# Public Key Service

## Schnittstellenbeschreibung OCSP-Responder

Version: 2.1
Stand: 19.09.2018
Status: Freigegeben

Business flexibility T · · Systems · · ·

# Impressum

**Herausgeber**

T-Systems International GmbH

| Dateiname | Dokumentennummer | Dokumentenbezeichnung |
|---|---|---|
| PKS OCSP-Responder v2.1.docx | | |

| Version | Stand | Status |
|---|---|---|
| 2.1 | 19.09.2018 | Freigegeben |

**Kurzinfo**

Dieses Dokument beschreibt die Schnittstelle zum OCSP-Responder der PKS Dienstleitung

# Inhaltsverzeichnis

# 1 Einleitung

Zur Überprüfung der Existenz und des Status eines Zertifikates ist es notwendig, eine vertrauenswürdige Instanz nach diesem Zertifikat zu befragen. Diese Schnittstelle einer Zertifizierungsstelle wird allg. als **Verzeichnisdienst** bezeichnet. Er hat die Aufgabe, dem Anwender Informationen über Zertifikate und (wenn freigegeben) die Zertifikate als solche über öffentliche Kommunikationsverbindungen zur Verfügung zu stellen. Damit dient der Verzeichnisdienst dem Kunden zum einen als verlässliche Quelle zur Abfrage von Zertifikaten in der Regel, zum anderen als Auskunftsdienst, ob und ab welchem Zeitpunkt ein bestimmtes Zertifikat gesperrt ist oder nicht.

Diese Dienste werden softwaretechnisch durch den LDAP-Server (***Lightweight Directory Access Protocoll-Server***) und den OCSP-Responder (***Online Certificate Status Protocol-Responder)*** realisiert.

In diesem Dokument werden die Funktion und das Front-End-Interface des OCSP-Responders beschrieben.

# 2 Der OCSP-Responder

Der OCSP-Responder übernimmt im Einzelnen folgende Aufgaben:

- Annahme und syntaktische Analyse von Anfragen an den OCSP-Responder
- Bereitstellen von Statusinformationen zu Signatur-, Verschlüsselungs-, Authentisierungs- und Attribut-Zertifikaten
- Erstellung einer kodierten Antwort. In diese Antwort wird die gesetzlich gültige Zeit eingebunden.
- Digitale Signatur dieses erzeugten Datenpaketes mit Hilfe eines privaten Schlüssels. Dieser private Schlüssel ist einem Signatur-Zertifikat zugeordnet, welches den Verzeichnisdienst der ZS authentisiert.
- Rücksendung der Antwort an den Kunden
- Erzeugung von Fehlermeldungen bei Problemen

## 2.1 Das OCSP-Protokoll

Alle Anfragen an den OCSP-Server werden mit Hilfe des http-Protokolls übertragen und müssen in einem definierten Format vorliegen (OCSP-Request) siehe dazu auch [COMMON-PKI].

Die Antwort des OCSP-Responders liefert für jedes Zertifikat eine der drei folgenden Statusinformationen:

- das Zertifikat ist nicht gesperrt und im Verzeichnisdienst vorhanden,
- das Zertifikat ist gesperrt,
- das Zertifikat befindet sich nicht in der Liste der von der ZS ausgestellten Zertifikate.

In die Antworten des OCSP-Responders an den Kunden wird jeweils die gesetzlich gültige Zeit eingebunden. Bei Fehlermeldungen erfolgt nur die Angabe des entsprechenden Fehlercodes ohne Einbindung eines Zeitstempels. Zur Gewähr der Integrität der Antwort als auch zur Angabe der Identität des Ausstellers wird die Antwort mit Hilfe eines privaten Schlüssels elektronisch signiert. Das zugehörige Signatur-Zertifikat des Verzeichnisdienstes wird der Antwort beigefügt.

### 2.1.1 HTTP als Transportprotokoll

In diesem Kapitel wird auf die Einbettung von OCSP-Anfragen und Antworten in HTTP eingegangen. Die Kommunikation zum Transport von OCSP-Anfragen über HTTP erfolgt über die beschriebenen Datenformate, die als Nachrichten über das Hypertext Transfer Protokoll (HTTP) an den OCSP-Responder übermittelt werden. Die Verwendung von HTTP [RFC2068] erlaubt eine einfache Erstellung von Software für den Zugriff auf Verzeichnisdienste unter Verwendung erprobter Programmbibliotheken. Weiterhin sind für HTTP Übergangsmöglichkeiten von firmeninternen Netzen in öffentliche Netze, sogenannten Firewalls, definiert und in der Form von HTTP Proxies bereits erprobt.

### 2.1.2 Definitionen für die Anfragen

Eine Anfrage wird mittels der Methode `POST` übertragen. Das HTTP Headerfeld „Content-Type“ mit dem Wert „application/ocsp-request“ muss verwendet werden. Die Länge der Anfrage muss im HTTP Headerfeld „Content-Length“ mit der Länge des Inhaltes der HTTP Anfrage belegt werden. Als Inhalt der HTTP Anfrage wird die DER-kodierte Anfrage an den Verzeichnisdienst verwendet. Die Anfrage soll binär übermittelt werden. Anfragen an den Verzeichnisdienst mit der Zugriffsmethode `POST` verwenden folgendes Format, wobei der base64-DER-kodierten Anfrage die ASN.1 Struktur ***OCSPRequest*** zugrunde liegt:

```
POST {url}
Content-Type: application/ocsp-request
Content-Length: ...
{binär kodierte Anfrage}
```

Außer den hier aufgeführten HTTP Header Feldern werden keine weiteren Felder ausgewertet, sondern nur stillschweigend ignoriert. Der OCSP-R verhält sich diesbezüglich wie ein HTTP/1.0 Server (siehe [RFC 1945], Anhang D).

#### 2.1.2.1 Definitionen für die Antworten

Eine HTTP Antwort besteht aus den üblichen HTTP Headern, der Inhalt des Dokumentes ist die DER-kodierte Antwort, wobei der DER-kodierten Antwort die ASN.1-Struktur OCSPResponse zugrunde liegt. Der Transport erfolgt binär.

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

## 2.1.3 OCSP-Datenformate

### 2.1.3.1 OCSP-Request

Der Verzeichnisdienst erlaubt das Abrufen von Statusinformationen von. Zur Abfrage werden durch den anfragenden Applikationsnutzer die nachfolgend genannten Informationen an den Verzeichnisdienst übergeben:

- Angabe über den verwendeten Hashalgorithmus (***hashAlgorithm***),
- Angabe über den Namen des Ausstellers (***issuerNameHash***),
- Angabe über das Ergebnis der Anwendung der Hashfunktion auf den Wert des Feldes ***subjectPublicKey*** (ohne den ASN.1-Tag und die Längenbeschreibung) aus dem Zertifikat des Ausstellers (***issuerKeyHash***),
- Angabe über die Seriennummer des betreffenden Zertifikats (***serialNumber***),

Die formelle Struktur der Anfrage im ASN1-Standard sieht wie folgt aus (unvollständige Darstellung der Struktur; vollständige Beschreibung ist in [COMMON-PKI] nachzulesen):

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

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

### 2.1.3.2 OCSP-Response

Die Antworten des Verzeichnisdienstes werden an den Applikationsnutzer zurückgesendet. Die Antwort des Dienstes kann für jedes angefragte Zertifikat grundsätzlich drei verschiedene Ergebnisse liefern. Die Antworten tragen eine qualifizierte elektronische Signatur, enthalten den Antwortzeitpunkt (Zeitstempel) und eine ***Statusinformation***.

Ist das angeforderte Zertifikat in der Datenbank vorhanden und nicht gesperrt, lautet die zugehörige ***Statusinformation*** „good".

Ist das angeforderte Zertifikat in der Datenbank vorhanden und gesperrt, lautet die zugehörige ***Statusinformation*** „revoked".

Ist das angeforderte Zertifikat nicht in der Datenbank vorgehalten, lautet die zugehörige ***Statusinformation*** „unknown".

Die Ergebnismeldungen des OCSP-Verzeichnisdienstes lauten:

| NAME | BEDEUTUNG |
|---|---|
| successful | erfolgreiche Bearbeitung einer Anfrage |
| malformedRequest | nicht erfolgreiche Bearbeitung einer Anfrage wegen fehlerhaftes Anfrage-Format |
| internalError | Auftreten eines internen Fehlers des Verzeichnisdienst |
| tryLater | temporäre Nichtverfügbarkeit des Verzeichnisdienst |

Anfragen zu unbekannten Ausstellerzertifikaten werden mit dem http Statuscode UNAUTHORIZED beantwortet.

Signierte Anfragen werden nicht unterstützt.

Die formelle Struktur der Anwort gem. [COMMON-PKI] im ASN1-Standard sieht wie folgt aus (unvollständige Darstellung der Struktur; vollständige Beschreibung ist in [COMMON-PKI] nachzulesen):

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

OCSPResponseStatus ::= ENUMERATED {
succesful (0),
malformedRequest (1),
internalError (2),
tryLater (3),
```

```
sigRequired (5),
unauthorized (6) }
```

Mögliche Fehlerwerte einer Antwort sind `malformedRequest`, `internalError` und `try-Later`.

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

id-pkix-ocsp OBJECT IDENTIFIER ::= { id-ad-ocsp }
id-pkix-ocsp-basic OBJECT IDENTIFIER ::= { id-pkix-ocsp 1 }

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

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

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

KeyHash ::= OCTET STRING

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

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

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

UnknownInfo ::= NULL


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

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

Folgende Extension wird benutzt um die Positivauskunft abzubilden:

```
id-isismtt-at-certHash OBJECT IDENTIFIER ::= {1 3 36 8 3 13 }
CertHash ::= SEQUENCE {
HashAlgorithm AlgorithmIdentifier,
certificateHash  OCTET STRING}
```

## 2.1.4 Übersicht zu den OCSP-Protokoll-Elementen

| (Sub-) Feld | Common-PKI | generiert | geparst | ausgewertet | Kommentar | |
|---|---|---|---|---|---|---|
| OCSPRequest | tbsRequest | | | X | X | |
| | optionalSignature | MAY | | X | - | |
| tbsRequest | version | | | X | X | muss v1 sein |
| | requestorName | MAY | | X | - | |
| | requestList | | | X | X | |
| | requestExtensions | MUST | | X | X | |
| request | reqCert | | | X | X | Element of requestList |
| | singleRequestExtensions | MUST | | X | X | |
| reqCert | hashAlgorithm | | | X | X | muss SHA1 oder SHA256 sein |
| | issuerNameHash | | | X | X | |
| | issuerKeyHash | | | X | X | |
| | serialNumber | | | X | X | |
| requestExtensions | Nonce | MAY, non-critical | | X | X | wird transparent in der Antwort durchgereicht |
| OCSPResponse | responseStatus | | X | | | |
| | responseBytes | MAY | X | | | Je nach Status |
| responseStatus | succesful | MUST | X | | | |
| | malformedRequest | MUST | X | | | |
| | internalError | MUST | X | | | |

| (Sub-) Feld | Common-PKI | generiert | geparst | ausgewertet | Kommentar |
|---|---|---|---|---|---|
| | tryLater | MUST | X | | |
| | sigRequired | MUST | (X) | | tritt nicht auf |
| | unauthorized | MUST | (X) | | falls eine Anfrage zu unbekannten Ausstellerzertifikat |
| responseBytes | responseType | (MUST) | X | | nur id-pkix-ocsp-basic |
| | response | MUST | X | | demnach Basic-Response |
| response | tbsResponseData | | X | | |
| | signatureAlgorithm | | X | | ecdsa-with-SHA256 (1.2.840.10045.4.3.2) |
| | signature | | X | | |
| | certs | MAY | X | | enthält Zertifikate des kompletten Pfades |
| responseData | version | | X | | immer v1 |
| | responderID | | X | | |
| | producedAt | | X | | |
| | responses | | X | | |
| | responseExtensions | MAY | X | | |
| responderID | byName | MAY | X | | Immer byname, Subject aus Signaturzertifikat |
| | byKey | MAY | - | | |
| singleResponse | certID | | X | | Element of responses |
| | certStatus | | X | | |
| | thisUpdate | | X | | gleich producedAt |
| | nextUpdate | MAY | - | | |
| | singleExtensions | MAY | X | | |
| certStatus | good | MUST | X | | certHash wird mitgeliefert |
| | revoked | MUST | X | | |

| (Sub-) Feld | Common-PKI | generiert | geparst | ausgewertet | Kommentar | |
|---|---|---|---|---|---|---|
| revokedInfo | unknown | | MUST | X | | |
| | revocationTime | | | X | | |
| | revocationReason | | MAY | X | | |
| unknownInfo | NULL | | (MUST) | X | | |
| revocationReason | MAY | | | | | |
| responseExtensions | Nonce | | MAY, non-crit | X | | nur wenn im Request vorhanden |
| singleExtensions | CRL entry extensions | | MAY, non-crit | - | | |
| | CertHash | | MAY, non-crit | X | | MUST, nur wenn Status = good/revoked |

# Quellenverzeichnis

[COMMON-PKI] Common-PKI COMMON PKI SPECIFICATIONS FOR INTEROPERABLE APPLICATIONS VERSION 2.0 – 20 JANUARY 2009

[RFC2560] M. Myers, R. Ankney, A. Malpani, S. Galperin, C. Adams, Online Certificate Status Protocol – OCSP, June 1999

[RFC 2068] R. Fielding, J. Getting, J. Mogul, H. Frystyk, T. Berners-Lee: Hypertext Transfer Protocol – HTTP / 1.1, Januar 1997

[RFC 2616] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, T. Berners-Lee: Hypertext Transfer Protocol – HTTP / 1.1, Juni 1999

[RFC 2396] T. Berners-Lee, R. Fielding, U.C. Irvine,L. Masinter, Uniform Resource Identifiers (URI): Generic Syntax, August 1998

[RFC 1945] T. Berners-Lee, R. Fielding, H. Frystyk: Hypertext Transfer Protocol – HTTP / 1.0, May 1996