich der Weiterleitung der erzeugten Schlüsselkomponenten (private und öffentliche Komponenten) an die jeweiligen Empfänger.
- **Zertifizierung** von Schlüsseln:  
Erst-, Verlängerungs- und Neuzertifizierung von Schlüsseln.
- **Sperrung** von Schlüsseln:  
Sperrung eines Schlüsseln aufgrund eines Sperrantrags.

Die Spezifikation enthält die folgenden Kapitel:

- Das Kapitel 2 enthält **Definitionen** von Begriffen, deren exakte Bedeutung für das Verständnis der Spezifikation wesentlich ist.
- Im Kapitel 3, „**Management von Zertifikaten und Schlüsseln**“, wird beschrieben, wie Schlüsselzertifizierungen und Sperrungen erfolgen. Es werden die verschiedenen Varianten dargestellt, für die die vorliegende Spezifikation geeignete Protokolle enthält. Anschließend wird eine Überblick über die Protokolle gegeben.
- Das Kapitel 4 enthält Definitionen der verwendeten allgemeinen „**Datenstrukturen**“ für den Austausch von Informationen zwischen den beteiligten Komponenten und die Initialisierung einer Root-CA.
- Unter Verwendung der spezifizierten Datenstrukturen werden dann in Kapitel 8 „**Profile für PKI-Nachrichten**“ dargestellt. In diesen Profilen werden die konkreten Feldinhalte angegeben. Die Summe aller dieser Profile ergibt das MailTrusT-Profil für das PKI-Management.

## 2 Definitionen

- **MailTrusT-konform**

Eine Komponente ist MailTrusT-konform, wenn sie alle Anforderungen der MailTrusT-Spezifikation erfüllt. Dabei ist zu beachten, daß die Spezifikation neben den Anforderungen auch Empfehlungen enthält, die stets als solche gekennzeichnet sind. Die Bestimmung der MailTrusT-Konformität bezieht sich nur auf die Anforderungen und nicht auf Empfehlungen.

- **PKI-Management**

Die Aufgabe des PKI-Managements besteht darin, eine funktionsfähige PKI aufzubauen und ihre Funktionsfähigkeit aufrecht zu erhalten. Zur Wahrnehmung dieser Aufgabe muß eine definierte Funktionalität der Komponenten gegeben sein. **MailTrusT-konforme** Komponenten müssen über diese Funktionalität verfügen. Die erforderliche Funktionalität reicht von die Intialisierung einer Root-CA über die Zertifizierung von CAs und Endteilnehmern bis zur Generierung von Sperrlisten.

- **PKI-Management-Protokolle**

Das **PKI-Management** benötigt Protokolle für online-Interaktionen zwischen den PKI-Komponenten. Hauptanwendungsbereich dieser Protokolle ist der Austausch von PKI-Nachrichten zwischen den Teilnehmer-Komponenten und einer CA. Die Protokolle werden jedoch auch für den Austausch von Informationen zwischen CAs verwendet, z.B. bei einer Cross-Zertifizierung.

- **Profil**

Durch die Bildung eines Profils werden Standards, die für eine Vielzahl unterschiedlicher Anwendungen entworfen wurden, auf einen konkreten Anwendungsbereich abgebildet. Standards bieten oft eine Vielzahl von Optionen, die sich nicht selten widersprechen. Für jeden Anwendungsbereich muß deshalb eine sorgfältige Auswahl der Optionen getroffen werden.

Außerdem muß allen Objekten des Standards durch das Profil eine eindeutige Semantik zugeordnet werden, sofern Mehrdeutigkeiten bestehen. Es muß auch festgelegt werden, ob die **Unterstützung** durch konforme Komponenten zwingend ist. Nur so kann der Standard interoperabel gemacht werden.

- **Unterstützung**

Die Bedeutung der Forderung nach Unterstützung eines bestimmten Objektes der Spezifikation durch eine bestimmte Komponente hängt von der Semantik ab, die mit der Verwendung dieses Objektes verbunden ist und von der Art der Komponente. Die Semantik bestimmt, was das Objekt bedeutet. Die Art der Komponente bestimmt, ob das Objekt generiert werden muß oder ob das Objekt erkannt und ausgewertet werden muß.

Die Forderung nach Unterstützung von Objekten kann sich z.B. auf Funktionen, Felder in Datenstrukturen oder Mechanismen beziehen. Sie bezieht sich in der Regel auf alle Komponenten, die am Austausch einer PKI-Nachricht beteiligt sind.

### 3 Management von Zertifikaten und Schlüsseln

Die vorliegende Spezifikation orientiert sich an etablierten und verbreiteten Standards, insbesondere den Management-Protokollen der PKIX-Arbeitsgruppe der IETF (Internet Engineering Task Force) [PKIX-CMP 98]. Die Protokolle wurden im wesentlichen unverändert in das ISO/IEC-Dokument „Specification of TTP Services to support the Application of Digital Signatures“ [ISO WD 15945 98] übernommen. Auf diesen Protokollen basiert auch die Spezifikation „Minimum Interoperability Specification for PKI Components“ [MISPC 97] des NIST (National Institute of Standards and Technology), einem nationalen US-Institut für Normen, deren technische Umsetzung und Anwendung.

In den genannten Spezifikationen werden die einzelnen Dienste der CA nicht isoliert voneinander betrachtet, sondern soweit wie möglich in einen Gesamtablauf integriert. Die Schlüsselgenerierung erfolgt z.B. stets im Rahmen einer Schlüsselzertifizierung. Es werden daher keine eigenen Abläufe für die Schlüsselgenerierung spezifiziert.

In Kapitel 4.2.5 der Version 1.1 der MailTrust-Spezifikation [MTRUST 96] wurden bereits Schnittstellen zwischen Teilnehmer und CA beschrieben. Es wurde eine Zertifizierungsanfrage des Teilnehmers und die zugehörige Antwort der CA spezifiziert, die beide auf PEM basieren [RFC 1422 93]. Aus Gründen der Kompatibilität zur Version 1.1 der Spezifikation müssen CA-Komponenten diese Schnittstellen weiterhin unterstützen.

Im folgenden wird das PKI-Management bei der Generierung und Zertifizierung von Schlüsseln und der Sperrung von Zertifikaten beschrieben.

#### 3.1 Schlüsselgenerierung und -zertifizierung

Die Zertifizierung eines Teilnehmers<sup>1</sup> ist ein Prozeß, an dem mehrere PKI-Komponenten beteiligt sind. Falls der Teilnehmer zum ersten Mal in Kontakt zur PKI tritt, beginnt der Prozeß mit der initialen Registrierung des Teilnehmers, die in der Regel bei einer Registrierungsstelle (RA) erfolgt. Falls der Teilnehmer bereits über einen zertifizierten gültigen Signaturschlüssel verfügt, kann dieser Schritt entfallen.

Nach der Registrierung wird stets eine Zertifizierungsanfrage an die Zertifizierungsstelle (CA) gerichtet. Die CA wertet die Anfrage aus, generiert u.a. ein Zertifikat und erstellt eine Zertifizierungsantwort. Der Prozeß endet damit, daß die CA das Zertifikat in einem Verzeichnis (DIR) bereitstellt.<sup>2</sup>

Am Prozeß der Schlüssel- und Zertifikatserzeugung können alle PKI-Komponenten beteiligt sein, die in dem Dokument „Gesamtkonzeption, Aufbau und Komponenten einer PKI“ beschrieben sind, d.h. CA, RA, DIR und die Teilnehmer-Komponente.

Die Beteiligung unterschiedlicher Komponenten und die unterschiedlichen Anforderungen an den Prozeß ergeben eine Vielzahl von Varianten, die im folgenden dargestellt werden. Es wird beschrieben, welche dieser Varianten in der Spezifikation berücksichtigt sind. Anschließend werden die Protokolle beschrieben, die der Spezifikation zugrunde liegen und von MailTrust-konformen Komponenten unterstützt werden müssen.

##### 3.1.1 Varianten

Produkte sollen einerseits möglichst flexibel gestaltet werden, so daß viele Varianten unterstützt werden, andererseits muß eine Beschränkung der von Produkten zu unterstützenden Varianten aus Komplexitäts- und Aufwandsgründen erfolgen.

---

<sup>1</sup> Teilnehmer kann ein Endteilnehmer, eine RA oder eine CA sein.

<sup>2</sup> Der Teilnehmer erhält das Zertifikat bereits vor der Bereitstellung im Verzeichnis.

Die zu unterstützenden Varianten wurden anhand folgender Kriterien bestimmt:

1. Modell der PKI, d.h. zentraler Modell, dezentrales Modell oder Mischform
2. Zuständige Instanz für die Generierung von Zertifizierungsanfragen, d.h. Teilnehmer, RA oder CA
3. Unterstützung unterschiedlicher Typen von Zertifizierungsanfragen
4. Verfahren zur Authentisierung von PKI-Management-Nachrichten
5. Zuständige Instanz für die Schlüsselgenerierung, d.h. Teilnehmer, RA oder CA
6. Kontrolle des Zertifizierungsprozesses durch den Teilnehmer
7. Nachweis des Besitzes des privaten Schlüssels
8. Zustimmung des Teilnehmers zur Veröffentlichung des Zertifikats durch den Verzeichnisdienst

#### **ad 1: Modell der PKI**

Die vorliegende Spezifikation ist nicht auf ein einziges Modell ausgerichtet, sondern unterstützt alle Modelle, die im Dokument „Gesamtkonzeption, Aufbau und Komponenten einer PKI“ beschrieben sind.

#### **ad 2: Zuständige Instanz für die Generierung von Zertifizierungsanfragen**

Eine Zertifizierungsanfrage kann technisch durch den Teilnehmer, die RA oder die CA erstellt werden. Dies ist für die Spezifikation unerheblich. Entscheidend ist der Absender der Zertifizierungsanfrage, da an diesen die Zertifizierungsantwort der CA gesendet wird.

Während des Prozesses der Zertifizierung findet der Austausch der in der Spezifikation definierten PKI-Management-Nachrichten ausschließlich zwischen dem Absender der Zertifizierungsanfrage und der CA statt.<sup>3</sup>

#### **ad 3: Unterstützung unterschiedlicher Typen von Zertifizierungsanfragen**

Die Spezifikation unterscheidet drei Typen von Zertifizierungsanfragen. Bei einem Erstantrag verfügt der Teilnehmer noch nicht über ein Signaturschlüssel-Zertifikat. Es muß eine Initialisierung stattfinden, bevor der Antrag gestellt werden kann.

Sobald der Teilnehmer über ein gültiges Signaturschlüssel-Zertifikat verfügt, sind Vereinfachungen möglich. Die Initialisierung muß nicht wiederholt durchgeführt werden. Deshalb werden Erstanträge anders behandelt als Folgeanträge.

Bei Folgeanträgen wird danach unterschieden, ob ein Zertifikat des Teilnehmers bei gleichbleibendem Schlüssel durch ein anderes Zertifikat ersetzt werden soll (Verlängerungsantrag) oder ob ein neuer Schlüssel für den Teilnehmer zertifiziert werden soll (Neuantrag). Bei einem Verlängerungsantrag entfällt somit die Notwendigkeit, einen neuen Schlüssel zu generieren.

#### **ad 4: Verfahren zur Authentisierung von PKI-Management-Nachrichten**

Bei einem Erstantrag verfügt der Teilnehmer noch nicht über einen zertifizierten Schlüssel, den er für Zwecke der Authentisierung verwenden könnte. Die Spezifikation bestimmt ein

---

<sup>3</sup> Der Teilnehmer kann eine Zertifizierungsanfrage direkt an die CA richten. Der Teilnehmer kann auch eine RA damit beauftragen, die Anfrage für ihn zu stellen oder die Anfrage zu prüfen. Die Prüfung der Anfrage kann z.B. dem Zweck dienen, die CA zu entlasten, da die Anzahl fehlerhafter Anträge reduziert wird.

Nicht unterstützt wird die Variante, daß ein Teilnehmer eine Zertifizierungsanfrage an die RA richtet und die RA die Inhalte dieser Anfrage dann in möglicherweise modifizierter Form für eine eigene Anfrage an die CA verwendet.

spezielles symmetrisches Verfahren zur Authentisierung des Teilnehmers gegenüber CA/RA, das für Erstanträge, Sperranträge und zum Nachweis des Besitzes eines privaten Verschlüsselungsschlüssels verwendet wird.

Jeder Teilnehmer erhält bei der initialen Registrierung, die in der Regel durch die RA erfolgt, mindestens einen symmetrischen Schlüssel, den er für Sperranträge verwenden kann. Daneben erhält er einen weiteren symmetrischen Schlüssel, falls er den Zertifizierungsantrag direkt an die CA richten will.<sup>4</sup> Falls die RA Absender des Zertifizierungsantrags ist, wird dieser Schlüssel nicht benötigt.

#### **ad 5: Zuständige Instanz für die Schlüsselgenerierung**

Die Schlüsselgenerierung kann durch CA, RA oder durch den Teilnehmer erfolgen. Im Rahmen der Spezifikation werden Zertifizierungsanträge danach unterschieden, ob eine zentrale Schlüsselgenerierung durch die CA oder eine dezentrale Schlüsselgenerierung durch den Antragsteller erfolgt.

#### **ad 6: Kontrolle des Zertifizierungsprozesses durch den Teilnehmer**

Die Spezifikation sieht vor, daß der Teilnehmer eine Kontrolle über den Zertifizierungsprozeß ausüben kann, der mit seinem Zertifizierungsantrag beginnt.

Für alle spezifizierten Abläufe ist vorgesehen, daß eine Zertifizierungsantwort der CA stets einer Bestätigungsnachricht bedarf, bevor das Zertifikat im Verzeichnis veröffentlicht werden darf. Mit dieser Nachricht gibt der Teilnehmer ein Zertifikat zur Veröffentlichung im Verzeichnis frei (falls es veröffentlicht werden soll).<sup>5</sup>

Der Teilnehmer kann die Veröffentlichung seines Zertifikats verhindern, wenn er Fehler im Zertifikat oder andere Unregelmäßigkeiten feststellt.<sup>6</sup>

#### **ad 7: Nachweis des Besitzes des privaten Schlüssels**

Der Nachweis des Besitzes des privaten Schlüssels verhindert bestimmte Angriffe, indem er es der CA ermöglicht, die Bindung zwischen Teilnehmer und Schlüsselpaar zu prüfen. Diese Prüfung hat die CA stets durchzuführen. Je nach Verwendungszweck des Schlüsselpaars sind unterschiedliche Verfahren zum Nachweis spezifiziert.<sup>7</sup>

Im Gegensatz zu den o.g. Spezifikationen muß nach der vorliegende Spezifikation die Prüfung des Besitzes stets durch die CA erfolgen.<sup>8</sup>

#### **ad 8: Zustimmung des Teilnehmers zur Veröffentlichung des Zertifikats**

Alle Zertifikate werden regelmäßig in einem Verzeichnis veröffentlicht. Der Teilnehmer kann sich jedoch auch vorbehalten, Zertifikate selbst zu verteilen. In diesem Fall kann er in der Zertifizierungsanfrage angeben, daß er keine Veröffentlichung wünscht.

---

<sup>4</sup> Dieser Schlüssel darf nur für den Erstantrag verwendet werden. Der Teilnehmer sollte deshalb den Erstantrag für ein Signaturschlüssel-Zertifikat stellen. Dieses Zertifikat kann er dann für die Authentisierung von Folgeanträgen verwenden.

<sup>5</sup> Die Bestätigungsnachricht dient daneben auch dem Nachweis des Besitzes des privaten Verschlüsselungsschlüssels, falls dieser nicht von der CA generiert worden ist.

<sup>6</sup> Im Falle der Verwendung von Chipkarten als Token wird er dies z.B. dann tun, wenn er feststellt, daß das Token bereits verwendet wurde.

<sup>7</sup> Im Falle von Signaturschlüsseln verifiziert die CA eine mit dem Schlüssel signierte Datenstruktur innerhalb des Zertifizierungsantrags. Bei Verschlüsselungsschlüsseln dient die Bestätigungsnachricht dem Nachweis des Besitzes.

<sup>8</sup> Falls die Prüfung durch die RA erfolgt, muß die Durchführung der Prüfung der CA angezeigt werden. Hierfür wäre ein eigenes Protokoll erforderlich, auf das verzichtet wurde.

### 3.1.2 Protokolle

Bevor ein Teilnehmer einen Zertifizierungsantrag stellen kann, muß er registriert werden. Während der Registrierung, die in der Regel durch eine RA erfolgt, erhält er einen geheimen symmetrischen Schlüssel, falls er die Zertifizierungsanfrage direkt an die CA richten will.

Der symmetrische Schlüssel wird initialer Authentisierungsschlüssel (Initial Authentication Key, IAK) genannt.<sup>9</sup> Zu jedem IAK gehört ein Referenzwert (Reference Value), der innerhalb von PKI-Nachrichten zur Identifikation des Teilnehmers verwendet wird. Er muß deshalb für jeden Teilnehmer eindeutig sein.

CA-Komponenten müssen die Generierung von IAK und Referenzwert unterstützen. Alle Komponenten müssen die Anwendung von IAK und Referenzwert unterstützen.

Sobald der Teilnehmer im Besitz von IAK und Referenzwert ist, kann er selbständig Zertifizierungsanfragen stellen und dadurch den Zertifizierungsprozeß initiieren.

Für den Zertifizierungsprozeß wird ein Drei-Wege-Protokoll verwendet. Es werden die folgenden drei PKI-Nachrichten ausgetauscht, in denen alle Informationen enthalten sind, die für die Zertifizierung benötigt werden:

- Durch die Zertifizierungsanfrage wird das Zertifikat spezifiziert, das von der CA generiert werden sollen. Durch Angabe weiterer Kontrollinformationen kann der Teilnehmer z.B. bestimmen, ob sein Zertifikat im Verzeichnis veröffentlicht werden soll.
- In einer Zertifizierungsantwort wird das generierte Zertifikat oder eine Fehlermeldung übermittelt. Die Zertifizierungsantwort kann darüber hinaus den privaten Schlüssel des Teilnehmers und weitere Informationen enthalten.
- Mit einer Bestätigungsnachricht muß der Teilnehmer eine erfolgreiche Zertifizierung elektronisch bestätigen.<sup>10</sup>

Während die Grundstruktur des Zertifizierungsprozesses stets gleich ist, sind die PKI-Nachrichten je nach Zertifizierungs-Variante unterschiedlich. Es werden sechs Hauptvarianten unterschieden. Jede dieser Varianten kombiniert eine Art der Schlüsselerzeugung (zentral oder dezentral) mit einem Typ für die Zertifizierungsanfrage (Erstantrag, Verlängerungsantrag oder Neuantrag).

Bei zentraler Schlüsselerzeugung wird der private Schlüssel dem Teilnehmer innerhalb der Zertifizierungsantwort übermittelt, falls er nicht z.B. in eine Chipkarte eingebracht wird. Bei dezentraler Schlüsselerzeugung muß der Schlüssel nicht transportiert werden, jedoch muß sich die CA davon überzeugen, daß der Teilnehmer im Besitz des privaten Schlüssel ist.

Bei Erstanträgen kann die Authentisierung mittels des IAK erfolgen. Zertifizierungsanfragen des Teilnehmers müssen bei einem Erstantrag unter Verwendung des Schlüssels und Angabe des Referenzwertes gestellt werden. Folgeanträge sind dagegen stets digital signiert.

Die Authentisierung mittels IAK soll jedoch ausschließlich für Zertifizierungsanfragen von Endteilnehmern verwendet werden. Aufgrund des erhöhten Sicherheitsbedarfs sollen CAs, die ein Zertifikat einer übergeordneten CA oder ein Cross-Zertifikat benötigen, den Zertifizierungsantrag stets bei der Registrierung stellen bzw. abgeben. Dies gilt auch für RAs.

Einen Verlängerungsantrag stellt der Teilnehmer, wenn er einen bereits zertifizierten Schlüssel weiterhin verwenden will, also kein Schlüsselwechsel erfolgen soll. In diesem Fall

---

<sup>9</sup> Der Schlüssel dient der Bildung eines MAC-Wertes für die Authentisierung von PKI-Nachrichten.

<sup>10</sup> Die Bestätigungsnachricht als abschließende Nachricht im Rahmen des Drei-Wege-Protokolls ist neu gegenüber der Version 1.1 der Spezifikation.

ist keine Schlüsselgenerierung erforderlich. Der Besitz des privaten Schlüssels soll jedoch von der CA im dezentralen Fall noch einmal geprüft werden.

Bei einem Neuantrag muß wie beim Erstantrag stets ein neuer Schlüssel generiert werden, der Antrag muß jedoch stets digital signiert werden.

Zu diesen sechs Hauptvarianten gibt es aufgrund des Drei-Wege-Protokolls jeweils drei PKI-Nachrichten, insgesamt somit 18 verschiedene Nachrichten. Für jede dieser Nachrichten wird ein Profil gebildet.

Die Profile werden in Kapitel 5.1, „Profile für Zertifizierungen“, im Detail dargestellt.

## 3.2 Sperrung von Zertifikaten

Es kann erforderlich sein, daß ein Zertifikat eines Teilnehmers innerhalb des Gültigkeitszeitraums für ungültig erklärt werden muß. Dieser Vorgang wird Sperrung genannt.

Für die Sperrung eines Zertifikats gibt es unterschiedliche Gründe, die sich in dringliche und nicht dringliche unterscheiden lassen. Ein dringlicher Sperrgrund liegt vor, wenn die Gefahr besteht, daß unechte digitale Signaturen erstellt oder verschlüsselte Daten unberechtigt entschlüsselt werden können.

Zu den dringlichen Sperrgründen zählen

- die Kompromittierung eines Teilnehmer-Schlüssels und der Verdacht der Kompromittierung („keyCompromise“). Dieser Sperrgrund liegt z.B. vor, wenn ein Teilnehmer seine Chipkarte (möglicherweise inklusive Paßwort) verloren hat.
- die Kompromittierung eines CA-Schlüssels und der Verdacht der Kompromittierung („cACompromise“).

Zu den nicht dringlichen Sperrgründen zählen

- die Änderung des Namen oder anderer im Zertifikat ausgewiesener Informationen über den Inhaber („affiliationChanged“). Dieser Sperrgrund liegt z.B. vor, falls sich die im Zertifikat ausgewiesene E-Mail Adresse geändert hat.
- die Ersetzung des Zertifikats durch ein anderes („superseded“). Dieser Sperrgrund liegt z.B. vor, wenn ein Verlängerungszertifikat generiert wurde und das Ende des Gültigkeitszeitraums des ersetzten Zertifikates noch nicht erreicht ist.
- die Beendigung der Nutzung des Zertifikats, da es nicht länger benötigt wird („cessationOfOperation“). Dieser Sperrgrund liegt z.B. vor, wenn der Zertifikats-Inhaber das Unternehmen verlassen hat, als dessen Vertreter er Signaturen erstellt hat.

In diesen Fällen ist die Sperrung in der Regel nicht dringlich, weil weder der Zertifikats-Inhaber noch ein unberechtigter Dritter unechte digitale Signaturen erstellen oder verschlüsselte Daten unberechtigt entschlüsselt kann.

Ein besonderer Fall ist die vorläufige Sperrung eines Zertifikats („certificateHold“). Sie gehört in der Regel zu den dringlichen Sperrgründen, da nur bei diesen eine Unklarheit darüber besteht, ob das Zertifikat gesperrt werden muß.

Mit Ausnahme des Sperrgrundes „cACompromise“, der den Schlüssel der CA betrifft, erfolgt eine Sperrung in der Regel auf Antrag des Zertifikats-Inhabers (bzw. eines hierzu Berechtigten). Im Rahmen der Spezifikation wird davon ausgegangen, daß der Antrag direkt

gegenüber der CA gestellt wird.<sup>11</sup> (CA steht hier und im folgenden für die Instanz, die Sperrlisten signiert. Dies muß nicht die CA sein, die das Zertifikat generiert hat.)

Die CA muß den Teilnehmer authentisieren können, um unberechtigte Sperrungen auszuschließen, die „denial of service“-Angriffe erlauben. Falls der Teilnehmer über die Möglichkeit verfügt, den Antrag digital zu signieren, soll er hiervon Gebrauch machen.<sup>12</sup> Falls er jedoch z.B. seine Chipkarte verloren hat, soll er den symmetrischen Schlüssel verwenden, den er bei der Registrierung für diesen Zweck erhalten hat.<sup>13</sup>

Im Sperrantrag müssen alle Sperrinformationen enthalten sein, die später in die Sperrliste eingetragen werden sollen, z.B. auch der Sperrgrund und der Ungültigkeitszeitpunkt.

Falls der Sperrantrag gültig ist, generiert die CA eine oder mehrere neue Sperrlisten, in denen das Zertifikat als gesperrt ausgewiesen wird.

Die CA generiert für jeden eingegangenen Sperrantrag eine Antwortnachricht, die den Status des Sperrantrag, Fehlermeldungen und ggf. die neuen Sperrlisten umfaßt.

Für den Fall einer Sperrung, die nicht durch den Zertifikats-Inhaber beantragt wurde, sieht die vorliegende Spezifikation eine Informationsnachricht vor, mit der der Zertifikats-Inhaber über die bevorstehende oder bereits erfolgte Sperrung in Kenntnis gesetzt wird (Revocation Notice).

Die Sperrliste wird durch den Verzeichnisdienst veröffentlicht. Sperrlisten müssen in regelmäßigen Zeitabständen publiziert werden. Der Publikationszeitraum sollte nicht zu lang gewählt werden, um die Aktualität der Zertifikatsinformationen nicht zu stark einzuschränken, aber auch nicht zu kurz, um den Aufwand zu begrenzen.<sup>14</sup>

Die Profile für die Sperrnachrichten werden im Kapitel 5.2, „Profile für Sperrungen“, im Detail dargestellt.

---

<sup>11</sup> Dadurch wird nicht ausgeschlossen, daß eine RA einen Antrag für den Teilnehmer stellt oder der Antrag des Teilnehmers über die RA gerootet wird.

<sup>12</sup> Die CA muß in diesem Fall nur prüfen, ob die digitale Signatur mathematisch korrekt ist, d.h. die Signatur muß nicht gültig sein.

<sup>13</sup> Nicht authentisierte Sperranträge werden im Rahmen der vorliegenden Spezifikation nicht behandelt. Es ist eine Frage der Sperrlichtlinie der CA, wie sie mit nicht authentisierten Sperranträgen umgeht. Sie kann in diesem Fall z.B. eine vorläufige Sperrung durchführen.

<sup>14</sup> Folgende doppelte Vorgehensweise kann empfohlen werden: Sperrlisten erhalten einen vernünftigen Gültigkeitszeitraum (z.B. 1-2 Wochen), werden jedoch regelmäßig kurzfristig aktualisiert bereitgestellt (z.B. täglich oder bei jeder Änderung). (d.h. überlappende Sperrlisten!) So können Standardprüfungen bei Bedarf durch höherwertige Prüfungen ersetzt werden, bei denen statt der lokal vorhandenen noch gültigen, z.B. die tagesaktuelle Sperrliste abgerufen und zugrunde gelegt wird.

## 4 Datenstrukturen

In diesem Kapitel werden die Datenstrukturen für PKI-Nachrichten spezifiziert. Zunächst wird die übergeordnete Datenstruktur dargestellt, die für alle PKI-Nachrichten verwendet wird. Danach werden die allgemeinen Datenstrukturen dargestellt, die nicht spezifisch für einen bestimmten Typ einer PKI-Nachricht sind. Anschließend werden Datenstrukturen für Zertifizierungen, Sperrungen und die Initialisierung einer Root-CA beschrieben.

### 4.1 Übergeordnete Datenstrukturen

Für alle PKI-Management-Nachrichten wird folgende allgemeine Struktur verwendet:

```
PKIMessage ::= SEQUENCE {
    header      PKIHeader,
    body        PKIBody,
    protection  [0] PKIProtection OPTIONAL,
    extraCerts  [1] SEQUENCE SIZE (1..MAX) OF Certificate
                OPTIONAL }
```

#### 4.1.1 Der Kopf einer PKI-Nachricht

Das Feld „header“ enthält allgemeine Informationen wie z.B. Adressen und Identifikatoren für Transaktionen.

Ein PKI-Header besteht aus folgender Datenstruktur:

```
PKIHeader ::= SEQUENCE {
    pvno          INTEGER { ietf-version2 (1) },
    sender        GeneralName,
    -- Dient der Identifikation des Absenders
    recipient     GeneralName,
    -- Dient der Identifikation des Adressaten
    messageTime  [0] GeneralizedTime          OPTIONAL,
    -- Information über den Zeitpunkt der Generierung der Nachricht
    protection    [1] AlgorithmIdentifer     OPTIONAL,
    -- Algorithmus, der für den Schutz der Nachricht verwendet wird
    senderKID    [2] KeyIdentifer            OPTIONAL,
    recipKID     [3] KeyIdentifer            OPTIONAL,
    -- Dienen der Identifizierung von Schlüsseln für den Schutz
    transactionID [4] OCTET STRING           OPTIONAL,
    -- Dient der Identifizierung der Transaktion
    senderNonce  [5] OCTET STRING            OPTIONAL,
    recipNonce   [6] OCTET STRING            OPTIONAL,
    -- Dienen dem Schutz vor einem Wiedereinspielen von Nachrichten
    freeText     [7] PKIFreeText             OPTIONAL,
    -- Dient der Angabe von kontext-spezifischen Instruktionen
    -- (zur manuellen Auswertung)
    generalInfo  [8] SEQUENCE SIZE (1..MAX) OF
                InfoTypeAndValue            OPTIONAL
    -- Dient der Angabe von kontext-spezifischen Informationen
    -- (zur automatischen Auswertung) }
```

```
PKIFreeText ::= SEQUENCE SIZE (1..MAX) OF UTF8String
    -- Beliebiger als UTF-8-String kodierter Text
```

MailTrusT-konforme Komponenten müssen alle Felder der Datenstruktur mit Ausnahme der optionalen Felder „messageTime“, „freeText“ und „generalInfo“ unterstützen. Falls diese optionalen Felder vorhanden sind, dürfen sie ignoriert werden.

#### 4.1.1.1 Versionsnummer

Das Feld „pvno“ dient der Angabe der Version der spezifizierten Datenstrukturen. Es ist stets der feste Integer-Wert „1“ zu verwenden.<sup>15</sup>

#### 4.1.1.2 Absender

Das Feld „sender“ wird (zusammen mit dem Feld „senderKID“) zur Identifikation des Absenders verwendet, wobei es zwei Varianten gibt. Falls dem Absender zertifiziert ist, muß er seinen X.500-Distinguished-Name verwenden. Andernfalls ist der Referenzwert zu verwenden, den er bei der Registrierung zusammen mit dem IAK erhalten hat. Dieser Referenzwert muß stets im Feld „senderKID“ eingetragen werden.

Im Feld „sender“ kann somit entweder der X.500-Distinguished-Name angegeben werden oder kein Name, falls der Absender nicht über die Möglichkeit verfügt, die Nachricht mit einem gültigen Schlüssel digital zu signieren. Im letztgenannten Fall enthält das Feld den Wert „NULL“, d.h. die „SEQUENCE OF RelativeDistinguishedName“ hat die Länge Null. Das Feld „senderKID“ muß in diesem Fall den Referenzwert enthalten.

#### 4.1.1.3 Empfänger

Für das Feld „recipient“ gilt das gleiche wie für das Feld „sender“. In diesem Feld muß der X.500-Distinguished-Name des Empfängers angegeben werden. Es bleibt ebenfalls leer, falls der Empfänger noch nicht über einen X.500-Distinguished-Name verfügt, z.B. bei einer Antwort auf eine Erstantrag. In diesem Fall muß der Referenzwert im Feld „recipientKID“ angegeben werden.

#### 4.1.1.4 Zeitpunkt der Generierung der Nachricht

Die Verwendung des Feldes „messageTime“ ist optional. Die Information über den Zeitpunkt der Generierung der Nachricht ist ohne Bedeutung. Die Zeitangabe kann jedoch z.B. erfolgen, damit Teilnehmer ihre lokale Zeit mit der Zeit der CA abstimmen können. MailTrusT-konforme Komponenten müssen dieses Feld aber nicht auswerten können. Falls der Zeitpunkt angegeben wird, ist das Zeitformat zu verwenden, das auch in Zertifikaten Verwendung findet.<sup>16</sup>

#### 4.1.1.5 Algorithmus

Das Feld „protectionAlg“ enthält einen OID für einen Algorithmus, der zum Schutz der Nachricht verwendet wird. (Die zulässigen Algorithmen werden zusammen mit den OIDs im Dokument „Algorithmen“ beschrieben.)

#### 4.1.1.6 Schlüssel des Absenders

Das Feld „senderKID“ muß zur Angabe des Referenzwertes verwendet werden, wenn die Nachricht durch den IAK geschützt wird.

---

<sup>15</sup> In der vorliegenden Spezifikation wurden ausschließlich IETF-Datenstrukturen verwendet und deshalb die aktuelle Versionsnummer der IETF übernommen. Für den Fall, daß diese Strukturen von der IETF in einer Weise geändert werden, die von MailTrusT nicht akzeptiert werden kann, besteht der Vorbehalt, eine eigene MailTrusT-Versionsnummer zu vergeben.

Produkte müssen auf eine Änderung dieser Versionsnummer vorbereitet sein. Sie muß jederzeit ohne wesentlichen Aufwand durchgeführt werden können.

<sup>16</sup> S. Kapitel 3.1.5, „Gültigkeitsdauer“, des Dokumentes „Profile für Zertifikate und Sperrlisten“.

#### **4.1.1.7 Schlüssel des Empfängers**

Das Feld „recipKID“ ist nur dann von Bedeutung, wenn der Empfänger der Nachricht (noch) nicht seinen X.500-Distinguished-Name kennt aber den Referenzwert, z.B. bei einem Erstantrag. In diesem Fall wird die CA den Referenzwert in dieses Feld eintragen und der Empfänger kann so erkennen, daß die Nachricht für ihn bestimmt ist.

#### **4.1.1.8 Transaktionskennzeichen**

Das Feld „transactionID“ dient der Zuordnung von Nachrichten, die zu einer Transaktion gehören, z.B. Zertifizierungsanfrage, Zertifizierungsantwort und Bestätigungsnachricht. Es kann vom Absender verwendet werden, um die Antwort auf die Nachricht zuordnen zu können. Aus diesem Grunde muß die „transactionID“ aus Sicht des Absenders des ersten PKI-Nachricht der Transaktion eindeutig sein.

#### **4.1.1.9 Nonce des Absenders**

Das Feld „senderNonce“ enthält einen Wert, der zum Schutz gegen das Wiedereinspielen von Nachrichten verwendet werden kann, z.B. eine Zufallszahl.

#### **4.1.1.10 Nonce des Empfängers**

Das Feld „recipientNonce“ enthält den Wert, den der Empfänger zuvor bei einer Nachricht an den Absender als „senderNonce“ angegeben hat.

#### **4.1.1.11 Freitext**

Das Feld „freeText“ kann der Übermittlung einer Information dienen, die zur manuellen Auswertung bestimmt ist. MailTrusT-konforme Produkte müssen dieses Feld nicht auswerten können.<sup>17</sup>

#### **4.1.1.12 Allgemeine Informationen**

Das Feld „generallInfo“ kann der Übermittlung einer Information dienen, die zur automatischen Auswertung bestimmt ist. MailTrusT-konforme Komponenten müssen dieses Feld nicht auswerten können.

---

<sup>17</sup> Da alle MailTrusT-Komponenten E-Mail unterstützen müssen, kann eine Freitext-Meldung stets als eigene E-Mail versandt werden. In der Regel wird eine PKI-Management-Nachricht ebenfalls per E-Mail versandt, so das die Freitext-Meldung in diese Nachricht integriert werden kann. Eines eigenen Feldes innerhalb der PKI-Management-Nachricht bedarf es daher nicht.

### 4.1.2 Der Rumpf einer PKI-Nachricht

Nachrichtenspezifische Informationen werden im Feld „body“ angegeben. Die folgenden nachrichtenspezifischen Elemente werden in der vorliegenden Spezifikation verwendet:

```
PKIBody ::= CHOICE {
    -- nachrichtenspezifische Elemente
    ir [0] CertReqMessages, -- Initialization Request
    -- Für Zertifizierungsanfrage bei Erstanträgen
    ip [1] CertRepMessage, -- Initialization Replay
    -- Für Zertifizierungsantworten auf Erstanträge
    cr [2] CertReqMessages, -- Certification Request
    -- Für Zertifizierungsanfragen bei Verlängerungsanträgen
    cp [3] CertRepMessage, -- Certification Replay
    -- Für Zertifizierungsantworten bei Verlängerungsanträgen
    kur [7] CertReqMessages, -- Key Update Request
    -- Für Zertifizierungsanfragen bei Neuanträgen (key update)
    kup [8] CertRepMessage, -- Key Update Response
    -- Für Zertifizierungsantworten bei Neuanträgen
    rr [11] RevReqContent, -- Revocation Request
    -- Für Sperranträge
    rp [12] RevRepContent, -- Revocation Response
    -- Für Antworten auf Sperranträge
    rann [17] RevAnnContent, -- Revocation Announcement
    -- Für Hinweise auf eine durchgeführte oder bevorstehende
    -- Sperrung }

```

Weitere nachrichtenspezifische Elemente wurden durch die IETF spezifiziert [PKIX-CMP 98]. Von den insgesamt 24 Elementen wurden aus Gründen der Übersichtlichkeit nur die aufgeführt, die in der Spezifikation genutzt werden. Bis auf das Element „Revocation Announcement“ müssen alle diese Elemente unterstützt werden.

### 4.1.3 Der Schutz einer PKI-Nachricht

Die Integrität aller PKI-Nachrichten wird durch Informationen im Feld „protection“ geschützt. Bei der Bestätigungsnachricht wird dieses Feld gleichzeitig auch zum Nachweis der Besitzes des privaten Schlüssels verwendet, wenn ein vom Teilnehmer generierter Verschlüsselungsschlüssel zertifiziert werden soll.

Die Verwendung dieses Feldes ist optional. Es wird nicht benötigt, wenn andere als die hier spezifizierten Schutzmechanismen verwendet werden. MailTrust-konforme Komponenten müssen jedoch die hier spezifizierten Mechanismen unterstützen.

Es wird die folgende Syntax verwendet:

```
PKIProtection ::= BIT STRING
```

Der Input für den Algorithmus zur Berechnung von „PKIProtection“ ist der DER-kodierte Wert folgender Datenstruktur:

```
ProtectedPart ::= SEQUENCE {
    header PKIHeader,
    body PKIBody }

```

Es werden sowohl symmetrische als auch asymmetrische Verfahren zur Berechnung des Bitstrings für „PKIProtection“ verwendet. Digitale Signaturverfahren können eingesetzt werden, wenn der Absender einer Nachricht über einen zertifizierten gültigen Signaturschlüssel verfügt. Andernfalls muß er ein symmetrisches Verfahren verwenden.<sup>18</sup>

---

<sup>18</sup> Die zulässigen Algorithmen werden im Dokument „Algorithmen“ beschrieben.

#### 4.1.4 Beigefügte Zertifikate in einer PKI-Nachricht

Das optionale Feld „extraCerts“ enthält ggf. Zertifikate für den Empfänger, die der Nachricht beigefügt werden können. Das Feld kann eine CA z.B. verwenden, damit ein Teilnehmer ein von ihr generiertes Zertifikat verifizieren kann, ohne Verzeichnisabfragen durchführen zu müssen.

Das Feld muß nicht unterstützt werden.

## 4.2 Allgemeine Datenstrukturen

Bevor die nachrichtenspezifischen Elemente beschrieben werden, erfolgt zunächst die Spezifikation von Datenstrukturen, die für mehrere dieser Elemente verwendet werden.

### 4.2.1 Datenstrukturen zur Spezifikation eines Zertifikates

Es gibt mehrere PKI-Nachrichten, in denen eine Schablone (template) für die Felder eines Zertifikates benötigt wird. In einer Zertifizierungsanfrage werden z.B. Daten angegeben, die im Zertifikat enthalten sein sollen. Die Datenstruktur für die Schablone ist nahezu identisch mit der Datenstruktur für ein Zertifikat. Der Unterschied besteht lediglich darin, daß in dieser Datenstruktur alle Felder optional sind.

Es wird folgende Syntax verwendet:

```

CertTemplate ::= SEQUENCE {
    version          [0] Version          OPTIONAL,
    serialNumber     [1] INTEGER          OPTIONAL,
    signingAlg       [2] AlgorithmIdentifer OPTIONAL,
    issuer           [3] Name              OPTIONAL,
    validity         [4] OptionalValidity OPTIONAL,
    subject          [5] Name              OPTIONAL,
    publicKey        [6] SubjectPublicKeyInfo OPTIONAL,
    issuerUID        [7] UniqueIdentifier  OPTIONAL,
    subjectUID       [8] UniqueIdentifier  OPTIONAL,
    extensions       [9] Extensions       OPTIONAL }

OptionalValidity ::= SEQUENCE {
    notBefore        [0] Time              OPTIONAL,
    notAfter         [1] Time              OPTIONAL
    -- Es muß mindestens ein Wert angegeben werden }

```

### 4.2.2 Verschlüsselte Werte

Eine PKI-Nachricht muß nicht insgesamt gegen unerlaubte Kenntnisnahme geschützt sein. Es besteht jedoch die Notwendigkeit, Schlüssel und Zertifikate in PKI-Nachrichten vertraulich zu übermitteln. Hierfür wird die folgende „EncryptedValue“-Syntax verwendet:

```

EncryptedValue ::= SEQUENCE {
  intendedAlg      [0] AlgorithmIdentifier OPTIONAL,
  -- Der Algorithmus, für den der Wert „encValue“ verwendet
  -- werden soll
  symmAlg          [1] AlgorithmIdentifier OPTIONAL,
  -- Der symmetrische Algorithmus zur Verschlüsselung des Wertes
  encSymmKey       [2] BIT STRING          OPTIONAL,
  -- Der verschlüsselte symmetrische Schlüssel zur
  -- Verschlüsselung des Wertes
  keyAlg           [3] AlgorithmIdentifier OPTIONAL,
  -- Der Algorithmus zur Verschlüsselung des symmetrischen
  -- Schlüssels
  valueHint        [4] OCTET STRING        OPTIONAL,
  -- Eine kurze Beschreibung des Inhalts oder ein Bezeichner des
  -- Wertes
  encValue         BIT STRING
  -- Der verschlüsselte Wert }

```

Die Verwendung dieser Datenstruktur setzt voraus, daß die Kommunikationspartner über einen gemeinsamen symmetrischen Schlüssel verfügen oder daß der Empfänger der PKI-Nachricht über einen privaten Schlüssel verfügt, der für die Zwecke der Entschlüsselung verwendet werden kann.

Mit diesem Schlüssel kann in beiden Fällen ein symmetrischer Session-Key verschlüsselt werden, der zur Verschlüsselung der Nutzdaten dient.

Alle Optionen mit Ausnahme des Feldes „valueHint“ müssen unterstützt werden.<sup>19</sup>

### 4.2.3 Status und Fehlerinformationen

Alle Antworten auf PKI-Nachrichten enthalten stets Statusinformationen zur Anfrage. Die allgemeine Information über den Status setzt sich aus folgenden drei Feldern zusammen:

```

PKIStatusInfo ::= SEQUENCE {
  status           PKIStatus,
  statusString     PKIFreeText   OPTIONAL,
  failInfo         PKIFailureInfo OPTIONAL }

```

Jede Antwort muß einen Statuswert enthalten. Der Status wird durch genau einen der folgenden Integer-Werte angegeben:

```

PKIStatus ::= INTEGER {
  granted           (0),
  -- Der Anfrage konnte entsprochen werden
  grantedWithMods  (1),
  -- Der Anfrage konnte nur zum Teil entsprochen werden.
  rejection        (2),
  -- Der Anfrage konnte nicht entsprochen werden
  waiting          (3),
  -- Der Anfrage konnte noch nicht bearbeitet werden
  revocationWarning (4),
  -- Diese Nachricht enthält eine Warnung, da eine Sperrung
  -- unmittelbar bevorsteht
  revocationNotification (5),
  -- Diese Nachricht enthält eine Sperranzeige
  keyUpdateWarning (6)
  -- Warnung, daß eine Schlüsselerneuerung für den in der Anfrage
  -- bezeichneten Schlüssel bereits erfolgt ist }

```

---

<sup>19</sup> Dem optionalen Feld „valueHint“ kommt für die MailTrusT-Spezifikation keine Bedeutung zu.

Im Fehlerfall können diesem Statuswert weitere Informationen hinzugefügt werden. Es kann ein Freitextfeld „statusString“ und ein Feld für strukturierte Informationen verwendet werden. Für das Freitextfeld wird folgende Datenstruktur verwendet:

```
PKIFreeText ::= SEQUENCE SIZE (1..MAX) OF UTF8String
  -- Beliebiger als UTF-8-String codierter Text
```

Strukturierte Fehlerinformationen können mittels „failInfo“ übermittelt werden. Dazu wird folgende Syntax verwendet:

```
PKIFailureInfo ::= BIT STRING {
  badAlg          (0),
  -- Unbekannter oder nicht unterstützter Bezeichner für einen
  -- verwendeten Algorithmus
  badMessageCheck (1),
  -- Die Verifikation der Nachricht ist fehlgeschlagen
  badRequest      (2),
  -- Die Transaktion ist nicht erlaubt oder wird nicht
  -- unterstützt
  badTime         (3),
  -- Die in der Nachricht angegebene Zeit liegt nicht im
  -- zulässigen Zeitfenster
  badCertId       (4),
  -- Es wurde kein Zertifikat gefunden, das den angegebenen
  -- Kriterien entspricht
  badDataFormat   (5),
  -- Die Daten haben das falsche Format
  wrongAuthority  (6),
  -- Die Empfängerinstanz, die in der Anfrage angegeben wurde
  -- entspricht nicht der Instanz, die die Antwort erstellt hat
  incorrectData   (7),
  -- Die Daten des Anfragenden sind falsch
  missingTimeStamp (8)
  -- Es fehlt ein erforderlicher Zeitstempel }
```

Mittels dieser Information können bestimmte Fehlertypen einer PKI-Nachricht dargestellt werden. Es können mehrere Fehler gleichzeitig auftreten, so daß auch mehrere Fehlerbits angegeben werden können.

MailTrusT-konforme Komponenten müssen die Bits „0“ und „2“ der Datenstruktur „PKIStatus“ unterstützen. Die Unterstützung der weiteren Bits und der optionalen Datenstrukturen „PKIFreeText“ und „PKIFailureInfo“ wird empfohlen, soweit sie für die vorliegende Spezifikation von Bedeutung sind.

#### 4.2.4 Identifikatoren für Zertifikate

Zur Identifikation von Zertifikaten wird die folgende Syntax verwendet:

```
CertID ::= SEQUENCE {
  issuer      GeneralName,
  serialNumber INTEGER }
```

Als Name der ausstellenden CA wird stets der X.500-Distinguished-Name (DN) verwendet, d.h. als „GeneralName“ kommt nur ein „directoryName“ in Betracht.<sup>20</sup> Jedes Zertifikat hat eine eindeutige Seriennummer. Durch den DN der ausstellenden CA und die Seriennummer wird jedes Zertifikat eindeutig identifiziert.

---

<sup>20</sup> S. Kapitel 3.2.7, „Alternative Namen für Zertifikats-Inhaber“, des Dokumentes „Profile für Zertifikate und Sperrlisten.“

MailTrusT-konforme Komponenten müssen die Identifikation von Zertifikaten über diese Datenstruktur unterstützen.

### 4.3 Datenstrukturen für Zertifizierungen

Unter Verwendung der allgemeinen Datenstrukturen werden die Datenstrukturen für Zertifizierungen spezifiziert, d.h. Zertifizierungsanfrage, Zertifizierungsantwort und Bestätigungsnachricht.

Die dargestellte Syntax für die Datenstrukturen würde es erlauben, mehrere Zertifikate über eine Zertifizierungsanfrage zu beantragen. Aus Gründen der Vereinfachung wurde für alle Profile jedoch eine Beschränkung auf ein Zertifikat pro Anfrage vorgenommen. Die Beantragung mehrerer Zertifikate durch einen Antrag muß nicht unterstützt werden.

#### 4.3.1 Datenstrukturen für Zertifizierungsanfragen

Der Teilnehmer muß in einer Zertifizierungsanfrage („CertRequest“) die Informationen angeben, die zur Ausstellung des Zertifikates erforderlich sind und er muß nachweisen, daß er im Besitz des mit dem zu zertifizierenden öffentlichen Schlüssel korrespondierenden privaten Schlüssel ist, falls er den Schlüssel selbst erzeugt hat („Proof of Possession“).

Für den Rumpf einer Zertifizierungsanfrage wird folgende Syntax verwendet:

```
CertReqMessages ::= SEQUENCE SIZE (1..MAX) OF CertReqMsg

CertReqMsg ::= SEQUENCE {
    certReq      CertRequest,
    pop          ProofOfPossession OPTIONAL
    regInfo     SEQUENCE SIZE (1..MAX) of
                AttributeTypeAndValue OPTIONAL }

```

Das Feld „regInfo“ kann für die Aufnahme zusätzlicher Informationen wie z.B. Vertragskonditionen oder Abrechnungsdaten dienen. Dies Feld muß nicht unterstützt werden. Zu unterstützen sind jedoch die Datenstrukturen „CertRequest“ und „ProofOfPossession“.

##### 4.3.1.1 CertRequest

Für „CertRequest“ wird folgende Syntax verwendet:

```
CertRequest ::= SEQUENCE {
    certReqID    INTEGER
    certTemplate CertTemplate
    controls     Controls OPTIONAL }

Controls ::= SEQUENCE SIZE(1..MAX) OF
            AttributeTypeAndValue

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

AttributeType ::= OBJECT IDENTIFIER

AttributeValue ::= ANY DEFINED BY AttributeType

```

Das Feld „certReqID“ enthält einen Integer-Wert, der als Index dient. Der Wert identifiziert ein Zertifikat in Zertifizierungsanfrage, Zertifizierungsantwort und Bestätigungsnachricht.<sup>21</sup>

Das Feld „certTemplate“ enthält die bereits beschriebene Schablone eines Zertifikates (s. Kapitel 4.2.1, „Datenstrukturen zur Spezifikation eines Zertifikates“). In die Felder der Schablone werden die Daten eingetragen, die im Zertifikat enthalten sein sollen.

Die Eintragungen sind für die CA nicht verbindlich. Die endgültige Entscheidung über den Inhalt von Zertifikaten trifft die CA. Sie kann deshalb die Daten ändern, entfernen oder weitere Daten in das beantragte Zertifikat aufnehmen.<sup>22</sup> Der Teilnehmer kann die Entscheidung und das Zertifikat akzeptieren oder ablehnen.

Über das Feld „controls“ kann der Antragssteller die Art und Weise der Bearbeitung des Antrags beeinflussen. MailTrust-konforme Komponenten müssen folgende Typen unterstützen:<sup>23</sup>

Typen	Verwendungszweck
„pkiPublicationInfo“	Dient der Steuerung der Veröffentlichung des Zertifikats <sup>24</sup>
„protocolEncrKey“	Dient der Verschlüsselung von Informationen für den Antragsteller

Tabelle 1: „controls“

Es wird folgende Syntax verwendet:

```
PKIPublicationInfo ::= SEQUENCE {
    action          INTEGER {
        dontPublish      (0),
        pleasePublish    (1) },
    pubInfos        SEQUENCE SIZE (1..MAX) OF
                    SinglePubInfo OPTIONAL }

SinglePubInfo     ::= SEQUENCE {
    pubMethod       INTEGER {
        dontCare        (0),
        x500            (1),
        web             (2),
        ldap           (3) },
    pubLocation     GeneralName OPTIONAL }
```

Durch Angabe des Wertes „0“ für „dontPublish“ wird der CA verboten, das Zertifikat zu veröffentlichen. In diesem Fall darf das Teilfeld „pubInfos“ nicht vorhanden sein.

Durch Angabe der Wertes „1“ für „pleasePublish“ oder durch Weglassen der „PKIPublicationInfo“ wird das Zertifikat zur Veröffentlichung freigegeben. Mittels der Datenstruktur „SinglePubInfo“ kann der Antragsteller im erstgenannten Fall bestimmen, wo das Zertifikat veröffentlicht werden soll. Durch Angabe von „dontCare“ gibt er zu verstehen,

<sup>21</sup> Für die vorliegende Spezifikation hat dieser Index keine Bedeutung, da sich ein Zertifizierungsantrag stets nur auf ein Zertifikat bezieht. Diese Beschränkung kann jedoch in späteren Spezifikationen aufgehoben werden. Ein Integer-Wert muß stets angegeben werden, da das Feld nicht optional ist.

<sup>22</sup> CA-Komponenten müssen daher diese Möglichkeit vorsehen.

<sup>23</sup> In der IETF-Spezifikation [PKIX-CRMF 98] werden weitere Typen definiert, die jedoch für die vorliegende Spezifikation keine Bedeutung haben und deshalb aus Gründen der Übersichtlichkeit weggelassen werden.

<sup>24</sup> Der Teilnehmer kann darüber entscheiden, ob sein Zertifikat durch die CA in einem öffentlichen Verzeichnis zum Abruf bereitgestellt wird oder ob er sich vorbehält, das Zertifikat selbst bereitzustellen, falls es z.B. für Verifikationszwecke benötigt wird.

daß ihm die Art der Veröffentlichung egal ist. Dieser Wert wird stets angenommen, wenn keine „SinglePubInfo“ vorhanden ist.

MailTrusT-konforme Komponenten müssen dem Antragsteller die Möglichkeit einräumen, die Bereitstellung des Zertifikates insgesamt zu verhindern. Sie müssen jedoch keine Auswahl hinsichtlich bestimmter Arten der Veröffentlichung unterstützen. Das Feld „pkiPublicationInfo“ muß somit unterstützt werden, nicht jedoch die Option „pubInfos“.<sup>25</sup>

Die gesicherte Übertragung eines durch die CA generierten privaten Schlüssels oder eines Zertifikates, das nicht veröffentlicht werden soll, erfolgt durch Verschlüsselung mit einem öffentlichen Verschlüsselungsschlüssel aus dem Feld „protocolEncrKey“. Der Teilnehmer generiert einen temporären asymmetrischen Verschlüsselungsschlüssel, der von der CA später zur Verschlüsselung des privaten Schlüssels verwendet wird, und trägt ihn in das Feld ein. Die Kodierung entspricht der Kodierung des Zertifikatsfeldes „SubjectPublicKeyInfo“.<sup>26</sup>

#### 4.3.1.2 ProofOfPossession

Die CA muß sich davon überzeugen, daß der Teilnehmer im Besitz des mit dem öffentlichen Schlüssel korrespondierenden privaten Schlüssels ist. Abhängig vom Verwendungszweck des Schlüssels werden verschiedene Verfahren verwendet.

Bei Signaturschlüsseln weist der Teilnehmer den Besitz dadurch nach, daß er eine Signatur durchführt. Bei Entschlüsselungsschlüsseln übermittelt die CA das für den Teilnehmer generierte Zertifikat in verschlüsselter Form. Der Teilnehmer kann das Zertifikat nur dann entschlüsseln, wenn er im Besitz des Schlüssels ist.

Für den Nachweis wird folgende Syntax verwendet:

```
ProofOfPossession ::= CHOICE {
    raVerified          [0] NULL,
    signature           [1] POPOSigningKey,
    keyEncipherment    [2] POPOPrivKey,
    keyAgreement       [3] POPOPrivKey }
```

Den Auswahlfeldern „raVerified“ und „keyAgreement“ kommt in der MailTrusT-Spezifikation keine Bedeutung zu.<sup>27</sup> Die übrigen Felder müssen unterstützt werden.

Das Feld „signature“ wird zur Prüfung des Besitzes eines Signaturschlüssels verwendet. Dabei wird folgende Syntax verwendet:

```
POPOSigningKey ::= SEQUENCE {
    poposkInput          [0] POPOSigningKeyInput OPTIONAL,
    algorithmIdentifier  AlgorithmIdentifier,
    signature            BIT STRING }
```

<sup>25</sup> Eine CA muß ein Zertifikat, das veröffentlicht werden soll, stets in allen Verzeichnissen zum Abruf bereitstellen, falls sie neben dem LDAP-Verzeichnis weitere Verzeichnistypen unterstützt. Durch diese Forderung wird verhindert, daß eine Suchoperation in einem Verzeichnis mit einem negativen Ergebnis endet, weil der Teilnehmer das „falsche“ Verzeichnis durchsucht hat.

<sup>26</sup> S. Kapitel 3.1.7, „Öffentlicher Schlüssel des Zertifikats-Inhabers“ des Dokumentes „Profile für Zertifikate und Sperrlisten“.

<sup>27</sup> Das Feld „raVerified“ ist nur dann von Bedeutung, wenn die Prüfung des Besitzes sowohl durch eine CA als auch eine RA erfolgen kann und die RA der CA das Ergebnis der Prüfung in diesem Feld mitteilt.

Dem Feld „keyAgreement“ kommt ebenfalls keine Bedeutung zu, da das Diffie-Hellman-Verfahren zur Schlüsselvereinbarung nicht unterstützt werden muß (s. Dokument „Algorithmen“).

```

POPOSigningKeyInput ::= SEQUENCE {
    authInfo CHOICE {
        sender [0] GeneralName,
        publicKeyMAC PKMACValue },
    publicKey SubjectPublicKeyInfo }

PKMACValue ::= SEQUENCE {
    algId AlgorithmIdentifier,
    value BIT STRING }

```

Das Feld „signature“ enthält die Signatur über das DER-kodierte Feld „certreq“ der Datenstruktur „CertReqMsg“, die mit dem zu zertifizierenden privaten Schlüssel generiert wird. Dabei wird der im Feld „algorithmIdentifier“ angegebenen Signaturalgorithmus verwendet.<sup>28</sup>

Das Feld „keyEncipherment“ wird verwendet, wenn es sich um einen Entschlüsselungsschlüssel handelt. In diesem Fall wird folgende Syntax verwendet:

```

POPOPrivKey ::= CHOICE {
    thisMessage [0] BIT STRING,
    subsequentMessage [1] SubsequentMessage,
    dhMAC [2] BIT STRING }

SubsequentMessage ::= INTEGER {
    encrCert (0),
    challengeResp (1) }

```

Dabei muß die Variante „subsequentMessage“ gewählt werden. Die Prüfung des Besitzes erfolgt in diesem Fall über eine weitere PKI-Nachricht.<sup>29</sup> Durch die Angabe des Integer-Wertes „0“ wird angezeigt, daß das in der Zertifizierungsantwort enthaltene Zertifikat mit dem öffentlichen Verschlüsselungsschlüssel verschlüsselt werden soll. In der Bestätigung der Zertifizierungsantwort muß dann nachgewiesen werden, daß das Zertifikat entschlüsselt werden konnte.

Die Variante „challengeResp“ muß von MailTrusT-konformen Komponenten nicht unterstützt werden.<sup>30</sup>

### 4.3.2 Datenstrukturen für Zertifizierungsantworten

Eine Zertifizierungsantwort enthält im PKI-Rumpf die Datenstruktur „CertRepMessage“, die für das beantragte Zertifikat einen Statuswert, das Zertifikat für den Teilnehmer und optional Zertifikate der CA, Fehlerinformationen und einen verschlüsselten privaten Schlüssel enthält.

<sup>28</sup> Das optionale Feld „poposInput“ wird nicht benötigt.

<sup>29</sup> Die Variante „thisMessage“ erfordert, daß der private Schlüssel der Zertifizierungsanfrage in verschlüsselter Form beigefügt wird. Sie führt zu einer Vereinfachung des Verfahrens. Aufgrund des Transportes des privaten Schlüssels erhöht diese Variante jedoch das Risiko für eine Kompromittierung des privaten Schlüssels. Diese Variante sollte deshalb nur dann in Betracht gezogen werden, wenn die CA den Schlüssel für andere Zwecke ohnehin benötigt, z.B. für ein Key-Backup.

Die Variante „dhMAC“ ist ohne Bedeutung, da das Diffie-Hellmann-Verfahren zur Vereinbarung von Schlüsseln nicht unterstützt werden muß (s. Dokument „Algorithmen“).

<sup>30</sup> Die Variante wurde in die IETF-Spezifikation aufgenommen, um ein Szenario zu unterstützen, in dem ein Teilnehmer eine Zertifizierungsanfrage stellt, eine RA diese Anfrage hinsichtlich des Besitzes des privaten Schlüssels prüft und anschließend eine weitere Zertifizierungsanfrage im Auftrag des Teilnehmers an die CA richtet. Das hierfür verwendete Protokoll umfaßt insgesamt 8 PKI-Nachrichten (s. Kapitel 3.2.8 [PKIX-CMP 98]). Da die CA stets die Prüfung des Besitzes durchführen muß besteht für diese Variante kein Bedarf.

Es wird folgende Syntax verwendet:

```
CertRepMessage ::= SEQUENCE {
    caPubs          [1] SEQUENCE SIZE (1..MAX) OF Certificate
    OPTIONAL,
    response        SEQUENCE OF CertResponse }
```

Ein oder mehrere Zertifikate der CA können optional im Feld „caPubs“ angegeben werden. Das Feld muß nicht unterstützt werden.

Das Feld „response“ enthält einen Wert vom Typ „CertResponse“, für den folgende Syntax verwendet wird:

```
CertResponse ::= SEQUENCE {
    certReqId      INTEGER,
    status         PKIStatusInfo,
    certifiedKeyPair CertifiedKeyPair OPTIONAL,
    rspInfo        OCTET STRING      OPTIONAL }
```

Das Feld „certReqId“ entspricht dem gleichnamigen Feld des Zertifizierungsantrags „CertRequest“. Es ist der Wert aus dem Zertifizierungsantrag anzugeben. Jedoch ist der Wert „-1“ ist zu verwenden, falls im „CertRequest“ kein Wert angegeben wurde.

Das Feld „status“ enthält einen Wert vom Typ „PKIStatusInfo“ (s. Kapitel 4.2.3, „Status und Fehlerinformationen“). Für Zertifizierungsantworten besteht die Beschränkung, daß immer nur ein Grund dafür angegeben werden darf, weshalb dem Antrag nicht entsprochen werden kann.

Das Feld „rspInfo“ entspricht dem Feld „regInfo“ der Zertifizierungsanfrage (s. Kapitel 4.3.1, „Datenstrukturen für Zertifizierungsanfragen“). Dieses Feld muß ebenfalls nicht unterstützt werden.

Das Feld „certifiedKeyPair“ enthält das Zertifikat, das stets Teil der Zertifizierungsantwort sein muß.<sup>31</sup> Für das Feld „certifiedKeyPair“ wird folgende Syntax verwendet:

```
CertifiedKeyPair ::= SEQUENCE {
    certOrEncCert  CertOrEncCert,
    privateKey     [0] EncryptedValue      OPTIONAL,
    publicationInfo [1] PKIPublicationInfo OPTIONAL }
```

```
CertOrEncCert ::= CHOICE {
    certificate     [0] Certificate,
    encryptedCert  [1] EncryptedValue }
```

Das Feld „certOrEncCert“ enthält entweder das Zertifikat oder das verschlüsselte Zertifikat. Im letztgenannten Fall wird die Syntax verwendet, die im Kapitel 4.2.2, „Verschlüsselte Werte“, beschrieben ist.

Die CA hat das Zertifikat zu verschlüsseln, wenn der Teilnehmer im Feld „publicationInfo“ der Zertifizierungsanfrage (s. Kapitel 4.3.1.1, „CertRequest“) angegeben hat, daß das Zertifikat nicht veröffentlicht werden soll. Das Zertifikat ist auch zu verschlüsseln, wenn es für einen Verschlüsselungsschlüssel erstellt wurde und das Schlüsselpaar durch den Teilnehmer erstellt wurde. Die Verschlüsselung dient in diesem Fall dem Nachweis des Besitzes des privaten Entschlüsselungsschlüssels.

<sup>31</sup> Die IETF-Spezifikation stellt es der CA grundsätzlich frei, ob sie das Zertifikat dem Teilnehmer übermittelt. Falls sie das Zertifikat nicht übermittelt, muß es der Teilnehmer aus dem öffentlichen Verzeichnis abrufen. Die vorliegende Spezifikation fordert demgegenüber, daß das Zertifikat vom Teilnehmer geprüft wird, bevor es im Verzeichnis veröffentlicht werden darf. Aus diesem Grunde muß das Zertifikat stets Teil der Zertifizierungsantwort sein.

Zur Verschlüsselung wird eine Kombination eines asymmetrischen mit einem symmetrischen Verfahrens verwendet. Das Zertifikat wird durch einen zufällig gewählten symmetrischen Session-Key und dieser dann mit einem öffentlichen Schlüssel des Teilnehmers verschlüsselt.

Falls der Teilnehmer die Zertifizierung eines von ihm generierten Verschlüsselungsschlüssel beantragt hat, muß stets dieser Schlüssel von der CA zur Verschlüsselung des Session-Key verwendet werden, damit der Teilnehmer durch die Entschlüsselung des Session-Key den Besitz des privaten Entschlüsselungsschlüssels nachweisen kann. Andernfalls verwendet die CA den „protocolEncrKey“, den der Teilnehmer der Zertifizierungsanfrage beifügen muß, damit das nicht zu veröffentlichende Zertifikat geschützt werden kann.

Das Teilfeld „privateKey“ enthält den ebenfalls mit dem „protocolEncrKey“ verschlüsselten privaten Schlüssel, falls die Schlüsselgenerierung durch die CA erfolgt ist.

Das Teilfeld „publicationInfo“ kann von der CA verwendet werden, um dem Teilnehmer anzuzeigen, wo sein Zertifikat veröffentlicht worden ist. Die Struktur wurde bereits im Kapitel 4.3.1.1, „CertRequest“, beschrieben.

Bis auf das optionale Teilfeld „publicationInfo“ muß die Datenstruktur insgesamt unterstützt werden.

### 4.3.3 Datenstrukturen für Bestätigungsnachrichten

Der PKI-Rumpf einer Bestätigungsnachricht enthält wie bei der Zertifizierungsantwort die Datenstruktur „CertRepMessage“ (s. Kapitel 4.3.2, „Datenstrukturen für Zertifizierungsantworten“). Über den Statuswert gibt der Antragsteller an, ob er das Zertifikat in der Form akzeptiert, wie es von der CA generiert worden ist, oder ob er die Bereitstellung des Zertifikates ablehnt.<sup>32</sup>

Von der Datenstruktur „CertRepMessage“ wird allein das Feld „response“ mit den Teilfeldern „certReqlid“ und „status“ verwendet. Wie bei einer Zertifizierungsantwort müssen Informationen über den Status und der Wert zur Identifizierung des Zertifikates angegeben werden.

Falls der Teilnehmer einen zu zertifizierenden Verschlüsselungsschlüssel selbst generiert hat, muß er über die Bestätigungsnachricht den Besitz des privaten Schlüssels nachweisen. Für diesen Nachweis ist das Feld „protection“ mit einem speziellen Wert zu belegen. Dazu muß stets ein symmetrisches Verfahren mit einem speziellen Schlüssel verwendet werden.

Der Teilnehmer entschlüsselt mit seinem privaten Schlüssel, für den er ein Zertifikat beantragt hat, den Session-Key aus der Zertifizierungsantwort, mit dem das Zertifikat verschlüsselt wurde. Der Session-Key ist der Schlüssel, der für das symmetrische Verfahren zum Schutz der Bestätigungsnachricht verwendet wird.

## 4.4 Datenstrukturen für die Sperrung von Zertifikaten

Die Sperrung von Zertifikaten erfolgt durch einen Sperrantrag (Revocation Request) auf den die CA mit einer Antwortnachricht (Revocation Response) reagiert. Die CA kann den Teilnehmer durch eine Sperrnachricht (Revocation Notice) über eine erfolgte oder bevorstehende Sperrung informieren.

---

<sup>32</sup> Die anderen Spezifikationen sehen vor, daß der Rumpf einer Bestätigungsnachricht stets leer ist. Diese Bestätigungsnachricht („PKI Confirmation“) dient allein dem Zweck, den Nachweis der Besitzes zu ermöglichen. Im MailTrusT-Profil wurde die Zweckbestimmung der Bestätigungsnachricht erweitert.

#### 4.4.1 Sperrantrag

Ein Sperrantrag hat die folgende Syntax:

```
RevReqContent ::= SEQUENCE OF RevDetails

RevDetails ::= SEQUENCE {
    certDetails      CertTemplate,
    -- hier werden entweder nur die Seriennummer des Zertifikats
    -- (und im Fall indirekter CRLs der Name der ausstellenden CA)
    -- oder alle bekannten Angaben zum Zertifikat eingetragen,
    -- falls die Seriennummer nicht bekannt ist.
    revocationReason ReasonFlags OPTIONAL,
    -- hier kann der Rückrufgrund angegeben werden
    badSinceDate     GeneralizedTime OPTIONAL,
    -- hier kann der Ungültigkeitszeitpunkt eingetragen werden.
    crlEntryDetails  Extensions OPTIONAL
    -- gewünschte Extensions für den Sperreintrag }

```

Das zu sperrende Zertifikat wird in der Regel über die Seriennummer identifiziert. Die Angabe von Rückrufgrund und Ungültigkeitszeitpunkt sind optional. Die Datenstruktur muß bis auf das Feld „crlEntryDetails“ unterstützt werden. Entgegen der angegebenen Syntax bezieht sich ein Sperrantrag immer nur auf ein Zertifikat.

Der Sperrantrag wird durch ein symmetrisches oder asymmetrisches kryptographisches Verfahren authentisiert.

#### 4.4.2 Antwort auf Sperrantrag

Ein Sperrantrag wird von der CA mit einer Nachricht beantwortet, die folgenden Inhalt hat:

```
RevRepContent ::= SEQUENCE {
    status          SEQUENCE SIZE (1..MAX) OF PKIStatusInfo,
    -- nur Statusangabe zu einem Sperrantrag
    revCerts       [0] SEQUENCE SIZE (1..MAX) OF CertId OPTIONAL,
    -- ID für die der Rückruf gewünscht wurde
    crls           [1] SEQUENCE SIZE (1..MAX) OF CertificateList
                  OPTIONAL
    -- die resultierende(n) Sperrliste(n) }

```

Das Feld „status“ enthält einen Wert des Typs „PKIStatusInfo“ (s. Kapitel 4.2.3, „Status und Fehlerinformationen“). Wie bei Zertifizierungsantworten besteht die Beschränkung, daß immer nur ein Grund dafür angegeben werden darf, weshalb dem Antrag nicht entsprochen werden kann.

#### 4.4.3 Sperrnachricht

Wird ein Zertifikat nicht auf Wunsch des Zertifikats-Inhaber, sondern auf Initiative einer CA oder einer berechtigten Person gesperrt, kann der Zertifikats-Inhaber darüber benachrichtigt werden. Dazu wird eine Nachricht verwendet, die folgenden Inhalt hat:

```
RevAnnContent ::= SEQUENCE {
    status          PKIStatus,
    certId          CertId,
    willBeRevokedAt GeneralizedTime,
    badSinceDate   GeneralizedTime,
    crlDetails     Extensions OPTIONAL
    -- zusätzliche CRL-Angaben (z.B. Nummer der CRL, Rückrufgrund
    -- etc.) }

```

Sperrnachrichten müssen nicht unterstützt werden. Ihre Unterstützung wird jedoch empfohlen.

## 4.5 Datenstrukturen für die Initialisierung einer Root-CA

Eine Root-CA generiert für sich selbst ein selbstsigniertes Zertifikat. Alle Teilnehmer der PKI müssen sich davon überzeugen, daß dieses Zertifikat tatsächlich von der Root-CA stammt. Dies ist eine Voraussetzung für jede Verifikation.

Dies wird dadurch vereinfacht, daß die Root-CA einen Fingerprint (Hashwert) des Zertifikates generiert und allen Teilnehmern bekannt macht, so daß diese sich von der Authentizität des Zertifikates überzeugen können. MailTrusT-konforme Komponenten müssen die Generierung dieses Hashwertes bzw. den Abgleich mit diesem Werte unterstützen.

Für den Fingerprint wird folgende Syntax verwendet:

```
OOBCertHash ::= SEQUENCE {
    hashAlg      [0] AlgorithmIdentifier OPTIONAL
    certID       [1] CertID                OPTIONAL
    hashVal      BIT STRING }
```

Der Hashwert wird mittels des durch „hashAlg“ spezifizierten Algorithmus gebildet. Die Root-CA wird durch das Feld „certID“ identifiziert (s. Kapitel 4.2.4, „Identifikatoren für Zertifikate“).

Die Root-CA muß den Hashalgorithmus, den Identifikator und den Hashwert bekanntmachen, wenn auch nicht zwingend mittels der Datenstruktur „OOBCertHash“. In der Datenstruktur muß nur der Hashwert enthalten sein.

Der DER-kodierte Wert sollte stets hexadezimal kodiert werden, damit er in geeigneter Form veröffentlicht und manuell abgeglichen werden kann. Die Teilnehmer-Komponente muß den Wert deshalb ebenfalls in hexadezimal kodierter Form anzeigen.

Für das Wurzel-Zertifikat gelten folgende Besonderheiten:

- Die Signatur muß mittels der Informationen im Feld „subjectPublicKeyInfo“ verifizierbar sein, d.h. das Zertifikat ist selbstsigniert.
- Die Felder „issuer“ und „subject“ enthalten beide den DN der Root-CA.
- Die Werte aller Extensions des Zertifikates müssen für ein selbstsigniertes Zertifikat geeignet gewählt sein.

## 5 Profile für PKI-Nachrichten

Dieses Kapitel enthält die detaillierten Profile für die PKI-Nachrichten und PKI-Operationen, die von MailTrust-konformen Produkten unterstützt werden müssen.

### 5.1 Profile für Zertifizierungen

Es werden zwei Schemata für Zertifizierungen definiert, ein dezentrales und ein zentrales, die beide von MailTrust-konformen Komponenten unterstützt werden müssen.<sup>33</sup>

Das dezentrale Schema besitzt folgende Merkmale:

- Die erste PKI-Nachricht wird durch den Teilnehmer generiert.<sup>34</sup>
- Eine Authentisierung dieser Nachricht ist erforderlich.
- Die Schlüsselgenerierung erfolgt beim Teilnehmer.
- Eine Bestätigung der Antwort auf diese Nachricht wird gefordert.

Das dezentrale Schema umfaßt den folgenden Ablauf:

Teilnehmer		CA
Schlüsselgenerierung		
Generierung einer Zertifizierungsanfrage		
Schutz der Anfrage		
	→ Übermittlung der Anfrage →	
		Verifizierung der Anfrage
		Bearbeitung der Anfrage
		Generierung der Antwort
		Schutz der Antwort
	← Übermittlung der Antwort ←	
Verifizierung der Antwort		
Generierung der Bestätigung		
Schutz der Bestätigung		
	→Übermittlung der Bestätig.→	
		Verifizierung der Bestätig.

Tabelle 2: Ablauf beim dezentralen Schema

<sup>33</sup> Die anderen Spezifikationen enthalten nur ein Profil für das dezentrale Schema. Dies Profil wird von IETF und ISO als „basic authenticated scheme“ bezeichnet. Die IETF hat die Unterstützung dieses Schemas für verbindlich erklärt, die ISO hat die Entscheidung darüber, ob Schemata für verbindlich erklärt werden sollen, noch offen gelassen. Im NIST-Dokument wird eine Abart dieses Schemas für verbindlich erklärt, die „ORA-Generated Registration Requests“ genannt wird.

<sup>34</sup> Alle Aktionen der Teilnehmers können auch durch eine RA im Auftrag des Teilnehmers oder durch die CA erfolgen.

Das zentrale Schema besitzt folgende Merkmale:

- Die erste PKI-Nachricht wird durch den Teilnehmer generiert.
- Eine Authentisierung dieser Nachricht ist erforderlich.
- Die Schlüsselgenerierung erfolgt durch die CA.
- Eine Bestätigung der Antwort auf diese Nachricht wird gefordert.

Das zentrale Schema unterscheidet sich von dezentralen Schema allein bei der Schlüsselgenerierung.<sup>35</sup>

Das zentrale Schema umfaßt folgenden Ablauf:

Teilnehmer		CA
Generierung einer Zertifizierungsanfrage		
Schutz der Anfrage		
	→ Übermittlung der Anfrage →	
		Verifizierung der Anfrage
		Bearbeitung der Anfrage
		Schlüsselgenerierung
		Generierung der Antwort
		Schutz der Antwort
	← Übermittlung der Antwort ←	
Verifizierung der Antwort		
Generierung der Bestätigung		
Schutz der Bestätigung		
	→Übermittlung der Bestätig.→	
		Verifizierung der Bestätig.

Tabelle 3: Ablauf beim zentralen Schema

Die vorliegende Spezifikation enthält jeweils Profile für Erstanträge, Verlängerungsanträge und Neuanträge, die von MailTrust-konformen Komponenten unterstützt werden müssen.

Für alle Profile sind die folgenden Voraussetzungen, Regeln und Konventionen zu beachten:

- CA und Teilnehmer müssen über einen gemeinsamen symmetrischen Schlüssel verfügen, falls der Teilnehmer selbst einen Erstantrag stellen will.<sup>36</sup> Falls die RA/CA

<sup>35</sup> In der IETF-Spezifikation wurde ein zentrales Schema beschrieben, daß von dem hier beschriebenen Schema wesentlich abweicht. Dieses Schema sieht lediglich vor, daß eine einzige PKI-Nachricht von der CA an den Teilnehmer übermittelt wird. Dies hat z.B. den Nachteil, daß eine elektronische Beantragung eines Zertifikates nicht möglich ist und dem Teilnehmer nicht die Möglichkeit eingeräumt wird, das Zertifikat freizugeben, bevor es im Verzeichnis bereitgestellt wird.

<sup>36</sup> Dieser Schlüssel wird in der Regel während der Registrierung bei einer RA dem Teilnehmer übergeben. Auch die CA muß über diesen Schlüssel verfügen. Die Art und Weise der Austausches des Schlüssels zwischen RA und CA ist nicht Gegenstand der Spezifikation.

einen Antrag für den Teilnehmer stellt, verwendet sie nicht das Profil für einen Erstantrag, so daß diese Anträge stets digital signiert sein müssen.

- Der Teilnehmer darf das Profil für den Erstantrag nur dann verwenden, wenn er nicht bereits über einen gültigen Signaturschlüssel verfügt, mit dem er den Antrag digital signieren kann.
- Jeder Teilnehmer soll stets im Besitz eines gültigen Signaturschlüssel-Zertifikats sein. Bevor dieses Zertifikat ungültig wird, soll er ein neues Zertifikat beantragen, damit er stets in der Lage ist, PKI-Nachrichten durch eine digital Signatur zu schützen.
- Der Teilnehmer soll mit seinem Erstantrag stets ein Signaturschlüssel-Zertifikat beantragen. Falls er auch ein Verschlüsselungsschlüssel-Zertifikat benötigt, soll er dies anschließend mit einem Neuantrag anfordern, der mit dem zertifizierten Signaturschlüssel geschützt ist. Ohne gültiges Signaturschlüssel-Zertifikat kann der Teilnehmer nur Erstanträge stellen, da der IAK immer nur für einen Zertifizierungsantrag verwendet werden darf.
- Sobald der Teilnehmer mit der Zertifizierungsantwort auf seinen Erstantrag über ein Signaturschlüssel-Zertifikat verfügt, soll er den zertifizierten Schlüssel bereits zur Signatur der Bestätigungsnachricht verwenden.
- Die CA muß stets über einen gültigen Signaturschlüssel verfügen, mit dem sie Zertifizierungsantworten digital signiert.
- Der Teilnehmer muß digitale Signaturen der CA verifizieren können. Er muß deshalb auch im Besitz einer authentischen Kopie des Wurzelzertifikates sein.<sup>37</sup>
- Falls ein beantragtes Zertifikat nicht ausgestellt werden kann, muß eine neue Zertifizierungsanfrage gestellt werden. Es sind keine besonderen Abläufe für die Behandlung von Fehlerfällen vorgesehen.
- Optionale Felder, die in den Profilen nicht erwähnt werden, dürfen nur dann verwendet werden, wenn sie von der Empfänger-Komponente unterstützt werden. Die Empfänger-Komponente darf andernfalls eine PKI-Nachricht als syntaktisch unkorrekt zurückweisen. Andere optionale Felder darf die Empfänger-Komponente ignorieren. Die PKI-Nachricht darf in diesem Fall jedoch nicht als syntaktisch unkorrekt zurückgewiesen werden.
- Die Identifizierer für Algorithmen werden im Dokument „Algorithmen“ beschrieben. Im Profil wird nur der Typ des Algorithmus angegeben, d.h. MSG\_MAC\_ALG für den MAC-Algorithmus und MSG\_SIG\_ALG für den Signatur-Algorithmus.
- Falls in einem Feld ein X.500-Distinguished-Name angegeben werden muß, ein solcher jedoch nicht zur Verfügung steht, muß der „NULL-DN“ verwendet werden.<sup>38</sup> Der Null-DN ist eine „SEQUENCE OF RelativeDistinguishedNames“ der Länge Null (DER-Kodierung: '3000'H).
- In den Profilen wird zur eindeutigen Identifikation der Teilfelder eine „Punkt“-Notation verwendet, wobei z.B. „certTemplate.subject“ das Teilfeld „subject“ des Feldes „certTemplate“ identifiziert.

### 5.1.1 Profile für das dezentrale Schema

Für das dezentrale Schema sind Profile für Erstanträge, Verlängerungsanträge und Neuanträge zu unterstützen.

---

<sup>37</sup> Die Art und Weise, wie er den Besitz erlangt, ist nicht Gegenstand der Spezifikation.

<sup>38</sup> Falls der Teilnehmer z.B. seinen DN noch nicht kennt.

### 5.1.1.1 Profil für Erstanträge

Die folgende Tabelle gibt die Belegung der Felder einer Zertifizierungsanfrage mit Werten wieder:

Feld	Wert	Kommentar
recipient	DN der CA	Der X.500-Distinguished-Name der CA, die das Zertifikat erstellen soll
protectionAlg	MSG_MAC_ALG	Es ist nur der symmetrische Algorithmus erlaubt, der den IAK verwendet
senderKID	referenceNum	Die Referenznummer, die dem Teilnehmer bei der Registrierung zusammen mit dem IAK ausgehändigt wurde
transactionID	vorhanden	Ein implementationsspezifischer Wert, der für den Antragsteller von Bedeutung ist <sup>39</sup>
senderNonce	vorhanden	128-Bits, die zufällig gewählt werden müssen
body	ir (CertReqMessages)	Nur eine „CertReqMsg“ erlaubt
certReqMsg	vorhanden	CertReqMsg wird im folgenden als crm bezeichnet
crm.certReq. certReqID	fester Wert („0“)	Der Index wird für das Profil zwar nicht benötigt, das Feld ist jedoch nicht optional
crm.certReq. certTemplate	Zertifikats-Schablone muß vorhanden sein	Die Zertifikats-Schablone muß mindestens den öffentliche Schlüssel enthalten <sup>40</sup>
crm.certReq. controls. publicationInfo	optional vorhanden	Muß vorhanden sein, wenn das Zertifikat nicht veröffentlicht werden soll
crm.certReq. controls. protocolEncKey	optional vorhanden	Muß vorhanden sein, wenn ein Signaturschlüsselzertifikat beantragt wird, das nicht veröffentlicht werden soll <sup>41</sup>
crm.pop. signature	optional vorhanden	Die Datenstruktur dient dem Nachweis des Besitzes des Signaturschlüssels <sup>42</sup>
protection	vorhanden	Der Schutz muß mittels MSG_MAC_ALG erfolgen

Tabelle 4: Zertifizierungsanfrage bei Erstanträgen

Die folgende Tabelle gibt die Belegung der Felder einer Zertifizierungsantwort mit Werten wieder:

- 
- <sup>39</sup> Der Antragsteller hat selbst dafür zu sorgen, daß er eindeutige Transaktionsnummern vergibt, um mehrere Transaktionen voneinander unterscheiden zu können. Eine Prüfung der Transaktionsnummer durch die CA ist nicht erforderlich.
- <sup>40</sup> Weitere Vorgaben für die Schablone bleiben der Policy überlassen. Zu beachten ist, daß die Schablone nicht gegen unbefugte Kenntnisnahme gesichert wird.
- <sup>41</sup> Dient der Verschlüsselung des durch die CA generierten Zertifikates. Im Falle eines Zertifikats für einen Verschlüsselungsschlüssel wird das Zertifikat mit dem Schlüssel selbst verschlüsselt.
- <sup>42</sup> Bei Verschlüsselungsschlüsseln erfolgt der Nachweis durch die Bestätigungsnachricht.

Feld	Wert	Kommentar
sender	DN der CA	Der X.500-Distinguished-Name der CA, die die Nachricht generiert hat
protectionAlg	MSG_SIG_ALG <sup>43</sup>	Es muß der Signaturalgorithmus verwendet werden
recipKID	referenceNum	Die Referenznummer, die dem Teilnehmer bei der Registrierung zusammen mit dem IAK ausgehändigt wurde
transactionID	vorhanden	Der Wert aus der Zertifizierungsanfrage
senderNonce	vorhanden	128-Bits, die zufällig gewählt werden müssen
recipNonce	vorhanden	Der Wert des Feldes „senderNonce“ der Zertifizierungsanfrage
body	ip (CertRepMessage)	Nur eine Antwort erlaubt
CertResponse	vorhanden	CertResponse wird im folgenden als crc bezeichnet
crc.certReqID	fester Wert ( „0“)	Gleicher Wert wie in Anfrage
crc.status. status	vorhanden	Zur Anzeige eines positiven Ergebnisses ist der Wert „granted“, bei negativem Ergebnis der Wert „rejection“ zu verwenden
crc.status. failInfo	optional vorhanden	Kann optional vorhanden sein, wenn der Statuswert „rejection“ ist <sup>44</sup>
crc.certified KeyPair	bei positivem Ergebnis vorhanden	Muß dann und darf nur dann vorhanden sein, wenn der Statuswert „granted“ ist
certificate	abhängig von „publicationInfo“ und Art des Schlüssels	Muß dann und darf nur dann vorhanden sein, wenn das Zertifikat für einen zu veröffentlichenden Signaturschlüssel beantragt wurde. <sup>45</sup>
encryptedCert	abhängig von „publicationInfo“ und Art des Schlüssels	Muß dann und darf nur dann vorhanden sein, falls das Zertifikat nicht veröffentlicht werden soll und/oder ein Verschlüsselungsschlüssel zertifiziert wurde.
protection	vorhanden	Der Schutz erfolgt durch MSG_SIG_ALG

Tabelle 5: Zertifizierungsantwort bei Erstanträgen

<sup>43</sup> Die anderen Spezifikationen lassen nur MSG\_MAC\_ALG zu. Die MailTrustT-Spezifikation bevorzugt asymmetrische Verfahren zum Schutz der PKI-Nachrichten gegenüber symmetrischen. Symmetrische Verfahren sollen daher immer nur dann verwendet werden, wenn der Schutz nicht auch durch asymmetrische Verfahren erreicht werden kann.

Der Teilnehmer muß bereits in der Lage sein, die digitale Signatur der CA für Zertifikate zu verifizieren, wenn er eine Zertifizierungsantwort prüft. Deshalb kann und muß die PKI-Nachricht durch eine digitale Signatur geschützt werden.

<sup>44</sup> Im Gegensatz zum IETF-Profil muß das Bit für „PKIFailureInfo“ nicht angegeben werden, da es nicht unterstützt werden muß.

<sup>45</sup> Die anderen Spezifikationen lassen es zu, daß ein Zertifikat, das nicht veröffentlicht werden soll, unverschlüsselt übertragen wird. Das MailTrustT-Profil verbietet dies.

Die folgende Tabelle gibt die Belegung der Felder einer Bestätigungsnachricht mit Werten wieder:

Feld	Wert	Kommentar
sender	DN des Teilnehmers oder NULL-DN	Der X.500-Distinguished-Name aus dem für den Teilnehmer generierten Zertifikat soll bevorzugt verwendet werden <sup>46</sup>
recipient	DN der CA	Der X.500-Distinguished-Name der CA
protectionAlg	MSG_SIG_ALG oder MSG_MAC_ALG	Der Signaturalgorithmus soll bevorzugt verwendet werden, falls ein Signaturschlüsselzertifikat beantragt wird. <sup>47</sup>
senderKID	referenceNum, falls NULL-DN im Feld „sender“	Die Referenznummer, die dem Teilnehmer bei der Registrierung zusammen mit dem IAK ausgehändigt wurde
transactionID	vorhanden	Der Wert aus Zertifizierungsanfrage und Zertifizierungsantwort
senderNonce	vorhanden	Der Wert der „senderNonce“ der Zertifizierungsanfrage
recipNonce	vorhanden	Der Wert der „senderNonce“ der Zertifizierungsantwort
body	ip (CertRepMessage)	Nur eine Bestätigungsnachricht erlaubt
CertResponse	vorhanden	CertResponse wird im folgenden als crc bezeichnet
crc.certReqID	fester Wert ( „0“)	Gleicher Wert wie in Anfrage und Antwort
crc.status. status	vorhanden	Zur Anzeige eines positiven Ergebnisses ist der Wert „granted“, bei negativem Ergebnis der Wert „rejection“ zu verwenden
crc.status. failInfo	optional vorhanden	Kann optional vorhanden sein, wenn der Statuswert „rejection“ ist
protection	vorhanden	Der Schutz erfolgt mittels MSG_SIG_ALG oder MSG_MAC_ALG

Tabelle 6: Bestätigungsnachricht bei Erstanträgen

### 5.1.1.2 Profil für Verlängerungsanträge

Der Teilnehmer kann einen Folgeantrag statt eines Erstantrages stellen, wenn er über einen gültigen Signaturschlüssel verfügt. Er benötigt in diesem Fall keinen IAK, um Nachrichten zu schützen.

<sup>46</sup> Das Feld wird in den anderen Profile nicht verwendet, da dort stets das symmetrische Verfahren zu Sicherung der Nachricht verwendet und der Absender über den Referenzwert identifiziert wird. Der Teilnehmer soll die Bestätigungsnachricht digital signieren, falls er ein Signaturschlüssel-Zertifikat beantragt hat, daß er mit der Bestätigungsnachricht akzeptieren will.

<sup>47</sup> Der Teilnehmer soll die Bestätigungsnachricht digital signieren, falls er ein Signaturschlüssel-Zertifikat beantragt hat, das er mit der Bestätigungsnachricht akzeptieren will. Falls ein Zertifikat für einen Entschlüsselungsschlüssel beantragt wird, muß der MAC-Algorithmus verwendet werden, um den Besitz des privaten Schlüssels nachzuweisen.

Ein Verlängerungsantrag ist ein Folgeantrag, bei dem kein Schlüsselwechsel erfolgt. Die folgende Tabelle gibt die Belegung der Felder einer Zertifizierungsanfrage mit Werten wieder:

Feld	Wert	Kommentar
sender	DN des Teilnehmers	Der X.500-Distinguished-Name des Teilnehmers
recipient	DN der CA	Der X.500-Distinguished-Name der CA
protectionAlg	MSG_SIG_ALG	Es muß der Signaturalgorithmus verwendet werden
transactionID	vorhanden	Ein implementationsspezifischer Wert, der für den Antragsteller von Bedeutung ist
senderNonce	vorhanden	128-Bits, die zufällig gewählt werden müssen
body	cr (CertReqMessages)	Nur eine „CertReqMsg“ erlaubt
certReqMsg	vorhanden	CertReqMsg wird im folgenden als crm bezeichnet
crm.certReq. certReqID	fester Wert („0“)	Der Index wird für das Profil zwar nicht benötigt, das Feld ist jedoch nicht optional
crm.certReq. certTemplate	Zertifikats-Schablone muß vorhanden sein	Die Zertifikats-Schablone muß mindestens den öffentliche Schlüssel enthalten sein
crm.certReq. controls. publicationInfo	optional vorhanden	Muß vorhanden sein, wenn das Zertifikat nicht veröffentlicht werden soll
crm.certReq. controls. protocolEncKey	optional vorhanden	Muß vorhanden sein, wenn ein Signaturschlüsselzertifikat beantragt wird, das nicht veröffentlicht werden soll
crm.pop. signature	optional vorhanden	Die Datenstruktur dient dem Nachweis des Besitzes des Signaturschlüssels
protection	vorhanden	Der Schutz muß mittels MSG_SIG_ALG erfolgen

Tabelle 7: Zertifizierungsanfrage bei Verlängerungsanträgen

Die folgende Tabelle gibt die Belegung der Felder einer Zertifizierungsantwort mit Werten wieder:

Feld	Wert	Kommentar
sender	DN der CA	Der X.500-Distinguished-Name der CA, die die Nachricht generiert hat
recipient	DN des Teilnehmers	Der X.500-Distinguished-Name des Teilnehmers
protectionAlg	MSG_SIG_ALG	Es muß der Signaturalgorithmus verwendet werden
transactionID	vorhanden	Der Wert aus der Zertifizierungsanfrage
senderNonce	vorhanden	128-Bits, die zufällig gewählt werden müssen
recipNonce	vorhanden	Der Wert des Feldes „senderNonce“ der Zertifizierungsanfrage
body	cp (CertRepMessage)	Nur eine Antwort erlaubt
CertResponse	vorhanden	CertResponse wird im folgenden als crc bezeichnet
crc.certReqID	fester Wert („0“)	Gleicher Wert wie in Anfrage
crc.status. status	vorhanden	Zur Anzeige eines positiven Ergebnisses ist der Wert „granted“, bei negativem Ergebnis der Wert „rejection“ zu verwenden
crc.status. failInfo	optional vorhanden	Kann optional vorhanden sein, wenn der Statuswert „rejection“ ist
crc.certified KeyPair	bei positivem Ergebnis vorhanden	Muß dann und darf nur dann vorhanden sein, wenn der Statuswert „granted“ ist
certificate	abhängig von „publicationInfo“ und Art des Schlüssels	Muß dann und darf nur dann vorhanden sein, wenn das Zertifikat für einen zu veröffentlichenden Signaturschlüssel beantragt wurde
encryptedCert	abhängig von „publicationInfo“ und Art des Schlüssels	Muß dann und darf nur dann vorhanden sein, falls das Zertifikat nicht veröffentlicht werden soll und/oder ein Verschlüsselungsschlüssel zertifiziert wurde
protection	vorhanden	Der Schutz erfolgt durch MSG_SIG_ALG

Tabelle 8: Zertifizierungsantwort bei Verlängerungsanträgen

Die folgende Tabelle gibt die Belegung der Felder einer Bestätigungsnachricht mit Werten wieder:

Feld	Wert	Kommentar
sender	DN des Teilnehmers	Der X.500-Distinguished-Name des Teilnehmers
recipient	DN der CA	Der X.500-Distinguished-Name der CA
protectionAlg	MSG_SIG_ALG oder MSG_MAC_ALG	Der Signaturalgorithmus wird verwendet, falls ein Signaturschlüsselzertifikat beantragt wird, andernfalls der MAC-Algorithmus <sup>48</sup>
transactionID	vorhanden	Der Wert aus Zertifizierungsanfrage und Zertifizierungsantwort
senderNonce	vorhanden	Der Wert der „senderNonce“ der Zertifizierungsanfrage
recipNonce	vorhanden	Der Wert der „senderNonce“ der Zertifizierungsantwort
body	cp (CertRepMessage)	Nur eine Bestätigungsnachricht erlaubt
CertResponse	vorhanden	CertResponse wird im folgenden als crc bezeichnet
crc.certReqID	fester Wert („0“)	Gleicher Wert wie in Anfrage und Antwort
crc.status. status	vorhanden	Zur Anzeige eines positiven Ergebnisses ist der Wert „granted“, bei negativem Ergebnis der Wert „rejection“ zu verwenden
crc.status. failInfo	optional vorhanden	Kann optional vorhanden sein, wenn der Statuswert „rejection“ ist
protection	vorhanden	Der Schutz erfolgt mittels MSG_SIG_ALG oder MSG_MAC_ALG

Tabelle 9: Bestätigungsnachricht bei Verlängerungsanträgen

### 5.1.1.3 Profil für Neuanträge

Der Teilnehmer muß einen Neuantrag stellen, wenn er die Zertifizierung eines neuen Schlüssels durch die CA beantragt, der den bisher verwendeten Schlüssel ersetzen soll. Dadurch unterscheidet sich der Neuantrag vom Verlängerungsantrag. Der Teilnehmer muß auch dann einen Neuantrag stellen, wenn er ein Zertifikat für einen zusätzlichen Schlüssel beantragt.

Gegenüber dem Profil für Verlängerungsanträge ergibt sich nur die Änderungen, daß der Typ des Body „kur“ oder „kup“ ist.

<sup>48</sup> Der MAC-Algorithmus muß verwendet werden, um den Besitz des privaten Entschlüsselungsschlüssels nachzuweisen. Obwohl der Besitz bereits einmal nachgewiesen wurde muß dieser Nachweis wiederholt werden, um folgendem Angriff zu begegnen.

Ein Angreifer ist zwar im Besitz des Signaturschlüssels, aber nicht im Besitz des Entschlüsselungsschlüssels. Der Angreifer beantragt ein neues Zertifikat für den Entschlüsselungsschlüssel. Die fremde Identität täuscht er durch eine digitale Signatur vor.

## 5.1.2 Profile für das zentrale Schema

Für das zentrale Schema sind ebenfalls drei Profile vorgesehen. In den Profilen wurde berücksichtigt, daß der durch die CA generierte private Schlüssel und das Zertifikat nicht stets Teil der Zertifizierungsantwort sind, sondern z.B. auf einer Chipkarte dem Teilnehmer übermittelt werden.

### 5.1.2.1 Profil für Erstanträge

Die folgende Tabelle gibt die Belegung der Felder einer Zertifizierungsanfrage mit Werten wieder:

Feld	Wert	Kommentar
recipient	DN der CA	Der X.500-Distinguished-Name der CA, die das Zertifikat erstellen soll
protectionAlg	MSG_MAC_ALG	Es ist nur der symmetrische Algorithmus erlaubt, der den IAK verwendet
senderKID	referenceNum	Die Referenznummer, die dem Teilnehmer bei der Registrierung zusammen mit dem IAK ausgehändigt wurde
transactionID	vorhanden	Ein implementationsspezifischer Wert, der für den Antragsteller von Bedeutung ist
senderNonce	vorhanden	128-Bits, die zufällig gewählt werden müssen
body	ir (CertReqMessages)	Nur eine „CertReqMsg“ erlaubt
certReqMsg	vorhanden	CertReqMsg wird im folgenden als crm bezeichnet
crm.certReq. certReqID	fester Wert („0“)	Der Index wird für das Profil zwar nicht benötigt, das Feld ist jedoch nicht optional
crm.certReq. certTemplate	optional vorhanden	Es bleibt der Policy überlassen, ob die Zertifikats-Schablone vorhanden sein muß
crm.certReq. controls. publicationInfo	optional vorhanden	Muß vorhanden sein, wenn das Zertifikat nicht veröffentlicht werden soll
crm.certReq. controls. protocolEncKey	optional vorhanden	Dient der Verschlüsselung des durch die CA generierten privaten Schlüssels und des Zertifikats, wenn es nicht veröffentlicht werden soll <sup>49</sup>
protection	vorhanden	Der Schutz muß mittels MSG_MAC_ALG erfolgen

Tabelle 10: Zertifizierungsanfrage bei Erstanträgen

<sup>49</sup> Muß nicht vorhanden sein, wenn Schlüssel und Zertifikat nicht in der Antwort übertragen werden sollen, weil sie z.B. auf einer Chipkarte gespeichert werden.

Die folgende Tabelle gibt die Belegung der Felder einer Zertifizierungsantwort mit Werten wieder:

Feld	Wert	Kommentar
sender	DN der CA	Der X.500-Distinguished-Name der CA
protectionAlg	MSG_SIG_ALG	Es muß der Signaturalgorithmus verwendet werden
recipKID	referenceNum	Die Referenznummer, die dem Teilnehmer bei der Registrierung zusammen mit dem IAK ausgehändigt wurde
transactionID	vorhanden	Der Wert aus der Zertifizierungsanfrage
senderNonce	vorhanden	128-Bits, die zufällig gewählt werden müssen
recipNonce	vorhanden	Der Wert des Feldes „senderNonce“ der Zertifizierungsanfrage
body	ip (CertRepMessage)	Nur eine Antwort erlaubt
CertResponse	vorhanden	CertResponse wird im folgenden als crc bezeichnet
crc.certReqID	fester Wert ( „0“)	Gleicher Wert wie in Anfrage
crc.status. status	vorhanden	Zur Anzeige eines positiven Ergebnisses ist der Wert „granted“, bei negativem Ergebnis der Wert „rejection“ zu verwenden
crc.status. failInfo	optional vorhanden	Kann optional vorhanden sein, wenn der Statuswert „rejection“ ist
crc.certified KeyPair	optional vorhanden	Muß dann und darf nur dann vorhanden sein, falls der Statuswert „granted“ ist und ein Zertifikat Teil der Antwort sein soll
certificate	optional vorhanden	Darf nicht vorhanden sein, falls das Zertifikat nicht veröffentlicht werden soll
encryptedCert	optional vorhanden	Muß vorhanden sein, falls das Zertifikat nicht veröffentlicht werden und Teil der Nachricht sein soll
privateKey	optional vorhanden	Der von der CA generierte verschlüsselte private Schlüssel, falls er Teil der Nachricht sein soll
protection	vorhanden	Der Schutz erfolgt durch MSG_SIG_ALG

Tabelle 11: Zertifizierungsantwort bei Erstanträgen

Die folgende Tabelle gibt die Belegung der Felder einer Bestätigungsnachricht mit Werten wieder:

Feld	Wert	Kommentar
sender	DN des Teilnehmers oder NULL-DN	Der X.500-Distinguished-Name aus dem für den Teilnehmer generierten Zertifikat soll bevorzugt verwendet werden <sup>50</sup>
recipient	DN der CA	Der X.500-Distinguished-Name der CA
protectionAlg	MSG_SIG_ALG oder MSG_MAC_ALG	Der Signaturalgorithmus soll bevorzugt verwendet werden. <sup>51</sup>
senderKID	referenceNum, falls NULL-DN im Feld „sender“	Die Referenznummer, die dem Teilnehmer bei der Registrierung zusammen mit dem IAK ausgehändigt wurde
transactionID	vorhanden	Der Wert aus Zertifizierungsanfrage und Zertifizierungsantwort
senderNonce	vorhanden	Der Wert der „senderNonce“ der Zertifizierungsanfrage
recipNonce	vorhanden	Der Wert der „senderNonce“ der Zertifizierungsantwort
body	ip (CertRepMessage)	Nur eine Bestätigungsnachricht erlaubt
CertResponse	vorhanden	CertResponse wird im folgenden als crc bezeichnet
crc.certReqID	fester Wert („0“)	Gleicher Wert wie in Anfrage und Antwort
crc.status. status	vorhanden	Zur Anzeige eines positiven Ergebnisses ist der Wert „granted“, bei negativem Ergebnis der Wert „rejection“ zu verwenden
crc.status. failInfo	optional vorhanden	Kann optional vorhanden sein, wenn der Statuswert „rejection“ ist
protection	vorhanden	Der Schutz erfolgt mittels MSG_SIG_ALG oder MSG_MAC_ALG

Tabelle 12: Bestätigungsnachricht bei Erstanträgen

<sup>50</sup> Das Feld wird in den anderen Profilen nicht verwendet, da dort stets das symmetrische Verfahren zur Sicherung der Nachricht verwendet und der Absender über den Referenzwert identifiziert wird. Der Teilnehmer soll die Bestätigungsnachricht digital signieren, falls er ein Signaturschlüssel-Zertifikat beantragt hat, daß er mit der Bestätigungsnachricht akzeptieren will.

<sup>51</sup> Der Teilnehmer soll die Bestätigungsnachricht digital signieren, falls er ein Signaturschlüssel-Zertifikat beantragt hat, daß er mit der Bestätigungsnachricht akzeptieren will.

### 5.1.2.2 Profil für Verlängerungsanträge

Die folgende Tabelle gibt die Belegung der Felder einer Zertifizierungsanfrage mit Werten wieder:

Feld	Wert	Kommentar
sender	DN des Teilnehmers	Der X.500-Distinguished-Name des Teilnehmers
recipient	DN der CA	Der X.500-Distinguished-Name der CA
protectionAlg	MSG_SIG_ALG	Es muß der Signaturalgorithmus verwendet werden
transactionID	vorhanden	Ein implementationsspezifischer Wert, der für den Antragsteller von Bedeutung ist
senderNonce	vorhanden	128-Bits, die zufällig gewählt werden müssen
body	cr (CertReqMessages)	Nur eine „CertReqMsg“ erlaubt
certReqMsg	vorhanden	CertReqMsg wird im folgenden als crm bezeichnet
crm.certReq. certReqID	fester Wert („0“)	Der Index wird für das Profil zwar nicht benötigt, das Feld ist jedoch nicht optional
crm.certReq. certTemplate	Zertifikats-Schablone muß vorhanden sein	In der Zertifikats-Schablone muß mindestens der öffentliche Schlüssel enthalten sein
crm.certReq. controls. publicationInfo	optional vorhanden	Muß vorhanden sein, wenn das Zertifikat nicht veröffentlicht werden soll
crm.certReq. controls. protocolEncKey	optional vorhanden	Dient der Verschlüsselung des Zertifikats, wenn es nicht veröffentlicht werden soll
protection	vorhanden	Der Schutz muß mittels MSG_SIG_ALG erfolgen

Tabelle 13: Zertifizierungsanfrage bei Verlängerungsanträgen

Die folgende Tabelle gibt die Belegung der Felder einer Zertifizierungsantwort mit Werten wieder:

Feld	Wert	Kommentar
sender	DN der CA	Der X.500-Distinguished-Name der CA, die die Nachricht generiert hat
recipient	DN des Teilnehmers	Der X.500-Distinguished-Name des Teilnehmers
protectionAlg	MSG_SIG_ALG	Es muß der Signaturalgorithmus verwendet werden
transactionID	vorhanden	Der Wert aus der Zertifizierungsanfrage
senderNonce	vorhanden	128-Bits, die zufällig gewählt werden müssen
recipNonce	vorhanden	Der Wert des Feldes „senderNonce“ der Zertifizierungsanfrage
body	cp (CertRepMessage)	Nur eine Antwort erlaubt
CertResponse	vorhanden	CertResponse wird im folgenden als crc bezeichnet
crc.certReqID	fester Wert („0“)	Gleicher Wert wie in Anfrage
crc.status. status	vorhanden	Zur Anzeige eines positiven Ergebnisses ist der Wert „granted“, bei negativem Ergebnis der Wert „rejection“ zu verwenden
crc.status. failInfo	optional vorhanden	Kann optional vorhanden sein, wenn der Statuswert „rejection“ ist
crc.certified KeyPair	bei positivem Ergebnis vorhanden	Muß dann und darf nur dann vorhanden sein, wenn der Statuswert „granted“ ist
certificate	optional vorhanden	Darf nur vorhanden sein, wenn das Zertifikat veröffentlicht werden soll
encryptedCert	optional vorhanden	Muß vorhanden sein, falls das Zertifikat nicht veröffentlicht werden und in der Antwort enthalten sein soll
protection	vorhanden	Der Schutz erfolgt durch MSG_SIG_ALG

Tabelle 14: Zertifizierungsantwort bei Verlängerungsanträgen

Die folgende Tabelle gibt die Belegung der Felder einer Bestätigungsnachricht mit Werten wieder:

Feld	Wert	Kommentar
sender	DN des Teilnehmers	Der X.500-Distinguished-Name des Teilnehmers
recipient	DN der CA	Der X.500-Distinguished-Name der CA
protectionAlg	MSG_SIG_ALG	Es muß der Signaturalgorithmus verwendet werden
transactionID	vorhanden	Der Wert aus Zertifizierungsanfrage und Zertifizierungsantwort
senderNonce	vorhanden	Der Wert der „senderNonce“ der Zertifizierungsanfrage
recipNonce	vorhanden	Der Wert der „senderNonce“ der Zertifizierungsantwort
body	cp (CertRepMessage)	Nur eine Bestätigungsnachricht erlaubt
CertResponse	vorhanden	CertResponse wird im folgenden als crc bezeichnet
crc.certReqID	fester Wert („0“)	Gleicher Wert wie in Anfrage und Antwort
crc.status. status	vorhanden	Zur Anzeige eines positiven Ergebnisses ist der Wert „granted“, bei negativem Ergebnis der Wert „rejection“ zu verwenden
crc.status. failInfo	optional vorhanden	Kann optional vorhanden sein, wenn der Statuswert „rejection“ ist
protection	vorhanden	Der Schutz erfolgt mittels MSG_SIG_ALG

Tabelle 15 Bestätigungsnachricht bei Verlängerungsanträgen

### 5.1.2.3 Profil für Neuanträge

Gegenüber dem Profil für Verlängerungsanträge ergeben sich folgende Änderungen:

- Der Typ des Body ist „kur“ oder „kup“
- Der „protocolEncKey“ wird auch zur Verschlüsselung des durch die CA generierten privaten Schlüssels verwendet. Er muß deshalb im Feld „crm.certReq.controls.protocolEncKey“ des Antrags stets vorhanden sein, wenn der Schlüssel transportiert werden soll.
- Das Feld „privateKey“ muß in der Antwort enthalten sein, wenn der Schlüssel transportiert werden soll.

## 5.2 Profile für Sperrungen

Die Profile für Sperrungen bestehen aus einem Profil für den Sperrantrag (Revocation Request) und einem Profil für die Antwort auf den Sperrantrag (Revocation Response).

### 5.2.1 Profil für Sperrantrag

Ein Sperrantrag kann entweder digital signiert oder mit einem symmetrischen Verfahren gesichert werden.

Die folgende Tabelle gibt die Belegung der Felder des Revocation Request mit Werten wieder:

Feld	Wert	Kommentar
sender	DN des Teilnehmers oder NULL-DN	Der DN der Teilnehmers, falls MSG_SIG_ALG verwendet wird
recipient	DN der CA	Der X.500-Distinguished-Name der CA, die die Sperrung durchführen soll
protectionAlg	MSG_MAC_ALG oder MSG_SIG_ALG	Es ist der symmetrische Algorithmus zu verwenden, falls der Antrag nicht digital signiert werden kann <sup>52</sup>
senderKID	referenceNum falls NULL-DN in „sender“	Die Referenznummer bei Verwendung des symmetrischen Algorithmus
transactionID	vorhanden	Ein implementationsspezifischer Wert, der für den Antragsteller von Bedeutung ist
senderNonce	vorhanden	128-Bits, die zufällig gewählt werden müssen
body	rr (RevReqContent)	Profil erlaubt nur den Rückruf eines Zertifikats pro Antrag
RevDetails	vorhanden	RevDetails wird im folgenden als rd bezeichnet
rd.certDetails	vorhanden	Angabe von Daten, die das zu sperrende Zertifikat eindeutig identifizieren
rd.revocation Reason	optional vorhanden	Optionale Angabe des Rückrufgrundes
rd.badSince Date	optional vorhanden	Optionale Angabe des Ungültigkeitszeitpunkts (invalidityDate)
protection	vorhanden	Der Schutz muß durch MSG_MAC_ALG oder MSG_SIG_ALG erfolgen

Tabelle 16: Profil für Sperrantrag

<sup>52</sup> Der Antrag soll auch dann digital signiert werden, wenn ein dringlicher Sperrgrund vorliegt. Auch im Falle der Kompromittierung des Signaturschlüssels besteht nicht die Gefahr eines „denial of service“-Angriffes.

### 5.2.2 Profil für die Antwort auf einen Sperrantrag

Die folgende Tabelle gibt die Belegung der Felder für die Antwort auf einen Sperrantrag mit Werten wieder:

Feld	Wert	Kommentar
sender	DN der CA	Der X.500-Distinguished-Name der CA
recipient	DN des Teilnehmers oder Null-DN	Der DN des Teilnehmers, falls er den Antrag digital signiert hat
protectionAlg	MSG_SIG_ALG	Es muß ein Signaturalgorithmus verwendet werden
recipKID	referenceNum falls NULL-DN in „sender“	Die Referenznummer bei Verwendung des symmetrischen Algorithmus für den Antrag
transactionID	vorhanden	Der Wert aus dem Antrag
recipNonce	vorhanden	Der Wert des Feldes „senderNonce“ des Antrags
body	cp (RevRepContent)	RevRepContent wird im folgenden als cp bezeichnet
cp.status.status	vorhanden	Zur Anzeige eines positiven Ergebnisses ist der Wert „granted“, bei negativem Ergebnis der Wert „rejection“ zu verwenden
cp.status.failInfo	optional vorhanden	Kann optional vorhanden sein, wenn der Statuswert „rejection“ ist
cp.crls	optional vorhanden	Sperrlisten können optional beigefügt werden
protection	vorhanden	Der Schutz erfolgt durch MSG_SIG_ALG

Tabelle 17: Profil für Antwort auf Sperrantrag

## 6 Literaturverzeichnis

- [Chokhani] A Security Flaw in the X.509 Standard; S. Chokhani; CygnaCom Solutions, Inc.; <http://www.cygnacom.com>
- [CCITT X.208 88] CCITT X.208: Specification of Abstract Syntax Notation One (ASN.1); 1988
- [ISO WD 15945 98] Specification of TTP Services to support the Application of Digital Signatures; ISO/IEC WD 15945; 26.06.1998
- [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.509 97] ITU-T X.509: Information Technology - Open Systems Interconnection - The Directory: Authentication Framework; 1997
- [ITU-T X.520 95] ITU-T X.521: Information Technology - Open Systems Interconnection - The Directory: Selected Attribute Types; 1995
- [ITU-T X.681 94] ITU-T X.681: Information Technology - ASN.1 Encoding Rules - Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER); 1994
- [ITU-T X.690 94] ITU-T X.690: Information Technology - Abstract Syntax Notation One (ASN.1): Information object specification; 1994
- [MISPC 97] Minimum Interoperability Specification for PKI Components; W. Burr; D. Dodson, N. Nazario, W. Polk; Version 1, 03.09.1997
- [MTRUST 96] F. Bauspieß: MailTrusT-Spezifikation; Version 1.1; Dezember 1996
- [PKCS7 93] RSA Laboratories: The Public-Key Cryptography Standards (PKCS), RSA Data Security Inc.; Redwood City, California; November 1993
- [PKCS10 93] RSA Laboratories: The Public-Key Cryptography Standards (PKCS), RSA Data Security Inc., Redwood City, California, November 1993
- [PKIX-CMP 98] C. Adams, S. Farrell: Internet X.509 Public Key Infrastructure - Certificate Management Protocols; Mai 1998
- [PKIX-CRMF 98] M. Myers, C. Adams, D. Solo, D. Kemp: Internet X.509 Certificate Request Message Format; Mai 1998
- [PKIX-PRO 98] R. Housely, W. Ford, W. Polk, D. Solo: Internet X.509 Public Key Infrastructure - X.509 Certificate and CRL Profile; September 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
- [RFC 1422 93] S. Kent: Privacy Enhancement for Internet Electronic Mail: Part II: Certificate-Based Key Management; Internet Engineering Task Force; RFC1422; Februar 1993.
- [SIGI-ZERT 99] Signatur-Interoperabilitätsspezifikation (SigI); Zertifikate; Version 3.0; 31. Januar. 1999