Signed Meter Values in OCPP

v1.0, 2025-02-10

Table of Contents

OCA Application Note

Relevant for OCPP version: 1.6, 2.0.1 and 2.1.

Version History

Version	Date	Author	Description
1.0	2025-02-10	Scott Thomson (Shell) Robert Schlabbach (Ubitricity) Patrick Rademakers (Shell)	First edition

1. Introduction

This document outlines the standardized method for sharing signed meter values between a Charging Station (CS) and a Charging Station Management System (CSMS). The purpose of which is to provide consumers a way of validating the validity of chargeable measurand(s) (Wh, Time, etc), in compliance with applicable laws or regulations in different jurisdictions. Those include German Mess- & Eichrecht laws.

Full compliance with these laws can include requirements that are out of scope for this whitepaper, which include:

- Unambiguously identifying a customer
- Handling of ad-hoc payments
- Handling of signed meter data between Charging Station Management System and Mobility Service Provider
- How the Charge Station Operator (CSO/CPO) or Mobility Operator must provide the details (including transaction IDs, public keys etc) to customers
- How the customer may verify the correctness of the measured/billable values

For more information on the out-of-scope elements, refer to guidance on local laws for the jurisdiction of interest. This whitepaper is limited to the transport of signed meter values between the Charging Station and the Charging Station Management System.

The measurements exchanged via OCPP can become part of Charge Detail Records (CDRs) that are exchanged via OCPI or a similar protocol, but the CDR exchange itself via those protocols is out of scope for this document.

Terminology & acronyms

Term	Meaning
CDR	Charge Detail Record
CSMS	Charging Station Management System
CS	Charging Station (which may container a number of EVSEs)
CSO	Charging Station Operators
EDL	Energie Dienst Leistung (Energy Service)
EVSE	Electric Vehicle Supply Equipment, the equipment needed to charge one vehicle
M&E	Mess- & Eichrecht
MSP	Mobility Service Provider (sometimes also referred to as EMSP)
OCA	Open Charge Alliance
OCMF	Open Charge Metering Format
OCPI	Open Charge Point Interface
OCPP	Open Charge Point Protocol
PTB	Physikalisch-Technische Bundesanstalt (Physical-Technical Federal Institute)

Term	Meaning
SAFE	Software Alliance For E-mobility

References

A list of references relevant for this paper:

- [BNetzA], Bundesnetzagentur database (German) with data for all commissioned charging stations – including Public Keys , Bundesnetzagentur, https://www.bundesnetzagentur.de/DE/Fachthemen/ElektrizitaetundGas/E-Mobilitaet/start.html
- [EICH_A7], https://www.gesetze-im-internet.de/messev/anlage_7.html
- [GCIR], German Charging Infrastructure Regulations (Source: Netherlands Enterprise Agency), Maarten Venselaar, Harm-Jan Idema, Thomas Endriß, https://www.rvo.nl/sites/default/files/2019/04/German%20charging%20infrastructure%20regulations%20report%20march%202019_0.pdf
- [OCMF], https://github.com/SAFE-eV/OCMF-Open-Charge-Metering-Format/blob/master/OCMF-en.md
- [OCPP16J], OCPP 1.6 edition 2, Open Charge Alliance, https://openchargealliance.org/protocols/open-charge-point-protocol/
- [OCPP201], OCPP 2.0.1 edition 3, Open Charge Alliance, https://openchargealliance.org/protocols/open-charge-point-protocol/
- [OCPP21], OCPP 2.1 edition 1, Open Charge Alliance, https://openchargealliance.org/protocols/open-charge-point-protocol/
- [S.A.F.E.], https://safe-ev.org/en/

2. Concepts

To ensure compliance with the applicable laws & regulations surrounding validated metering of transactions including metered consumption, the following must be achieved:

a. Ensure trustworthy measurements.

b. Ensure those measurements are communicated throughout the ecosystem in a tamper-proof way,

c. Provide customers with a method to verify measurements that are the underpinning for their invoices.

For a), see section 2.2 for informative description, and for b), see the rest of this document. Item c) is out of scope for this whitepaper.

Section 2.1 provides an overview of how a Charging Station operator may wish to configure signed meter values for different use cases.

2.1. Configuration flexibility & use-cases

To provide flexibility of use-cases, the configuration of the Charging Station may be changed based on configuration keys (OCPP 1.6) or variables (OCPP 2.0.1/2.1). In sections 3 and 4 the specific requirements for each version of OCPP are described. Appendix 5.1 provides a matrix of how the configurations operate.

For example, by setting only SampledDataSignReadings to true, this will enable signed start and end meter values to be placed within the StopTransaction.req (OCPP 1.6) or TransactionEventRequest(Ended) (OCPP 2.0.1/2.1). Setting SampledDataSignStartedReadings to true, in addition, would result in a signed start meter value being sent to the Central System within either a MeterValues.req (OCPP 1.6) or TransactionEventRequest(Started) (OCPP 2.0.1/2.1). To be compliant with Eichrecht, at minimum signed start and stop value must be shared with the Central System – this could be done via just the StopTransaction.req/TransactionEventRequest(Ended) message, which would require only setting SampledDataSignedReadings to true.

The variables are split between Read-Writeable (for switching the signed meter values on and off globally) and Read-Writeable or Read-Only, depending on the capabilities of the Charging Station. For example, if a charging station is unable, based on hardware capability, to sign intermediate values, then SampledDataSignUpdatedReadings or AlignedDataSignUpdatedReadings may be set to false and be read-only variables.

2.2. Correctness of measurement

To ensure the correct measurement, the meter must be certified and calibrated. If the certification/calibration can expire, then the CSO must have a process to re-certify/calibrate the meter in a maintenance cycle. It is up to the CSO to ensure this is done at the right time. If not, if the certification/calibration expires, conformance to the applicable laws would be lost. This means that from that point onward, the rules for having a non-certified solution apply. Depending on what category the charging solution falls under (see [EICH_A7] for possibilities within Germany) the certification validity period can vary from 4 to 16 years.

Not only is a calibrated meter needed, as a charging station, communication is required internally to get the values from the meter and send them to the CSMS. However, different meters communicate in a unique way

and have their own way of signing data to protect against tampering. So, it is important to understand what protocol the meter speaks, because it will affect the data sent in our OCPP metering messages. It should be noted that it is possible to be Eichrecht compliant without sharing the meter values with the CSMS – they can be stored locally and provided on a display based on a customer prompt. This is out of the scope of this whitepaper or OCPP.

2.2.1. Meter information

The meter (or more technically for Eichrecht, a "Messkapsel" or measuring capsule, which could be a system including a meter and a separate signing module) itself must be a certified, and finally the entire charging station including the meter must be certified for compliance with local laws, as needed. The public key linked to the private key that certified the meter value must then be made available to the charging station to share with the CSMS, as needed. There are different potential coding methods for the meter values that are to be shared with the Central System. The most common two at time of writing are OCMF or EDL, however these are not prescribed by the whitepaper.

2.2.2. OCMF: Open Charge Metering Format

The most common type of encoding method supported by different manufacturers in Germany is OCMF (Open Charge Metering Format). The aim of OCMF is to describe an independent and generally usable data format for recording meter readings from charging stations that are relevant under the German calibration law. The full OCMF definition can be found via [OCMF].

Note that the OCMF document talks about OCPP 1.5, not OCPP 1.6 or 2.0.1/2.1, but the elements of 1.5 it is referring to, still exist in later versions, and are still applicable.

2.2.3. EDL

EDL stands for Energie Dienst Leistung, or Energy Service, an encoding method that is common in the German market and supported by assorted brands, used to encode the meter values.

2.2.4. Other meter formats

There are several other meter formats available and possible now and in the future. How to handle these different formats is outlined in sections 3.2.1 (OCPP 1.6J) and 4.2.2 (OCPP 2.x).

2.2.5. Meter Public Key Verification

To enable a user/customer or Charge Point Operator to validate that the public key is correct, the Bundesnetzagentur (Federal Network Agency) requires that all commissioned charging stations are published on their databases. This includes the public keys for a given EVSE. See references for link.

2.2.6. Public key

The public key is specific to the meter that does the signing of the measurements. It should only change when the meter is physically replaced or modified, since the public key is reported on the body of the meter (for M&E in Germany, this must be visible to the customer via a window in the charging station or as part of the digital display, however that 2nd option then requires the screen and its software to be part of the certification process).

During installation/configuration of the charging station this key must initially be configured in the charging station. The public key can be transmitted with the signed meter values. Alternatively, the CSMS can retrieve the public key from a configuration key/variable to reduce transferred data usage.

Charging stations typically have one meter for each EVSE, so a charging station with multiple EVSEs will also have to provide multiple public keys for retrieval. See MeterPublicKey[ConnectorID] (3.3.1 or 4.3.2) for more information on this.

2.3. Transactions & CDRs

2.3.1. Transaction flow

To start a charging session six requirements will have to be satisfied (OCPP 1.6 described):

1. The driver must be allowed to charge; done locally via RFID, via an Authorize request to the CSMS, using cached Authorize results, the local list pushed by the CSMS, or a proprietary integration to a payment terminal. Alternatively, the authorization can be done remotely via a RemoteStartTransaction.req by the CSMS.
2. The cable must be properly plugged in, to both the car and the charging station. Should this be applied to a wireless charging session then this step would mean the wireless connection is operational.
3. The charging station must configure the energy meter by sending at least a user identification or a session identification. (internally to CS)
4. The time of the energy meter must be verified. (internally to CS)
5. Maybe a charging tariff must be configured. (internally to CS)
6. Where supported, the official eMI3 EVSE Id must be configured as additional metadata within the energy meter, so that the EV driver can easily verify the public key against a public key database (see 2.1.4). (internally to CS)

When these requirements are met, the charging station takes the initial measurement from the certified meter and sends a StartTransaction.req to the CSMS. This marks the beginning of the charging session. During the charging session the charging station, if so configured, may send intermediate *meterValue* updates to the CSMS. Depending on the use cases and settings of the configuration keys outlined in this whitepaper, these intermediate values may be signed or unsigned type.

When one of the two start requirements is revoked, e.g., by unplugging, swiping the RFID card again, pushing a stop button, or via a RemoteStopTransaction.req, the charging station will stop energy transfer, get the last meter value, and report this in the StopTransaction.req message that signals the end of the transaction to the CSMS.

NOTE	StopTransaction.req can only be sent after reading the last certified meter value. In most cases this will not cause any noticeable delay, but it has been reported that some meters in exceptional situations can cause a delay of up to 30 seconds. At the time of writing, it is not known which meter(s) specifically have this delay.

2.3.2. Transaction duration

The OCPP transaction duration is the time between the message timestamp of the StartTransaction.req (TransactionEventRequest(Started) for OCPP 2.x) and the message timestamp of the StopTransaction.req (TransactionEventRequest(Ended) for OCPP 2.x). Be aware that the cost calculation typically must be done on timestamps in the signed meter data between the first and last signed value.

The timestamp of the StopTransaction.req (TransactionEventRequest(Ended) for OCPP 2.x) message will be the time that the charge was stopped. The timestamp of the *sampledValue* item with a signed meter value will be the time that the MID meter is requested to create the signed meter value.

2.3.3. Charge Detail Record

German Eichrecht laws require CDRs to include signed meter values for the start and the end of the transaction. Signed intermediate meter values are optional depending on if the charging session has a change in tariff during the transaction, then it is mandatory. The CSMS combines all the charging session information into a CDR.

2.3.4. Direct/Ad-hoc Payments

This whitepaper outlines how to manage the transport of messages from typical RFID activated transactions. For full compliance with Eichrecht and direct payment methods, it may be required to inject the tariff information into the signed data. A method for doing this within OCPP 1.6 & OCPP 2.x is under consideration.

2.4. Notes in this document

Notes are used to provide additional information to help in understanding or using this document. These are informative only, and some notes will be intended primarily for operators (CSO – Charging Station Operators), and some notes will be intended primarily for implementors (Charging Station Manufacturers).

3. OCPP 1.6 Implementation

This section covers how to extend OCPP 1.6 to support signed meter data in a standardized way. Subsection 3.2 explains a new datatype borrowed from OCPP 2.x called SignedMeterValueType. This contains the details of the payload of a signed meter value according to this whitepaper. Subsection 3.3 contains additional configuration settings (see 2.1 for more detail).

3.1. Messages

3.1.1. StartTransaction.req

As explained in section 2.3.1 (Transaction flow), a transaction start is communicated to the central system via a StartTransaction.req. This message has an integer field called *meterStart* to pass the initial *meterValue* but only as unsigned data (and as an integer Wh value), without the option to include signed data as well. The sending of a signed meter value is the topic of the next section.

3.1.2. MeterValues.req – for delivering a signed meter value at the start of the transaction

The use of this message at the start of a transaction is optional for this whitepaper. Its intent is to enable a signed meter value to be sent to the CSMS at the start of the transaction, immediately following the StartTransaction.req message. See the keys in 3.3.4 for sampled data.

3.1.3. MeterValues.req – for delivering intermediate signed meter values

Some use cases may require additional MeterValues.req to receive updated intermediate values based on the settings of the keys:

- MeterValuesSampledData
- MeterValuesSampledDataMaxLength
- MeterValueSampledInterval
- ClockAlignedDataInterval
- MeterValuesAlignedData
- MeterValuesAlignedDataMaxLength

Typically, these are unsigned, but with the new configuration key SampledDataSignUpdatedReadings (3.3.6) or AlignedDataSignUpdatedReadings (3.3.8) these can be set to be of a signed type.

NOTE	It may take some time for the metering device to sign a value, therefore, higher frequency intervals may not be possible. It is recommended that an implementation of the approach described in this whitepaper, documents the highest possible frequency of signed meter values.

3.1.4. StopTransaction.req

By setting SampledDataSignReadings (3.3.3) or AlignedDataSignReadings (3.3.6), the charging station SHALL

send signed values with the CSMS within the StopTransaction.req message.

If the signed meter data to populate the *sampledValue* field of the *transactionData* in the StopTransaction.req are in separate data containers, (e.g., start and stop signed meter values are in separate OCMF containers), then:

- The signed data for the start meter value SHALL go in a *sampledValue* (of SampledValue type) with context Transaction.Begin.
- The signed data for the stop meter value SHALL go in a *sampledValue* (of SampledValue type) with context Transaction.End
- And both placed in the StopTransaction.req *transactionData* field.

If the signed meter data are in a single data container (e.g., both start and stop signed meter values are in one OCMF container), then:

- The signed data for the meter values SHALL go in a *sampledValue* (of SampledValueType) with context Transaction.End
- And placed in the StopTransaction.req *transactionData* field.

NOTE This is to minimize data duplication in case the meter sends the signed meter values in one data container to the Charging Station.

3.2. Data types

This section explains the additional data types used for implementation in OCPP 1.6.

3.2.1. SignedMeterValueType – Reused from OCPP 2.x

For compatibility, the data structure that is already specified in OCPP 2.x: SignedmeterValueType SHALL be used to format the signed meter data. This is used in StopTransaction.req as the form taken by the value within *sampledValue* of the *transactionData* field. It is also used in the *sampledValue* of the *meterValue* field in a MeterValues.req. Figure 1 below is an extract from the OCPP 2.x specification, and Table 1 outlines the requirements per this whitepaper.

Field Name	Field Type	Card.	Description
signedMeterData	string[0..2500]	1..1	Required. Base64 encoded, contains the signed data which might contain more then just the meter value. It can contain information like timestamps, reference to a customer etc.
signingMethod	string[0..50]	1..1	Required. Method used to create the digital signature.
encodingMethod	string[0..50]	1..1	Required. Method used to encode the meter values before applying the digital signature algorithm.
publicKey	string[0..2500]	1..1	Required. Base64 encoded, sending depends on configuration variable *PublicKeyWithSignedMeterValue*.

Figure 1. OCPP2.0.1 Edition 3 Part 2 - Specification, chapter 2.46, page 413

Table 1. Requirements for SignedMeterValueType

Field Name	Requirements
signedMeterData	SHALL be populated based on the requirements in Figure 1.
signingMethod	May already be included in the *signedMeterData* block (depending on the encoding format of the signed meter data). If it is already included in the *signedMeterData*, then this SHALL be an empty string. If not included in *signedMeterData*, then this field SHALL be populated. See Table 13 for value definitions.
encodingMethod	SHALL be populated based on the format of the meter (e.g., OCMF, EDL, etc).
publicKey	SHALL be included based on the configuration of the PublicKeyWithSignedMeterValue (3.3.2) setting. If the public key is not sent with a message, it SHALL be sent as an empty string. If it is sent, it SHALL contain a Base64 encoded string with the content as specified in 3.2.2.

3.2.2. *publicKey* field content specification (for both OCPP 1.6 and OCPP 2.x)

The Base64 encoded content of the *publicKey* field shall be a colon-separated string:

<marker>:<encoding>:<content-type>:<printed-public-key>

Where:

- <marker> identifies the content format specified in this document. It is **oca**.
- <encoding> specifies the encoding of the <printed-public-key>, see Table 2.
- <content-type> specifies the type of content **after** decoding <printed-public-key>. See Table 3.
- <printed-public-key> contains the public key **as printed on the certified meter**.

NOTE The public key representation as printed on the certified meter was chosen for easiest matching of the public key visible to the customer to the one transmitted with the signed meter values. Since there is no standardized format for printing the public key on a certified meter, the additional information above is needed to be able process the public key for signature validation.

Table 2. publicKey encoding values

Encoding value	**Content of <printed-public-key>**
base16	A case-insensitive string containing a hexadecimal representation of the content. Non-hexadecimal character strings (i.e. other than 0-9a-fA-F) and a hexadecimal prefix (0x) SHALL be ignored.
base64	A Base64 encoded string.

Table 3. publicKey content-type values

content-type value	**Decoded content of <printed-public-key>**
asn1	A binary ASN.1 structure containing the signature algorithm, its parameters and the actual public key.

NOTE The above tables are intended to be extended with additional values as needed. It is strongly encouraged for vendors needing any additional value to contact the OpenChargeAlliance for inclusion, so that additional values can be specified in a uniform way.

NOTE See 5.3 for an example of how to put together the *publicKey* value.

3.3. Configuration settings

The configuration keys described in this section are based on the naming convention of OCPP 2.x, to provide consistency across versions.

3.3.1. MeterPublicKey[ConnectorID]

Table 4. Configuration Key MeterPublicKey[ConnectorID]

Variable name	**MeterPublicKey[ConnectorID]**
Required/optional (for this whitepaper)	Required
Accessibility	RO
Type	string
Description	Configuration key that can be used to retrieve the public key for a meter connected to a specific connector/EVSE. The value of this configuration key SHALL be as specified in 3.2.2. Examples: It is expected that each EVSE will have its own meter, therefore, each configuration of charging station will be slightly different. For a charging station with a single physical connector, this would always be MeterPublicKey1. For a charging station with a dual physical connector, that functions as a single EVSE (can only charge one vehicle at a time), MeterPublicKey1 and MeterPublicKey2 would return the same value. For a charging station that has two EVSEs with one connector on each, the configuration keys MeterPublicKey1 and MeterPublicKey2 would return different values.

NOTE The value of a configuration key in OCPP 1.6 is limited in length to 500 bytes. Consequently, this cannot be used for public keys longer than that.

3.3.2. PublicKeyWithSignedMeterValue

The purpose of using this configuration key is to define when a charging station needs to include the public key in a signed meter value. It is re-used from OCPP 2.x and defined in Table 5.

Table 5. Configuration Key PublicKeyWithSignedMeterValue

Key name	**PublicKeyWithSignedMeterValue**
Required/optional (for this whitepaper)	Required
Accessibility	RW
Type	Enum String
Description	The value may be set to the following values: • Never: i.e., would not be sent automatically with signed meter values. CSMS already knows the value or requests it from configuration variable "MeterPublicKey". • OncePerTransaction • EveryMeterValue: public key in every *meterValue*, including the ones in the *transactionData* field in the StopTransaction.req

NOTE If this configuration is set to Never then in case of a meter swap, public keys associated with historical transactions (made before the meter change) will no longer be available by a GetConfiguration.req. This could be a compliance issue.

3.3.3. SampledDataSignReadings

Configuration Key SampledDataSignReadings

Key name	**SampledDataSignReadings**
Required/optional (for this whitepaper)	Required
Accessibility	RW
Type	Boolean
Description	If set to true, the Charging Station SHALL include signed meter values in the StopTransaction.req to the CSMS for those measurands configured in StopTxnSampledData, which can be signed by the certified meter, and optionally in additional messages, as configured by the configuration settings SampledDataSignStartedReadings (see 3.3.5) and SampledDataSignUpdatedReadings (see 3.3.6).

NOTE The description of SampledDataSignReadings is extended from the OCPP 2.x specification. It now globally enables or disables the transmission of signed sampled meter values. Which messages will contain signed sampled meter values, in addition to StopTransaction.req, is controlled by the additional configuration keys SampledDataSignStartedReadings and SampledDataSignUpdatedReadings, specified in this document.

3.3.4. StartTxnSampledData

With this new configuration key, when set to any measurands, the Charging Station SHALL send a MeterValues.req right after the StartTransaction.req, with SampledValues for the configured measurands with the context "Transaction.Begin". If it is not implemented or the value is an empty list, this extra MeterValues.req will not be sent (standard OCPP 1.6 behaviour).

Table 6. Configuration Key StartTxnSampledData

Key name	**StartTxnSampledData**
Required/optional (for this whitepaper)	Optional
Accessibility	RW or RO
Type	CSL of Measurands
Description	If the CSL is not empty, the Charging Station SHALL send a MeterValues.req with SampledValues having the context "Transaction.Begin" to the CSMS for the measurands contained in the CSL, right after the StartTransaction.req.

3.3.5. SampledDataSignStartedReadings

With this new configuration key, when set to true, the start meter values (according to 3.3.4) sent from the Charging Station to the Central System SHALL be signed, with format "signedData" and the value field formatted according to 3.2.1. If set to false, start meter values SHALL be of an unsigned format.

Table 7. Configuration Key SampledDataSignStartedReadings

Key name	**SampledDataSignStartedReadings**
Required/optional (for this whitepaper)	Optional
Accessibility	RW or RO
Type	Boolean
Description	If set to true, the Charging Station SHALL include signed meter values in the *meterValue* field of the MeterValues.req sent to the CSMS at the start of the transaction for those measurands configured in StartTxnSampledData which can be signed by the certified meter. This setting only has an effect if SampledDataSignReadings is set to true and StartTxnSampledData is implemented and contains any signable measurands.

3.3.6. SampledDataSignUpdatedReadings

With this new configuration key, when set to true, subsequent intermediate meter values (those not according to 3.3.5) sent from the Charging Station to the Central System SHALL be signed. If this key is set to false, intermediate values SHALL be according to the settings of the configuration variables outlined in 3.1.3 and of an unsigned format.

Table 8. Configuration Key SampledDataSignUpdatedReadings

Key name	**SampledDataSignUpdatedReadings**
Required/optional (for this whitepaper)	Required
Accessibility	RW or RO
Type	Boolean
Description	If set to true, the Charging Station SHALL include signed meter values in the *meterValue* field in the MeterValues.req to the CSMS for those measurands configured in MeterValuesSampledData which can be signed by the certified meter. This setting only has an effect if SampledDataSignReadings is set to true.

3.3.7. AlignedDataSignReadings

Table 9. Configuration Key AlignedDataSignReadings

Key name	**AlignedDataSignReadings**
Required/optional (for this whitepaper)	Required
Accessibility	RW
Type	Boolean
Description	If set to true, the Charging Station SHALL include signed meter values in the StopTransaction.req to the CSMS for those measurands configured in StopTxnAlignedData which can be signed by the certified meter, and optionally in other messages, as per the AlignedDataSignUpdatedReadings configuration key (see 3.3.7).

NOTE AlignedDataSignReadings globally enables or disables the transmission of signed aligned meter values. Which messages will contain signed sampled meter values, in addition to StopTransaction.req, is controlled by the additional configuration key AlignedDataSignUpdatedReadings, specified in this document.

3.3.8. AlignedDataSignUpdatedReadings

With this new configuration key, when set, clock-aligned intermediate meter values sent from the Charging Station to the Central System SHALL be signed. If this key is not set, intermediate values SHALL be according to the settings of the configuration variables outlined in 3.1.3 and of an unsigned format.

Table 10. Configuration Key AlignedDataSignUpdatedReadings

Key name	**AlignedDataSignUpdatedReadings**
Required/optional (for this whitepaper)	Required
Accessibility	RW or RO
Type	Boolean
Description	If set to true, the Charging Station SHALL include signed meter values in the *meterValue* field in the MeterValues.req messages to the CSMS for those measurands configured in MeterValuesAlignedData which can be signed by the certified meter. This variable only has an effect if AlignedDataSignReadings (3.3.7) is set to true.

4. OCPP 2.x Implementation

OCPP 2.0.1/2.1 were developed with native support for signed meter value functionality (and is the source of much being used for the 1.6 section of this whitepaper). To enable the use cases outlined in section 2.1, configuration variables (see section 4.3) are specified to standardize them.

4.1. Messages

4.1.1. TransactionEventRequest (*eventType* = Started and *eventType* = Updated)

These messages are used to notify about the start of a transaction (*eventType* = Started) or provide an update to an ongoing transaction (*eventType* = Updated). Within OCPP 2.x both start and updated natively can provide signed meter values to the CSMS, depending on the settings of the Charging Station. The configuration variables in section 4.3 are used to control when signed meter values are to be provided to the CSMS, including transaction-related start and intermediate meter values.

Depending on the configured TxStartPoint and the time taken for the certified meter (or Messkapsel) to provide the signed meter data to the charging station, the location of the first signed meter value can be sent either in TransactionEventRequest (*eventType* = Started) or TransactionEventRequest (*eventType* = Updated, *triggerReason* = DataSigned).

To deliver intermediate meter values, the TransactionEventRequest (*eventType* = Updated) message is used. The use of these messages is optional, based on the settings of these configuration variables (found in OCPP 2.x specification):

- SampledDataTxUpdatedMeasurands
- SampledDataTxUpdatedInterval
- AlignedDataTxUpdatedMeasurands
- AlignedDataTxUpdatedInterval

To sign these intermediate values, the following variables may be set to true (AlignedDataSignReadings or SampledDataSignReadings are required in addition to enable the signed meter value function):

- AlignedDataSignUpdatedReadings (4.3.4)
- SampledDataSignUpdatedReadings (4.3.7)

4.1.2. TransactionEventRequest (*eventType* = Ended)

This message is used to notify about the completion of a transaction. When providing signed meter values (i.e., configuration variables AlignedDataSignReadings or SampledDataSignReadings are set to true) it is mandatory that within this message at least two signed meter values are present – one for the start meter value and one for the stop meter value.

If the signed meter data to populate the *signedMeterData* field of the signed meter value of the TransactionRequestEvent (*eventType* = Ended) are in separate data containers (e.g., start and stop signed meter

values are in separate OCMF containers), then:

- The signed data for the start meter value SHALL go in a *sampledValue* (of SampledValueType) with context Transaction.Begin.
- The signed data for the stop meter value SHALL go in a *sampledValue* (of SampledValueType) with context Transaction.End
- And both placed in the TransactionEventRequest(Ended) *meterValue* field.

If the signed meter data is in a single data container (e.g., both start and stop signed meter values are in one OCMF container), then:

- The signed data for the meter values SHALL go in a *sampledValue* (of SampledValueType) with context Transaction.End
- And placed in the TransactionEventRequest(Ended) *meterValue* field.

NOTE Due to the complexity of meter configurations, CSMS should be configured to flexibly handle receiving the Start and Stop meter values in the TransactionEventRequest (*eventType* = Ended) in the combinations outlined above.

4.2. Data Types

4.2.1. SignedMeterValueType

This data type is to be used to complete the signedMeterValue field of the *sampledValue* field of the *meterValue* field within a TransactionEventRequest message. Figure 2 below is an extract from the OCPP 2.x specification, and Table 12 outlines the requirements per this whitepaper.

Field Name	Field Type	Card.	Description
signedMeterData	string[0..2500]	1..1	Required. Base64 encoded, contains the signed data which might contain more then just the meter value. It can contain information like timestamps, reference to a customer etc.
signingMethod	string[0..50]	1..1	Required. Method used to create the digital signature.
encodingMethod	string[0..50]	1..1	Required. Method used to encode the meter values before applying the digital signature algorithm.
publicKey	string[0..2500]	1..1	Required. Base64 encoded, sending depends on configuration variable *PublicKeyWithSignedMeterValue*.

Figure 2. OCPP2.0.1 Edition 3 Part 2 - Specification, chapter 2.46, page 413

Table 11. Requirements for SignedMeterValueType

Field name	Formatting Requirements
signedMeterData	SHALL be populated based on the requirements in Figure 2.
signingMethod	May already be included in the *signedMeterData* block (depending on the encoding format of the signed meter data). If it is already included in the *signedMeterData*, then this SHALL be an empty string. If not included in *signedMeterData*, then this field SHALL be populated. See Table 13 for value definitions.
encodingMethod	SHALL be populated based on the format of the meter (e.g., OCMF, EDL, etc.).
publicKey	SHALL be included based on the configuration of the PublicKeyWithSignedMeterValue (4.3.1) setting. If the public key is not sent with a message, it SHALL be sent as an empty string. If it is sent, it SHALL contain a Base64 encoded string with the content as specified in 3.2.2.

Table 12. signingMethod values

signingMethod value	Algorithm	Curve	Key Length	Hash Algorithm
ECDSA-secp192k1-SHA256	ECDSA	secp192k1	192 bits	SHA-256
ECDSA-secp256k1-SHA256	ECDSA	secp256k1	256 bits	SHA-256
ECDSA-secp192r1-SHA256	ECDSA	secp192r1	192 bits	SHA-256
ECDSA-secp256r1-SHA256	ECDSA	secp256r1	256 bits	SHA-256
ECDSA-brainpool256r1-SHA256	ECDSA	brainpool256r1	256 bits	SHA-256
ECDSA-secp384r1-SHA256	ECDSA	secp384r1	384 bits	SHA-256
ECDSA-brainpool384r1-SHA256	ECDSA	brainpool384r1	384 bits	SHA-256

4.3. Configuration Variables

4.3.1. PublicKeyWithSignedMeterValue

Table 13. Configuration Variable PublicKeyWithSignedMeterValue

Variable name	**PublicKeyWithSignedMeterValue**
Component	OCPPCommCtrlr
Required/optional (for this whitepaper)	Required
Accessibility	RW
Type	Enum (OptionList)
Description	The value may be set to the following values: • Never: i.e., would not be sent automatically with signed meter values. CSMS already knows the value or requests it from configuration variable "PublicKey". • OncePerTransaction • EveryMeterValue: public key in every *meterValue* of each TransactionEventRequest

NOTE If this variable is set to Never then in case of a meter swap, public keys associated with historical transactions (made before the meter change) will no longer be available by a GetVariablesRequest. This could be a compliance issue.

4.3.2. PublicKey

Table 14. Configuration Variable PublicKey

Variable name	**PublicKey**
Component	FiscalMetering
Required/optional (for this whitepaper)	Required
Accessibility	RO
Type	String
Description	Configuration variable that can be used to retrieve the public key for a meter connected to a specific EVSE. The value of this configuration variable SHALL be as specified in 3.2.2.

NOTE The **FiscalMetering** component will typically be at the EVSE-tier level. There may be one at the Charging Station tier level for the overall input of the Charging Station.

4.3.3. AlignedDataSignReadings

Table 15. Configuration Variable AlignedDataSignReadings

Variable name	**SignReadings**
Component	AlignedDataCtrlr
Required/optional (for this whitepaper)	Required
Accessibility	RW
Type	Boolean
Description	If set to true, the Charging Station SHALL include signed meter values in the TransactionEventRequest (*eventType* = Ended) to the CSMS for those measurands configured in AlignedDataTxEndedMeasurands which can be signed by the certified meter, and optionally in other messages controlled by AlignedDataSignUpdatedReadings (see 4.3.4).

NOTE

AlignedDataSignReadings globally enables or disables the transmission of signed aligned meter values. Which messages will contain signed sampled meter values, in addition to TransactionEventRequest (*eventType* = Ended), is controlled by the additional configuration variable AlignedDataSignUpdatedReadings, specified in this document.

4.3.4. AlignedDataSignUpdatedReadings

Table 16. Configuration Variable AlignedDataSignUpdatedReadings

Variable name	**SignUpdatedReadings**
Component	AlignedDataCtrlr
Required/optional (for this whitepaper)	Required
Accessibility	RW or RO
Type	Boolean
Description	If set to true, the Charging Station SHALL include signed meter values in the TransactionEventRequest (*eventType* = Updated) messages to the CSMS for those measurands configured in AlignedDataTxUpdatedMeasurands which can be signed by the certified meter. This variable only has an effect if AlignedDataSignReadings (4.3.3) is set to true.

4.3.5. SampledDataSignReadings

Table 17. Configuration Variable SampledDataSignReadings

Variable name	**SignReadings**
Component	SampledDataCtrlr
Required/optional (for this whitepaper)	Required
Accessibility	RW
Type	Boolean
Description	If set to true, the Charging Station SHALL include signed meter values in the TransactionEventRequest (*eventType* = Ended) message to the CSMS for those measurands configured in SampledDataTxEndedMeasurands which can be signed by the certified meter and optionally in other messages, as per SampledDataSignStartedReadings (see 4.3.6) and SampledDataSignUpdatedReadings (see 4.3.7).

NOTE SampledDataSignReadings globally enables or disables the transmission of signed sampled meter values. Which messages will contain signed sampled meter values, in addition to TransactionEventRequest (*eventType* = Started), is controlled by the configuration variables SampledDataSignStartedReadings and SampledDataSignUpdatedReadings, specified in this document.

4.3.6. SampledDataSignStartedReadings

Table 18. Configuration Variable SampledDataSignStartedReadings

Variable name	**SignStartedReadings**
Component	SampledDataCtrlr
Required/optional (for this whitepaper)	Required
Accessibility	RW or RO
Type	Boolean
Description	If set to true, the Charging Station SHALL include signed meter values in the TransactionEventRequest (*eventType* = Started or Updated) to the CSMS for those measurands configured in SampledDataTxStartedMeasurands which can be signed by the certified meter. This setting only has an effect if SampledDataSignReadings is set to true.

4.3.7. SampledDataSignUpdatedReadings

Table 19. Configuration Variable SampledDataSignUpdatedReadings

Variable name	**SignUpdatedReadings**
Component	SampledDataCtrlr
Required/optional (for this whitepaper)	Required
Accessibility	RW or RO
Type	Boolean
Description	If set to true, the Charging Station SHALL include signed meter values in the TransactionEventRequest (*eventType* = Updated) messages to the CSMS for those measurands configured in SampledDataTxUpdatedMeasurands which can be signed by the certified meter. This setting only has an effect if SampledDataSignReadings is set to true.

5. Appendix

5.1. Configuration Key/Variable Settings Matrix

Table 21 describes how the variables, when set and in what combination, affect which meter values should be signed. The upper section of the table lists the variables, and the middle section defines the output. The bottom section provides more detail on the messages involved.

Table 20. Configuration Settings Matrix

Configuration settings							
SampledDataSignReadings		FALSE	TRUE	TRUE	TRUE	TRUE	TRUE
AlignedDataSignReadings		FALSE	TRUE	TRUE	TRUE	TRUE	TRUE
SampledDataSignStartedReadings		N/A	FALSE	TRUE	TRUE	TRUE	TRUE
SampledDataSignUpdatedReadings		N/A	FALSE	FALSE	TRUE	FALSE	TRUE
AlignedDataSignUpdatedReadings		N/A	FALSE	FALSE	FALSE	TRUE	TRUE
Which meter values are signed							
Meter Value Context	Message	None	End Only	Start & End	Periodic Intermediate	Aligned Intermediate	Both Intermediate
Transaction.Begin	**2.x**: TxEvent (Started) OR 1st (Updated) **1.6**: MeterValuesReq	-	-	Y	Y	Y	Y
Sample.Periodic	**2.x**: TxEvent (Updated) **1.6**: MeterValuesReq	-	-	-	Y	-	Y
Sample.Clock	**2.x**: TxEvent (Updated) **1.6**: MeterValuesReq	-	-	-	-	Y	Y
Transaction.Begin & Transaction.End	**2.x**: TxEvent (Ended) **1.6**: StopTransactionReq	-	Y	Y	Y	Y	Y

5.2. OCPP 1.6 Example Message (Informative)

An example of a StopTransaction.req with the signed meter data for both the start and end of a transaction within one OCMF container in one sampled value with context Transaction.End. The public key is also shared.

[2, "729491009", "StopTransaction", \{"meterStop": 108814, "timestamp": "2023-05-19T13:55:48Z", "idTag": "HRWWBX8", "reason": "Local", "transactionData": [\{"timestamp": "2023-05-19T13:55:48Z", "sampledValue": [\{"format": "SignedData", "value": "\{\"signedMeterData\":\"T0NNRnx7IkZWIiA6ICIxLjAiLCJHSSIgOiAiRFpHLUdTSDAxLjFLMkwiLCJHUyIgOiAiMURaRzAwMjgyMjUxNzkiLCJHViIgOiAiMjMwIiwiUEciIDogIlQ5NiIsIk1WIiA6ICJEWkciLCJNTSIgOiAiR1NIMDEuMUsyTCIsIk1TIiA6ICIxRFpHMDAyODIyNTE3OSIsIk1GIiA6ICIyMzAiLCJJUyIgOiB0cnVlLCJJVCIgOiAiQ0VOVFJBTF8xIiwiSUQiIDogIkhSV1dCWDgiLCJDVCIgOiAiRVZTRUlEIiwiQ0kiIDogIjIyQlozMTc4QTAiLCJSRCIgOiBbeyJUTSIgOiAiMjAyMy0wNS0xOVQxNTo1MjozOSwwMDArMDIwMCBJIiwiVFgiIDogIkkiLCJSViIgOiAiMC4wMDAiLCJSSSIgOiAiMDEtMDA6OTguMDguMDAuRkYiLCJSVSIgOiAia1doIiwiUlQiIDogIkRDIiwiRUYiIDogIlsiNUIiA6ICJHIn0seyJUTSIgOiAiMjAyMy0wNS0xOVQxNTo1MzozOCwwMDArMDIwMCBJIiwiVFgiIDogIkUiLCJSViIgOiAiMC42MzYiLCJSSSIgOiAiMDEtMDA6OTguMDguMDAuRkYiLCJSVSIgOiAia1doIiwiUlQiIDogIkRDIiwiRUYiIDogIlsiNUIiA6ICJHIn0seyJUTSIgOiAiMjAyMy0wNS0xOVQxNTo1MjozOSwwMDArMDIwMCBJIiwiVFgiIDogIkIiLCJSViIgOiAiMC4wMDIyIiwiUkkiIDogIjAxLTAwOjlDLjA4LjAwLkZGIiwiUlUiIDogIk9obSIsIlJUIiA6ICJEQyIsIkVGIiA6ICIiLCJTVCIgOiAiRyJ9LHsiVE0iIDogIjIwMjMtMDUtMTlUMTU6NTM6NTgsMDAwKzAyMDAgSSIsIlRYIiA6ICJFIiwiUlYiIDogIjc5IiwiUkkiIDogIjAxLTAwOjAwLjA4LjA2LkZGIiwiUlUiIDogInMiLCJSVCIgOiAiREMiLCJFRiIgOiAiIiwiU1QiIDogIkcifV19fHsiU0EiIDogIkVDRFNBLXNlY3AyNTZrMS1TSEEyNTYiLCJTRCIgOiAiMzA0NTAyMjEwMEQwM0YzMTlDN0FEMDhBRDRGNTA3Q0FGRUYxNjZGRkU1RkU1NTc3OEI4Njg2NzYyNjQxRkY2RERDMDg0RTMyQTcwMjIwNjM1QTg5MzZGRTZDNjFBQUNFQ0JGQURFOTY2MzYyQkQxNUIwOEFFRjEwOTM5ODk2NDBGQUJBREMzNDE0MkU1MiJ9\",\"encodingMethod\":\"OCMF\", \"publicKey\":\"MzA1NjMwMTAwNjA3MkE4NjQ4Q0UzRDAyMDEwNjA1MkI4MTA0MDAwQTAzNDIwMDA0MEE4ODUyN0UyM0VEODcxMTE3NDkxQkQ0MzVEQTA0ODA0MUFBRjlCMzcxRjZBNUM0QzA0OERDRDU5OUQ5NjlDM0EwRUNCRjc3MzcwRjIzMjA4RTdDQTAzQkQzNTMwN0NCNDJGNTkwNEE5Qzc1QkI3RDgxQjQxQzA1MzQ2N0Y1NTg=\"}", "location": "Outlet", "context": "Transaction.End", "measurand": "Energy.Active.Import.Register", "unit": "Wh"}]}], "transactionId": 1745412560}]

5.3. Example *publicKey* Value Composition (Informative)

1. Public key printed on the certified meter (ASN.1 structure containing key algorithm and parameters as well as the actual key):

3056301006072a8648ce3d020106052b8104000a03420004460a02ba2766d9c44f023ecc0e4e58644a87add1aadd6317e5fe4dccdb29b163a01d8a6297c84bc530f86431e92f8d46ab37830247c05cbd92fac252929e7f61

2. Prefix with metadata:

oca:base16:asn1:3056301006072a8648ce3d020106052b8104000a03420004460a02ba2766d9c44f023ecc0e4e58644a87add1aadd6317e5fe4dccdb29b163a01d8a6297c84bc530f86431e92f8d46ab37830247c05cbd92fac252929e7f61

3. Base64 encode to the string to put in the *publicKey* field:

b2NhOmJhc2UxNjphc24xOjMwNTYzMDEwMDYwNzJhODY0OGNlM2QwMjAxMDYwNTJiODEwNDAwMGEwMzQyMDAwNDQ2MGEwMmJhMjc2NmQ5YzQ0ZjAyM2VjYzBlNGU1ODY0NGE4N2FkZDFhYWRkNjMxN2U1ZmU0ZGNjZGIyOWIxNjNhMDFkOGE2Mjk3Yzg0YmM1MzBmODY0MzFlOTJmOGQ0NmFiMzc4MzAyNDdjMDVjYmQ5MmZhYzI1MjkyOWU3ZjYx