

Industry Contribution for OmniAir WG Review / Recommendation for Standards / Specification, Testing, and Certification

Contribution Type: [Standard Change]

Specification Affected: [IEEE Std 1609.2.1-2022 Section 10.1.4.1]

Description Affected: [appPermissions in enrollment certificates]

Applicable OmniAir WG: [CyberSecurity]

WG Review: [Initial 2024-02-23]

WG Recommendation: [Posted on website until Standards Consideration]

Certification Test Spec Effect: [Incorporated 1609.2.1-2026 published 2026-07-03]

Formatted for Public Posting: [Roebuck]

Document Identifier: [1609.2.1 Guidance Note 1]

Industry Informational Submission to OmniAir WG and its review/recommendation for future consideration as public website posting. OmniAir Consortium does not take any responsible or liability for industry contribution.

Archived 20260707

IEEE Std 1609.2.1-2022 Guidance Note 1: appPermissions in enrollment certificates.

Virendra Kumar and William Whyte, Qualcomm Incorporated

Version 0.4, 2023-05-22

1 Introduction and problem statement

This guidance note is provided by the OmniAir CyberSecurity Working Group to support implementers in the interpretation of the 1609 series of specifications. These notes reflect the understanding and assumptions of the OmniAir Working Group while developing test cases and test plans intended to be used for device certification. OmniAir guidance notes will be revised or eliminated as new versions of the 1609 standards are published by IEEE. This document is advisory only, it is not an extension or modification of any IEEE standard.

The presence or absence of appPermissions field in enrollment certificates is inconsistently specified in IEEE Std 1609.2.1-2022.

The inconsistent text concerns appPermissions in enrollment certificates. In particular, the appPermissions field in the ASN.1 structure EeEcaCertRequest is set to be ABSENT (see 7.3.29 in 1609.2.1-2022), and the enrollment certificate profile states that the appPermissions field shall be present (see 7.6.3.7 in 1609.2.1-2022). The correction presented in this document is to change the appPermissions field in EeEcaCertRequest from ABSENT to PRESENT. This change in turn requires the value of the version field to be incremented from 2 to 3. For future versions of IEEE Std 1609.2.1, the version field value 2 will be deprecated, i.e., the version field will only have values 3 or higher.

The proposed change is to update the descriptions in 7.3.29.

2 Proposed change

7.3.29 EeEcaCertRequest

```
EeEcaCertRequest ::= SEQUENCE {  
    version                Uint8 (3),  
    generationTime         Time32,  
    type                   CertificateType,  
    tbsCert                 ToBeSignedCertificate (WITH COMPONENTS {  
        ...,  
        id (WITH COMPONENTS {  
            ...,  
            linkageData          ABSENT  
        }  
    ),  
    cracaId ('000000'H),  
    crlSeries (0),  
    appPermissions         PRESENT,  
    certIssuePermissions   ABSENT,  
    certRequestPermissions PRESENT,  
    verifyKeyIndicator (WITH COMPONENTS {
```

Industry Informational Submission to OmniAir WG and its review/recommendation for future consideration as public website posting. OmniAir Consortium does not take any responsible or liability for industry contribution.

Archived 20260707

```

        verificationKey
    })
  }),
  canonicalId          IA5String OPTIONAL,
  ...
}

```

This structure contains parameters needed to request an enrollment certificate from the ECA. The ECA may, subject to policy, issue an enrollment certificate with different contents than the contents requested. An overview of this structure is as follows:

- `version` contains the current version of the structure.
- `generationTime` contains the generation time of `EeEcaCertRequest`.
- `type` indicates whether the request is for an explicit or implicit certificate (see 4.1.1 and 4.1.4.3.2).
- `tbsCert` contains the parameters used by the ECA to generate the enrollment certificate.
- **`tbsCert.appPermissions` is set consistent with the enrollment certificate profile in 7.6.3.7.**
- **`tbsCert.certRequestPermissions` is set consistent with the enrollment certificate profile in 7.6.3.7.**
- **`tbsCert.canRequestRollover` is set consistent with the enrollment certificate profile in 7.6.3.7, i.e. it is set if the enrollment certificate is intended to be able to request successor enrollment certificates, see 4.1.4.3.4 and 4.1.4.3.5 for details on successor enrollment certificates including how to request and download them.**
- `tbsCert.verifyKeyIndicator verificationKey` contains the public key information sent by the requester. The `verifyKeyIndicator` field indicates the choice `verificationKey` even if `type` is implicit, as this allows the requester to indicate which signature algorithm and curve they are requesting. The value in this field is used as the verification key in the certificate if the certificate issued in response to this request is explicit, and as the input public key value for implicit certificate generation if the certificate issued in response to this request is implicit.
- `canonicalId` shall be present and contain the canonical identifier for the device per 4.1.4.2 if the enclosing `EeEcaCertRequestSpdu` was signed by the canonical private key. The receiver is intended to use the `canonicalId` to look up the canonical public key to verify the certificate request.

NOTE 1—The `tbsCert.cracaId` and `tbsCert.crlSeries` are set to the indicated values in the corresponding `EeEcaCertRequest`. In the issued enrollment certificate, they may have different values, set by the ECA.

NOTE 2—The `EE` uses the `type` field to indicate whether it is requesting an explicit or an implicit enrollment certificate. A policy is anticipated that determines what type of certificate is appropriate for a given set of circumstances (such as PSIDs, other end entity information, and locality) and that if the `EE` has requested a kind of certificate that is not allowed by policy, the ECA returns an error to the `EE`.

NOTE 3—**In principle the enrollment certificate contents can be set by the ECA to be anything regardless of the contents of `EeEcaCertRequest`. This document recommends that the end entity make a request whose permissions would be of a valid enrollment certificate.**

Industry Informational Submission to OmniAir WG and its review/recommendation for future consideration as public website posting. OmniAir Consortium does not take any responsible or liability for industry contribution.

Archived 20260707

Industry Contribution for OmniAir WG Review / Recommendation for Standards / Specification, Testing, and Certification

Contribution Type: [Standard Change]

Specification Affected: [IEEE Std 1609.2-2022]

Description Affected: [(no) constraints on eeType appearing in certRequestPermissions]

Applicable OmniAir WG: [CyberSecurity]

WG Review: [Initial 2024-02-27]

WG Recommendation: [Posted on website until Standards Consideration]

Certification Test Spec Effect: [Incorporated 1609.2-2025 published 2025-12-10]

Formatted for Public Posting: [Roebuck]

Document Identifier: [1609.2 Guidance Note 2]

Industry Informational Submission to OmniAir WG and its review/recommendation for future consideration as public website posting. OmniAir Consortium does not take any responsible or liability for industry contribution.

Archived 20260707

IEEE Std 1609.2-2022 Guidance Note 2: (no) constraints on eeType appearing in certRequestPermissions

Virendra Kumar and William Whyte, Qualcomm Incorporated

Version 0.1, 2023-07-31

1 Introduction and problem statement

This guidance note is provided by the OmniAir CyberSecurity Working Group to support implementers in the interpretation of the 1609 series of specifications. These notes reflect the understanding and assumptions of the OmniAir Working Group while developing test cases and test plans intended to be used for device certification. OmniAir guidance notes will be revised or eliminated as new versions of the 1609 standards are published by IEEE.

This note identifies an area in IEEE Std 1609.2.1-2022 that has been reported as inconsistent and gives guidance on addressing that inconsistency in the standard.

This note identifies an area in IEEE Std 1609.2-2022 that would benefit from more explicit explanation.

In IEEE Std 1609.2, a certificate can contain the field `certIssuePermissions` which is an instance of `PsidGroupPermissions`.

`PsidGroupPermissions` contains an `eeType` field of type `EndEntityType`. This is a bit string that is constrained to contain at least one non-zero bit and where the non-zero bits are named `app` and `enroll`.

`PsidGroupPermissions` also contains the fields `minChainLength` and `chainLengthRange`, which constrain whether the certificate is allowed to issue an end-entity certificate or a CA certificate (or both)

If `certIssuePermissions.eeType` has the `app` bit set, and `minChainLength` and `chainLengthRange` permit a chain of length 0, then the certificate can issue an end-entity authorization certificate, i.e. a certificate containing `appPermissions`.

If `certIssuePermissions.eeType` has the `enroll` bit set, and `minChainLength` and `chainLengthRange` permit a chain of length 0, then the certificate can issue an end-entity enrollment certificate, i.e. a certificate containing `certRequestPermissions`. The field `certRequestPermissions`, like the field `certIssuePermissions`, is of type `PsidGroupPermissions`.

Industry Informational Submission to OmniAir WG and its review/recommendation for future consideration as public website posting. OmniAir Consortium does not take any responsible or liability for industry contribution.

If minChainLength and chainLengthRange permit a chain of length greater than 0, then the certificate can issue a CA certificate, i.e. a certificate containing certIssuePermissions. In this case, the eeType field in the subordinate (i.e. the issued) CA certificate must be consistent with the eeType field in the issuing CA certificate, i.e. an eeType bit (app or enroll) can only be set in the certIssuePermissions in the subordinate certificate if it is also set in the issuing certificate.

This naturally raises the question: **what are the consistency conditions between eeType in the certIssuePermissions of a CA certificate, and eeType in the certRequestPermissions of an enrollment certificate issued by that CA?**

The answer to this question is: Neither 1609.2 nor 1609.2.1 specifies any such consistency conditions, and so they should not be applied.

However, this answer is not stated explicitly in either 1609.2 or 1609.2.1 and it would be useful for implementers to have such an explicit implementation. The purpose of this Guidance Note is to provide such an explicit statement for 1609.2.

The proposed change is to include the new material in the definition of the EndEntityType structure. It could potentially be included in other locations, for example in ToBeSignedCertificate, but EndEntityType is the most likely location for implementers to look for this information.

2 Proposed change

6.4.33 EndEntityType

```
EndEntityType ::=
    BIT STRING {app (0), enroll (1)} (SIZE (8)) (ALL EXCEPT {})
```

This type indicates which type of permissions may appear in end entity certificates the chain of whose permissions passes through the PsidGroupPermissions field containing this value. If app is indicated, the end entity certificate may contain an appPermissions field. If enroll is indicated, the end entity certificate may contain a certRequestPermissions field.

In the case where the end entity certificate contains a certRequestPermissions field, the values that may appear in the eeType field of that certRequestPermissions field are not constrained by the eeType field in the certIssuePermissions field of the issuing certificate. Any values of eeType in the certRequestPermissions field of the subordinate certificate are consistent with the certIssuePermissions field in the issui

Industry Informational Submission to OmniAir WG and its review/recommendation for future consideration as public website posting. OmniAir Consortium does not take any responsible or liability for industry contribution.

Industry Contribution for OmniAir WG Review / Recommendation for Standards / Specification, Testing, and Certification

Contribution Type: [Standard Change]

Specification Affected: [IEEE Std 1609.2-2022]

Description Affected: [validity period end]

Applicable OmniAir WG: [CyberSecurity]

WG Review: [Initial 2024-02-27]

WG Recommendation: [Posted on website until Standards Consideration]

Certification Test Spec Effect: [Incorporated 1609.2-2025 published 2025-12-10]

Formatted for Public Posting: [Roebuck]

Document Identifier: [1609.2 Guidance Note 03]

IEEE Std 1609.2-2022 Guidance Note 3: validity period end

Virendra Kumar and William Whyte, Qualcomm Incorporated

Version 0.1, 2023-10-04

1 Introduction and problem statement

This guidance note is provided by the OmniAir CyberSecurity Working Group to support implementers in the interpretation of the 1609 series of specifications. These notes reflect the understanding and assumptions of the OmniAir Working Group while developing test cases and test plans intended to be used for device certification. OmniAir guidance notes will be revised or eliminated as new versions of the 1609 standards are published by IEEE.

This note identifies an area in IEEE Std 1609.2.1-2022 that has been reported as inconsistent and gives guidance on addressing that inconsistency in the standard.

This note identifies an area in IEEE Std 1609.2-2022 that is ambiguous and would benefit from clarification and an example.

In IEEE Std 1609.2, the end of a validity period is not clearly specified in clause 6.4.14.

Clause 6.4.14 says “The start of the validity period is given by start and the end is given by start + duration.” Here, start is of type Time32, which is defined as the number of seconds since the epoch; and duration is of type Duration, which is defined as units of time ranging from microseconds to years, and duration contains an indication of the number of those units. This definition does not address whether a time value equal to the end time is inside or outside the validity period.

This can be illustrated using the following examples:

Example 1:

- start = 3 seconds, duration = 5 seconds (i.e., duration unit = seconds)
- Is the time “8 seconds” inside or outside the validity period?
 - If 8 seconds is inside the validity period, then any time value is within the validity period if, when considered with 1-second granularity, it is equal to 8. This means that 8.999... seconds is inside the validity period, meaning that the validity period is effectively 6 seconds rather than 5.

Example 2:

- start = 3 seconds, duration = 5000 milliseconds (i.e., duration unit = milliseconds)
- Is the time “8000 milliseconds” inside or outside the validity period?

- If 8000 milliseconds is inside the validity period, then 8000.9999 milliseconds is within the validity period if it is considered with 1-millisecond granularity. This means that the validity period is effectively 5.001 seconds rather than 5 seconds.

Both these examples illustrate that if a time equal to the end time is in the validity period, then the length of the validity period depends on the units it is expressed in – “1 second” and “1000 milliseconds” mean different durations.

Therefore, as stated, the only interpretation that makes sense is that the end time value of the validity period is not in the validity period.

The proposed changes are two-fold:

1. Explicitly state that the validity period end is exclusive, i.e., does not include the end time value.
2. Provide an example.

2 Proposed change

6.4.14 ValidityPeriod

```
ValidityPeriod ::= SEQUENCE {  
    start      Time32,  
    duration   Duration  
}
```

This type gives the validity period of a certificate. The start of the validity period is given by `start`, and the end is given by `start + duration`. **A time value is within the validity period if it is greater than or equal to the start time, and strictly less than the end time.**

For example, if `start` = 3 seconds, `duration` = 5 seconds,

- **Within the validity period: 3.0 seconds through 7.999... seconds,**
- **Outside the validity period: any time less than 3.0 seconds, 8.0 seconds, any time greater than 8.0 seconds.**

Industry Contribution for OmniAir WG Review / Recommendation for Standards / Specification, Testing, and Certification

Contribution Type: [Standard Change]

Specification Affected: [IEEE Std 1609.2.1-2022 Section 10.1.4.1]

Description Affected: [end entity authentication in API commands]

Applicable OmniAir WG: [CyberSecurity]

WG Review: [Initial 2024-02-23]

WG Recommendation: [Posted on website until Standards Consideration]

Certification Test Spec Effect: [Incorporated 1609.2.1-2026 published 2026-07-03]

Formatted for Public Posting: [Roebuck]

Document Identifier: [1609.2.1 Guidance Note 4]

IEEE Std 1609.2.1-2022 Guidance Note 4: end entity authentication in API commands.

Virendra Kumar and William Whyte, Qualcomm Incorporated

Version 0.2, 2023-07-25

1 Introduction and problem statement

This guidance note is provided by the OmniAir CyberSecurity Working Group to support implementers in the interpretation of the 1609 series of specifications. These notes reflect the understanding and assumptions of the OmniAir Working Group while developing test cases and test plans intended to be used for device certification. OmniAir guidance notes will be revised or eliminated as new versions of the 1609 standards are published by IEEE. This document is advisory only, it is not an extension or modification of any IEEE standard.

This note identifies an area in IEEE Std 1609.2.1-2022 that has been reported to be confusing and also possibly creating a race condition resulting in the end entity (EE) not being able to verify certificates.

This guidance note addresses two issues around EE authentication:

1. The availability of EE authentication options listed in the standard for different application programming interface (API) commands depends on the provider, and the current language in the standard can be confusing to the reader.
2. If a provider requires EE authentication for downloading security management information, then under certain circumstances the EE may not be able to verify certificates.

For EE authentication in different API commands, the standard currently says something like “An EE that wants to use this API command shall authenticate using one of these options.”. However, a Security Credential Management System (SCMS) component that provides the API command may not support all the authentication options listed in the standard. This is discussed in clause 6.3.1.7. This guidance note aims to clarify that end entity needs to pick from the options supported by the SCMS component.

The second issue is around an EE being able to download security management information. If authorization certificates of the EE are revoked, the corresponding enrollment certificate may be blocked. The standard does not require EE authentication for security management information download services: 6.3.5.7 – 6.3.5.14. For example, see 6.3.5.10 that states “An EE that wants to use this API command may authenticate using one of these options.”. However, a SCMS component that provides the API command can impose additional restrictions, e.g., requiring EE authentication using enrollment certificate. If implementations require EE authentication using enrollment certificate, and if the EE’s enrollment certificate is blocked, EE

Industry Informational Submission to OmniAir WG and its review/recommendation for future consideration as public website posting. OmniAir Consortium does not take any responsible or liability for industry contribution.

Archived 20260707

authentication will fail, and the EE won't be able to download the certificate revocation list (CRL), or any other security management information. Therefore, the EE will not be able to verify certificates. This guidance note aims to address this issue by encouraging SCMS implementations to accept blocked enrollment certificates.

Finally, in 6.3.5.1 the standard lists the services provided by the RA, but that list is incomplete. This guidance note amends the list with missing services.

2 Proposed change

4.1.4.3.1 General—Enrollment certificate registration and blocking—Relationship between EE and RA

The assumption in this document is that the enrollment certificate is used as the primary identifier of an EE to the SCMS, in particular to the RA. To prevent an EE from using its enrollment certificate to request certificates via multiple RAs and so receiving more certificates than it is entitled to, the assumption is that RAs will only process requests from EEs whose enrollment certificates are registered with that RA. If access to the RA is authorized using OAuth access tokens (ATs) as specified in 6.2, it is assumed that the OAuth authorization server (OAS) also has access to a list of registered EEs and will only issue ATs to EEs on that list. The list may use the enrollment certificate or any other appropriate identifier to identify registered EEs. This document does not specify interfaces to be used to register the EE or the enrollment certificate with an RA or other SCMS component. An enrollment certificate that is not registered with an RA is referred to as an *unregistered enrollment certificate*.

If an EE is determined to be misbehaving, or otherwise to need certificate privileges to be withdrawn, the RA may be instructed to "block" that EE by marking the EE's enrollment certificate as blocked. If an EE is blocked, the RA shall deny certificate requests and certificate download requests for authorization and successor enrollment certificates. **The RA should allow a blocked EE to download security management information as defined in 4.1.1, as this enables the blocked EE to determine the trust status of other certificates, learn if it has itself been revoked, etc.** The RA may restrict a **blocked** EE from other activities. In this case, the EE's enrollment certificate is referred to as a *blocked enrollment certificate*.

NOTE—How an EE is determined to be misbehaving and who instructs the RA to block the EE are out of scope for this document.

5.4.3 Protocol parameters

The secure session based on ISO/TS 21177 is characterized by the following values of the session security parameters:

- a) *session-scmsAuth* = *iso21177*
- b) *session-eeAuth* =
 - 1) **iso21177-authorization**, if authentication is done by the EE using a valid IEEE 1609.2 authorization certificate
 - 2) **iso21177-enrollment**, if authentication is done by the EE using a valid IEEE 1609.2 enrollment certificate

6.3.5.1 General

The RA provides the following services:

- a) EE-specific services:

Industry Informational Submission to OmniAir WG and its review/recommendation for future consideration as public website posting. OmniAir Consortium does not take any responsible or liability for industry contribution.

- 1) Authorization certificate request (see 6.3.5.2)
 - 2) Authorization certificate download (see 6.3.5.3)
 - 3) Successor enrollment certificate request (see 6.3.5.4)
 - 4) Successor enrollment certificate download (see 6.3.5.5)
 - 5) Misbehavior report submission (see 6.3.5.6)
- b) Security management information download services:
- 1) CCF download (a collection of trusted certificates plus the CTL); see 6.3.5.7)
 - 2) CRL download (a collection of CRLs plus the CTL; see 6.3.5.8)
 - 3) Individual CA certificate download (see 6.3.5.9)
 - 4) Individual CRL download (see 6.3.5.10)
 - 5) CTL download (see 6.3.5.11)
 - 6) RA certificate download (see 6.3.5.12)
 - 7) **MA certificate download (see 6.3.5.13)**
 - 8) **Certificate management information status download (see 6.3.5.14)**

6.3.5.2.2 EE authentication

The specification of this command supports the following options for EE authentication at the:

- a) Session level:
- 1) *session-eeAuth = iso21177-enrollment*
 - 2) *session-eeAuth = tls1.2-x.509*
 - 3) *session-eeAuth = tls1.3-x.509*
- b) Web API level: *webApi-eeAuth = oAuth2.0*
- c) SCMS REST API v3 level:
- 1) *scmsV3-eeAuth = enrollment*
 - 2) *scmsV3-eeAuth = x509*

An individual SCMS component that offers this API command shall support authentication at least at the SCMS REST API v3 level and may additionally support any other of these mechanisms. See 6.3.1.7 for further discussion. An EE that uses this API command shall authenticate at least at the SCMS REST API v3 level and may additionally authenticate using any other mechanism supported by the SCMS component.

6.3.5.3.2.2 EE authentication

The specification of this command supports the following options for EE authentication at the:

- a) Session level:
- 1) *session-eeAuth = iso21177-enrollment*
 - 2) *session-eeAuth = tls1.2-x.509*

Industry Informational Submission to OmniAir WG and its review/recommendation for future consideration as public website posting. OmniAir Consortium does not take any responsible or liability for industry contribution.

Archived 20260707

- 3) *session-eeAuth = tls1.3-x.509*
- b) Web API level: *webApi-eeAuth = oAuth2.0*
- c) SCMS REST API v3 level: *scmsV3-eeAuth = enrollment*

An individual SCMS component that offers this API command shall support authentication using at least one of these mechanisms. See 6.3.1.7 for further discussion. An EE that uses this API command shall authenticate using any mechanism supported by the SCMS component.

6.3.5.3.2 EE authentication

The specification of this command supports the following options for EE authentication at the:

- a) Session level:
 - 1) *session-eeAuth = iso21177-enrollment*
 - 2) *session-eeAuth = tls1.2-x.509*
 - 3) *session-eeAuth = tls1.3-x.509*
- b) Web API level: *webApi-eeAuth = oAuth2.0*
- c) SCMS REST API v3 level: *scmsV3-eeAuth = enrollment*

An individual SCMS component that offers this API command shall support authentication using at least one of these mechanisms. See 6.3.1.7 for further discussion. An EE that uses this API command shall authenticate using any mechanism supported by the SCMS component.

6.3.5.4.2 EE authentication

The specification of this command supports the following options for EE authentication at the:

- a) Session level:
 - 1) *session-eeAuth = iso21177-enrollment*
 - 2) *session-eeAuth = tls1.2-x.509*
 - 3) *session-eeAuth = tls1.3-x.509*
- b) Web API level: *webApi-eeAuth = oAuth2.0*
- c) SCMS REST API v3 level: *scmsV3-eeAuth = enrollment*

An individual SCMS component that offers this API command shall support authentication at least at the SCMS REST API v3 level and may additionally support any other of these mechanisms. See 6.3.1.7 for further discussion. An EE that uses this API command shall authenticate at least at the SCMS REST API v3 level and may additionally authenticate using any other mechanism supported by the SCMS component.

The EE shall not use the `canonicalId` field in the `EeEcaCertRequest` within the `EeRaSuccessorEnrollmentCertRequestSpdu` for authentication and the `canonicalId` field shall be absent for this use case.

6.3.5.5.2 EE authentication

The specification of this command supports the following options for EE authentication at the:

- a) Session level:
 - 1) *session-eeAuth = iso21177-enrollment*

Industry Informational Submission to OmniAir WG and its review/recommendation for future consideration as public website posting. OmniAir Consortium does not take any responsible or liability for industry contribution.

Archived 20260707

- 2) *session-eeAuth = tls1.2-x.509*
- 3) *session-eeAuth = tls1.3-x.509*
- b) Web API level: *webApi-eeAuth = oAuth2.0*
- c) SCMS REST API v3 level: *scmsV3-eeAuth = enrollment*

An individual SCMS component that offers this API command shall support authentication using at least one of these mechanisms. See 6.3.1.7 for further discussion. An EE that uses this API command shall authenticate using any mechanism supported by the SCMS component.

6.3.5.6.2 EE authentication

The specification of this command supports the following options for EE authentication at the:

- Session level: *session-eeAuth = iso21177-authorization*
- Web API level: *webApi-eeAuth = oAuth2.0*
- SCMS REST API v3 level: *scmsV3-eeAuth = enrollment*

An individual SCMS component that offers this API command shall support authentication using at least one of these mechanisms. See 6.3.1.7 for further discussion. An EE that uses this API command shall authenticate using any mechanism supported by the SCMS component.

If the end entity authenticates at the session level using ISO/TS 21177 and an authorization certificate (*session-eeAuth = iso21177-authorization*), the authorization certificate shall contain a misbehavior reporting PSID and that PSID shall appear in the headerInfo of the Ieee1609Dot2Data of type signedData contained in the TLS CertificateVerify message.

6.3.5.7.2 EE authentication

The specification of this command supports the following options for EE authentication at the:

- a) Session level:
 - 1) *session-eeAuth = iso21177-enrollment*
 - 2) *session-eeAuth = tls1.2-x.509*
 - 3) *session-eeAuth = tls1.3-x.509*
- b) Web API level: *webApi-eeAuth = oAuth2.0*

An individual SCMS component that offers this API command may choose not to require authentication or may support one or more of these options. See 6.3.1.7 for further discussion. If the SCMS component supports session-level authentication, it should accept a blocked enrollment certificate if the certificate passes all other verification checks. An EE that uses this API command may authenticate using any mechanism supported by the SCMS component.

6.3.5.8.2 EE authentication

The specification of this command supports the following options for EE authentication at the:

- a) Session level:
 - 1) *session-eeAuth = iso21177-enrollment*
 - 2) *session-eeAuth = tls1.2-x.509*
 - 3) *session-eeAuth = tls1.3-x.509*

- b) Web API level: *webApi-eeAuth = oAuth2.0*

An individual SCMS component that offers this API command may choose not to require authentication or may support one or more of these options. See 6.3.1.7 for further discussion. If the SCMS component supports session-level authentication, it should accept a blocked enrollment certificate if the certificate passes all other verification checks. An EE that uses this API command may authenticate using any mechanism supported by the SCMS component.

6.3.5.9.2 EE authentication

The specification of this command supports the following options for EE authentication at the:

- a) Session level:
- 1) *session-eeAuth = iso21177-enrollment*
 - 2) *session-eeAuth = tls1.2-x.509*
 - 3) *session-eeAuth = tls1.3-x.509*
- b) Web API level: *webApi-eeAuth = oAuth2.0*

An individual SCMS component that offers this API command may choose not to require authentication or may support one or more of these options. See 6.3.1.7 for further discussion. If the SCMS component supports session-level authentication, it should accept a blocked enrollment certificate if the certificate passes all other verification checks. An EE that uses this API command may authenticate using any mechanism supported by the SCMS component.

6.3.5.10.2 EE authentication

The specification of this command supports the following options for EE authentication at the:

- a) Session level:
- 1) *session-eeAuth = iso21177-enrollment*
 - 2) *session-eeAuth = tls1.2-x.509*
 - 3) *session-eeAuth = tls1.3-x.509*
- b) Web API level: *webApi-eeAuth = oAuth2.0*

An individual SCMS component that offers this API command may choose not to require authentication or may support one or more of these options. See 6.3.1.7 for further discussion. If the SCMS component supports session-level authentication, it should accept a blocked enrollment certificate if the certificate passes all other verification checks. An EE that uses this API command may authenticate using any mechanism supported by the SCMS component.

6.3.5.11.2 EE authentication

The specification of this command supports the following options for EE authentication at the:

- a) Session level:
- 1) *session-eeAuth = iso21177-enrollment*
 - 2) *session-eeAuth = tls1.2-x.509*
 - 3) *session-eeAuth = tls1.3-x.509*

- b) Web API level: *webApi-eeAuth* = *oAuth2.0*

An individual SCMS component that offers this API command may choose not to require authentication or may support one or more of these options. See 6.3.1.7 for further discussion. If the SCMS component supports session-level authentication, it should accept a blocked enrollment certificate if the certificate passes all other verification checks. An EE that uses this API command may authenticate using any mechanism supported by the SCMS component.

6.3.5.12.2 EE authentication

The specification of this command supports the following options for EE authentication at the:

- a) Session level:
- 1) *session-eeAuth* = *iso21177-enrollment*
 - 2) *session-eeAuth* = *tls1.2-x.509*
 - 3) *session-eeAuth* = *tls1.3-x.509*
- b) Web API level: *webApi-eeAuth* = *oAuth2.0*

An individual SCMS component that offers this API command may choose not to require authentication or may support one or more of these options. See 6.3.1.7 for further discussion. If the SCMS component supports session-level authentication, it should accept a blocked enrollment certificate if the certificate passes all other verification checks. An EE that uses this API command may authenticate using any mechanism supported by the SCMS component.

6.3.5.13.2 EE authentication

The specification of this command supports the following options for EE authentication at the:

- a) Session level:
- 1) *session-eeAuth* = *iso21177-enrollment*
 - 2) *session-eeAuth* = *tls1.2-x.509*
 - 3) *session-eeAuth* = *tls1.3-x.509*
- b) Web API level: *webApi-eeAuth* = *oAuth2.0*

An individual SCMS component that offers this API command may choose not to require authentication or may support one or more of these options. See 6.3.1.7 for further discussion. If the SCMS component supports session-level authentication, it should accept a blocked enrollment certificate if the certificate passes all other verification checks. An EE that uses this API command may authenticate using any mechanism supported by the SCMS component.

6.3.5.14.2 EE authentication

The specification of this command supports the following options for EE authentication at the:

- a) Session level:
- 1) *session-eeAuth* = *iso21177-enrollment*
 - 2) *session-eeAuth* = *tls1.2-x.509*
 - 3) *session-eeAuth* = *tls1.3-x.509*

- b) Web API level: *webApi-eeAuth = oAuth2.0*

An individual SCMS component that offers this API command may choose not to require authentication or may support one or more of these options. See 6.3.1.7 for further discussion. If the SCMS component supports session-level authentication, it should accept a blocked enrollment certificate if the certificate passes all other verification checks. An EE that uses this API command may authenticate using any mechanism supported by the SCMS component.

Archived

Industry Contribution for OmniAir WG Review / Recommendation for Standards / Specification, Testing, and Certification

Contribution Type: [Standard Change]

Specification Affected: [IEEE Std 1609.2.1-2022 Section 10.1.4.1]

Description Affected: [eeType in enrollment certificates]

Applicable OmniAir WG: [CyberSecurity]

WG Review: [Initial 2024-02-23]

WG Recommendation: [Posted on website until Standards Consideration]

Certification Test Spec Effect: [Incorporated 1609.2.1-2026 published 2026-07-03]

Formatted for Public Posting: [Roebuck]

Document Identifier: [1609.2.1 Guidance Note 05]

IEEE Std 1609.2.1-2022 Guidance Note 5: eeType in enrollment certificates.

Virendra Kumar and William Whyte, Qualcomm Incorporated

Version 0.2, 2023-08-02

1 Introduction and problem statement

This guidance note is provided by the OmniAir CyberSecurity Working Group to support implementers in the interpretation of the 1609 series of specifications. These notes reflect the understanding and assumptions of the OmniAir Working Group while developing test cases and test plans intended to be used for device certification. OmniAir guidance notes will be revised or eliminated as new versions of the 1609 standards are published by IEEE. This document is advisory only, it is not an extension or modification of any IEEE standard.

This note identifies an area in IEEE Std 1609.2.1-2022 that has been reported as limiting for enrollment certificates issued under informal documents preceding this standard, and gives guidance on removing that limitation.

Certificate management interfaces specified prior to IEEE Std 1609.2.1 had less stringent consistency conditions for successor enrollment certificate requests than what is currently specified in 10.1.4.2. In particular, enrollment certificates that were issued following those prior specifications didn't have the eeType value of enroll, which means those enrollment certificates can't be used for successor enrollment certificate requests as per the consistency condition listed in 10.1.4.2 bullet a.1.iii: eeType permits the end entity type enroll.

This consistency condition is sensible on the merits, because it is appropriate to be rigorous about the use of the statements made in certificates. However, because existing enrollment certs did not enforce this condition, if CAs were to enforce the condition now, it would require all existing enrolled devices to obtain new enrollment certificates, which would be expensive and not create a significant benefit in security. As such, it is appropriate to allow current practice but to encourage CAs to migrate towards issuing enrollment certificates that will meet these consistency conditions.

The proposed change is to update the descriptions in 7.6.3.7 and 10.1.4.1.

2 Proposed change

7.6.3.7 Enrollment (IEEE 1609.2) certificate profile

The toBeSigned field within an IEEE 1609.2 enrollment certificate shall meet the following conditions:

Industry Informational Submission to OmniAir WG and its review/recommendation for future consideration as public website posting. OmniAir Consortium does not take any responsible or liability for industry contribution.

Archived 20260707

- a) The `appPermissions` field shall be present and shall contain at least one entry with `PSID = SecurityMgmtPsid` and `SSP = SecurityMgmtSsp` of type `EeSsp`.
- b) The `certRequestPermissions` field shall be present and shall contain at least one entry with the range of permitted PSIDs/SSPs, where the following occurs:
 - 1) `minChainLength` shall be equal to 0.
 - 2) `chainLengthRange` shall be greater than or equal to 0.
 - 3) `eeType` shall include `app` **and may include enroll**.

This entry (or these entries) indicate the maximum set of application permissions for which the enrollment certificate may be used to request authorization certificates.

If the enrollment certificate is to be used to request successor enrollment certificates, then it shall meet the following additional condition:

- c) The `canRequestRollover` field shall be present.

10.1.4.1 General

The successor enrollment certificate request is a `Ieee1609Dot2Data-SignedCertRequest` containing a `SignedCertificateRequest` with `signer` field of type `certificate`. The contents of the `SignedCertificateRequest` are an `EeRaInterfacePdu.eeRaSuccessorEnrollmentCertRequest`, which is of type `EeEcaCertRequestSpdu`. The `EeEcaCertRequestSpdu` is another `Ieee1609Dot2Data-SignedCertRequest` with a `signer` field of type `self`. The contents of the inner `EeEcaCertRequestSpdu` are a `SignedCertificateRequest` that in turn contains an `EcaEeInterfacePdu.eeEcaCertRequest`, which is of type `EeEcaCertRequest`. The `EeEcaCertRequest` contains a `ToBeSignedCertificate` along with other request data. Subclause 10.1.4 defines consistency and other validity conditions for successor enrollment certificate requests.

For a successor enrollment certificate request to be consistent with the signing enrollment certificate (the certificate that signs the outer `SignedCertificateRequest`), the following conditions shall hold:

- a) The `EeEcaCertRequest.generationTime` field shall indicate a time that is within the validity period of the signing certificate.
- b) The `ToBeSignedCertificate.certRequestPermissions` field shall be consistent with the signing certificate's `ToBeSignedCertificate.certRequestPermissions` field as defined below.
- c) The `ToBeSignedCertificate.certRequestPermissions` field shall contain at least one field with `PSID = SecurityMgmtPsid`.
- d) The `ToBeSignedCertificate.region` field shall be equal to or wholly contained within the signing certificate's `ToBeSignedCertificate.region` field.
- e) The request shall meet the common validity conditions of 10.1.2, other than those that are specific to certificate requests signed by certificates.
- f) The contents of the `ToBeSignedCertificate` shall be consistent with the enrollment certificate profile as given in 7.6.3.7. Consistency with the first set of conditions in 7.6.3.7 is required and consistency with the second set of conditions is optional for purposes of determining the validity of the request.
- g) The contents of the signing enrollment certificate shall be consistent with the enrollment certificate profile as given in 7.6.3.7, including both the first and the second set of conditions in that subclause.

There may be a policy addressing the contents of the `ToBeSignedCertificate` in the request. The policy may constrain the presence or absence of any optional field, and may also constrain the values that may appear in fields. For example, the policy may constrain the requested `type`, or the requested `certRequestPermissions`, or the form of the `id` field. As noted above, the definition of the policy and how it is communicated to EEs and to SCMS components is out of scope of this document.

NOTE—CA providers have discretion about which consistency conditions to enforce, and in particular a CA provider has the option not to enforce the `eeType = enroll` consistency condition in 10.1.4.2. This makes it more straightforward to rollover enrollment certificates from ones issued under informal documents preceding this standard to ones more fully compliant with this standard. CAs issuing enrollment certificates in future are encouraged to enable them to conform with the consistency conditions in this standard.

Archived

Industry Contribution for OmniAir WG Review / Recommendation for Standards / Specification, Testing, and Certification

Contribution Type: [Standard Change]

Specification Affected: [IEEE Std 1609.2.1-2022 Section 10.1.4.1]

Description Affected: [raEeCertInfo in aEeEnrollmentCertAckSpdu]

Applicable OmniAir WG: [CyberSecurity]

WG Review: [Initial 2024-02-23]

WG Recommendation: [Posted on website until Standards Consideration]

Certification Test Spec Effect: [Incorporated 1609.2.1-2026 published 2026-07-03]

Formatted for Public Posting: [Roebuck]

Document Identifier: [1609.2.1 Guidance Note 06]

IEEE Std 1609.2.1-2022 Guidance Note 6: raEeCertInfo in aEeEnrollmentCertAckSpdu

Virendra Kumar and William Whyte, Qualcomm Incorporated

Version 0.1, 2023-07-31

1 Introduction and problem statement

This guidance note is provided by the OmniAir CyberSecurity Working Group to support implementers in the interpretation of the 1609 series of specifications. These notes reflect the understanding and assumptions of the OmniAir Working Group while developing test cases and test plans intended to be used for device certification. OmniAir guidance notes will be revised or eliminated as new versions of the 1609 standards are published by IEEE. This document is advisory only, it is not an extension or modification of any IEEE standard.

This note identifies an area in IEEE Std 1609.2.1-2022 that is incorrect and gives guidance on fixing that error.

The ASN.1 structure RaEeEnrollmentCertAckSpdu is incorrectly specified in that the field of EeRaInterfacePdu that should be included in RaEeEnrollmentCertAckSpdu is raEeCertAck instead of the currently specified raEeCertInfo. This was a typo in the standard that can be corrected by changing raEeCertInfo to raEeCertAck in 7.4.21.

Furthermore, the standard as currently specified states only that raEeCertAck is used to acknowledge the receipt of EeRaCertRequestSpdu, when in fact it can also be used to acknowledge the receipt of an EeEcaCertRequestSpdu used to request a successor enrollment certificate.

The proposed change is to update the descriptions in 7.3.32, 7.3.37, and 7.4.21.

2 Proposed change

7.3.32 EeRaInterfacePdu

```
EeRaInterfacePdu ::= CHOICE {  
    eeRaCertRequest      EeRaCertRequest,  
    raEeCertAck           RaEeCertAck,  
    raEeCertInfo          RaEeCertInfo,  
    eeRaDownloadRequest  EeRaDownloadRequest,  
    eeRaSuccessorEnrollmentCertRequest EeEcaCertRequestSpdu,  
    ...  
}
```

This is the parent structure for all structures exchanged between the EE and the RA. An overview of this structure is as follows:

- `eeRaCertRequest` contains the certificate generation request sent by the EE to the RA.
- `raEeCertAck` contains the RA's acknowledgement of the receipt of certificate request: EeRaCertRequestSpdu or EeEcaCertRequestSpdu.
- `raEeCertInfo` contains the information about certificate download.
- `eeRaDownloadRequest` contains the download request sent by the EE to the RA.
- `eeRaSuccessorEnrollmentCertRequest` contains a self-signed request for an enrollment certificate, identical in format to the one submitted for an initial enrollment certificate. (This becomes a request for a successor enrollment certificate by virtue of being signed by the current enrollment certificate.)

7.3.37 RaEeCertAck

```
RaEeCertAck ::= SEQUENCE {
    version            Uint8 (2),
    generationTime     Time32,
    requestHash        HashedId8,
    firstI             IValue OPTIONAL,
    nextDlTime         Time32,
    ...
}
```

This structure is used to create the acknowledgement for certificate requests, as follows:

- `version` contains the current version of the structure.
- `generationTime` contains the generation time of `RaEeCertAck`.
- `requestHash` contains the hash of the corresponding request: EeRaCertRequestSpdu or EeEcaCertRequestSpdu.
- `firstI` shall be present and contain the i-value that will be associated with the first certificate or the certificate batch that will be made available to the EE if the corresponding `EeRaCertRequest` uses the butterfly key mechanism as indicated in the field `additionalParams`. The EE uses this to form the download filename for the download request as specified in 8.2.2.
- `nextDlTime` contains the time after which the EE should connect to the RA to download the certificates.

7.4.21 RaEeEnrollmentCertAckSpdu

```
RaEeEnrollmentCertAckSpdu ::= Ieee1609Dot2Data-Signed {
    ScmsPdu-Scoped {
        EeRaInterfacePdu (WITH COMPONENTS {
            raEeCertInforaEeCertAck (WITH COMPONENTS {
                acpeTreeIdfirstI ABSENT
            })
        })
    },
    SecurityMgmtPsid
}
```

This structure is the SPDU used to send a signed ~~RaEeCertInfo~~RaEeCertAck. For the signature to be valid, the signing certificate shall conform to the RA certificate profile given in 7.6.3.10.

Industry Informational Submission to OmniAir WG and its review/recommendation for future consideration as public website posting. OmniAir Consortium does not take any responsible or liability for industry contribution.

Archived

Industry Contribution for OmniAir WG Review / Recommendation for Standards / Specification, Testing, and Certification

Contribution Type: [Standard Change]

Specification Affected: [IEEE Std 1609.2.1-2022 Section 10.1.4.1]

Description Affected: [(no) constraints on eeType appearing in certRequestPermissions]

Applicable OmniAir WG: [CyberSecurity]

WG Review: [Initial 2024-02-23]

WG Recommendation: [Posted on website until Standards Consideration]

Certification Test Spec Effect: [Incorporated 1609.2.1-2026 published 2026-07-03]

Formatted for Public Posting: [Roebuck]

Document Identifier: [1609.2.1 Guidance Note 07]

IEEE Std 1609.2.1-2022 Guidance Note 7: (no) constraints on eeType appearing in certRequestPermissions

Virendra Kumar and William Whyte, Qualcomm Incorporated

Version 0.1, 2023-07-31

1 Introduction and problem statement

This guidance note is provided by the OmniAir CyberSecurity Working Group to support implementers in the interpretation of the 1609 series of specifications. These notes reflect the understanding and assumptions of the OmniAir Working Group while developing test cases and test plans intended to be used for device certification. OmniAir guidance notes will be revised or eliminated as new versions of the 1609 standards are published by IEEE. This document is advisory only, it is not an extension or modification of any IEEE standard.

This note identifies an area in IEEE Std 1609.2-2022 that would benefit from more explicit explanation.

In IEEE Std 1609.2, a certificate can contain the field `certIssuePermissions` which is an instance of `PsidGroupPermissions`.

`PsidGroupPermissions` contains an `eeType` field of type `EndEntityType`. This is a bit string that is constrained to contain at least one non-zero bit and where the non-zero bits are named `app` and `enroll`.

`PsidGroupPermissions` also contains the fields `minChainLength` and `chainLengthRange`, which constrain whether the certificate is allowed to issue an end-entity certificate or a CA certificate (or both)

If `certIssuePermissions.eeType` has the `app` bit set, and `minChainLength` and `chainLengthRange` permit a chain of length 0, then the certificate can issue an end-entity authorization certificate, i.e. a certificate containing `appPermissions`.

If `certIssuePermissions.eeType` has the `enroll` bit set, and `minChainLength` and `chainLengthRange` permit a chain of length 0, then the certificate can issue an end-entity enrollment certificate, i.e. a certificate containing `certRequestPermissions`. The field `certRequestPermissions`, like the field `certIssuePermissions`, is of type `PsidGroupPermissions`.

If `minChainLength` and `chainLengthRange` permit a chain of length greater than 0, then the certificate can issue a CA certificate, i.e. a certificate containing `certIssuePermissions`. In this Industry Informational Submission to OmniAir WG and its review/recommendation for future consideration as public website posting. OmniAir Consortium does not take any responsible or liability for industry contribution.

Archived 20260707

case, the eeType field in the subordinate (i.e. the issued) CA certificate must be consistent with the eeType field in the issuing CA certificate, i.e. an eeType bit (app or enroll) can only be set in the certIssuePermissions in the subordinate certificate if it is also set in the issuing certificate.

This naturally raises the question: **what are the consistency conditions between eeType in the certIssuePermissions of a CA certificate, and eeType in the certRequestPermissions of an enrollment certificate issued by that CA?**

The answer to this question is: Neither 1609.2 nor 1609.2.1 specifies any such consistency conditions, and so they should not be applied.

However, this answer is not stated explicitly in either 1609.2 or 1609.2.1 and it would be useful for implementers to have such an explicit implementation. The purpose of this Guidance Note is to provide such an explicit statement for 1609.2.

The proposed change is to note this in clause 10.1.2 of 1609.2.1, "Common Validity Conditions".

2 Proposed change

10.1.2 Common validity conditions

For a signed certificate request of any type to be valid, it shall meet the following conditions:

- a) It is correctly formed using the data structures of this document.
- a) The signature on the certificate request verifies with the appropriate certificate or public key, as specified in IEEE Std 1609.2. See 7.3.44 and 7.3.45 for more details on signature calculation.
- b) If the certificate request was signed by an IEEE 1609.2 certificate:
 - 1) The certificate has not expired.
 - 2) The certificate has not been revoked or otherwise determined to be untrustworthy. (How the ECA or RA determines whether a certificate has been determined to be untrustworthy without using revocation lists is out of scope of this standard.)
 - 3) There is a certificate chain that leads from the signing certificate to a known trust anchor, constructed as specified in IEEE Std 1609.2 subclause 5.1.2.1, such that:
 - i) All of the certificates in the chain are correctly formed using the data structures of IEEE Std 1609.2 Clause 6.
 - ii) All certificates in the chain pass cryptographic verification with the appropriate public keys as specified in IEEE Std 1609.2 subclause 5.1.2.3.
 - iii) The certificate chain is internally consistent as specified in IEEE Std 1609.2 subclause 5.1.2.4. (**see note below**)
 - iv) None of the certificates in the chain have been revoked as specified in IEEE Std 1609.2 subclause 5.1.3.

NOTE--- As specified in IEEE 1609.2, there are no consistency conditions between the eeType field(s) in the certRequestPermissions field of the enrollment certificate and the eeType field(s) in the certIssuePermissions field of the issuing CA certificate. Any value of eeType that appears in the certRequestPermissions field of

Archived

Industry Contribution for OmniAir WG Review / Recommendation for Standards / Specification, Testing, and Certification

Contribution Type: [Standard Change]

Specification Affected: [IEEE Std 1609.2.1-2022 Section 10.1.4.1]

Description Affected: [no such thing as PsidSsp with value all]

Applicable OmniAir WG: [CyberSecurity]

WG Review: [Initial 2024-02-23]

WG Recommendation: [Posted on website until Standards Consideration]

Certification Test Spec Effect: [Incorporated 1609.2.1-2026 published 2026-07-03]

Formatted for Public Posting: [Roebuck]

Document Identifier: [1609.2.1 Guidance Note 08]

IEEE Std 1609.2.1-2022 Guidance Note 8: no such thing as PsidSsp with value all

Virendra Kumar and William Whyte, Qualcomm Incorporated

Version 0.1, 2023-08-08

1 Introduction and problem statement

This guidance note is provided by the OmniAir CyberSecurity Working Group to support implementers in the interpretation of the 1609 series of specifications. These notes reflect the understanding and assumptions of the OmniAir Working Group while developing test cases and test plans intended to be used for device certification. OmniAir guidance notes will be revised or eliminated as new versions of the 1609 standards are published by IEEE. This document is advisory only, it is not an extension or modification of any IEEE standard.

This note fixes an incorrect part of IEEE Std 1609.2.1-2022.

1609.2.1 clause 10.1.7 defines consistency conditions between the appPermissions field in an authorization certificate request, and the certRequestPermissions field in the enrollment certificate that signs the request.

Clause 10.1.7.2 defines the concept of an “atomic PsidSsp”, which is a single “permissions instance” in the request. This concept is used to establish consistency: a requirement for consistency between the request and the signing enrollment certificate is that every atomic PsidSsp in the request is permitted by the enrollment certificate.

Clause 10.1.7.2.1, and clause 10.1.7.2.2, address consistency in the case where an atomic PsidSsp in the request takes the value “all”. This is where the error comes in: in 1609.2 there is no way for an appPermissions field to indicate the value “all”. (“All” would presumably mean that the certificate holder is entitled to do any conceivable activity within any conceivable application, so it makes sense that it isn’t defined as a PsidSsp value).

Since there is no way for an atomic PsidSsp to take the value “all”, the material saying what to do in this case cannot be implemented and is confusing to implementers. Therefore it should be removed.

This Guidance Note proposes changes that remove this material.

(For the reader’s information: there are two similar structures in IEEE 1609.2 which can take the value “all”. PsidSspRange.subjectPermissions can take the value “all”, indicating that the issuing CA certificate or requesting enrollment certificate can issue or request any PSID and SSP value.

Industry Informational Submission to OmniAir WG and its review/recommendation for future consideration as public website posting. OmniAir Consortium does not take any responsible or liability for industry contribution.

Archived 20260707

And PsidSspRange.sspRange can take the value “all”, indicating the issuing CA certificate or requesting enrollment certificate can issue or request any SSP value associated with the specified PSID. These cases of “all” are correctly handled in 1609.2.1. It seems that confusion with these cases is what led to the error corrected in this guidance note).

2 Proposed change

2.1 Summary

The change is:

- Remove the bullet in 10.1.7.2.1 that lists “all” as an option for an atomic PsidSsp
- Remove the contents of subclause 10.1.7.2.2 and replace the title of subclause 10.1.7.2.2 with “(void)”. This keeps the subclause number as a referenceable value, meaning that there is no need to renumber subsequent subclauses (and risk breaking references elsewhere in the document).

This removes all references to PsidSsp taking the value “all”.

2.2 Changes specified as markup

10.1.7.2 Atomic PsidSsp

10.1.7.2.1 General

An *atomic PsidSsp* is one of the following:

- “All”
- A single PSID with SSP equal to “all”
- A single PSID with a BitmapSsp
- A single PSID with a single opaque SSP value

Whether an atomic PsidSsp *ps* is permitted by a signing PsidGroupPermission *spg* is determined as per 10.1.7.2.

10.1.7.2.2 ~~*ps* is “all”~~ (void)

~~If *ps* is “all”: *ps* is permitted by *spg* if *spg.subjectPermissions* is all.~~

Industry Contribution for OmniAir WG Review / Recommendation for Standards / Specification, Testing, and Certification

Contribution Type: [Standard Change]

Specification Affected: [IEEE Std 1609.2.1-2022 Section 10.1.4.1]

Description Affected: [validity periods of authorization certificates]

Applicable OmniAir WG: [CyberSecurity]

WG Review: [Initial 2024-02-23]

WG Recommendation: [Posted on website until Standards Consideration]

Certification Test Spec Effect: [Incorporated 1609.2.1-2026 published 2026-07-03]

Formatted for Public Posting: [Roebuck]

Document Identifier: [1609.2.1 Guidance Note 09]

IEEE Std 1609.2.1-2022 Guidance Note 9: validity periods of authorization certificates

Virendra Kumar and William Whyte, Qualcomm Incorporated

Version 0.2, 2023-09-06

1 Introduction and problem statement

This guidance note is provided by the OmniAir CyberSecurity Working Group to support implementers in the interpretation of the 1609 series of specifications. These notes reflect the understanding and assumptions of the OmniAir Working Group while developing test cases and test plans intended to be used for device certification. OmniAir guidance notes will be revised or eliminated as new versions of the 1609 standards are published by IEEE. This document is advisory only, it is not an extension or modification of any IEEE standard.

This note identifies an area in IEEE Std 1609.2.1-2022 that is confusing and gives guidance on addressing that confusion.

The guidance given in clause 7.3.33 of the standard for use of validityPeriod is confusing. It currently says “validityPeriod contains the requested validity period of the first batch of certificates.”. From this description, it is not clear:

- How does this validity period map to the subsequent certificate batches?
- What if there are multiple different validity periods within a batch or i-Period?

Furthermore, the validity period for pseudonym certificates will typically be set by the system policy rather than by individual end entities, so that differences in validity period duration can't be used to distinguish between senders. There may be systemwide policy on when certificate batches can start.

Finally, the ASN.1 structure EeRaCertRequest can be used to request both individual certificates and a series of batches of certificates via the butterfly key mechanism as specified in 9.3.

The proposed change is to add the clarifying text to the bullet corresponding to the validityPeriod.

2 Proposed change

7.3.33 EeRaCertRequest

```
EeRaCertRequest ::= SEQUENCE {  
    version          Uint8 (2),  
    generationTime   Time32,
```

Industry Informational Submission to OmniAir WG and its review/recommendation for future consideration as public website posting. OmniAir Consortium does not take any responsible or liability for industry contribution.

Archived 20260707

```

type          CertificateType,
tbsCert       ToBeSignedCertificate (WITH COMPONENTS {
    ...,
    cracaId ('000000'H),
    crlSeries (0),
    appPermissions PRESENT,
    certIssuePermissions ABSENT,
    certRequestPermissions ABSENT,
    verifyKeyIndicator (WITH COMPONENTS {
        verificationKey
    })
}),
additionalParams AdditionalParams OPTIONAL,
...
}

```

This structure contains parameters needed to request different types of authorization certificates. An overview of this structure is as follows:

- `version` contains the current version of the structure.
- `generationTime` contains the generation time of `EeRaCertRequest`.
- `type` indicates whether the request is for an explicit or implicit certificate (see 4.1.1 and 4.1.4.3.2).
- `tbsCert` contains the parameters to be used by the ACA to generate authorization certificate(s).
 - 1) `id` contains the identity information sent by the requester. If the type is `LinkageData`, the contents of the field are chosen by the EE using any appropriate means. RA replaces that in the certificates with the linkage values generated with the help of the LAs and the ACA; see Annex F.
 - 2) `validityPeriod` contains the requested validity period of the first batch of certificates:
 - i) **If the request is for an individual certificate, the `validityPeriod` in the request is the `validityPeriod` that is being requested to appear in the certificate.**
 - ii) **If the request is for a series of certificate batches to be generated using the butterfly key mechanism as specified in 9.3, the `validityPeriod` in the request is the `validityPeriod` that is being requested to appear in the first certificate in the batch. This shall be consistent with the time period parameters. The end entity that is making the request is expected to know or be able to derive the validity periods that are acceptable for inclusion in the request. If the `validityPeriod` in the request is not correct, the RA is expected to return the corresponding error code.**
 - 3) `region`, `assuranceLevel`, `canRequestRollover`, and `encryptionKey`, if present, contain the information sent by the requester for the requested certificates.
 - 4) `verifyKeyIndicator.verificationKey` contains the public key information sent by the requester. The `verifyKeyIndicator` field indicates the choice `verificationKey` even if type is implicit, as this allows the requester to indicate which signature algorithm and curve they are requesting.
 - i) If the certificate issued in response to this request is explicit and butterfly expansion is not used, the value in this field is the verification key that appears in that certificate.
 - ii) If the certificate issued in response to this request is implicit and butterfly expansion is not used, the value in this field is the input public key value for implicit certificate generation.

- iii) If butterfly expansion is used, that is, if one of (original, unified, or compactUnified) options is present in the field `additionalParams`, the value in this field is combined with the values in the `additionalParams` field as specified in 9.3.
- `additionalParams` shall be present and contain relevant parameters if the requested certificates are to be generated using the butterfly key mechanism as specified in 9.3, or if the requested certificates are to be encrypted without using the butterfly key mechanism. If present, the field `tbsCert.verifyKeyIndicator` shall be used as the caterpillar public key for signing in the butterfly key mechanism.

The definition of validity for a certificate request, including constraints on the fields in this structure, is specified in 10.1.

NOTE 1—In the case where the butterfly key mechanism is used to derive the certificate encryption key, the value j is not communicated to the ACA. However, the EE that receives the certificate response can only decrypt the response if it knows j . The RA is therefore anticipated to store j so that it can be associated with the appropriate certificate response.

NOTE 2—If the type of `id` is `LinkageData`, the contents of the field in the request are replaced by random data by the RA when it sends the individual certificate requests to the ACA. The ACA then in turn replaces that data with the linkage values generated with the help of the LAs; see Annex F.

NOTE 3—This standard does not specify a means to apply butterfly key expansion to encryption keys. If a certificate request contains an encryption key, that encryption key appears in all certificates associated with the request. In the case where a butterfly key certificate request is being made to obtain multiple certificates per time period for unlinkability reasons, this means that a requester is advised not to submit a request containing an encryption key, as the encryption key will be constant across all certificates and allow them to be linked. In the case where a butterfly key certificate request is being made to obtain a single certificate per time period, there is no unlinkability concern related to the inclusion of an encryption key in the request and a requester can include or not include an encryption key according to the needs of the application.

Industry Contribution for OmniAir WG Review / Recommendation for Standards / Specification, Testing, and Certification

Contribution Type: [Standard Change]

Specification Affected: [IEEE Std 1609.2.1-2022 Section 10.1.4.1]

Description Affected: [CTL download]

Applicable OmniAir WG: [CyberSecurity]

WG Review: [Initial 2024-02-23]

WG Recommendation: [Posted on website until Standards Consideration]

Certification Test Spec Effect: [Incorporated 1602.2.1-2026 published 2026-07-03]

Formatted for Public Posting: [Roebuck]

Document Identifier: [1609.2.1 Guidance Note 10]

Industry Informational Submission to OmniAir WG and its review/recommendation for future consideration as public website posting. OmniAir Consortium does not take any responsible or liability for industry contribution.

Archived 20260707

IEEE Std 1609.2.1-2022 Guidance Note 10: CTL download.

Virendra Kumar and William Whyte, Qualcomm Incorporated

Version 0.1, 2023-09-07

1 Introduction and problem statement

This guidance note is provided by the OmniAir CyberSecurity Working Group to support implementers in the interpretation of the 1609 series of specifications. These notes reflect the understanding and assumptions of the OmniAir Working Group while developing test cases and test plans intended to be used for device certification. OmniAir guidance notes will be revised or eliminated as new versions of the 1609 standards are published by IEEE. This document is advisory only, it is not an extension or modification of any IEEE standard.

This note identifies an area in IEEE Std 1609.2.1-2022 that is ambiguous and gives guidance on removing that ambiguity.

For the CTL download service specified in 6.3.5.11, the query parameter `ctlSequenceNumber` is optional, i.e., the requesting EE can omit it in the request. If the CTL download request does not contain the `ctlSequenceNumber`, the standard is not clear on which CTL file is returned by the server.

The proposed change is to clarify that if the `ctlSequenceNumber` is absent from the request, the server returns the most recent CTL (that otherwise matches all other query parameter values) known to the server.

2 Proposed change

6.3.5.11.1 General

The API details for the CTL download service are presented in Table 20. In the table, the query parameter `ctlSeriesId` is the 16 characters long hex encoding of the `CtlSeriesId` of the CTL that the end entity is requesting, and the query parameter `ctlSequenceNumber` is the minimum length integer encoding of the `CtlSequenceNumber` of the CTL that the end entity is requesting. The query parameter `ctlSequenceNumber` is optional. **If `ctlSequenceNumber` is absent from the request, the CTL download server shall return the most recent CTL (that otherwise matches all other query parameter values) known to the server.** The encoding does not include spaces and the API command is not case-sensitive.

Example: `ctlSeriesId` = 0123456789ABCDEF, `ctlSequenceNumber` = 1.

If this API command is made available over TLS, all certificates presented as part of the TLS authentication shall conform with the certificate acceptance policy as discussed in 5.3.1.

Industry Informational Submission to OmniAir WG and its review/recommendation for future consideration as public website posting. OmniAir Consortium does not take any responsible or liability for industry contribution.

Archived 20260707

Table 20—CTL download

Path	<code>ctl?ctlSeriesId = [CtlSeriesId of the [CTL]&ctlSequenceNumber = [CtlSequenceNumber of the CTL].</code> The query parameter <code>sequenceNumber</code> is optional, so the path without the <code>ctlSequenceNumber</code> will look like: <code>ctl?ctlSeriesId = [CtlSeriesId of the CTL].</code>
HTTP method	GET.
HTTP request body	Empty.
HTTP request headers	If <i>webApi-eeAuth = oAuth2.0</i> : — Authorization: Bearer {access-token/AT}.
HTTP response body	The response depends on whether or not the query parameter <code>ctlSequenceNumber</code> was provided in the request: — If the <code>ctlSequenceNumber</code> was provided in the request, a CTL file as defined in 8.6, where the <code>multiSignedCtl.tbsCtl.FullIeeeTbsCtl.ctlSeriesId</code> shall be equal to the query parameter <code>ctlSeriesId</code> and <code>multiSignedCtl.tbsCtl.FullIeeeTbsCtl.sequenceNumber</code> shall be equal to the query parameter <code>ctlSequenceNumber</code> provided in the request path. — If the <code>ctlSequenceNumber</code> was not provided in the request, the most recent CTL file known to the server as defined in 8.6, where the <code>multiSignedCtl.tbsCtl.FullIeeeTbsCtl.ctlSeriesId</code> shall be equal to the query parameter <code>ctlSeriesId</code> provided in the request path.
HTTP response header	If <i>scmsV3-error = fine</i> and the RA returns 400 or 403, the <i>Ieee-1609.2.1-Error</i> header indicates the issue: — 400-81 wrong RA.
Errors	a) If <i>scmsV3-error = coarse</i> : 500 internal server error. b) If <i>scmsV3-error = fine</i> : 1) 400 bad request: see response header for details. 2) 401 unauthorized: the authorization mechanism used was not accepted (can be generated for the OAuth token for profiles with <i>webApi-eeAuth = oAuth2.0</i>). 3) 403 forbidden: see response header for details. 4) 404 not found: the resource at the given URL does not exist. 5) 405 method not allowed: retry with HTTP GET. 6) 408 request timeout: retry after <i>download-minWait</i> and up to <i>download-maxReqs</i> times. 7) 416 requested range not satisfiable: retry without HTTP Range. 8) 429 too many requests. 9) 500 internal server error. 10) 503 service unavailable: retry after <i>download-minWait</i> and up to <i>download-maxReqs</i> times.

Industry Contribution for OmniAir WG Review / Recommendation for Standards / Specification, Testing, and Certification

Contribution Type: [Standard Change]

Specification Affected: [IEEE Std 1609.2.1-2022 Section 10.1.4.1]

Description Affected: [no dashes in butterfly certificate filenames]

Applicable OmniAir WG: [CyberSecurity]

WG Review: [Initial 2024-02-23]

WG Recommendation: [Posted on website until Standards Consideration]

Certification Test Spec Effect: [Incorporated 1609.2.1-2026 published 2026-07-03]

Formatted for Public Posting: [Roebuck]

Document Identifier: [1609.2.1 Guidance Note 11]

Industry Informational Submission to OmniAir WG and its review/recommendation for future consideration as public website posting. OmniAir Consortium does not take any responsible or liability for industry contribution.

Archived 20260707

IEEE Std 1609.2.1-2022 Guidance Note 11: no dashes in butterfly certificate filenames

Virendra Kumar and William Whyte, Qualcomm Incorporated

Version 0.1, 2023-09-22

1 Introduction and problem statement

This guidance note is provided by the OmniAir CyberSecurity Working Group to support implementers in the interpretation of the 1609 series of specifications. These notes reflect the understanding and assumptions of the OmniAir Working Group while developing test cases and test plans intended to be used for device certification. OmniAir guidance notes will be revised or eliminated as new versions of the 1609 standards are published by IEEE. This document is advisory only, it is not an extension or modification of any IEEE standard.

This note identifies an area in IEEE Std 1609.2.1-2022 that is confusing and gives guidance on removing that confusion.

The naming convention for butterfly certificate files specified in 8.2.2 says that the filename is formed as `h_i.zip`, where `h` and `i` are the certificate request hash and the `i`-value, respectively. That is, the certificate request hash and the `i`-value in the filename are separated by an underscore (`_`). However, the filename example given in Table 11 and Table 12, “AB09281C9867DE53-9AB.zip”, has a dash (`-`) between the certificate request hash and the `i`-value.

The proposed changes are to correct the example given in Table 11 and Table 12 by replacing the dash with an underscore.

2 Proposed change

6.3.5.3.2.3 SCMS component authentication

The specification of this command supports the following options for SCMS component authentication at the session level:

- `session-scmsAuth = iso21177`
- `session-scmsAuth = tls1.2`
- `session-scmsAuth = tls1.3`

An SCMS component that offers this API command shall authenticate using at least one of these options.

Industry Informational Submission to OmniAir WG and its review/recommendation for future consideration as public website posting. OmniAir Consortium does not take any responsible or liability for industry contribution.

Archived 20260707

Table 11—Authorization certificate download without filename in URL

Path	authorization-certificate.
HTTP method	GET.
HTTP request body	Empty.
HTTP request headers	a) Depending on <i>webApi-eeAuth</i>: 1) If <i>webApi-eeAuth</i> = <i>oAuth2.0</i>: Authorization: Bearer {access-token/AT}. 2) If <i>webApi-eeAuth</i> = <i>none</i>: no headers due to this parameter. b) Depending on <i>scmsV3-eeAuth</i>: 1) If <i>scmsV3-eeAuth</i> = <i>enrollment</i>: IEEE-1609.2.1-Authorization: EeRaDownloadRequestSpdu. 2) If <i>scmsV3-eeAuth</i> = <i>none</i>: IEEE-1609.2.1-Authorization: EeRaDownloadRequestPlainSpdu.
HTTP response body	An authorization certificate download file as defined in 8.2, where the filename shall be equal to the <i>eeRaDownloadRequest.filename</i> field of the IEEE-1609.2.1-Authorization provided in the request headers.
HTTP response header	The Content-Disposition header is present and has type “attachment”. It is used to convey the name of the file being downloaded. The filename is derived as specified in 8.2, for example, Content-Disposition: attachment; filename = “AB09281C9867DE53_9AB.zip” “AB09281C9867DE53_9AB.zip”. If <i>scmsV3-error</i> = <i>fine</i> and the RA returns 400 or 403, the IEEE-1609.2.1-Error header indicates the issue: — 400-10 absent encryption. — 400-20 blocked enrollment certificate. — 400-21 unregistered enrollment certificate. — 400-40 failed certificate chain verification. — 400-41 failed decryption. — 400-42 failed parsing. — 403-43 failed signature verification. — 400-50 future generationTime. — 400-62 invalid enrollment certificate. — 400-74 outside <i>download-maxAge</i> generationTime. — 400-75 outside <i>download-maxReqs</i>. — 400-76 within <i>download-minWait</i>. — 400-712 outside enrollment certificate’s validityPeriod. — 400-81 wrong RA.
Errors	a) If <i>scmsV3-error</i> = <i>coarse</i>: 500 internal server error. b) If <i>scmsV3-error</i> = <i>fine</i>: 1) 400 bad request: see response header for details. 2) 401 unauthorized: the authorization mechanism used was not accepted (can be generated for the OAuth token for profiles with <i>webApi-eeAuth</i> = <i>oAuth2.0</i>, or for failures of SPDU-level authorization for profiles with <i>scmsV3-eeAuth</i> = <i>enrollment</i>). 3) 403 forbidden: see response header for details. 4) 404 not found: the resource at the given URL does not exist. Either <i>requestHash</i> or <i>time-iPeriod</i> in the <i>EeRaDownloadRequest</i> is wrong or the requested file is not generated yet. 5) 405 method not allowed: retry with HTTP GET. 6) 408 request timeout: retry after <i>download-minWait</i> and up to <i>download-maxReqs</i> times. 7) 416 requested range not satisfiable: retry without HTTP Range. 8) 429 too many requests. 9) 500 internal server error. 10) 503 service unavailable: retry after <i>download-minWait</i> and up to <i>download-maxReqs</i> times.

Industry Informational Submission to OmniAir WG and its review/recommendation for future consideration as public website posting. OmniAir Consortium does not take any responsible or liability for industry contribution.

6.3.5.3.3.3 SCMS component authentication

The specification of this command supports the following options for SCMS component authentication at the session level:

- *session-scmsAuth* = *iso21177*
- *session-scmsAuth* = *tls1.2*
- *session-scmsAuth* = *tls1.3*

An SCMS component that offers this API command shall authenticate using at least one of these options.

NOTE—In the cases where *scmsV3-eeAuth* is not *enrollment*, there is no need for an SPDU indicating the filename because the filename is in the URL.

Table 12—Authorization certificate download with filename in URL

Path	authorization-certificate?filename = [filename value], where filename value is as specified in 8.2.
HTTP method	GET.
HTTP request body	Empty.
HTTP request headers	a) Depending on <i>webApi-eeAuth</i>: 1) If <i>webApi-eeAuth</i> = <i>oAuth2.0</i>: Authorization: Bearer {access-token/AT}. 2) If <i>webApi-eeAuth</i> = <i>none</i>: no headers due to this parameter. b) Depending on <i>scmsV3-eeAuth</i>: 1) If <i>scmsV3-eeAuth</i> = <i>enrollment</i>: IEEE-1609.2.1-Authorization: EeRaDownloadRequestSpdu. The request is invalid if the <i>eeRaDownloadRequest.filename</i> field is not equal to the query parameter filename provided in the request path. 2) If <i>scmsV3-eeAuth</i> = <i>none</i>: no headers due to this parameter.
HTTP response body	An authorization certificate download file as defined in 8.2, where the filename shall be equal to the query parameter filename provided in the request path.
HTTP response header	The Content-Disposition header is present and has type "attachment". It is used to convey the name of the file being downloaded. The filename is derived as specified in 8.2, for example, Content-Disposition: attachment; filename = "~~AB09281C9867DE53-9AB.zip~~" "AB09281C9867DE53_9AB.zip". If <i>scmsV3-error</i> = <i>fine</i> and the RA returns 400 or 403, the IEEE-1609.2.1-Error header indicates the issue: — 400-10 absent encryption. — 400-20 blocked enrollment certificate. — 400-21 unregistered enrollment certificate. — 400-40 failed certificate chain verification. — 400-41 failed decryption. — 400-42 failed parsing. — 403-43 failed signature verification. — 400-50 future generationTime. — 400-62 invalid enrollment certificate. — 400-74 outside <i>download-maxAge</i> generationTime. — 400-75 outside <i>download-maxReqs</i>. — 400-76 within <i>download-minWait</i>. — 400-712 outside enrollment certificate's validityPeriod. — 400-81 wrong RA.
Errors	a) If <i>scmsV3-error</i> = <i>coarse</i>: 500 internal server error. b) If <i>scmsV3-error</i> = <i>fine</i>: 1) 400 bad request: see response header for details. 2) 401 unauthorized: the authorization mechanism used was not accepted (can be generated for the OAuth token for profiles with <i>webApi-eeAuth</i> = <i>oAuth2.0</i>, or for failures of SPDU-level authorization for profiles with <i>scmsV3-eeAuth</i> = <i>enrollment</i>). 3) 403 forbidden: see response header for details. 4) 404 not found: the resource at the given URL does not exist. Either <i>requestHash</i> or <i>time-iPeriod</i> in the URL is wrong or the requested file is not generated yet. 5) 405 method not allowed: retry with HTTP GET. 6) 408 request timeout: retry after <i>download-minWait</i> and up to <i>download-maxReqs</i> times. 7) 416 requested range not satisfiable: retry without HTTP Range. 8) 429 too many requests. 9) 500 internal server error. 10) 503 service unavailable: retry after <i>download-minWait</i> and up to <i>download-maxReqs</i> times.

Industry Informational Submission to OmniAir WG and its review/recommendation for future consideration as public website posting. OmniAir Consortium does not take any responsible or liability for industry contribution.

the filename to the most recent one known to the generator of the file.

Archived

Industry Informational Submission to OmniAir WG and its review/recommendation for future consideration as public website posting. OmniAir Consortium does not take any responsible or liability for industry contribution.

Archived 20260707

Industry Contribution for OmniAir WG Review / Recommendation for Standards / Specification, Testing, and Certification

Contribution Type: [Standard Change]

Specification Affected: [IEEE Std 1609.2.1-2022 Section 10.1.4.1]

Description Affected: [zip file directory structure]

Applicable OmniAir WG: [CyberSecurity]

WG Review: [Initial 2024-02-23]

WG Recommendation: [Posted on website until Standards Consideration]

Certification Test Spec Effect: [Incorporated 1609.2.1-2026 published 2026-07-03]

Formatted for Public Posting: [Roebuck]

Document Identifier: [1609.2.1 Note 12]

Industry Informational Submission to OmniAir WG and its review/recommendation for future consideration as public website posting. OmniAir Consortium does not take any responsible or liability for industry contribution.

Archived 20260707

IEEE Std 1609.2.1-2022 Guidance Note 12: zip file directory structure.

Virendra Kumar and William Whyte, Qualcomm Incorporated

Version 0.1, 2023-10-04

1 Introduction and problem statement

This guidance note is provided by the OmniAir CyberSecurity Working Group to support implementers in the interpretation of the 1609 series of specifications. These notes reflect the understanding and assumptions of the OmniAir Working Group while developing test cases and test plans intended to be used for device certification. OmniAir guidance notes will be revised or eliminated as new versions of the 1609 standards are published by IEEE. This document is advisory only, it is not an extension or modification of any IEEE standard.

This note identifies an area in IEEE Std 1609.2.1-2022 that is ambiguous and would benefit from clarification.

In IEEE Std 1609.2.1, the .zip file directory structure is not clearly specified. The following three clauses specify inconsistently what a .zip file is supposed to produce when unzipped.

- **8.2.3.1 Nonbutterfly certificate file contents**
For nonbutterfly certificate files, the contents of the .zip file is *a single file ...*
- **8.2.3.2 Butterfly certificate file contents**
For butterfly certificate files, the content of authorization certificate download .zip files is organized as *a directory containing files and directories ...*
- **8.6 CTL files**
The content is *a single directory ...*

Where the standard talks about a directory, it is not clear whether the .zip file is supposed to produce as its top-level output a directory containing the contents (as illustrated in bullet #1 below), or just the contents (as illustrated in bullet #2 below).

1. abc.zip -> directory -> [set of files/folders]
2. abc.zip -> [set of files/folders]

A survey among the implementors of the standard revealed that the majority of the survey participants have interpreted/implemented the standard as illustrated in bullet #2 above.

The proposed changes are to clarify in clauses 8.2.3.1, 8.2.3.2, and 8.6 that the .zip file when unzipped produces the contents directly, not contained within a directory.

Industry Informational Submission to OmniAir WG and its review/recommendation for future consideration as public website posting. OmniAir Consortium does not take any responsible or liability for industry contribution.

Archived 20260707

2 Proposed change

8.2.3.1 Nonbutterfly certificate file contents

For nonbutterfly certificate files, the ~~contents of the~~ .zip file **when unzipped produces** is a single file, named h without any file extension, where h is the 16 characters long hex encoding of the HashedId8 of the submitted authorization certificate request `EeRaCertRequestSpdu`, as returned in the `requestHash` field of `RaEeCertAckSpdu.raEeCertAck`.

8.2.3.2 Butterfly certificate file contents

For butterfly certificate files, the ~~content of authorization certificate download~~ .zip files is organized as a directory containing **when unzipped produces** files and directories that contain the certificates and metadata for a single $iPeriodLength$. The total number of files is $j_{max} + 1$.

The certificates in the file need not all have the same start and end validity times. See 4.3.2 for further discussion.

The contents of the zip file are as follows:

- a) ~~The first file in the root directory is the~~ **At the top level, a** .info file with name $x.info$, where x is the minimal length hex encoding of the i -value for this certificate batch. The contents of the .info file is a C-OER encoded `RaEeCertInfoSpdu` if the file does not contain ACPC information, and a C-OER encoded `RaEeCertAndAcpcInfoSpdu` if the file contains ACPC information (specifically, an `AcpcTreeId`).
- b) ~~The second file onwards are the certificate files. The content of each individual certificate file is an appropriate C-OER encoded~~ `AcaRaCertResponse.acaResponse`. The naming format for the individual certificate files is x_y without any file extension, where the following occurs:
 - 1) ~~x is the i -value.~~
 - 2) ~~y is the j -value.~~
 - 3) ~~Example certificate file names for $iCert = 30, j_{max} = 60$: 1E_0, 1E_1, ..., 1E_3B, 1E_3C.~~
- c) ~~The root directory also contains certificate files for certificates whose validity period starts at the same time as the i -period.~~
- c) **Also at the top level, all of the certificate files for certificates whose validity period starts at the same time as the i -period. The contents of each individual certificate file is an appropriate C-OER encoded** `AcaRaCertResponse.acaResponse`. **The naming format for the individual certificate files is x_y without any file extension, where the following occurs:**
 - 1) **x is the i -value.**
 - 2) **y is the j -value.****Example certificate file names for $iCert = 30, j_{max} = 60$: 1E_0, 1E_1, ..., 1E_3B, 1E_3C.**
- d) **There may also be one or more sub-directories for certificate files for certificates whose validity period does not start at the same time as the i -period. These are organized as follows.**
 - 1) **All certificates with the same start validity time are contained in a single sub-directory within the zip file.**
 - 2) **The sub-directory with the earliest start validity time is called “overlap”, the sub-directory with the second earliest start validity time is called “overlap1”, and the sub-directory with the k th earliest start validity time is called “overlap< $k-1$ >”, where “< $k-1$ >” represents the minimal-length encoding of the decimal representation of the number $k - 1$.**

Industry Informational Submission to OmniAir WG and its review/recommendation for future consideration as public website posting. OmniAir Consortium does not take any responsible or liability for industry contribution.

- 3) **Each subdirectory contains the files for the certificates with the start validity times appropriate for that directory. The files are named per the convention given in step c) above.**

8.6 CTL files

The CTL file is used to respond to a CTL download request as specified in 6.3.5.11. It is a zip file without compression that follows the PKWare Zip File Format specification version 6.3.6. ~~The content is a single directory containing~~ **.zip file when unzipped produces** individual CTL files. The content of each individual CTL file is a C-OER encoded MultiSignedCtlSpdu. Each CTL file name follows the convention *ctl_ctlSeriesId_sequenceNumber.oer*. The zip file name is *ctl_ctlSeriesId_sequenceNumber.zip*, where *sequenceNumber* is the lowest sequence number value among the included CTLs. The zip file contains all the CTLs for that *CtlSeriesId* value, from the one with the sequence number indicated in the filename to the most recent one known to the generator of the file.