Using ISO 15118 Plug & Charge with OCPP 1.6

v1.0, 2020-09-16

OCA Application Note:

Using ISO 15118 Plug & Charge with OCPP 1.6.

Relevant for OCPP version: 1.6J.

Version History

VERSION	DATE	AUTHOR	DESCRIPTION
1.0	2020-09-16	Milan Jansen *OCA* Lonneke Driessen *ElaadNL* Paul Klapwijk *OCA*	Final version

1. Introduction

1.1. Reason for this Application Note

The OCPP 2.0 specification (first published in April 2018 and revised by OCPP 2.0.1 in April 2020) supports the use of ISO 15118 edition 1 between a Charge Point and an Electric Vehicle. All functionality that requires Charge Point to Charge Point Management System (CPMS) communication is supported: managing ISO 15118 certificates in both the Charge Point and the EV, handling Plug and Charge authorization and smart charging using ISO 15118.

The OCPP version 1.6 (published October 2015) does not support ISO 15118 messages. However, some OCPP implementers have asked the Open Charge Alliance if an intermediate solution, combining both OCPP 1.6 and the ISO 15118 Plug & Charge functionality would be possible.

1.2. The Data Transfer Mechanism

The Data Transfer Mechanism in OCPP
The default mechanism in OCPP to accommodate features that are not yet part of the OCPP standard is the 'Data Transfer Mechanism'. The DataTransfer.req message allows for the exchange of data or messages and as such, it offers a framework within OCPP for experimental functionality that may find its way into future OCPP versions. Experimenting can be done without creating new (possibly incompatible) OCPP dialects. Secondly, it offers a possibility to implement vendor-specific custom extensions.

Limitations of the Data Transfer Mechanism
The use of DataTransfer messages for functionality that overlaps with functionality that is already available in OCPP is not recommended, since this could cause interoperability issues with other implementations, will disturb a smooth upgrade to newer versions of the protocol and could prevent an implementation from being OCPP compliant according to the rules of the OCPP Certification Program.

1.3. About this Application Note

This application note describes how to use the data transfer mechanism in OCPP 1.6 to exchange messages for ISO 15118 certificate management and Plug and Charge (*excluding* the smart charging functionality). The messages used in the DataTransfer are taken from OCPP 2.0.1, are slightly simplified and are wrapped in a DataTransfer message and must be used only for the use cases described in this application note. Please refer to Generic DataTransfer fields for an example how this is done. This extension describes the use cases only on a higher level. For detailed descriptions, please refer to the corresponding ISO 15118 use cases in the OCPP 2.0.1 Specification. As a general recommendation it is advised to not only read the use cases in the OCPP 2.0.1 specification this Application Note refers to to, but also the introduction chapters of the functional blocks where the use cases are located for better understanding.

Please note: This application note describes combining OCPP 1.6 and ISO 15118 Plug and Charge. Implementers that want to implement the ISO 15118 Smart Charging functionality must upgrade to OCPP 2.0.1, this functionality is not part of this application note. This application note could be implemented in combination with the *"Improved Security for OCPP 1.6J"* whitepaper, although it will not lead to the most elegant solution. Messages in this application note that would overlap with the security related messages in the *"Improved Security for OCPP 1.6J"* whitepaper are wrapped in DataTransfer messages and will therefore not cause direct conflicts. As mentioned earlier, having overlapping functionalities could have caused, among others, interoperability issues. In OCPP 2.0.1 support for both OCPP security and ISO 15118 is added as part of the specification and these functionalities are merged together. Implementers that want to implement both OCPP security and ISO 15118 Plug & Charge are therefore strongly advised to use OCPP 2.0.1.

Note: For readability, in the following chapters, we will use the *<message name>* from OCPP 2.0.1 in the descriptions, instead of using "*a DataTransfer message containing a <message name>*".

2. ISO 15118 support in OCPP 1.6J

2.1. Determining and enabling ISO 15118 PnC support

Enabling support for ISO 15118 Plug and Charge can be done with the configuration key `ISO15118PnCEnabled`. This configuration key can also be used to determine whether the Charge Point supports this functionality. If the configuration key is present, the Central System can consider this functionality supported by the Charge Point.

If the Charge Point supports this application note and sends a SignCertificate.req (wrapped in a DataTransfer message) for getting a V2G certificate and the Central System does *not* support this application note, the Central System will respond with a DataTransfer.conf with *status* `Rejected`, `UnknownVendorId` or `UnknownMessageId`, the Charge Point SHALL not resend the SignCertificate.req message unless triggered by a TriggerMessage.req.

2.2. Generic DataTransfer fields

As explained in the introduction, the DataTransfer mechanism is used to create vendor-specific custom extensions. To support this application note *and* to be interoperable with other implementations of this application note, it is important that the DataTransfer.req message fields are used in the same way:

FIELD NAME	FIELD TYPE	DESCRIPTION
messageId	string[0..50]	The OCPP message name from the following list: - Authorize - CertificateSigned - DeleteCertificate - Get15118EVCertificate - GetCertificateStatus - GetInstalledCertificateIds - InstallCertificate - SignCertificate - TriggerMessage
data	Text Length undefined	The message in JSON format.
vendorId	string[0..255]	fixed value: org.openchargealliance.iso15118pnc

An example DataTransfer.req message with an ISO 15118 OCPP message could look like this:

```
[2,
"123456",
"DataTransfer",
    {
    "messageId": "SignCertificate",
    "data": "{\"csr\": \"<certificate signing request>\",\"certificateType\":
\"V2GCertificate\"}",
    "vendorId": "org.openchargealliance.iso15118pnc"
    }
]
```

The message name is included in the *messageId* and the *vendorId* contains a fixed value, as explained in Generic DataTransfer fields. The data field contains the entire JSON message. For readability the actual csr was not included.

An example flow of the messages is displayed in the figure below, where the notation DataTransfer.req(SignCertificate.req(csr)) refers to a DataTransfer.req message, containing a messageId "SignCertificate", data field with a *SignCertificate.req* message with a "csr" field as defined in Messages :

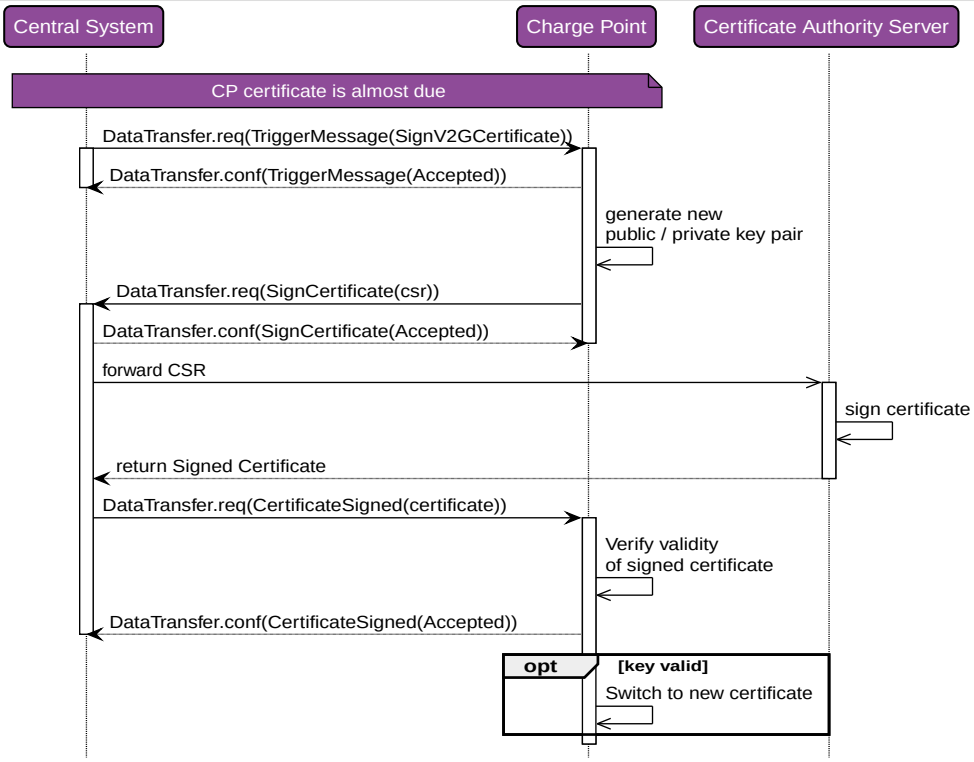

Figure 1. Example (happy) flow of messages (Certificate Installation initiated by Central System)

2.3. Using this application note with *"Improved Security for OCPP 1.6J"*

When combining this application note with the *"Improved Security for OCPP 1.6J"* whitepaper, the messages from this application note wrapped in the DataTransfer messages SHALL only be used for ISO 15118 related use cases and *not* for functionality that is described in the *"Improved Security for OCPP 1.6J"* whitepaper.
For example, when deleting an ISO 15118 related certificate, e.g. an MORootCertificate, the DeleteCertificate.req message wrapped in a DataTransfer.req message MUST be used. When deleting an CentralSystemRootCertificate the DeleteCertificate.req message from the *"Improved Security for OCPP 1.6J"* (*not* wrapped in a DataTransfer.req message) MUST be used.

Similarly, when using this application note, messages from this application note wrapped in the DataTransfer messages SHALL only be used for ISO 15118 related use cases and *not* for functionality that is described in the *OCPP 1.6 edition 2*.
For example, the Authorize.req message wrapped in a DataTransfer.req message is *only* used for Plug and Charge authorization, in all other cases the regular Authorize.req from *OCPP 1.6 edition 2* MUST be used.

3. Certificate Installation

3.1. ISO 15118 Certificate Installation Charge Point

The installation of an ISO 15118 Charge Point certificate can be installed by the initative of a Charge Point or the Central System. These variants are described in the following 2 paragraphs.

3.1.1. Initiated by the Central System

USE CASE(S) OCPP 2.0.1	A02
Scenario description	The installation of an ISO 15118 Charge Point certificate by the initative of the Central System consists of the following steps: 1. Central System sends a TriggerMessage.req to trigger a SignCertificate.req message to the Charge Point. 2. The Charge Point generates a new public / private key pair. 3. Charging station sends a SignCertificate.req message to the Central System. 4. The Central System requests a Certificate Authority (CA) to sign the Certificate Signing Request that was sent by the Charge Point. 5. The Central System sends the certificate it has acquired from the CA to the Charge Point, using the CertificateSigned.req message. 6. The Charge Point verifies the Signed Certificate and if valid, it installs it as its ISO 15118 Charging Point certificate. If the certificate is invalid the Charge Point must reject the CertificateSigned.req message by sending a CertificateSigned.conf message with status *Rejected*. If the *"Improved Security for OCPP 1.6J"* whitepaper is implemented, the Charge Point must also report a SecurityEvent with security event *InvalidChargePointCertificate*. *See OCPP 2.0.1 Specification – A. Security Use Case A02 for further details.* Please note: in addition to the requirements in OCPP 2.0.1 for this use case, an OCPP 2.0.1 erratum is written, adding two configuration variables, CertSigningWaitMinimum and CertSigningRepeatTimes, with the purpose of providing a back off mechanism for the case when the Charge Point never receives the signed certificate generated from the provided CSR.
Messages wrapped in DataTransfer messages	TriggerMessage, SignCertificate, CertificateSigned

3.1.2. Initiated by the Charge Point

USE CASE(S) OCPP 2.0.1	A03
Scenario description	The installation of an ISO 15118 Charge Point certificate by the initative of the Charge Point consists of the following steps: 1. The Charge Point generates a new public / private key pair. 2. Charging station sends a SignCertificate.req message to the Central System. 3. The Central System requests a Certificate Authority (CA) to sign the Certificate Signing Request that was sent by the Charge Point. 4. The Central System sends the certificate it has acquired from the CA to the Charge Point, using the CertificateSigned.req message. 5. The Charge Point verifies the Signed Certificate and if valid, it installs it as its ISO 15118 Charging Point certificate. If the certificate is invalid the Charge Point must reject the CertificateSigned.req message by sending a CertificateSigned.conf message with status *Rejected*. If the *"Improved Security for OCPP 1.6J"* whitepaper is implemented, the Charge Point must also report a SecurityEvent with security event *InvalidChargePointCertificate*. *See OCPP 2.0.1 Specification – A. Security Use Case A03 for further details.* Please note: in addition to the requirements in OCPP 2.0.1 for this use case, an OCPP 2.0.1 erratum is written, adding two configuration variables, CertSigningWaitMinimum and CertSigningRepeatTimes, with the purpose of providing a back off mechanism for the case when the Charge Point never receives the signed certificate generated from the provided CSR.
Messages wrapped in DataTransfer messages	SignCertificate, CertificateSigned

3.2. ISO 15118 Certificate Installation / Update EV

USE CASE(S) OCPP 2.0.1	M01/M02
Scenario description	The installation or update of a ContractCertificate in an EV consists of the following steps: 1. The EV sends an CertificateInstallationReq message to the Charge Point. 2. Based on this, the Charge Point sends an Get15118EVCertificate.req message to Central System. In case of Certificate installation, the *action* must be 'Install', in case of a certificate update, the *action* must be 'Update' 3. The Central System retrieves the right certificate and responds with a Get15118EVCertificate.conf message to the Charge Point. In case that an error occurs when fetching the exiResponse by the Central System or a timeout occurs, the Central System will respond with a Get15118EVCertificate.conf message with a *status* `Failed` to the Charge Point. *See OCPP 2.0.1 Specification – M. ISO 15118 CertificateManagement Use Case M01/M02 for further details.*
Messages wrapped in DataTransfer messages	Get15118EVCertificate

4. Certificate Management

4.1. Retrieving a list of installed certificates

USE CASE(S) OCPP 2.0.1	M03
Scenario description	Retrieving a list of currently installed certificates consists of the following steps: 1. The Central System sends a GetInstalledCertificateIds.req message to the Charge Point. 2. The Charge Point responds with a GetInstalledCertificateIds.conf message containing the requested certificate hashes. *See OCPP 2.0.1 Specification – M. ISO 15118 CertificateManagement Use Case M03 for further details.*
Messages wrapped in DataTransfer messages	GetInstalledCertificateIds

4.2. Deleting a certificate

USE CASE(S) OCPP 2.0.1	M04
Scenario description	Deleting a certificate from the Charge Point consists of the following steps: 1. The Central System sends a DeleteCertificate.req message to the Charge Point. 2. The Charge Point responds with a DeleteCertificate.conf message. *See OCPP 2.0.1 Specification – M. ISO 15118 CertificateManagement Use Case M04 for further details.* . Please note: in addition to the requirements in OCPP 2.0.1 for this use case, an erratum is written, adding that a Charge Point may reject the request to prevent the deletion of a certificate, if it is the last one from its certificate type.
Messages wrapped in DataTransfer messages	DeleteCertificate

4.3. Installing a Root CA certificate

USE CASE(S) OCPP 2.0.1	M05
Scenario description	Installing a Root CA certificate in the Charge Point consists of the following steps: 1. The Central System sends an InstallCertificate.req message to the Charge Point. 2. The Charge Point responds with an InstallCertificate.conf message. *See OCPP 2.0.1 Specification – M. ISO 15118 CertificateManagement Use Case M05 for further details.*
Messages wrapped in DataTransfer messages	InstallCertificate

4.4. Getting an OCSP response

USE CASE(S) OCPP 2.0.1	M06
Scenario description	Retrieving an OCSP response containing the certificate status consists of the following steps: 1. The Charge Point sends an GetCertificateStatus.req message to the Central System. 2. The Central System responds with an GetCertificateStatus.conf message containing the OCSP response. *See OCPP 2.0.1 Specification – M. ISO 15118 CertificateManagement Use Case M06 for further details.*
Messages wrapped in DataTransfer messages	GetCertificateStatus

5. Plug and Charge authorization

USE CASE(S) OCPP 2.0.1	C07
Scenario description	Plug and Charge authorization consists of the following steps: 1. When it is plugged in, the EV sends a ContractCertificate and eMAID to the Charge Point. 2. The Charge Point checks this offline (see below) or it sends an Authorize.req message wrapped in DataTransfer.req to the Central System containing the eMAID and data needed for the contract certificate validation: this can be the information in the OCSPRequestDataType so that the Central System can do an OCSP request with regards to the ContractCertificate and Certificate Chain. Alternatively, if the Charge Point is not able to validate a contract certificate, because it does not have the associated root certificate and the configuration key `CentralContractValidationAllowed` is *true*, the Charge Point MUST pass the contract certificate to the Central System in the certificate attribute (in PEM format) for validation by the Central System. When using ISO 15118 Plug and Charge authorization the Authorize message wrapped in a DataTransfer replaces the regular Authorize.req message. 3. The Central System replies with an agreement or non-agreement, and the certificate status. *See OCPP 2.0.1 Specification – C. Authorization Use Case C07 for further details.* **Please note**: Using the DataTransfer.req with the wrapped Authorize message is mandatory, the regular command Authorize.req SHALL NOT be used and the (regular) StartTransaction.req MUST use the eMAID in the *idTag*. When using ISO 15118 with External Identification Means (EIM), the regular Authorize.req message MUST be used.
Messages wrapped in DataTransfer messages	Authorize
Offline behavior	When the Charge Point is offline, the contract certificate cannot be verified at the CSMS. Offline contract certificate validation is configurable on the Charge Point side. The Configuration key `ContractValidationOffline` is available for this. Basically it verifies the contract offline and then applies the regular behavior of the Local Authorization List and Authorization Cache to the eMAID. The following more detailed rules apply to this key:
When the Charge Point is offline AND `ContractValidationOffline` is false	*The Charge Point SHALL NOT allow charging.*
When Charge Point is offline AND `ContractValidationOffline` is true	*The Charge Point SHALL try to validate the contract certificate locally.*
In the situation that the Charge Point is offline and the ContractValidationOffline key is set to true, the Charge Point thus locally validates the contract certificate. If the certificate is valid, the Local Authorization List and Authorization Cache mechanisms can be used acccording to the regular OCPP 1.6 offline behaviour, using the eMAID. In more detail, the following rules apply:	
If LocalAuthorizeOffline is true	*The Charge Point SHALL lookup the eMAID in the Local Authorization List or the Authorization Cache.*
If LocalAuthorizeOffline is true and the eMAID is found in the Local Authorization List	*The Charge Point SHALL treat this as a regular OCPP 1.6 offline authorization through the Local Authorization List.*
If LocalAuthorizeOffline is true and the eMAID is found in the Authorization Cache	*The Charge Point SHALL treat this as a regular OCPP 1.6 offline authorization through the Authorization Cache*
If LocalAuthorizeOffline is true, the eMAID is not found and `AllowOfflineTxForUnknownId` = true	*The Charge Point SHALL allow charging according to regular behavior of this configuration key.*
Please note: since the eMAID is linked to a contract with an expiration date, it is up to the Central System to remove eMAIDs from the Local Authorization List once the validity has expired. Furthermore, OCPP 1.6 knows 1 IdToken type, so it is up to the Central System to take into account to only send IdTokens in the Local Authorization List that are supported by the Charge Point.	

6. Messages

6.1. Authorize

The Authorize message wrapped in a DataTransfer SHALL only be used for delivering an Authorize belonging to a ISO 15118 Plug & Charge session.

6.1.1. Authorize.req

This contains the field definition of the Authorize.req that is wrapped in the DataTransfer PDU sent by the Charge Point to the Central System.

Class

FIELD NAME	FIELD TYPE	CARD.	DESCRIPTION
certificate	string[0..5500]	0..1	Optional. The X.509 certificated presented by EV and encoded in PEM format.
idToken	IdTokenType	1..1	Required. This contains the identifier that needs to be authorized.
iso15118CertificateHashData	OCSPRequestDataType	0..4	Optional. Contains the information needed to verify the EV Contract Certificate via OCSP.

6.1.2. Authorize.conf

This contains the field definition of the Authorize.conf that is wrapped in the DataTransfer PDU sent by the Central System to the Charge Point in response to an Authorize.req.

Class

FIELD NAME	FIELD TYPE	CARD.	DESCRIPTION
certificateStatus	AuthorizeCertificateStatusEnumType	0..1	Optional. Certificate status information. - if all certificates are valid: return 'Accepted'. - if one of the certificates was revoked, return 'CertificateRevoked'.
idTokenInfo	IdTokenInfoType	1..1	Required. This contains information about authorization status, expiry and group id.

6.2. CertificateSigned

6.2.1. CertificateSigned.req

This contains the field definition of the CertificateSigned.req that is wrapped in the DataTransfer PDU sent by the Central System to the Charge Point.

Class

FIELD NAME	FIELD TYPE	CARD.	DESCRIPTION
certificateChain	string[0..10000]	1..1	Required. The signed PEM encoded X.509 certificate. This can also contain the necessary sub CA certificates. In that case, the order of the bundle should follow the certificate chain, starting from the leaf certificate. The Configuration Variable MaxCertificateChainSize can be used to limit the maximum size of this field.

6.2.2. CertificateSigned.conf

This contains the field definition of the CertificateSigned.conf that is wrapped in the DataTransfer PDU sent by the Charge Point to the Central System in response to a CertificateSigned.req that is wrapped in a DataTransfer PDU.

Class

FIELD NAME	FIELD TYPE	CARD.	DESCRIPTION
status	CertificateSignedStatusEnumType	1..1	Required. Returns whether certificate signing has been accepted, otherwise rejected.

6.3. DeleteCertificate

6.3.1. DeleteCertificate.req

Message wrapped in the DataTransfer PDU by the Central System to request deletion of an installed certificate on a Charge Point.

Class

FIELD NAME	FIELD TYPE	CARD.	DESCRIPTION
certificateHashData	CertificateHashDataType	1..1	Required. Indicates the certificate of which deletion is requested.

6.3.2. DeleteCertificate.conf

Response to a DeleteCertificate.req.

Class

FIELD NAME	FIELD TYPE	CARD.	DESCRIPTION
status	DeleteCertificateStatusEnumType	1..1	Required. Charge Point indicates if it can process the request.

6.4. Get15118EVCertificate

6.4.1. Get15118EVCertificate.req

This message is sent (wrapped in the DataTransfer PDU) by the Charge Point to the Central System if an ISO 15118 vehicle selects the service Certificate installation. NOTE: This message is based on CertificateInstallationReq Res from *ISO 15118-2*.

Class

FIELD NAME	FIELD TYPE	CARD.	DESCRIPTION
iso15118SchemaVersion	string[0..50]	1..1	Required. Schema version currently used for the 15118 session between EV and Charge Point. Needed for parsing of the EXI stream by the Central System.
action	CertificateActionEnumType	1..1	Required. Defines whether certificate needs to be installed or updated.
exiRequest	string[0..5600]	1..1	Required. Raw CertificateInstallationReq request from EV, Base64 encoded.

6.4.2. Get15118EVCertificate.conf

Response message (in a DataTransfer PDU) from Central System to Charge Point containing the status and optionally new certificate. NOTE: This message is based on CertificateInstallationReq Res from *ISO 15118-2*.

Class

FIELD NAME	FIELD TYPE	CARD.	DESCRIPTION
status	Iso15118EVCertificateStatusEnumType	1..1	Required. Indicates whether the message was processed properly.
exiResponse	string[0..5600]	1..1	Required. Raw CertificateInstallationRes response for the EV, Base64 encoded.

6.5. GetCertificateStatus

6.5.1. GetCertificateStatus.req

This contains the field definition of the GetCertificateStatus.req that is wrapped in the DataTransfer PDU sent by the Charge Point to the Central System.

Class

FIELD NAME	FIELD TYPE	CARD.	DESCRIPTION
ocspRequestData	OCSPRequestDataType	1..1	Required. Indicates the certificate of which the status is requested.

6.5.2. GetCertificateStatus.conf

This contains the field definition of the GetCertificateStatus.conf that is wrapped in the DataTransfer PDU sent by the Central System to the Charge Point.

Class

FIELD NAME	FIELD TYPE	CARD.	DESCRIPTION
status	GetCertificateStatusEnumType	1..1	Required. This indicates whether the charging station was able to retrieve the OCSP certificate status.

FIELD NAME	FIELD TYPE	CARD.	DESCRIPTION
ocspResult	string[0..5500]	0..1	Optional. OCSPResponse class as defined in *IETF RFC 6960*. DER encoded (as defined in *IETF RFC 6960*), and then base64 encoded. MAY only be omitted when status is not Accepted.

6.6. GetInstalledCertificateIds

6.6.1. GetInstalledCertificateIds.req

Used by the Central System to request an overview of the installed certificates on a Charge Point.

Class

FIELD NAME	FIELD TYPE	CARD.	DESCRIPTION
certificateType	GetCertificateIdUseEnumType	0..*	Optional. Indicates the type of certificates requested. When omitted, all certificate types are requested.

6.6.2. GetInstalledCertificateIds.conf

Response to a GetInstalledCertificateIDs.req (wrapped in a DataTransfer PDU).

Class

FIELD NAME	FIELD TYPE	CARD.	DESCRIPTION
status	GetInstalledCertificateStatusEnumType	1..1	Required. Charge Point indicates if it can process the request.
certificateHashDataChain	CertificateHashDataChainType	0..*	Optional. The Charge Point includes the Certificate information for each available certificate.

6.7. InstallCertificate

6.7.1. InstallCertificate.req

Used by the Central System to request installation of a certificate on a Charge Point.

Class

FIELD NAME	FIELD TYPE	CARD.	DESCRIPTION
certificateType	InstallCertificateUseEnumType	1..1	Required. Indicates the certificate type that is sent.
certificate	string[0..5500]	1..1	Required. A PEM encoded X.509 certificate.

6.7.2. InstallCertificate.conf

The response to a InstallCertificate.req that is wrapped in a DataTransfer PDU, sent by the Charge Point to the Central System.

Class

FIELD NAME	FIELD TYPE	CARD.	DESCRIPTION
status	InstallCertificateStatusEnumType	1..1	Required. Charge Point indicates if installation was successful.

6.8. SignCertificate

6.8.1. SignCertificate.req

Sent (wrapped in a DataTransfer PDU) by the Charge Point to the Central System to request that the Certificate Authority signs the public key into a certificate.

Class

FIELD NAME	FIELD TYPE	CARD.	DESCRIPTION
csr	string[0..5500]	1..1	Required. The Charge Point SHALL send the public key in form of a Certificate Signing Request (CSR) as described in RFC 2986 [22] and then PEM encoded, using the SignCertificate.req message.

6.8.2. SignCertificate.conf

Sent (wrapped in a DataTransfer PDU) by the Central System to the Charge Point in response to the SignCertificate.req message that was wrapped in a DataTransfer PDU.

Class

FIELD NAME	FIELD TYPE	CARD.	DESCRIPTION
status	GenericStatusEnumType	1..1	Required. Specifies whether the Central System can process the request.

6.9. TriggerMessage

6.9.1. TriggerMessage.req

This message does not contains any fields. This TriggerMessage.req SHALL only be used for triggering a certificate signing request for an ISO 15118 V2G certificate for the Charge Point.

6.9.2. TriggerMessage.conf

This contains the field definition of the TriggerMessage.conf that is wrapped in a DataTransfer PDU sent by the Charge Point to the Central System in response to TriggerMessage.conf.

Class

FIELD NAME	FIELD TYPE	CARD.	DESCRIPTION
status	TriggerMessageStatusEnumType	1..1	Required. Indicates whether the Charge Point will send the requested notification or not.

7. DataTypes

7.1. CertificateHashDataChainType

Class

FIELD NAME	FIELD TYPE	CARD.	DESCRIPTION
certificateType	GetCertificateIdUseEnumType	1..1	Required. Indicates the type of the requested certificate(s).
certificateHashData	CertificateHashDataType	1..1	Required. Information to identify a certificate.
childCertificateHashData	CertificateHashDataType	0..4	Optional. Information to identify the child certificate(s).

7.2. CertificateHashDataType

Class

FIELD NAME	FIELD TYPE	CARD.	DESCRIPTION
hashAlgorithm	HashAlgorithmEnumType	1..1	Required. Used algorithms for the hashes provided.
issuerNameHash	identifierString[0..128]	1..1	Required. Hashed value of the Issuer DN (Distinguished Name).
issuerKeyHash	string[0..128]	1..1	Required. Hashed value of the issuers public key
serialNumber	identifierString[0..40]	1..1	Required. The serial number of the certificate.

7.3. IdTokenType

Class

Contains a case insensitive identifier to use for the authorization.

FIELD NAME	FIELD TYPE	CARD.	DESCRIPTION
idToken	identifierString[0..20]	1..1	Required. IdToken is case insensitive. Might hold the hidden id of an RFID tag, but can for example also contain a UUID.

7.4. IdTokenInfoType

Class

Contains status information about an identifier.

FIELD NAME	FIELD TYPE	CARD.	DESCRIPTION
status	AuthorizationStatusEnumType	1..1	Required. Current status of the ID Token.

FIELD NAME	FIELD TYPE	CARD.	DESCRIPTION
cacheExpiryDateTime	dateTime	0..1	Optional. Date and Time after which the token must be considered invalid.

7.5. OCSPRequestDataType

Class

FIELD NAME	FIELD TYPE	CARD.	DESCRIPTION
hashAlgorithm	HashAlgorithmEnumType	1..1	Required. Used algorithms for the hashes provided.
issuerNameHash	identifierString[0..128]	1..1	Required. Hashed value of the Issuer DN (Distinguished Name).
issuerKeyHash	string[0..128]	1..1	Required. Hashed value of the issuers public key
serialNumber	identifierString[0..40]	1..1	Required. The serial number of the certificate.
responderURL	string[0..512]	1..1	Required. This contains the responder URL (Case insensitive).

8. Enumerations

8.1. AuthorizationStatusEnumType

Enumeration

Status of an authorization response.

VALUE	DESCRIPTION
Accepted	Identifier is allowed for charging.
Blocked	Identifier has been blocked. Not allowed for charging.
ConcurrentTx	Identifier is already involved in another transaction and multiple transactions are not allowed.
Expired	Identifier has expired. Not allowed for charging.
Invalid	Identifier is invalid. Not allowed for charging.

8.2. AuthorizeCertificateStatusEnumType

Enumeration

Status of the EV Contract certificate.

VALUE	DESCRIPTION
Accepted	Positive response
SignatureError	If the validation of the Security element in the message header failed.
CertificateExpired	If the OEMProvisioningCert in the CertificateInstallationReq, the Contract Certificate in the CertificateUpdateReq, or the ContractCertificate in the PaymentDetailsReq is expired.
CertificateRevoked	Used when the SECC or Central System matches the ContractCertificate contained in a CertificateUpdateReq or PaymentDetailsReq with a CRL and the Contract Certificate is marked as revoked, OR when the SECC or Central System matches the OEM Provisioning Certificate contained in a CertificateInstallationReq with a CRL and the OEM Provisioning Certificate is marked as revoked. The revocation status can alternatively be obtained through an OCSP responder.
NoCertificateAvailable	If the new certificate cannot be retrieved from secondary actor within the specified timeout
CertChainError	If the ContractSignatureCertChain contained in the CertificateInstallationReq message is not valid.
ContractCancelled	If the EMAID provided by EVCC during CertificateUpdateReq is not accepted by secondary actor.

8.3. CertificateActionEnumType

Enumeration

VALUE	DESCRIPTION
Install	Install the provided certificate.
Update	Update the provided certificate.

8.4. CertificateSignedStatusEnumType

Enumeration

Status in CertificateSigned.conf.

VALUE	DESCRIPTION
Accepted	Signed certificate is valid.
Rejected	Signed certificate is invalid.

8.5. DeleteCertificateStatusEnumType

Enumeration

VALUE	DESCRIPTION
Accepted	Normal successful completion (no errors).
Failed	Processing failure.
NotFound	Requested resource not found.

8.6. GenericStatusEnumType

Enumeration

Generic message response status

VALUE	DESCRIPTION
Accepted	Request has been accepted and will be executed.
Rejected	Request has not been accepted and will not be executed.

8.7. GetCertificateIdUseEnumType

Enumeration

VALUE	DESCRIPTION
V2GRootCertificate	Use for certificate of the V2G Root.
MORootCertificate	Use for certificate from an eMobility Service provider. To support PnC charging with contracts from service providers that not derived their certificates from the V2G root.
V2GCertificateChain	ISO 15118 V2G certificate chain (excluding the V2GRootCertificate).

8.8. GetCertificateStatusEnumType

Enumeration

VALUE	DESCRIPTION
Accepted	Successfully retrieved the OCSP certificate status.
Failed	Failed to retrieve the OCSP certificate status.

8.9. GetInstalledCertificateStatusEnumType

Enumeration

VALUE	DESCRIPTION
Accepted	Normal successful completion (no errors).
NotFound	Requested resource not found.

8.10. HashAlgorithmEnumType

Enumeration

VALUE	DESCRIPTION
SHA256	SHA-256 hash algorithm.
SHA384	SHA-384 hash algorithm.
SHA512	SHA-512 hash algorithm.

8.11. InstallCertificateStatusEnumType

Enumeration

VALUE	DESCRIPTION
Accepted	The installation of the certificate succeeded.
Rejected	The certificate is invalid and/or incorrect OR the CSO tries to install more certificates than allowed.
Failed	The certificate is valid and correct, but there is another reason the installation did not succeed.

8.12. InstallCertificateUseEnumType

Enumeration

VALUE	DESCRIPTION
V2GRootCertificate	Use for certificate of the V2G Root, a V2G Charge Point Certificate MUST be derived from one of the installed V2GRootCertificate certificates.
MORootCertificate	Use for certificate from an eMobility Service provider. To support PnC charging with contracts from service providers that not derived their certificates from the V2G root.

8.13. Iso15118EVCertificateStatusEnumType

Enumeration

Iso15118EVCertificateStatusEnumType is used by: get15118EVCertificate:Get15118EVCertificate.conf

VALUE	DESCRIPTION
Accepted	exiResponse included. This is no indication whether the update was successful, just that the message was processed properly.
Failed	Processing of the message was not successful, no exiResponse included.

8.14. TriggerMessageStatusEnumType

Enumeration

Status in TriggerMessage.conf.

TriggerMessageStatusEnumType is used by: triggerMessage:TriggerMessage.conf

VALUE	DESCRIPTION
Accepted	Requested message will be sent.
Rejected	Requested message will not be sent.

VALUE	DESCRIPTION
NotImplemented	Requested message cannot be sent because it is either not implemented or unknown.

9. Configuration keys

9.1. CentralContractValidationAllowed

Required/optional	optional
Accessibility	RW
Type	Boolean
Description	If this variable exists and has the value *true*, then the Charge Point can provide a contract certificate that it cannot validate to the Central System for validation as part of the Authorize.req.

9.2. CertificateSignedMaxChainSize

Required/optional	optional
Accessibility	R
Type	integer
Description	This configuration key can be used to limit the size of the 'certificateChain' field from the CertificateSigned.req PDU. The value of this configuration key has a maximum limit of 10.000 characters. It is RECOMMENDED to set at least a size of 5600. This will allow the Charge Point to support most security architectures.

9.3. CertSigningWaitMinimum

Required/optional	Optional
Accessibility	RW
Type	integer
Description	This configuration key defines how long the Charge Point has to wait (in seconds) before generating another CSR, in the case the Central System accepts the SignCertificate.req, but never returns the signed certificate back. This value will be doubled after every attempt. The amount of attempts is configured at CertSigningRepeatTimes. If the certificate signing process is slow, this setting allows the Central System to tell the Charge Point to allow more time. Negative values must be rejected. The value 0 means that the Charge Point does not generate another CSR (leaving it up to the Central System to trigger another certificate installation).

9.4. CertSigningRepeatTimes

Required/optional	Optional
Accessibility	RW
Type	integer

Description	This configuration key can be used to configure the amount of times the Charge Point SHALL double the previous back-off time, starting with the number of seconds configured at CertSigningWaitMinimum, every time the back-off time expires without having received the CertificateSigned.req containing the signed certificate based on the CSR generated. When the maximum number of increments is reached, the Charge Point SHALL stop resending the SignCertificate.req, until it is requested by the Central System using a TriggerMessage.req. Negative values must be rejected. The value 0 means that the Charge Point does not double the back-off time.

9.5. `CertificateStoreMaxLength`

Required/optional	optional
Accessibility	R
Type	integer
Description	Maximum number of Root/CA certificates that can be installed in the Charge Point.

9.6. `ContractValidationOffline`

Required/optional	Required
Accessibility	RW
Type	Boolean
Description	If this variable is *true*, then the Charge Point will try to validate a contract certificate when it is offline.

9.7. `ISO15118PnCEnabled`

Required/optional	Required
Accessibility	RW
Type	Boolean
Description	If this variable set to *true*, then the Charge Point supports ISO 15118 plug and charge messages via the DataTransfer mechanism as described in this application note.