ČESKÁ TECHNICKÁ NORMA

ICS 35.100.30 **Prosinec 2009**

ČSN 36 9791
ed. A

Information technology – Country Verifying Certification Authority Key Management Protocol for SPOC

Informační technologie – Protokol pro správu klíčů Národní ověřovací certifikační autotrity používaný SPOC

Technologies de I,information – Protocole d,échange de clés CVCA par un point unique de communication

Informationstechnik – Protokoll für SPOC zur Verwaltung von Schlüsseln der Country Verifying Certification Authority

This standard is the English version of the Czech Standard CSN 36 9791:2009. It was translated by Czech Office for Standards, Metrology and Testing. It has the same status as the official version.

Tato norma je anglickou verzí české normy ČSN 36 9791:2009. Překlad byl zajištěn Úřadem pro technickou normalizaci, metrologii a státní zkušebnictví. Má stejný status jako oficiální verze.

Contents

Introduction

Machine readable travel documents (MRTD) support advanced security mechanisms for the protection of the data stored in the MRTD. One of these mechanisms is the extended access control (EAC). If data stored in a MRTD is protected by EAC a terminal must be authenticated by the MRTD and must prove its right to the MRTD before the terminal can access the data. EAC as well as other advanced security mechanisms are described in [BSI-EAC].

The terminal authentication to be performed before reading protected data out of a MRTD is based on card verifiable (CV) certificates which can be verified by a MRTD. The access rights given to a terminal are coded within the CV certificate. After verifying the CV certificate the MRTD grants access to its data according to the access rights coded in the CV certificate. A public key infrastructure for the generation and distribution of the CV certificates is outlined in [BSI-EAC]. This EAC-PKI will be constructed by all member states of the EU. A common certificate policy for the entities of the EAC-PKI is given by [EUCP].

Within the EAC-PKI each member state operates its own root CA called country verifying CA (CVCA). The second level of the EAC-PKI is formed by CAs called Document Verifier (DV). Each DV is associated to the national CVCA of its own country. The DV gets its own CV certificates from that

national and foreign CVCAs and generates the CV certificates for inspection systems (IS) within its sphere of influence. From this point of view inspection systems are the holder of the end user certificates of the EAC-PKI.

1 Scope

This document specifies the key management protocol in operation *across international borders* between the Country Verifying Certification Authority (CVCA) components of the ePassport EAC architecture and the Document Verifier (DV) certification authority components.

This protocol is used to exchange keys and certificates, in order that:

- the DV can send a certification request to the foreign CVCA;
- the CVCA can send the issued certificate to the requesting DV;
- the DV and the CVCA can request for a list of valid certificates (a certificate chains) needed to read an ePassport from the foreign CVCA;
- general messages can be exchanged between EAC PKI entities.

The specification covers following channels to exchange data:

- manual exchange with the data stored on a removable media (CD-R,DVD+/-R, USB storage) or published on the internet;
- web services interface.

This specification does not cover:

- exchanges and communication which is internal to the country (domestic DV to domestic CVCA, IS to DV);
- exchanges related to the initial registration process except format of the data exchanged (media format, metadata content).

2 Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO 3166-1 Codes for the representation of names of countries and their subdivisions – Part 1: Country codes

ISO/IEC 9293:1994 Information technology – Volume and file structure of disk cartridges for information interchange

ISO 9660:1988 Information processing – Volume and file structure of CD-ROM for information interchange

3 Terms and definitions

Abbreviation	**Explanation**
SPOC	Single Point of Contact – the only EAC communication gate for a member state
SPOC CA	Certification authority for EAC communication security
SPOC TLS server certificate	Certificate representing SPOC identity when acting as TLS server
SPOC TLS client certificate	Certificate representing SPOC identity when acting as TLS client
SPOC URL	The base URL where the WSDL file describing SPOC web-service interface is available.

4 Abbreviations

Abbreviation	Explanation
DER	Distinguished Encoding Rules
BIG	Brussels Interoperability Group
CA	Certification authority
CP	Certification policy
CRL	Certificate revocation list
CSCA	Country signing CA
CSN	Czech technical standard
CVCA	Country Verifying Certification Authority
DH	Diffie-Hellman
DNS	Domain name service
DV	Document Verifier same as DVCA
EAC	Extended Access Control
EC	Elliptic Curves
ECDH	Elliptic Curves Diffie Hellman
ECDSA	Elliptic Curves Digital Signature Algorithm
HSM	Hardware Security Module
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secured / http over TLS
CHR	Certificate Holder Reference
ICAO	International Civil Aviation Organization
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronic Engineering
IS	Inspection System
ISO	International Organization for Standardization
MRTD	Machine Readable Travel Document
NIST	National Institute of Standards and Technology
PIN	Personal identification number
PKCS	Public Key Cryptography Standard
PKI	Public Key Infrastructure
SHA-1	Secure Hash Algorithm v1
SHA-2	Secure Hash Algorithm v2
SPOC	Single Point of Contact
SSL	Secure Socket Layer
TLS	Transport Layer Security
TLV	Tag Length Value

5 Overview

Each country implementing EAC PKI will maintain its internal infrastructure. The internal infrastructure is opaque and not visible from abroad. The only information available about the internal infrastructure is as follows:

- CVCA is presented by its identity and the certification policy (mainly for the purpose of registration which is not covered by this specification);
- NCP – the National Certification Policy as a base for the country DV operation;
- DV (one or more) is presented by its identity and the DV certification policy;
- SPOC – single point of contact is presented by its identity and a list of contact details.

SPOC is the interface exposed by the country EAC PKI and MUST be used for communication with foreign entities. CVCA SHALL utilize domestic SPOC to accept incoming certification requests and to send the resulting certificates or the failure notifications to the requestor. DV SHALL utilize domestic

SPOC to send certification request to the foreign CVCAs and to receive the resulting certificates or failure notifications. Every EAC PKI entity inside the country infrastructure MUST utilize SPOC as an intermediate to retrieve certificate chains needed to read fingerprints from the foreign eMRTD.

The SPOC MUST:

- Collect requests from inside of the country EAC PKI and forwards the requests to the other country SPOCs as needed.
- Collect requests and responses from other countries SPOCs and delivers them to relevant entities inside the country EAC PKI.

The internal country EAC PKI infrastructure is out of the scope of this specification. The external inter-SPOC protocol is described in this specification.

All interSPOC wire exchanges SHALL be protected by means of PKI. Web-service communication SHALL use HTTPS with TLS authentication both of the client and the server. Each SPOC registers separately with all other SPOCS of interest. During the registration of the SPOC its Certification Authority is also registered. The SPOC CA issues SPOC certificates.

All interSPOC communication MUST be conducted via internet. The SPOC MUST be reachable in publicly available IPv4 internet infrastructure. After 31 December 2011 the SPOC web-service SHALL be reachable using IPv6 technology in addition to IPv4.

Figure 1 SPOC context schema

6 Single point of contact (SPOC)

SPOC is an entity responsible for sending and receiving data during the EAC PKI key management operations between countries. SPOC is the only interface exposed by a country EAC PKI for foreign communication.

6.1 SPOC initial registration information

During the initial registration with each SPOC at least the following information MUST be registered:

- *SPOC country* – the country which the SPOC acts for;
- *SPOC URL* – URL of WSDL describing SPOC interface and service location;
- *SPOC CA root certificate* – certificates used to verify SPOC communication certificates.

7 Messages

7.1 RequestCertificate

Intended Use:

This message is used by a SPOC for requesting the generation of a new certificate for one of its DV from a foreign CVCA.

Input parameters:

- `callerID`: (MANDATORY)
  This parameter contains the identifier of originator country. The value SHALL be 2 letter country code

according to ISO 3166-1 ALPHA-2. The value of `callerID` SHALL be verified by the recipient SPOC with the value recorded for the originating SPOC during its registration.

- `messageID`: (MANDATORY)
  This parameter contains the identification of the message. It MUST identify the message uniquely within all messages from that originator. If a response message will be sent to the originator as a result of this message, the response message will contain the same `messageID`. Hence an incoming response message can be assigned to the correct original message. Construction and allocation of the `messageID` can be decided by the originator and is not verified by the receiving party.
- `certReq`: (MANDATORY)
  This parameter contains the actual certificate request. It must be constructed according to C.2 of [BSI-EAC]. The coding must follow the specifications in addendum D of [BSI-EAC].

Output parameters:

- `certificateSeq` (CONDITIONAL)
  This parameter will contain the result (one or more certificates) after processing this message, if the message has been processed successfully and synchronously by the receiver. It is REQUIRED if certificates have to be sent with the response. It MUST be absent if no certificates will be sent with the message. See Remarks for description of the certificates included in this parameter.

Return codes:

- `ok_cert_available`: The message has been processed successfully and synchronously. The output parameter `certificateSeq` contains one ore more certificates.
- `ok_reception_ack`: The reception of the message is acknowledged. No further verification of the message has been done yet. The processing of the message will be done asynchronously. The result of the processing will be sent to the registered URL using the message `SendCertificates`.
- `failure_inner_signature`: The verification of the inner signature of the actual certificate request failed.
- `failure_outer_signature`: The verification of the outer signature of the actual certificate request failed.
- `failure_syntax`: The message is syntactically not correct.
- `failure_request_not_accepted`: The message has been processed correctly but the request has not been accepted.
- `failure_internal_error`: error other than above

Remarks:

- According to C.2 of [BSI-EAC] the body of the actual certificate request SHOULD contain a certification authority reference to inform the certification authority about the private key that is expected by the originator of the message to be used to sign the certificate. If the certification authority reference in the request deviates from the certification authority reference in the issued certificate, the corresponding certificate of the certification authority SHALL also be provided to the originator in the response. In such a case, and if the message is processed synchronously, the certificate of the certification authority SHALL be part of the output parameter `certificateSeq`. The DV certificate SHALL be the first certificate in the sequence. CVCA certificates (root and/or link) SHALL be ordered by effective date (ascending) in the sequence.

7.2 SendCertificates

Intended use:

SPOC sends the new certificate or certificate chain to the requesting SPOC. This message SHALL be generated in response to:

- RequestCertificate: upon successful asynchronous request processing after the certificate is issued
- GetCACerticifates:
  In addition the message MUST be used when a new certificate is created (CVCA root and link) to push the certificates to registered foreign SPOC.

Input parameters:

- callerID*:* (MANDATORY)
  This parameter contains the identifier of originator country. The value SHALL be 2 letter country code according to ISO 3166-1 ALPHA-2. The value of callerID SHALL be verified by the recipient SPOC with the value recorded for the originating SPOC during its registration.
- messageID: (CONDITIONAL)
  When the message is generated in response to a request message the parameter MUST contain the same contents as the parameter messageID of the request message. When the message generation was triggered without external intervention (CVCA certificate rekey) The statusInfo value SHALL be new_cert_available_notification and the messageID parameter MAY be omitted and SHALL be ignored when present.
- statusInfo: (MANDATORY)
  This parameter contains a status code about the result of processing the corresponding message. Following statuses are possible:
- new_cert_available_notification: The originating SPOC wants to notify about new CVCA certificate(s) available without being requested.
- ok_cert_available: The request has been processed successfully. The input parameter certificateSeq contains one or more certificates.
- failure_inner_signature: The verification of the inner signature of the actual certificate request failed.
- failure_outer_signature: The verification of the outer signature of the actual certificate request failed.
- failure_syntax: The corresponding message is syntactically not correct.
- failure_request_not_accepted: The corresponding message has been processed correctly but the request has not been accepted.
- failure_internal_error: error other than above
- certificateSeq: (CONDITIONAL)
  This parameter is REQUIRED if certificates have to be sent with the message. It MUST be absent if no certificates will be sent with the message. The certificates SHALL be binary TLV DER encoded as defined in [BSI-EAC].

- When the message is generated in response to GetCACertificates or because there is a new certificate the sequence SHALL contain a list of CA certificates. The list SHALL be ordered. CVCA certificates (link and/or root) SHALL be ordered by effective date in the sequence. The list SHALL contain all public keys required to verify all passports currently in circulation. When the sequence contains certificates with different domain parameters at least one certificate with domain parameters included for each domain paramaters variant SHALL be present.
- When the message is generated in response to RequestCertificate the content of the sequence is the same as described for synchronous response of RequestCertificate.

Output parameters:

none

Return codes:

- ok_received_correctly: The message has been received correctly.
- failure_syntax: The message is syntactically not correct.
- failure_messageID_unknown: The contained messageID cannot be matched with a message formerly sent.
- failure_internal_error: error other than above

7.3 GetCACertificates

Intended use:

This message is sent by a SPOC to a foreign SPOC in order to get all valid CVCA certificates (link certificates and selfsigned certificates) of that country.

Input parameters:

- callerID*:* (MANDATORY)
  This parameter contains the identifier of the originator country. The value SHALL be 2 letter country code according to ISO 3166-1 ALPHA-2. The value of callerID SHALL be verified by the recipient SPOC with the value recorded for the originating SPOC during registration.
- messageID: (MANDATORY)
  This parameter contains the identification of the message. It MUST identify the message uniquely within all messages of the originator. If a response message will be send to the originator as a result of this message, the response message will contain the same messageID. Hence an incoming response message can be assigned to the correct original message. Construction and allocation of the messageID can be decided by the originator.

Output parameters:

- certificateSeq: (CONDITIONAL)
  This parameter will contain the result (one or more certificates) after processing this message, if the message has been processed successfully and synchronously by the receiver. It is REQUIRED if certificates have to be sent with the response. It MUST be absent if no certificates will be sent with the message.

Return codes:

- ok_cert_available: The message has been processed successfully and synchronously. The output parameter certificateSeq contains one ore more CA certificates.
- ok_reception_ack: The reception of the message is acknowledged. No further verification of the message has been done yet. The processing of the message will be done asynchronously. The result of the processing will be sent to the registered URL using the message SendCertificates.
- failure_request_not_accepted: The message has been processed correctly but the request has not been accepted
- failure_internal_error: error other than above

Remarks:

If the message is processed successfully and accepted the CVCA MUST send all valid CVCA within the response, either in the output parameter certificateSeq (synchronous processing) or in the corresponding response message SendCertificates (asynchronous processing).

7.4 GeneralMessage

Intended use:

This message is sent by a SPOC to a foreign SPOC in order to send notification or other general text human readable message.

Input parameters:

- `callerID`*:* (MANDATORY)
  This parameter contains the identifier of the originator country. The value SHALL be 2 letter country code according to ISO 3166-1 ALPHA-2. The value of `callerID` SHALL be verified with value recorded during registration including message security features (digital signature certificate/TLS client certificate is registered for respective country).
- `messageID`: (MANDATORY)
  This parameter contains the identification of the message. It MUST identify the message uniquely within all messages of the originator. If a response message will be send to the originator as a result of this message, the response message will contain the same `messageID`. Hence an incoming response message can be assigned to the correct original message. Construction and allocation of the `messageID` can be decided by the originator.
- `subject`: (MANDATORY)
  This parameter contains the subject of the message. The subject SHOULD briefly describe the content of the message body. English MUST be used for subject.
- `body`: (MANDATORY)
  This parameter contains the body of the message. The body SHALL be human readable plain text which is not intended for direct automated processing. English MUST be used for the body.

Return codes:

- `ok`: The message has been accepted for delivery.
- `failure_internal_error`: error other than above

8 Web service

The web service interface is the interface for the routine interSPOC wire data exchange. The interface SHALL use [SOAP] over [HTTPS] protocol. The SPOC web service interface SHALL conform to the WSDL specified in the Section 11.

8.1 SOAP usage

Pure [SOAP] over [HTTPS] SHALL be used to implement the Web-service interfaces. Any other SOAP extensions (e.g. WS-Addressing, WS-Security, WS-Secure Conversation, WS-Authorization, WS-Federation, WS-Authorization, WS-Policy, WS-Trust, WS-Privacy, WS-Test and other extensions of WS) SHALL NOT be used.

The intermediary SOAP node type SHALL NOT be used. Only a direct client SPOC to server SPOC configuration SHALL be used.

The SOAP fault element SHALL be used only when a transport layer processing error that is not covered by this specification occurs. Application level errors SHALL be communicated as normal SOAP responses using the error mechanism as described for each message.

It is RECOMMENDED that the web service interface is implemented in accordance to [WS-IBP] and

[WS-ISBBP].

The SPOC SOAP interface MUST conform to WSDL definition as described in section 11 of this document.

8.2 Security consideration

The SPOC web service communication SHALL use secure and authenticated channel. SOAP over HTTPS SHALL be used. TLS v1.0 SHALL be used as defined in [TLS10]. TLS 1.1 and TLS 1.2 MAY be supported through standard TLS protocol version negotiation. Both client and server SHALL be authenticated using asymmetric cryptography based on public keys stored in X.509 certificates. All certificates needed to build trusted chain up to the registered SPOC CA SHALL be included in the TLS negotiation.

The TLS client SHALL perform following verifications:

- the server certificate SHALL be fully validated according to [RFC5280] including revocation status
- the server certificate ExtKeyUsage extension MUST be present and SHALL contain the OIDs according to Table SPOC TLS server certificate
- the server certificate subject country SHALL be equal to the value of `callerID` parameter

In case of any failure the TLS client MUST close the connection.

The TLS server SHALL perform following verifications:

- the client SHALL be fully authenticated using a certificate
- the client certificate SHALL be fully validated according to [RFC5280] including revocation status
- the client certificate ExtKeyUsage extension MUST be present and SHALL contain the OIDs according to **Table 3** SPOC TLS client certificate
- the client certificate subject country SHALL correspond to the intended one

In case some of the verifications fail the request SHALL be rejected using `HTTP 401 Unauthorized` response code.

Fallback to weak (pre TLS) protocol SHALL NOT be allowed by the interface configuration. Only following cipher suites SHALL be supported and used:

Table1 – TLS Encryption suites

Cipher suite	**Certificate and key exchange algorithm**
TLS_RSA_WITH_AES_128_CBC_SHA	RSA
TLS_DHE_RSA_WITH_AES_128_CBC_SHA	DHE_RSA
TLS_RSA_WITH_AES_256_CBC_SHA	RSA
TLS_DHE_RSA_WITH_AES_256_CBC_SHA	DHE_RSA
TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA	ECDHE_ECDSA
TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA	ECDHE_ECDSA

In the scope of the TLS handshake negotiation client SHALL support all the TLS cipher suites defined in **Table 1**. Both the server and the client side SHALL support RSA and ECDSA based authentication. It is permissible for a server to request and also for the client to send a client certificate of a different type than the server certificate.

The use of the ECDHE_ECDSA key agreement in TLS handshake is in accordance with the additions defined in

[TLSECC] and [TLSEXT]. Both the client and the server SHALL support the appropriate Elliptic curves extensions as specified in [TLSECC] specification in the scope of TLS handshake. The supported Elliptic curves and EC Point formats are defined in Section 5 of [TLSECC]. The use of the supported TLS cipher suites defined in **Table 1**. which uses Advanced Encryption Standard (AES) for encryption SHALL be in accordance with [TLSAES] specification.

9 Out of band

Out of band communication is used to hand over the data during the registration process. It can be used as needed depending on bilateral agreement between parties. When removable media is used the content of the media MUST conform to the description in section 9.1. The data can also be published in an electronic form or distributed in an electronic form. In case the data is available electronically the metadata MUST be handed over by a trusted communication channel. The metadata MUST conform to description in section 9.2.

9.1 Media format and file naming convention

One of the following media types SHALL be used:

- CD-R with [ISO9660] file system with Joliete extension for long file names;
- DVD+/-R with [ISO9660] file system with Joliete extension for long file names;
- USB mass storage with FAT ([ISO/IEC 9293:1994]) or [FAT32] file system (requesters and responders MUST support the Microsoft Windows extension for long file names).

Media used for key management SHALL NOT contain any information unrelated to key management.

When a media contains CV certificates each certificate SHALL be in a separate file. Each file SHALL be named `CAR_CHR.cvcert`, where CAR and CHR are the certification authority reference and certificate holder reference values from the certificate contained in the file.

When a media contains a certificate request each certificate request SHALL be in a separate file. Each file SHALL be named `CHR.cvreq` – where CHR is the certificate holder reference value from the certificate request contained in the file.

Files SHALL contain the binary encoding of the data structures as specified in [BSI-EAC].

9.2 Metadata

The metadata SHALL contain:

- The filename or URL or other data source identifier;
- The size of the data in bytes;
- The hash value computed over the binary contents of the data, in the format: `[sha1|sha256]:xx:xx:xx:…:xx:xx`, where xx represents the hex encoded value of the corresponding byte of the hash value. One of either the SHA1 or SHA256 hash functions SHALL be used to compute the hash;
- a textual description of the data contents in English.

9.3 Security considerations

The metadata MUST be exchanged using an auditable chain of custody. Hash values of the data MUST be computed and compared to the metadata hash values. In case the hash values don,t match the data MUST be rejected as untrusted.

10 PKI for inter SPOC security

The SPOC CA root certificate SHALL be registered during the initial registration process. The SPOC MUST use the registered SPOC CA to validate certificates used in the communication. After SPOC CA rekey the new certificate SHALL be delivered to all SPOCs using the same trusted channel as used during the initial registration.

Following certificates SHALL be used by a SPOC

- *SPOC CA Root certificate* – selfsigned certificate of SPOC root CA (MANDATORY);
- *SPOC CA Intermediate certificate* – intermediate CA certificate (OPTIONAL);
- *SPOC TLS client certificate* – represents SPOC identity when acting as TLS client (MANDATORY);
- *SPOC TLS server certificate* – represents SPOC identity when acting as TLS server(MANDATORY).

The SPOC PKI certificate profile is defined in **Table 2**. The certificates belonging to SPOC are identified by extended key usage as specified in Table. In addition it is RECOMMENDED to use CN values as specified **Table 3.**

10.1 The SPOC certificate profiles

Following notation is used**:** m=mandatory, o=optional, x=not used; c=critical

Table 2 – SPOC Certificate profile – certificate body

Certificate Component	Section in RFC 5280	Presence	Comments
Certificate	4.1.1	m	
TBSCertificate	4.1.1.1	m	see next part of the table
signatureAlgorithm	4.1.1.2	m	value inserted here dependent on algorithm selected
signatureValue	4.1.1.3	m	value inserted here dependent on algorithm selected
TBSCertificate	4.1.2		
version	4.1.2.1	m	MUST be v3
serialNumber	4.1.2.2	m	
signature	4.1.2.3	m	value inserted here MUST match the OID in signatureAlgorithm
issuer	4.1.2.4	m	SPOC CA reference
validity	4.1.2.5	m	Implementations MUST specify using UTC time until 2049 from then on using GeneralizedTime
subject	4.1.2.6	m	C part of DN SHALL contain SPOC country code (see callerID parameter description); See **Table 3** for RECOMMENDED CN values; Other fields MAY be used with arbitrary values
subjectPublicKeyInfo	4.1.2.7	m	
issuerUniqueID	4.1.2.8	x	
subjectUniqueID	4.1.2.8	x	
extensions	4.1.2.9	m	see **Table 4** for extensions requirements

Table 3 – SPOC certificates CN and ExtendedKeyUsage values and

Certificate	CN (Optional)	Extended Key Usage OID (Mandatory)
SPOC TLS client certificate	SPOC TLS client	`id-csn-369791-tls-client`, `id-kp-clientAuth`
SPOC TLS server certificate	SPOC TLS server	`id-csn-369791-tls-server`, `id-kp-serverAuth`

Table 4 – SPOC certificate profile – extensions

Extension name	Section in RFC 5280	SPOC CA	TLS client	TLS server	Comments
AuthorityKeyIdentifier	4.2.1.1	m	m	m	
keyIdentifier		m	m	m	
authorityCertIssuer		o	o	o	
authorityCertSerialNumber		o	o	o	
PrivateKeyUsagePeriod	4.2.1.4	o	o	o	
SubjectKeyIdentifier	4.2.1.2	m	m	o	
subjectKeyIdentifier		m	m	m	
KeyUsage	4.2.1.3	mc	mc	mc	
digitalSignature		x	m	o	
nonRepudiation		x	x	x	
keyEncipherment		x	o	o	
dataEncipherment		x	x	x	
keyAgreement		x	o	m	
keyCertSign		m	x	x	
cRLSign		m	x	x	
encipherOnly		x	x	x	
decipherOnly		x	x	x	
CertificatePolicies	4.2.1.5	o	o	o	
PolicyInformation		m	m	m	
policyIdentifier		m	m	m	
policyQualifiers		o	o	o	
PolicyMappings	4.2.1.6	x	x	x	
SubjectAltName	4.2.1.7	x	o	m	Note 1
IssuerAltName	4.2.1.8	o	o	o	
SubjectDirectoryAttributes	4.2.1.9	o	o	o	
Basic Constraints	4.2.1.10	m	o	o	
cA		true	false	false	
PathLenConstraint		1 or 2	x	x	
NameConstraints	4.2.1.11	x	x	x	
PolicyConstraints	4.2.1.12	x	x	x	
ExtKeyUsage	4.2.1.13	x	m	m	Note 3
CRLDistributionPoints	4.2.1.14	m	m	m	Note 2
distributionPoint		m	m	m	
reasons		x	x	x	
cRLIssuer		x	x	x	

InhibitAnyPolicy	4.2.1.15	x	x	x	
FreshestCRL	4.2.1.16	x	x	x	
privateInternetExtensions	4.2.2	o	o	o	
other private extensions	N/A	o	o	o	

Note 1:

SPOC TLS server certificate SHALL contain SubjectAltName extension containing dNSName value. The dNSName value SHALL be the host part of the SPOC URL.

Note 2:

At least one distribution point MUST use HTTP as an access method and the URL MUST be publicly available over the internet. The CDP network service SHALL be reachable as defined in section 5.

Note 3:

See **Table 3** for mandatory extended key usage OIDs for each certificate.

11 WSDL for web service interface

```
<?xml version="1.0" encoding="UTF-8"?>

<wsdl:definitions

xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"

xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"

xmlns:xs="http://www.w3.org/2001/XMLSchema"

xmlns:SPOC="http://namespaces.unmz.cz/csn369791"

targetNamespace="http://namespaces.unmz.cz/csn369791">


<wsdl:types>

<xs:schema xmlns="http://namespaces.unmz.cz/csn369791"

targetNamespace="http://namespaces.unmz.cz/csn369791"

elementFormDefault="qualified" attributeFormDefault="unqualified">

<xs:element name="certificateSequence">

<xs:complexType>

<xs:sequence>

<xs:element name="certificate" type="xs:base64Binary" minOccurs="1"
maxOccurs="unbounded"/>

</xs:sequence>
```

```
</xs:complexType>
</xs:element>
<xs:element name="RequestCertificateRequest">
<xs:complexType>
<xs:sequence>
<xs:element name="callerID" type="xs:string"/>
<xs:element name="messageID" type="xs:string"/>
<xs:element name="certificateRequest" type="xs:base64Binary"/>
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="RequestCertificateResponse">
<xs:complexType>
<xs:sequence>
<xs:element ref="certificateSequence" minOccurs="0" maxOccurs="1"/>
<xs:element name="result">
<xs:simpleType>
<xs:restriction base="xs:string">
<xs:enumeration value="ok_cert_available"/>
<xs:enumeration value="ok_reception_ack"/>
<xs:enumeration value="failure_inner_signature"/>
<xs:enumeration value="failure_outer_signature"/>
<xs:enumeration value="failure_syntax"/>
<xs:enumeration value="failure_request_not_accepted"/>
<xs:enumeration value="failure_internal_error"/>
</xs:restriction>
</xs:simpleType>
</xs:element>
```

```
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="SendCertificatesRequest">
<xs:complexType>
<xs:sequence>
<xs:element name="callerID" type="xs:string"/>
<xs:element name="messageID" type="xs:string" minOccurs="0" maxOccurs="1"/>
<xs:element ref="certificateSequence" minOccurs="0" maxOccurs="1"/>
<xs:element name="statusInfo">
<xs:simpleType>
<xs:restriction base="xs:string">
<xs:enumeration value="new_cert_available_notification"/>
<xs:enumeration value="ok_cert_available"/>
<xs:enumeration value="failure_inner_signature"/>
<xs:enumeration value="failure_outer_signature"/>
<xs:enumeration value="failure_syntax"/>
<xs:enumeration value="failure_request_not_accepted"/>
<xs:enumeration value="failure_internal_error"/>
</xs:restriction>
</xs:simpleType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="SendCertificatesResponse">
<xs:complexType>
<xs:sequence>
```

```
<xs:element name="result">
<xs:simpleType>
<xs:restriction base="xs:string">
<xs:enumeration value="ok_received_correctly"/>
<xs:enumeration value="failure_syntax"/>
<xs:enumeration value="failure_messageID_unknown"/>
<xs:enumeration value="failure_internal_error"/>
</xs:restriction>
</xs:simpleType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="GetCACertificatesRequest">
<xs:complexType>
<xs:sequence>
<xs:element name="callerID" type="xs:string"/>
<xs:element name="messageID" type="xs:string"/>
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="GetCACertificatesResponse">
<xs:complexType>
<xs:sequence>
<xs:element ref="certificateSequence" minOccurs="0" maxOccurs="1"/>
<xs:element name="result">
<xs:simpleType>
<xs:restriction base="xs:string">
```

```
<xs:enumeration value="ok_cert_available"/>
<xs:enumeration value="ok_reception_ack"/>
<xs:enumeration value="failure_request_not_accepted"/>
<xs:enumeration value="failure_internal_error"/>
</xs:restriction>
</xs:simpleType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="GeneralMessageRequest">
<xs:complexType>
<xs:sequence>
<xs:element name="callerID" type="xs:string"/>
<xs:element name="messageID" type="xs:string"/>
<xs:element name="subject" type="xs:string"/>
<xs:element name="body" type="xs:string"/>
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="GeneralMessageResponse">
<xs:complexType>
<xs:sequence>
<xs:element name="result">
<xs:simpleType>
<xs:restriction base="xs:string">
<xs:enumeration value="ok"/>
<xs:enumeration value="failure_internal_error"/>
```

```
</xs:restriction>
</xs:simpleType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:schema>
</wsdl:types>

<wsdl:message name="RequestCertificateRequest">
<wsdl:part name="RequestCertificateRequest" element="SPOC:RequestCertificateRequest"/>
</wsdl:message>
<wsdl:message name="RequestCertificateResponse">
<wsdl:part name="RequestCertificateResponse" element="SPOC:RequestCertificateResponse"/>
</wsdl:message>

<wsdl:message name="SendCertificatesRequest">
<wsdl:part name="SendCertificatesRequest" element="SPOC:SendCertificatesRequest"/>
</wsdl:message>
<wsdl:message name="SendCertificatesResponse">
<wsdl:part name="SendCertificatesResponse" element="SPOC:SendCertificatesResponse"/>
</wsdl:message>

<wsdl:message name="GetCACertificatesRequest">
<wsdl:part name="GetCACertificatesRequest" element="SPOC:GetCACertificatesRequest"/>
</wsdl:message>
<wsdl:message name="GetCACertificatesResponse">
<wsdl:part name="GetCACertificatesResponse" element="SPOC:GetCACertificatesResponse"/>
```

```
</wsdl:message>

<wsdl:message name="GeneralMessageRequest">
<wsdl:part name="GeneralMessageRequest" element="SPOC:GeneralMessageRequest"/>
</wsdl:message>
<wsdl:message name="GeneralMessageResponse">
<wsdl:part name="GeneralMessageResponse" element="SPOC:GeneralMessageResponse"/>
</wsdl:message>

<wsdl:portType name="SPOCPortType">
<wsdl:operation name="RequestCertificate">
<wsdl:input message="SPOC:RequestCertificateRequest"/>
<wsdl:output message="SPOC:RequestCertificateResponse"/>
</wsdl:operation>
<wsdl:operation name="SendCertificates">
<wsdl:input message="SPOC:SendCertificatesRequest"/>
<wsdl:output message="SPOC:SendCertificatesResponse"/>
</wsdl:operation>
<wsdl:operation name="GetCACertificates">
<wsdl:input message="SPOC:GetCACertificatesRequest"/>
<wsdl:output message="SPOC:GetCACertificatesResponse"/>
</wsdl:operation>
<wsdl:operation name="GeneralMessage">
<wsdl:input message="SPOC:GeneralMessageRequest"/>
<wsdl:output message="SPOC:GeneralMessageResponse"/>
</wsdl:operation>
</wsdl:portType>
```

```
<wsdl:binding name="SPOCSOAPBinding" type="SPOC:SPOCPortType">
<soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
<wsdl:operation name="RequestCertificate">
<soap:operation soapAction="RequestCertificate"/>
<wsdl:input>
<soap:body parts="RequestCertificateRequest" use="literal"/>
</wsdl:input>
<wsdl:output>
<soap:body parts="RequestCertificateResponse" use="literal"/>
</wsdl:output>
</wsdl:operation>
<wsdl:operation name="SendCertificates">
<soap:operation soapAction="SendCertificates"/>
<wsdl:input>
<soap:body parts="SendCertificatesRequest" use="literal"/>
</wsdl:input>
<wsdl:output>
<soap:body parts="SendCertificatesResponse" use="literal"/>
</wsdl:output>
</wsdl:operation>
<wsdl:operation name="GetCACertificates">
<soap:operation soapAction="GetCACertificates"/>
<wsdl:input>
<soap:body parts="GetCACertificatesRequest" use="literal"/>
</wsdl:input>
<wsdl:output>
<soap:body parts="GetCACertificatesResponse" use="literal"/>
</wsdl:output>
```

```
</wsdl:operation>
<wsdl:operation name="GeneralMessage">
<soap:operation soapAction="GeneralMessage"/>
<wsdl:input>
<soap:body parts="GeneralMessageRequest" use="literal"/>
</wsdl:input>
<wsdl:output>
<soap:body parts="GeneralMessageResponse" use="literal"/>
</wsdl:output>
</wsdl:operation>
</wsdl:binding>

<wsdl:service name="SPOC">
<wsdl:port name="SPOCPort" binding="SPOC:SPOCSOAPBinding">
<soap:address location="http://spoc-server/SPOC"/>
</wsdl:port>
</wsdl:service>

</wsdl:definitions>
```

12 OID assignment

For purpose of identification of certificate/key usages the following OIDs are introduced:

```
id-csn-369791 OBJECT IDENTIFIER ::= {
        iso(1) member-body(2) cz(203) moi(7064)
        orpeg(1) cdbp(1) csn369791(369791)
}
id-csn-369791-tls-client ::= { id-csn-369791 1 }
id-csn-369791-tls-server ::= { id-csn-369791 2 }
```

Bibliography

[BSI-EAC] Technical Guideline TR-03110, Advanced Security Mechanisms for Machine Readable Travel Documents – Extended Access Control (EAC), Version 1.11, Bundesamt für Sicherheit in der Informationstechnik, 21 Feb 2008.

[EUCP] Common Certificate Policy for the Extended Access Control Infrastructure for Passports and Travel Documents Issued by EU Member States, Version 1.0, 22 April 2008

[FAT32] FAT32 File System Specification, Microsoft corporation, available at: http://www.microsoft.com/whdc/system/platform/firmware/fatgen.mspx

[RFC2119] S. Bradner, „Key words for use in RFCs to Indicate Requirement Levels“, BCP 14, RFC 2119, March 1997

[SOAP] SOAP Version 1.2 Part 1: Messaging framework (Second Edition), W3C Recommendation 27 April 2007

[WSI-BP] WS-I Basic Profile available at http://www.ws-i.org/Profiles/BasicProfile-1.1.html

[WSI- SSBP] WS-I Basic Bindingavailable at http://www.ws-i.org/Profiles/SimpleSoapBindingProfile-1.0.html

[TLS10] Dierks, T. and C. Allen, „The TLS Protocol Version 1.0“, RFC 2246, January 1999

[TLS11] Dierks, T. and E. Rescorla, „The Transport Layer Security (TLS) Protocol Version 1.1“, RFC 4346, April 2006

[TLS12] Dierks, T., and E. Rescorla, „The Transport Layer Security (TLS) Protocol, Version 1.2“, RFC 5246, August 2008

[TLSEXT] Blake-Wilson, S., Nystrom, M., Hopwood, D., Mikkelsen, J., and T. Wright, „Transport Layer Security (TLS) Extensions“, RFC 4366, April 2006

[TLSAES] Chown, P., „Advanced Encryption Standard (AES) Ciphersuites for Transport Layer Security (TLS)“, RFC 3268, June 2002

[TLSECC] Blake-Wilson, S., Bolyard, N., Gupta, V., Hawk, C., and B. Moeller, „Elliptic Curve Cryptography (ECC) Cipher Suites for Transport Layer Security (TLS)“, RFC 4492, May 2006

[HTTP] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, T. Berners-Lee, „Hypertext Transfer Protocol – HTTP/1.1.“, RFC2616, June 1999.

[HTTPS] E. Rescorla., „HTTP Over TLS.“, RFC 2818, May 2000.

[PKIX] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, W. Polk., Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. RFC 5820, May 2008.

Věstníku Úřadu pro technickou normalizaci, metrologii a státní zkušebnictví.

ČSN 36 9791 ed. A

Vydal Úřad pro technickou normalizaci, metrologii a státní zkušebnictví, Praha
Rok vydání 2009, 20 stran
85000 Cenová skupina 411

+!5J0JG3-ifaaaj!