OCF 1.0 Security CR - CR35 (1385/1303)

Legal Disclaimer

9.3.3 Asymmetric Authentication Key Credentials

Asymmetric authentication key credentials contain either a public and private key pair or only a public key. The private key is used to sign device authentication challenges. The public key is used to verify a device authentication challenge-response.

The PrivateData property in the /oic/sec/cred resource contains the private key.

The PublicData property contains the public key.

The OptionalData property may contain revocation status.

The OIC device implementer should apply hardened key storage techniques that ensure the PrivateData remains private.

Devices should generate asymmetric authentication key pairs internally to ensure the private key is only known by the device. See section 9.3.3.1 for when it is necessary to transport private key material between devices.The OIC device implementer should apply appropriate integrity protection of the /oic/sec/cred resources to prevent unauthorized modifications.

9.3.3.1 External Creation of Asymmetric Authentication Key Credentials

Devices should employ industry-standard high-assurance techniques when allowing off-device key pair creation and provisioning. Use of such key pairs should be minimized, particularly if the key pair is immutable and cannot be changed or replaced after provisioning.

When used as part of onboarding, these key pairs can be used to prove the device possesses the manufacturer-asserted properties in a certificate to convince an OBT or a user to accept onboarding the device. See section 7.3.6 for the owner transfer method that uses such a certificate to authenticate the device, and then provisions new network credentials for use.

9.3.5 Certificate Credentials

Certificate credentials are asymmetric keys that are accompanied by a certificate issued by a credential management service (CMS) or an external certificate authority (CA).

A certificate enrolment protocol is used to obtain a certificate and establish proof-of-possession.

The issued certificate is stored with the asymmetric key credential resource.

Other objects useful in managing certificate lifecycle such as certificate revocation status are associated with the credential resource.

Either an asymmetric key credential resource or a self-signed certificate credential is used to terminate a path validation.

The PrivateData property in the /oic/sec/cred resource contains the private key.

The PublicData property contains the issued certificate.

The OptionalData property may contain revocation status.

The OIC device implementer should apply hardened key storage techniques that ensure the PrivateData remains private.

The OIC device implementer should apply appropriate integrity protection of the /oic/sec/cred resources to prevent unauthorized modifications.

9.4.2 Certificate Format

An OIC certificate format is a subset of X.509 format (version 3 or above) as defined in [RFC5280].

9.4.2.1 Certificate Profile and Fields

The OIC certificate shall support the following fields; `version`, `serialNumber`, `signature`, `issuer`, `validity`, `subject`, `subjectPublicKeyInfo`, `extensions`, `signatureAlgorithm` and `signatureValue`.

- `version`: the version of the encoded certificate
- `serialNumber` : certificate serial number
- `signature`: the algorithm identifier for the algorithm used by the CA to sign this certificate
- `issuer`: the entity that has signed and issued certificates
- `validity`: the time interval during which CA warrants
- `subject`: the entity associated with the subject public key field (deviceID)
- `subjectPublicKeyInfo`: the public key and the algorithm with which key is used
- `extensions`: certificate extensions as defined in section 9.4.2.1.1
- `signatureAlgorithm`: the cryptographic algorithm used by the CA to sign this certificate
- `signatureValue`: the digital signature computed upon the ASN.1 DER encoded `OICtbsCertificate` (this signature value is encoded as a BIT STRING.)

The OIC certificate syntax shall be defined as follows;

```
OICCertificate  ::=  SEQUENCE  {

        OICtbsCertificate       TBSCertificate,
        signatureAlgorithm      AlgorithmIdentifier,
        signatureValue          BIT STRING
}
```

The `OICtbsCertificate` field contains the names of a subject and an issuer, a public key associated with the subject, a validity period, and other associated information. Per RFC5280, version 3 certificates use the value 2 in the version field to encode the version number; the below grammar does not allow version 2 certificates.

```
 OICtbsCertificate  ::=  SEQUENCE  {
        version           [0]  2 or above,
        serialNumber      CertificateSerialNumber,
        signature         AlgorithmIdentifier,
        issuer            Name,
        validity          Validity,
        subject           Name,
    subjectPublicKeyInfo  SubjectPublicKeyInfo,
        extensions    [3] EXPLICIT Extensions
}
```

```
subjectPublicKeyInfo  ::=  SEQUENCE  {
        algorithm            AlgorithmIdentifier,
        subjectPublicKey     BIT STRING
}

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

   Extension  ::=  SEQUENCE  {
        extnID      OBJECT IDENTIFIER,
        critical    BOOLEAN DEFAULT FALSE,
        extnValue   OCTET STRING
                    -- contains the DER encoding of an ASN.1 value
                    -- corresponding to the extension type identified
                    -- by extnID
        }
```

The table below shows the comparison between OIC and X.509 certificate fields.

Certificate Fields		Description	OIC	X.509
OICtbsCertificate	version	2 or above	Mandatory	Mandatory
	serialNumber	CertificateSerialNumber	Mandatory	Mandatory
	signature	AlgorithmIdentifier	1.2.840.10045.4.3.2(ECDSA algorithm with SHA256, Mandatory)	Specified in [RFC3279],[RFC4055], and [RFC4491]
	issuer	Name	Mandatory	Mandatory
	validity	Validity	Mandatory	Mandatory
	subject	Name	Mandatory	Mandatory
	subjectPublicKeyInfo	SubjectPublicKeyInfo	1.2.840.10045.2.1,1.2.840.10045.3.1.7(ECDSA algorithm with SHA256 based on secp256r1 curve, Mandatory)	Specified in [RFC3279],[RFC4055], and [RFC4491]
	issuerUniqueID	IMPLICIT UniqueIdentifier	Not supported	Optional
	subjectUniqueID	IMPLICIT UniqueIdentifier	Not supported	
	extensions	EXPLICIT Extensions	Mandatory	
signatureAlgorithm		AlgorithmIdentifier	1.2.840.10045.4.3.2(ECDSA algorithm with SHA256, Mandatory)	Specified in [RFC3279],[RFC4055], and [RFC4491]
signatureValue		BIT STRING	Mandatory	Mandatory

9.4.2.1.1 Supported Certificate Extensions

As these certificate extensions are a standard part of RFC 5280, this specification includes the section number from that RFC to include it by reference. Each extension is summarized here, and any modifications to the RFC definition are listed. Devices MUST implement and understand the extensions listed here; other extensions from the RFC are not included in this specification and therefore are not required. Section 10.3 describes what devices must implement when validating certificate chains, including processing of extensions, and actions to take when certain extensions are absent.

- Authority Key Identifier (4.2.1.1)

  The Authority Key Identifier (AKI) extension provides a means of identifying the public key corresponding to the private key used to sign a certificate. This specification makes the following modifications to the referenced definition of this extension:

  The authorityCertIssuer or authorityCertSerialNumber fields of the AuthorityKeyIdentifier sequence are not permitted; only keyIdentifier is allowed. This results in the following grammar definition:

```
id-ce-authorityKeyIdentifier OBJECT IDENTIFIER ::=  { id-ce 35 }

AuthorityKeyIdentifier ::= SEQUENCE {
      keyIdentifier             [0] KeyIdentifier           }

KeyIdentifier ::= OCTET STRING
```

- Subject Key Identifier (4.2.1.2)

  The Subject Key Identifier (SKI) extension provides a means of identifying certificates that contain a particular public key.

  This specification makes the following modification to the referenced definition of this extension:

  Subject Key Identifiers SHOULD be derived from the public key contained in the certificate's SubjectPublicKeyInfo field or a method that generates unique values. This specification RECOMMENDS the 256-bit SHA-2 hash of the value of the BIT STRING subjectPublicKey (excluding the tag, length, and number of unused bits). Devices verifying certificate chains must not assume any particular method of computing key identifiers, however, and must only base matching AKI's and SKI's in certification path constructions on key identifiers seen in certificates.

- Key Usage (4.2.1.3)

  The key usage extension defines the purpose (e.g., encipherment, signature, certificate signing) of the key contained in the certificate. The usage restriction might be employed when a key that could be used for more than one operation is to be restricted.

  This specification does not modify the referenced definition of this extension.

- Basic Constraints (4.2.1.9)

The basic constraints extension identifies whether the subject of the certificate is a CA and the maximum depth of valid certification paths that include this certificate. Without this extension, a certificate cannot be an issuer of other certificates.

This specification does not modify the referenced definition of this extension..

- Extended Key Usage (4.2.1.12)

  Extended Key Usage describes allowed purposes for which the certified public key may can be used. When a device receives a certificate, it determines the purpose based on the context of the interaction in which the certificate is presented, and verifies the certificate can be used for that purpose.

  This specification makes the following modifications to the referenced definition of this extension:

  CAs SHOULD mark this extension as critical.

  CAs MUST NOT issue certificates with the anyExtendedKeyUsage OID (2.5.29.37.0).

  The list of OCF-specific purposes and the assigned OIDs to represent them are:
  - Identity certificate W.X.Y.Z.1

9.4.5 Certificate Provisioning

The credential management service (e.g., a hub or a smart phone) issues certificates for new devices. The credential management service shall have its own certificate and key pair. The certificate is either a) self-signed if it acts as Root CA or b) signed by the upper CA in its trust hierarchy if it acts as Sub CA. In either case, the certificate shall have the format described in Section 9.4.2.

The CA in the credential management service shall retrieve a device's public key and proof of possession of the private key, generate a device's certificate signed by this CA certificate, and then the credential management service shall transfer them to the device including its CA certificate chain.

The sequence flow of a certificate transfer for a Client-directed model is described in Figure 17.

1. The credential management service retrieves information from the device that requests a certificate.

The credential management service shall transfer the issued certificate and CA chain to the designated device using the same credid, to maintain the association with the private key.

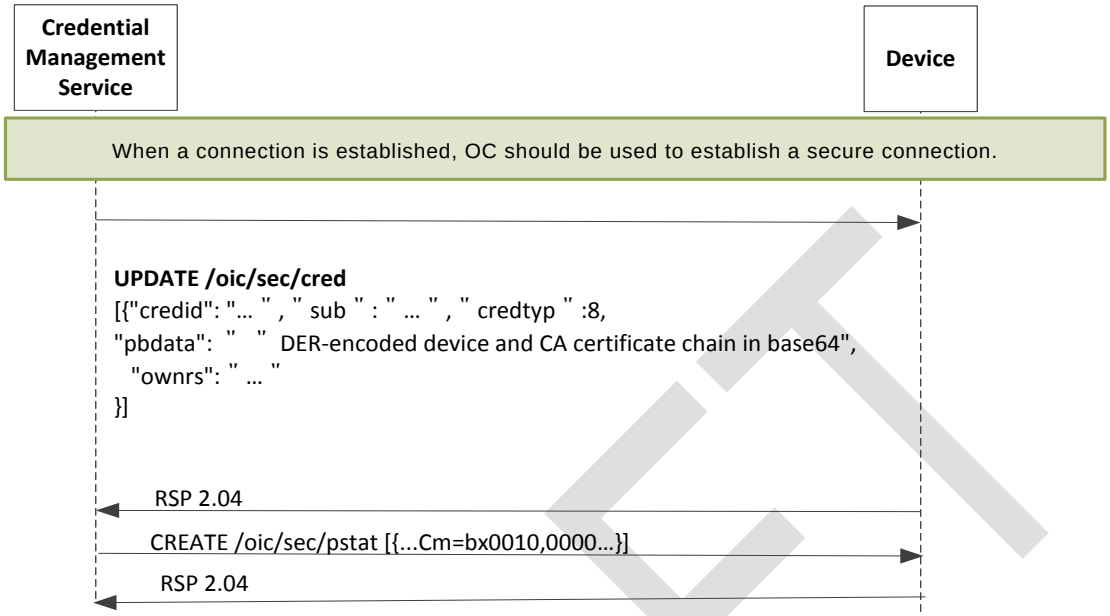

Figure 17 – Client-directed Certificate Transfer

10.3 Device Authentication with Certificates

When using certificates to authenticate, the client and server shall each include their certificate chain, as stored in the appropriate credential, as part of the selected authentication cipher suite. Each device shall validate the certificate chain presented by the peer device. Each certificate signature shall be verified until a public key is found within the /oic/sec/cred resource. Credential resources found in /oic/sec/cred are used to terminate certificate path validation.

Devices must follow the certificate path validation algorithm in section 6 of RFC 5280. In particular:

- For all non-end-entity certificates, devices shall verify that the basic constraints extension is present, and that the cA boolean in the extension is TRUE. If either is false, the certificate chain MUST be rejected. If the pathLenConstraint field is present, devices will confirm the number of certificates between this certificate and the end-entity certificate is less than or equal to pathLenConstraint. In particular, if pathLenConstraint is zero, only an end-entity certificate can be issued by this certificate. If the pathLenConstraint field is absent, there is no limit to the chain length.

- For all non-end-entity certificates, devices shall verify that the key usage extension is present, and that the keyCertSign bit is asserted.

- Devices may use the Authority Key Identifier extension to quickly locate the issuing certificate. Devices MUST NOT reject a certificate for lacking this extension, and must instead attempt validation with the public keys of possible issuer certificates whose subject name equals the issuer name of this certificate.

- The end-entity certificate of the chain shall be verified to contain an Extended Key Usage (EKU) suitable to the purpose for which it is being presented. An end-entity certificate which contains no EKU extension is not valid for any purpose and must be rejected. Any certificate which contains the anyExtendedKeyUsage OID (2.5.29.37.0) must be rejected, even if other valid EKUs are also present.

- Devices MUST verify “transitive EKU” for certificate chains. Issuer certificates (any certificate that is not an end-entity) in the chain MUST all be valid for the purpose for which the certificate chain is being presented. An issuer certificate is valid for a purpose if it contains an EKU extension and the EKU OID for that purpose is listed in the extension, OR it does not have an EKU extension. An issuer certificate SHOULD contain an EKU extension and a complete list of EKUs for the purposes for which it is authorized to issue certificates. An issuer certificate without an EKU extension is valid for all purposes; this differs from end-entity certificates without an EKU extension.

The list of purposes and their associated OIDs are defined in section 9.4.2.1.1.

If the device does not recognize an extension, it must examine the `critical` field. If the field is TRUE, the device MUST reject the certificate. If the field is FALSE, the device MUST treat the certificate as if the extension were absent and proceed accordingly. This applies to all certificates in a chain. Note: Certificate revocation mechanisms are currently out of scope of this version of the specification.