|  |
| --- |
| **CHANGE REQUEST** |
|  | ETSI TS 103 097 | **Version** | 1.3.1 | **CR** | 1 | **rev** | - |  |
|  |
| **CR Title** | Make IEEE 1609.2 HeaderInfo extensible in a way that reduces coordination burden |
|  |  |
| **Original Source** | ITS WG5 |
|  |  |
| **Work Item Ref** | RTS/ITS-00557 | **Submission date** | 16.06.2020 |
| **Approving TB**  | ITS | **Approval date** | 01-07-2020 |
| **Category:** | **B** | **Release** | 1 |  |
|  | Use **one** of the following categories:**F** (correction)**A** (correction in an earlier release)**B** (addition of feature) **C** (functional modification of feature)**D** (editorial modification) |  |
|  |  |
| **Reason for change** | TS 103 097 is based on IEEE 1609.2. IEEE 1609.2 provides ASN.1 structures that are extensible so that if stakeholder groups using 1609.2 determine that additional features are necessary, those groups can specify those features without breaking deployed implementations. However, the approach taken to extensibility in 1609.2 requires coordination between different groups that are specifying extensions, as otherwise there is a risk that different groups will specify ASN.1 extensions in parallel which will end up being assigned by the encoder with the same “extension identifier”, causing conflicts and interoperability issues. This CR proposes an approach for extending core structures in IEEE 1609.2. The approach should reduce coordination overhead by providing different “namespaces” for different stakeholder groups to use, ensuring that those groups’ extensions do not conflict. The CR additionally proposes two specific extensions, for use in requesting Certificate Revocation Lists (CRLs) and Certificate Trust Lists (CTLs). |
|  |  |
| **Consequence if not approved** | 1. All future CRs for IEEE 1609.2, will need to be developed in a way that requires more coordination with IEEE and other stakeholders. If the coordination process fails, the ETSI and IEEE versions of 1609.2 could become incompatible. Alternatively, some coordination mechanism could be developed that used a registration authority, but this has been observed to have a cost in terms of administrative overhead and financial fees charged by the registration authority.
2. It will not be possible to request peer-to-peer distribution of CRLs and CTLs.
 |
|  |  |
| **Summary of change** | Add an EtsiOriginatingExtensions field to the 1609.2 HeaderInfo type; define CRL and CTL request extensions for use in signed SPDUs. |
|  |  |
| **Clauses affected** | 4, 5.2, 7.1.1, 7.1.2, A.2.2 |
|  |  |
| **Linked Change Requests** | None |  |
|  |  |  |
|  |  |
| **Other comments** | Text below in ***bold italics*** is editing instructions and not part of the new technical material |
|  |  |

Introduction

***Insert text between second and third paragraph***

From time to time, new versions of the present document may be published that extend IEEE 1609.2 data types using ASN.1 extension mechanisms to define ETSI originated extensions that are not necessarily endorsed by IEEE.

4 Basic format elements

4.1 Based on the published version of IEEE 1609.2

***Include the current text of 103 097 v 1.3.1 section 4 here.***

4.2 Extensions

4.2.1 General process

NOTE: This section and the following section outline approaches for maintaining and extending this standard. The functionality to be implemented in these extensions is specified elsewhere in this document.

IEEE 1609.2 structures are extensible using ASN.1 extension mechanisms.

For all extensible IEEE 1609.2 data types other than HeaderInfo, extensions will be done by adding new fields after the extension marker in the underlying IEEE 1609.2 data type. To avoid conflicts that might arise if multiple stakeholder groups want to extend the same IEEE 1609.2 data type at the same time, the rapporteur of this document shall coordinate with the editor of IEEE 1609.2 and ensure that different extension identifiers are associated with each different extension that is being simultaneously developed.

NOTE: In the above paragraph, “extension identifier” refers to the numeric identifier that the ASN.1 encoder automatically associates with an extension field in an ASN.1 structure. The numeric identifier is assigned automatically by the encoder based on the index of the extension field in the list of extension fields in that structure; it is not an identifier that is assigned through a registration process and visible to a human reader of the ASN.1. It would be more narrowly correct above to use the phrasing “ensure that each different extension is given a distinct position within the list of extensions to that data type,” but the phrasing above is a better expression of the underlying requirement.

4.2.2 HeaderInfo extensions

**Background**. The HeaderInfo type defined in IEEE 1609.2 [1] has a structure including an ASN.1 “…” extension marker. The fields after the extension marker are the extension fields. The reader is referred to IEEE 1609.2 directly for the current IEEE definition of HeaderInfo.

In this standard, the IEEE 1609.2 type ToBeSignedData in the module IEEE1609dot2 shall have the component headerInfo of type HeaderInfo as defined in IEEE Std 1609.2 [1] clause 6.3.9, with the addition of:

* the component contributedExtensions as specified in A.2.2
	+ within the component contributedExtensions, an optional sequence of components of type EtsiOriginatingHeaderInfoExtension as specified in A.2.2

HeaderInfo extensions are included in the component contributedExtensions.

The component contributedExtensions is of type ContributedExtensionBlocks and is a sequence of single extension “blocks” of type ContributedExtensionBlock. Each extension block defined by an identified contributing organization. The ETSI TC ITS WG5 extension block shall be identified by the integer etsiHeaderInfoContributorId (2). Within the ETSI TC ITS WG5 extension block, each extension shall be of type EtsiOriginatingHeaderInfoExtension. ASN.1 implementing these design principles is specified in Annex A.2.2.

The type EtsiOriginatingHeaderInfoExtension is defined in the module EtsiTs103097ExtensionModule specified in Annex A.1 and composed of the component id and the component Extn. The component id shall be of type ExtId and shall uniquely identify the extension within the set of EtsiOriginatingHeaderInfoExtensions. The component Extn shall be associated to the related id according to the information object set EtsiTs103097HeaderInfoExtensions. The ETSI originated extensions shall be defined as information objects of the class EXTENSION and shall be listed in the information object set EtsiTs103097HeaderInfoExtensions.

NOTE: This approach allows ETSI to specify new extensions as necessary, using an identifier that is entirely under ETSI’s control (the EtsiTs103097HeaderInfoExtensionId) to identify those extensions and a separate module called EtsiTs103097ExtensionModule that can be updated by ETSI without a need to change the module IEEE1609dot2.

The data type Version in the module EtsiTs103097ExtensionModule shall indicate the version of the module EtsiTs103097ExtensionModule.

The data type EtsiTs103097ExtensionModuleVersion in the module EtsiTs103097Module shall contain the version number of the module EtsiTs103097ExtensionModule as contained in the data type Version.

ASN.1 implementing these design principles is specified in Annex A.1.

5.2 SignedData

***At the end of the bullet-pointed list describing HeaderInfo, add the following text following the bullet point concerning requestedCertificate:***

* In the component contributedExtensions, any component of type EtsiOriginatingHeaderInfoExtension identified in the Information Object Set EtsiTs103097HeaderInfoExtensions present or absent according to the specification of the message profiles in clause 7 and according to the specification of the particular extension in the document that specifies it:
	+ The extension EtsiTs102941CrlRequest, if present, shall indicate that the ITS-Station is requesting a CRL according to TS 103 601 [i.3], with format as defined in TS 102 941 [i.2]. The component issuerId shall indicate the issuer of the CRL and the component lastKnownUpdate ,if present, shall indicate the value of the thisUpdate field of the latest CRL that the ITS-Station has available.
	+ The extension EtsiTs102941DeltaCtlRequest, if present, shall indicate that the ITS-Station is requesting a delta CTL according to TS 103 601 [i.3], with format as defined in TS 102 941 [i.2], using the data structure EtsiTs102941CtlRequest. The component issuerId shall indicate the issuer of the CTL and the component lastKnownCtlSequence, if present, shall indicate the value of the ctlSequence field of the latest CTL that the ITS-Station has available.
* In the component contributedExtensions, any component of type other than EtsiOriginatingHeaderInfoExtension always absent.

NOTE: the present document does not specify contributedExtensions fields of type other than EtsiOriginatingHeaderInfoExtension and does not specify what an implementation that processes received secure data structures shall do based on such extensions. Anyhow compliance to the present document requires an implementation to correctly parse received secure data structures that contain those extensions

7.1.1 Security profile for CAMs

***At the end of the bullet-pointed list describing HeaderInfo, add the following text immediately before the final bullet point (which begins “all other components of the component tbsData.headerInfo…”):***

* Any component of type EtsiOriginatingHeaderInfoExtension appearing in contributedExtensions may be present, absent, present under specified conditions, or optional. As different types of EtsiOriginatingHeaderInfoExtension are specified in future versions of this document, those future versions will also state whether and under what circumstances those EtsiOriginatingHeaderInfoExtension types are included in CAMs.

7.1.2 Security profile for DENMs

***At the end of the bullet-pointed list describing HeaderInfo, add the following text immediately before the final bullet point (which begins “all other components of the component tbsData.headerInfo…”):***

* Any component of type EtsiOriginatingHeaderInfoExtension appearing in contributedExtensions may be present, absent, present under specified conditions, or optional. As different types of EtsiOriginatingHeaderInfoExtension are specified in future versions of this document, those future versions will also state whether and under what circumstances those EtsiOriginatingHeaderInfoExtension types are included in DENMs.

A.1 ETSI TS 103 097 ASN.1 Modules

***Provide a link to the modified TS 103 097 ASN.1 module on ETSI Forge***

***Update the OID of the module EtsiTs103097Module as follows:***

EtsiTs103097Module

{ itu-t(0) identified-organization(4) etsi(0) itsDomain(5) wg5(5) secHeaders(103097) core(1) version2(2) }

***Modify the following IMPRT Statement (as underlined)***

Ieee1609Dot2Data, ExplicitCertificate

FROM IEEE1609dot2 {iso(1) identified-organization(3) ieee(111)

standards-association-numbered-series-standards(2) wave-stds(1609)

dot2(2) base(1) schema(1) major-version-2(2) minor-version-3(3)}

***Add the following IMPORT statement***

Version

FROM EtsiTs103097ExtensionModule {itu-t(0) identified-organization(4) etsi(0) itsDomain(5) wg5(5) secHeaders(103097) extension(2) version1(1)};

***Add the new data type ExtensionModuleVersion as follows:***

ExtensionModuleVersion::= Version

***Define the following new module EtsiTs103097ExtensionModule to be provided through a link on ETSI Forge:***

EtsiTs103097ExtensionModule

{itu-t(0) identified-organization(4) etsi(0) itsDomain(5) wg5(5) secHeaders(103097) extension(2) version1(1)}

DEFINITIONS AUTOMATIC TAGS ::= BEGIN

IMPORTS

 HashedId8,

 Time32

FROM IEEE1609dot2BaseTypes {iso(1) identified-organization(3) ieee(111)

 standards-association-numbered-series-standards(2) wave-stds(1609)

 dot2(2) base(1) base-types(2) major-version-2 (2) minor-version-3(3)}};

Version::= INTEGER(1)

Extension {EXT-TYPE : ExtensionTypes} ::= SEQUENCE {

 id EXT-TYPE.&extId({ExtensionTypes}),

 content EXT-TYPE.&ExtContent({ExtensionTypes}{@.id})

}

EXT-TYPE ::= CLASS {

 &extId ExtId,

 &ExtContent

} WITH SYNTAX {&ExtContent IDENTIFIED BY &extId}

ExtId ::= INTEGER(0..255)

EtsiOriginatingHeaderInfoExtension ::= Extension{{EtsiTs103097HeaderInfoExtensions}}

EtsiTs103097HeaderInfoExtensionId ::= ExtId

 etsiTs102941CrlRequestId EtsiTs103097HeaderInfoExtensionId ::= 1 --'01'H

 etsiTs102941DeltaCtlRequestId EtsiTs103097HeaderInfoExtensionId ::= 2 --'02'H

EtsiTs103097HeaderInfoExtensions EXT-TYPE ::= {

 { EtsiTs102941CrlRequest IDENTIFIED BY etsiTs102941CrlRequestId } |

 { EtsiTs102941DeltaCtlRequest IDENTIFIED BY etsiTs102941DeltaCtlRequestId },

 ...

}

EtsiTs102941CrlRequest::= SEQUENCE {

 issuerId HashedId8,

 lastKnownUpdate Time32 OPTIONAL

 }

EtsiTs102941CtlRequest::= SEQUENCE {

 issuerId HashedId8,

 lastKnownCtlSequence INTEGER (0..255) OPTIONAL

 }

EtsiTs102941DeltaCtlRequest::= EtsiTs102941CtlRequest

END

A.2 IEEE 1609.2 ASN.1 modules

***Insert the following subclause heading before the current IEEE 1609.2 ASN.1 modules:***

### A.2.1 Actual IEEE 1609.2 ASN.1 modules

This clause provides the relevant ASN.1 modules from IEEE Std 1609.2 [1] (and its amendments), reprinted with permission from IEEE, Copyright © 2016

***Provide a link to the modules on ETSI Forge***

***Insert the following new subclause:***

### A.2.2 Provisional changes to the actual IEEE 1609.2 ASN.1 modules

***Provide a link to the changed modules on ETSI Forge. Insert the following:***

The following modifications to the actual IEEE 1609.2 schema module have been applied:

Modify the module’s OID as follows:

IEEE1609dot2 {iso(1) identified-organization(3) ieee(111)

standards-association-numbered-series-standards(2) wave-stds(1609)

dot2(2) base(1) schema(1) major-version-2(2) minor-version-3(3)}

Modify the IMPORTS statement to import the data type EtsiOriginatingHeaderInfoExtension:

EtsiOriginatingHeaderInfoExtension

FROM EtsiTs103097ExtensionModule {itu-t(0) identified-organization(4) etsi(0) itsDomain(5) wg5(5) secHeaders(103097) extension(2) version1(1)} WITH SUCCESSORS;

Extend the type HeaderInfo and add the following structure definitions after it, as defined below:

HeaderInfo ::= SEQUENCE {

 -- add the current elements of HeaderInfo here.

 contributedExtensions ContributedExtensionBlocks OPTIONAL

}

ContributedExtensionBlocks ::= SEQUENCE (SIZE(1..MAX)) OF ContributedExtensionBlock

ContributedExtensionBlock ::= SEQUENCE {

 contributorId IEEE1609DOT2-HEADERINFO-CONTRIBUTED-EXTENSION.

 &id({Ieee1609dot2HeaderInfoContributedExtensions}),

 extns SEQUENCE (SIZE(1..MAX)) OF IEEE1609DOT2-HEADERINFO-CONTRIBUTED-EXTENSION.

 &Extn({Ieee1609dot2HeaderInfoContributedExtensions}{@.contributorId})

}

IEEE1609DOT2-HEADERINFO-CONTRIBUTED-EXTENSION ::= CLASS {

 &id HeaderInfoContributorId UNIQUE,

 &Extn

} WITH SYNTAX {&Extn IDENTIFIED BY &id}

Ieee1609dot2HeaderInfoContributedExtensions

 IEEE1609DOT2-HEADERINFO-CONTRIBUTED-EXTENSION ::= {

 { EtsiTs103097ExtensionModule.EtsiOriginatingHeaderInfoExtension IDENTIFIED BY etsiHeaderInfoContributorId},

 ...

}