eCHARGE
4DRIVERS

Deliverable 4.1

Plug & Charge
guidance document

www.echarge4drivers.eu

This project has received funding from the
European Union's Horizon 2020 research and innovation programme
under grant agreement No. 875131 (Innovation Action)

Work Package 4	Interoperable enhanced software services for users
Task 4.1	Plug & Charge guidelines and implementation
Authors	Mahmoud Draz, Tim Kleeberg, Carl Vahrenholz, Fabian Stichtenoth (HUBJECT)
Dissemination Level	Public
Status	Final
Due date	31/05/2021
Document Date	30/07/2021
Version Number	1.0

Quality Control

	Name	Organisation	Date
Editor	Mahmoud Draz	Hubject	09/07/2021
Peer review 1	Cedric De Cauwer, Omar Hegazy, Thomas Geury	VUB	22/07/2021
Peer review 2	Philipp Schumann	RB	27/07/2021
Authorized by (Technical Coordinator)	Evangelos Karfopoulos	ICCS	29/07/2021
Authorized by (Quality Manager)	Alessandro Rinaldi	POLIBA	28/07/2021
Submitted by (Project Coordinator)	Angelos Amditis	ICCS	30/07/2021

Legal Disclaimer

Document History

Version	Date	Editor	Revisions
0.1	25/05/2021	Mahmoud Draz (Hubject)	Frist draft
0.2	27/05/2021	Alessandro Rinaldi (Poliba) Evangelos Karfopoulos (ICCS)	Feedback on the first draft
0.3	07/07/2021	Mahmoud Draz (Hubject), Fabian Stichtenoth (Hubject)	Integrating the feedback from Poliba and ICCS. Adding two sections on ISO15118, elaborating on the onboarding process to the Hubject Ecosystem for the CPO, OEM, and EMP.
0.4	08/07/2021	Tim Kleeberg (Hubject), Carl Vahrenholz (Hubject)	Hubject onboarding guidelines
0.5	23/07/2021	Cedric De Cauwer (VUB), Omar Hegazy (VUB), Thomas Geury (VUB)	First peer review
0.6	27/07/2021	Schumann Philipp (Bosh)	Second peer review
0.7	27/07/2021	Mahmoud Draz (Hubject)	Document updated based on the comments provided by the peer-reviewers
1.0		Angelos Amditis (ICCS)	Final for submission

TABLE OF CONTENTS

List of figures

List of tables

List of abbreviations and acronyms

Abbreviation	Meaning
CA	Certificate authority
CCP	Contract certificate pool
CN	Common name
CPMS	Charge Point Management System
CPO	Charge point operator
CPS	Certificate provisioning service
CRL	Certificate revocation list
CSR	Certificate signing request
DHPublicKey	Diffie-Helman public key
DN	Distinguished name
EMAID	e-mobility account identifier
EVSE	Electric vehicle supply equipment
EST	Enrollment over Secure Transport
HSM	Hardware secure module
MO	Mobility operator
OCPP	Open Charge Point Protocol
OCSP	Online certificate status protocol
OEM	Original equipment manufacturer
OICP	Open intercharge Protocol
PCP	Provisioning certificate pool
PE	Private environment
PCID	Provisioning certificate identifier
PKI	Public key infrastructure
PnC	Plug&Charge
QA	Quality Assurance
RCP	Root certificate pool
SECC	Supply equipment communication controller
V2G	Vehicle to grid
VDE	Verband der Elektrotechnik Elektronik Informationstechnik e. V.
VIN	Vehicle identification number
WMI	World manufacturer identifier

EXECUTIVE SUMMARY

This document serves as the deliverable of Task 4.1 “*Plug & Charge Guidelines and Implementation*”.

The objective of Task 4.1 is to derive guidelines on how to establish an authenticated, confidential, and integral connection between actors within the electromobility value chain and to support task members in implementing the ISO 15118 Plug & Charge feature.

This deliverable aims to provide the guidelines for the implementation of the ISO15118PnC feature by different emobility stakeholders. It offers an introduction to the ISO15118 standards. The document also briefly introduces the Plug & Charge technology and how it works by explaining the main concepts required to successfully implement the Plug&Charge technology. Finally, the document addresses the requirements of the individual players in the electromobility value chain to enable their system with Plug&Charge.

1. INTRODUCTION

1.1 Project intro

eCharge4Drivers is an H2020 project running from June 2020 to May 2024 and deployed by a consortium of 32 partners. Charging an electric vehicle (EV) is still not as convenient as refueling a conventional vehicle, potentially posing a barrier to increase the market uptake of EVs. eCharge4Drivers works to substantially improve the EV charging experience within cities and for long trips. The project will develop and demonstrate user-friendly charging stations and innovative charging solutions as well as smart charging services for the users. By capturing users' perceptions and expectations on the various charging options and their mobility and parking habits, eCharge4Drivers will organise demonstrations in 10 areas across Europe, including metropolitan areas and Trans-European Transport Network (TEN-T) corridors. Charging stations in these areas will offer user-friendly and convenient functionalities for EV drivers of passenger and light vehicles and motorcycles, such as direct payment methods and bigger, user-friendly displays. Using the knowledge generated, the project will also propose an EV Charging Location Planning Tool, fostering the broad implementation of charging infrastructure in Europe.

1.2 Purpose of the deliverable

The objectives related to this deliverable have been achieved in full and as scheduled. In the following table, we list the three main goals of the WP and also an assessment of each goal.

Table 1: The goals achieved of the deliverable 4.1.

Role:	Who:	Status
Developing Plug & Charge guidance	Hubject	Achieved
Providing the educational Materials and training required to implement Plug&Charge.	Hubject	Achieved
Implement ISO15118, Plug&Charge Feature	All other members of the Task,	This is still an ongoing task to be achieved by the end of M24.

1.3 Intended audience

Deliverable D4.1 is public. The document refers to Charging Point Operators (CPOs), Mobility Operators (MO) and vehicle manufacturers analysing the requirements and the processes to be adopted towards implementing ISO15118PnC.

1.4 Structure of the deliverable and its relationship with other work packages/deliverables

This report is structured as follows. We start with a brief introduction to ISO15118 and Plug&Charge technology in Subsection 2.1. We explain some important concepts related to ISO 15118 in subsections 2.20, and 2.4. The relevant roles and actors for Plug&Charge are then presented in Subsection 0. In the end, we give a summary and draw conclusions.

2. PLUG&CHARGE GUIDELINES

2.1 The concept

Power grids are critical infrastructures that must be secured to prevent possible tampering and cyber-attacks. Part of grid security is the security of all its gateways. Electric vehicle (EV) charging infrastructure is considered to be one of the critical gateways into the grid. Manipulating the mass behavior of EVs can lead to severe consequences including major blackouts. Plug&Charge technology secures access to such gateways by providing automatic authentication and authorization based on cryptographic encryption**. Secure communication is defined here by three notions: data confidentiality, data integrity, and data authenticity**. **Data confidentiality** ensures that the messages exchanged cannot be viewed by unauthorized persons. **Data integrity** prevents unauthorized changes during data exchange. **Data authenticity** ensures that the intended entity is addressed. This level of security is achieved through the use of cryptographic encryption techniques and digital certificates. The details of these techniques are defined in the ISO 15118 standards and the supplementary application guide (VDE-AR-E 2802-100) (VDE, 2019). Further explanations of both ISO 15118 and the VDE guide can be found in Subsections 2.20.

Plug&Charge also offers an efficient and more user-friendly alternative to RFID cards or mobile apps, which have mostly been used for charging authorization. Providing a secure authorization mechanism is the minimum function that ISO15118 offers. However, with the ISO 15118 protocol, more information can be exchanged between the EV and the charging station than just that needed for authentication. The additional information could be user preferences and energy needs, payment methods, and more. This would allow operators to create an optimal charging plan that potentially reduces costs or maximizes the use of green resources.

As a technology, Plug&Charge is enabled by an ecosystem that consists of several elements and includes multiple market participants. The first element necessary for Plug&Charge is the ISO 15118 communication stack with at least the mandatory message pairs and security requirements. The role of the ISO 15118 protocol is to standardize a secure bidirectional data flow between the EV on one side and the charging point on the other side. The second element of the Plug&Charge ecosystem is the Public Key Infrastructure (PKI). The PKI system regulates and manages the process of handling digital certificates between the actors involved. Examples of these certificates are those installed in the vehicle and at the charging station. Other important elements for the functioning of Plug&Charge are the databases or the certificate pools and the application programming interface (APIs) needed to store, approve or sign and exchange or provoke certificates. In the following, we will discuss these three elements in more detail.

2.2 ISO 15118 protocol

ISO15118 is a client-server-based protocol designed to secure the communication between the Electric Vehicle Communication Controller (EVCC) or the vehicle from one side and the Supply Equipment Communication Controller (SECC) from the other side. The protocol consists of several requests and response messages, combined with some security mechanisms to exchange them. Some of these messages are mandatory to provide basic functions (like Plug&Charge) and some of them are optional to provide additional services. Security mechanisms are applied to ensure the confidentiality, integrity, and authenticity of the message flow in between. This is achieved by using techniques such as asymmetric cryptography, digital certificates, and Transport Layer Security (TLS) handshake. To learn more about these concepts, we refer to ISO 15118 document (15118-2:2014, ISO, 2014)

According to ISO 15118, as depicted in Figure 1, eight use cases are defined with the designations A to H. Use case A is for establishing the first physical connection and applying the 5% pulse width modulation (PWM) signal to the connector's power line communication. Use case B is concerned with establishing the communication session after exchanging the IP address and port number. Exchanging or updating digital certificates is use case C. Checking the validity of the certificate and authorizing the EV to load it is part of use case group D. Once the SECC confirms the existence of a valid contract for the EVCC and authorizes the EV to charge, the EV informs the SECC of its charging parameters, such as the amount of energy needed to fully charge the EV's battery. This step is performed in use case group E. Now we have three more main use case groups, charging control, value-added services, and end of charge. The charging control use case group is used to monitor and renegotiate the charging schedule with the vehicle based on the network charging state. The value-added services deal with the services that can be offered to the vehicle during the charging process. Examples of these services are smart charging and ancillary services for the power grid. And the ends of charging are simply part of the messages to signal the end of the charging session.

2.3 Public Key Infrastructure (PKI)

The public key infrastructure (PKI) is the system used to store, delete, distribute and revoke digital certificates. A key player in the PKI system is the Certificate Authority (CA).

The CA operates the highest trust anchor (V2G root), which is directly or indirectly responsible for signing the leaf certificate. The leaf certificate is a signed public key of a device, which can be an EV or a charging point. The Leaf certificate sits at the bottom of the hierarchy of the certificate chain, which is either installed in the EVCC or the SECC. Between the V2G root and the Leaf certificate, there are two levels of authority. These levels are the Sub-CA 1 and the Sub-CA 2, which are operated by the actors that do not operate their own PKI to sign their leaf certificates. Some actors such as Mobility Operators (MOs) and Original Equipment Manufacturers (OEMs) may run their own PKI system. In this case, they not only operate the sub-CAs but also the corresponding roots. Figure 2 shows the signing hierarchies as described in the ISO15118 document.

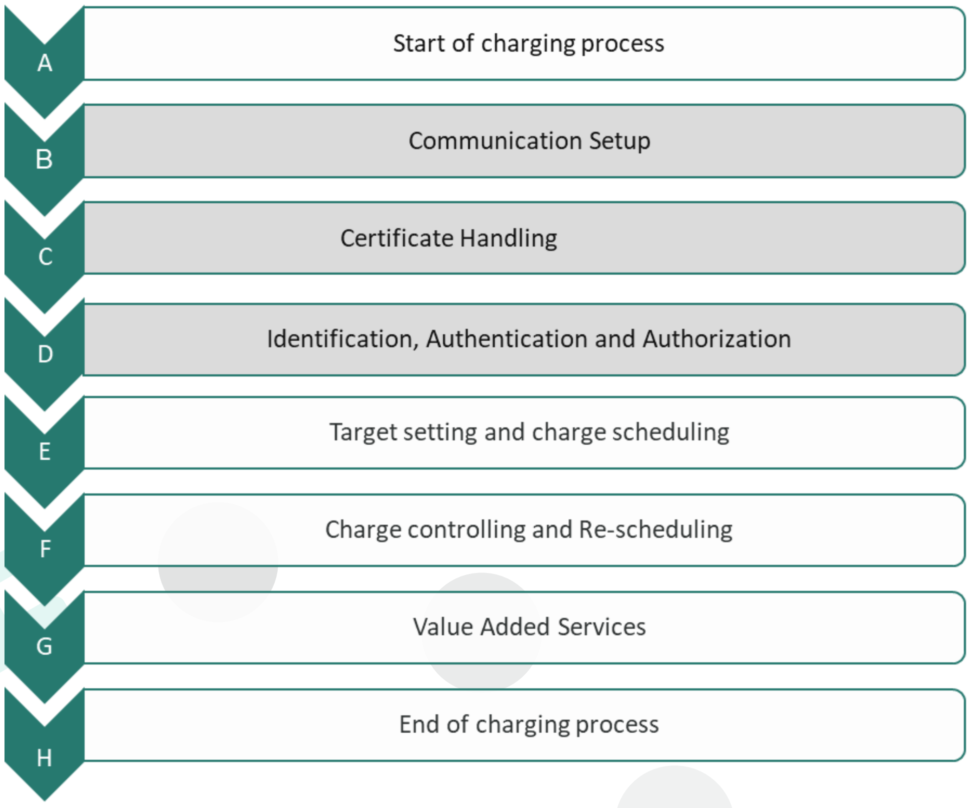

A
Start of charging process
B
Communication Setup
C
Certificate Handling
D
Identification, Authentication and Authorization
E
Target setting and charge scheduling
F
Charge controlling and Re-scheduling
G
Value Added Services
H
End of charging process

Figure 1: Use case groups defined in ISO 15118 standards

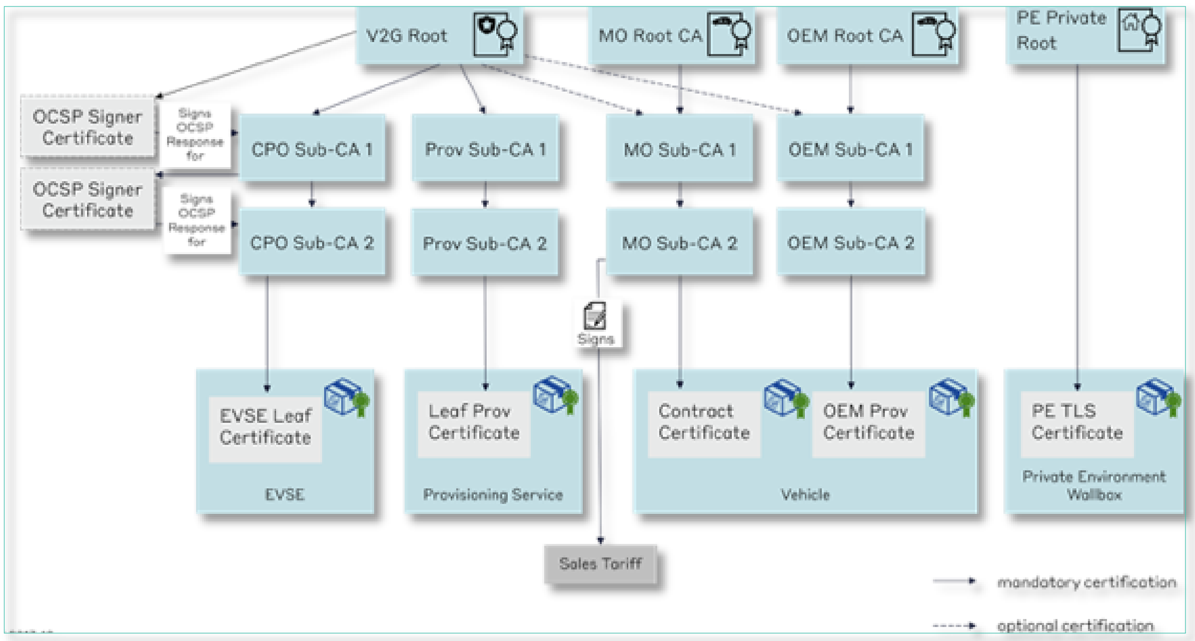


Figure 2: The hierarchy of the certificate singing according to the ISO 15118 document. Source: (15118-2:2014, ISO, 2014)

2.4 Certificate handling ecosystem

The ISO 18115 is, as already mentioned, the communication protocol that normalizes the communication between the EV and the charging station. ISO 15118 is essentially based on asymmetric cryptography and digital certificates. **The process by which these digital certificates are exchanged is not described in ISO 15118.** However, to make the Plug&Charge technology work, **the guide provided by the Association of German Electrical Engineers (VDE) has added a complementarity document to the protocol.** This document defines the certificate handling mechanism and its requirements. The certificate handling ecosystem provides an automated process for securing the exchange and processing of digital certificates between the participating actors. There are six main actors actively collaborate to make Plug&Charge possible, namely the car manufacturers (called also OEM), the charging point operator (CPO), the mobility operators (MO), the driver or EV, the certificate authority (CA), and the certificate handling ecosystem operator.

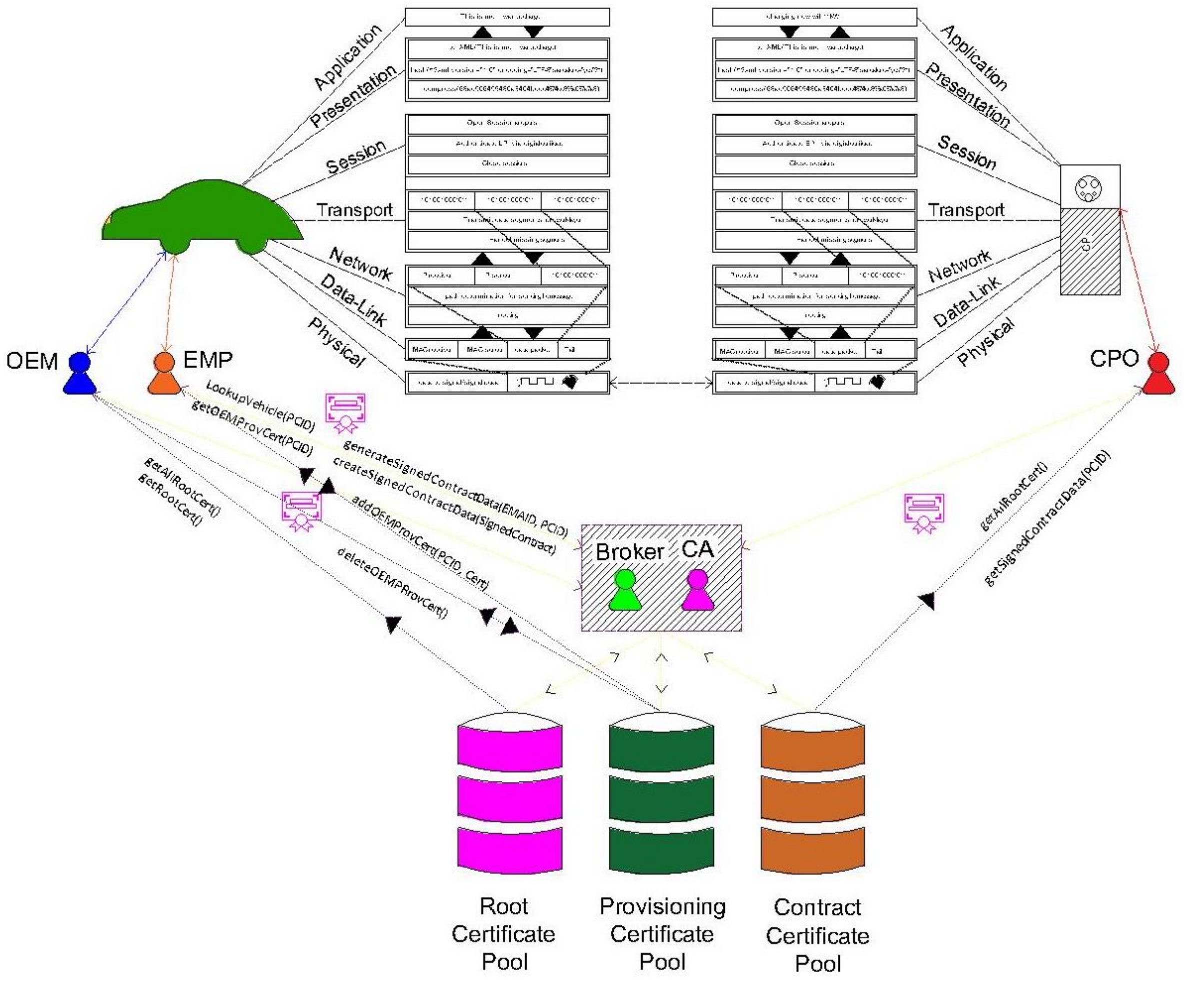


Figure 3: The OSI model layers for ISO 15118 and the Plug&Charge ecosystem.

The certificate handling ecosystem defines three main contract pools for storing the digital certificates and the number of application programming interfaces (APIs) for processing them. **The first pool stores the vehicle-to-grid root certificates (V2G Root CA)**. The V2G Root CA acts as the highest security anchor, signing all issued certificates directly or via a sub-certificate (sub-CA). **The second pool is the Contract Certificate Pool (CCP).** The CCP stores the digital contracts that are usually concluded between the user/car and the mobility operator (MO). **The third pool is the Provisioning Certificate Pool (PCP)**. In the CPC, the identifiers (PCIDs) of all EVs are stored. Usually, it is the OEM who creates the PCIDs as part of the provisioning certificate for the new vehicles and pushes them into the PCP pool as part of the manufacturing process.

In three main steps, the Plug&Charge VDE Guide describes the actions and the activities of the individual stakeholders. The first action is the responsibility of the OEM. An OEM should consider three main activities that are part of the EV production process. In the following, we will go into more details about these steps.

2.4.1 During the manufacturing process

An OEM must first create a provisioning certificate for each vehicle produced with a unique provisioning certificate identifier (PCID). This certificate PCID comes as part of the certificate to be installed in the EV. The PCID will also be shared with the PCP pool of the Plug & Charge ecosystem. The OEM should also install at least one V2G root certificate in the vehicle that can be retrieved from the ecosystem's

RCP for certificate management. The PCID is later used by the MO to enter into a contract with the driver. The V2G root installed in the vehicle along with the PCID is required for the authentication process at the beginning of each charging session. OEMs can install and update the digital certificates as needed either via a physical conductive connection or . Another important element that must be issued and installed in the vehicle is the vehicle's private key. This key is used to encrypt messages between the charging station and the vehicle.

2.4.2 Mobility Service Contract Concluding

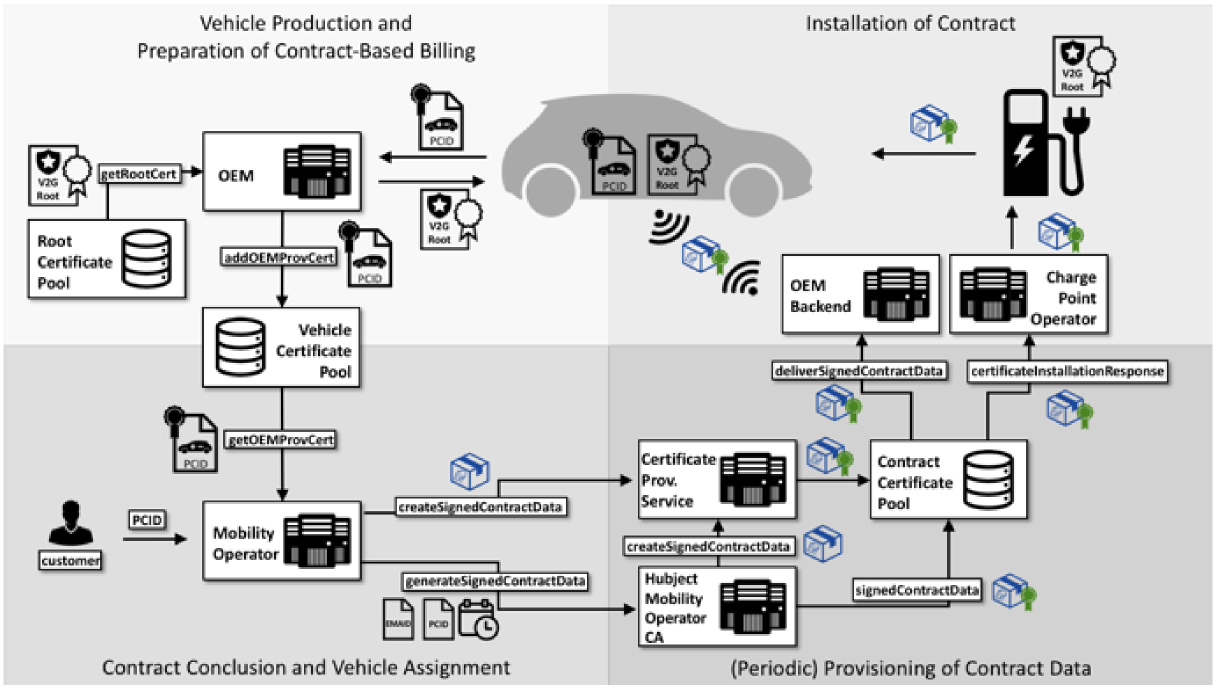
For the user to receive a charging service, he/she must normally enter into a contract with an MO. In the current system, this contract is traditionally made with the user or the owner of the vehicle, without limiting the use of the contract to a specific vehicle. In other words, the vehicle itself as a device is not part of the contract process with the MO. In the Plug&Charge ecosystem, the story is a little different. The user enters into a contract for their vehicle. This contract includes the PCID of the vehicle and the ID of the mobility operator account, the e-mobility account ID (eMAID).

2.4.3 Contract provisioning

After the contract is concluded, it is signed either by the CA root of the MO or by the CA root of another PKI provider. This is the case if the MO does not operate its own PKI. Once the contract has been concluded and signed, it should be pushed into the CCP pool of the certificate-processing ecosystem. The contract can then be installed, updated, or exchanged via ISO 15118 message pairs and the corresponding APIs of the certificate processing ecosystem.

2.4.4 Contract installation

An EV needs a valid digital contract certificate installed in the vehicle to use Plug&Charge, automate the billing service, and use other value-added services. This type of certificate is not installed during the production of the vehicle, since the user has not yet decided from whom to buy the charging service. For this reason, the contract certificate is installed postproduction and after selling the vehicle to the end-user. There are two ways to install this contract in the vehicle. The first way is to assume that the OEMs give the option to install it through the CPO backend by calling the certificate handling ecosystem. The other way is that the OEMs choose to have the sole right to install any certificates in the vehicle. In this case, the MO must pass the completed contracts to the appropriate OEM for installation in the vehicle. This is can be done via the Plug&Charge APIs and the certificate pools. The OEMs then can communicate directly with the vehicle, either through a physical connection or wireless via using telematic technologies.



Vehicle Production and Preparation of Contract-Based Billing
Installation of Contract
V2G Root
getRootCert
OEM
PCID
Root Certificate Pool
addOEMProvCert
Vehicle Certificate Pool
getOEMProvCert
OEM Backend
Charge Point Operator
deliverSignedContractData
certificateInstallationResponse
customer
Mobility Operator
createSignedContractData
Certificate Prov. Service
Contract Certificate Pool
generateSignedContractData
Hubject Mobility Operator CA
signedContractData
EMAID
Contract Conclusion and Vehicle Assignment
(Periodic) Provisioning of Contract Data

Figure 4: Plug&Charge ecosystem and certificate handling process.

As an example, Figure *4* shows Hubejct's ecosystem for handling certificates, or as we call it "Plug&Charge Eco-System". The figure shows all possible communications to the vehicle required to install or update the certificates.

3. PLUG&CHARGE REQUIREMENTS

In this section, we will present the requirement of each stakeholder in the ecosystem to enable Plug&Charge. These requirements will define mainly minimum hardware as well as the software specifications that are needed to implement the ISO15118 Plug&Charge feature. We also will describe the integration process which can be used by each player during the implementation. We will also provide a checklist by which a stakeholder can follow during the integration process of the Plug&Charge APIs.

3.1 Plug&Charge implementation – CPO

This subsection provides insights into the requirements for the CPO to implement Plug&Charge. Prerequisites. In the following, we are listing the main four requirements from the CPO to activate Plug&Charge functionality in their system.

Implementation of OCPP

For the ISO15118 Plug&Charge feature to work, communication with the CPO backend is required. This communication, as described in Figure *4*, is also necessary to communicate with the PKI and Plug&Charge ecosystem. The protocol currently used for communication between the CPO backend and the charging points is the open charge point protocol (OCCP) (Open Charge Alliance, 2021). The OCCP versions 1.6 and below do not have the necessary messages that support ISO 15118. The e-mobility community has adapted the 1.6-version and added the required messages for ISO 15118 Plug&Charge. For this reason, implementing the adapted version of OCCP 1.6 or higher is a necessary prerequisite to enable Plug&Charge technology for the CPO role. The required changes to the OCCP 1.6 to support ISO15118 are explained in (Open Charge Alliance, 2020).

Security measures

The ISO 15118 and Plug&Charge are all about secure and user-friendly communication between the power grid and electric vehicles. From the CPO's side, there are two security measures to be met. The first is hardware and the second is software. The hardware part consists of installing hardware security modules (HSM) in each charging point to secure the storage and use of the private key and digital certificates. The software security part on the other hand is responsible for securing the communication and data flow from the EV to the charging point and from the charging point to the CPO backend. The software security measures can be achieved by implementing the TLS handshake for both communications. The TLS handshake is well explained in (Oppliger, 2016) . We also refer to (Fraunhofer SIT, 2019) for more information on the technicalities of the HSM.

ISO15118 protocol

The implementation of at least the mandatory message pairs is essential for the process of implementing the Plug&Charge function. The implementation of the ISO communication stack could consist of the two-class sets. These sets can be divided into four categories based on the actor involved. The first category addresses the charging point side, which deals with the transport layer, the session handler, the SECC, the stream, the meter, and the interaction with the CPO backend and the CA authority. Similarly, on the vehicle side, the ISO 15118 protocol also deals with a transport layer, session handler,

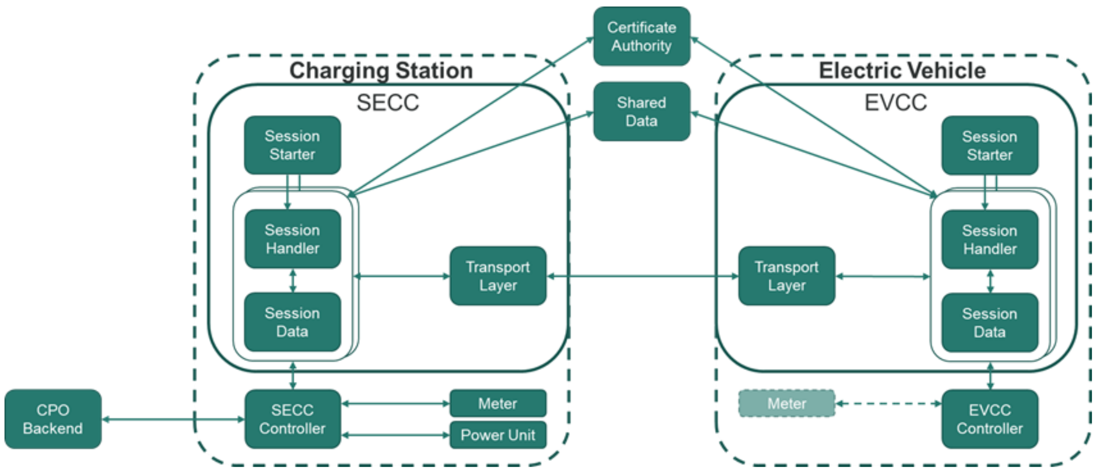


Figure 5: An overview of the ISO 15118 implementation as described in the open source V2G Rise.

EVCC, and optional communication with the OEM backend to handle the relevant certificates. This is all depicted in Figure *5*. Here we will go through a summary of the main ISO 15118 -2 messages and the order in which they are handled. A detailed description of these message pairs can be found in (15118-2:2014, ISO, 2014).

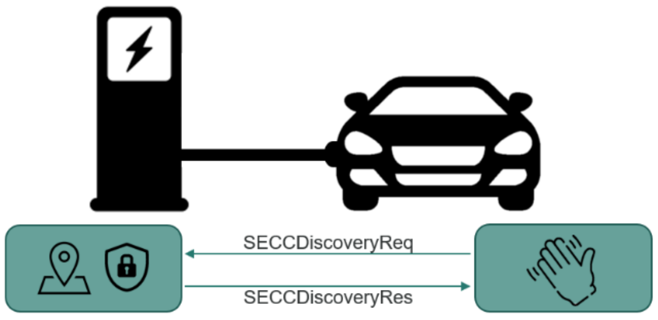


Figure 6: SECCDiscoveryReq/Res

When an electric vehicle wants to charge at a charging point via ISO 15118, a set of defined messages must be exchanged between the vehicle and the charging point to complete the authorization process. Some messages also require communication with other entities such as the CPO backend of the charging point, a Certificate Authority (CA), and in some cases, to the OEM backend. The exact sequence of exchanged messages thereby can vary slightly as some messages are optional and need to be triggered by certain events. The exchanged messages also sometimes depend on whether the charging point is AC or DC, and the payments method is a contract or ad-hoc. We will talk more about these kinds of changes in the sequence as we proceed with the charging process. Anyway, the overall structure of the charging process stays the same in the general case. In the following, the typical message flow for a DC charging session will be described.

Typically, the user wants to start the charging session by connecting the vehicle to a charging point via a charging cord. Once the cord is connected, the charging point state changes from A to C[1]. Now the

[1]State A (+12V): The EV is not connected to the charging station.
State B (+9V): The EV is connected to the charging station but not yet ready for charging.
State C (+6V): The EV is connected to the charging station and ready for charging (station can still open and close charging circuit)
State D (+3V): The EV is connected to the charging station and ready for charging (ventilation requested) (station can still open and close charging circuit)

first ISO15118 message pair is ready to be exchanged. As the ISO15118 is a client-server protocol, the client (in our case, the EV) initiates the communication always. Based on that, the EV will start by sending its first message (i.e., *SECCDiscoveryReq*). The charging point will reply to this message with a *SECCDiscoveryRes*. The aim of this message pair is to provide the EVCC with the IP addresses and the port number which will be used later to establish the communication session. The message also includes information regarding the security level the EVCC and SECC agree upon. The level of security here means whether the communication is secured by a TLS or just a TCP. For Plug&Chare, TLS must be chosen by both, the EVCC and SECC. In short, the SECC will reply with a *SECCDiscoveryRes* carrying three items: the IP address of the SECC; the port number; and an indication of an agreement on using a TLS.

In the next step, the EVCC and SECC need to agree on the version of the ISO15118 that will be used. This is achieved using the ***supportedAppProtocolReq/Res*** message pair. As usual, the EV initializes the communications. In this case, the EVCC will send its supported versions of the ISO 15118. Then the SECC will decide on one that both the EVCC and SECC support or have in common. The SECC should in this case take the latest version that they both can support. This kind of information is sent again to the EVCC carried on the *supportedAppProtocolRes* message.

To proceed further with setting up the charging session, another message pair is used. This message pair is the **SessionSetupReg/Res**. This pair is used to create a session ID, a unique identifier for the charging session. Using the ID, a previous paused session can also be resumed. Also, can be used to recall information regarding a specific session. To create an ID for the session, the *SessionSetupReq* is sent to the SECC with some information related to the Mac address of the EVCC as its ID. The SECC creates a session ID and sends it over a *SessionSetupRes* message which also includes additional information such as the ID of the SECC and the time stamp.

Until now, we can say that the EV and charging point is connected, EV knows the IP address and the port number of the SECC, and they agreed on 1) the security level – TLS –, 2) the ISO15118 version, 3) an ID for the session. Maybe also the charging point notified its charging point operator (CPO backend) over an OCPP message that an EV is connected. Until now the communication is happening on the unsecured UDP protocol. In other words, the TLS handshake has not happened yet. But before the EVCC and the SECC start to encrypt their communication, the EV wants to be informed about the services that the charging point can offer. These services can be, of course, next to the energy, internet connection, software updates, smart charging, grid ancillary. The EV uses the *ServiceDiscoveryReq/Res* message pair to get informed about the available services offered by the charging point where it wants to charge. A side note here, this message pair and the upcoming ones will contain the session ID created by the SECC in the previous message. Let's continue with the message pair regarding the services offered by the charging point. Now, the SECC must reply to the request sent by the EVCC over the *ServiceDiscoveryRes* a mandatory service – ChargeService—and a list – ServiceList— of, so far, optional services.

Each service is expected to have included some important information such as ID, name, category, and whether it is free or not. The EV can get more detailed information about a certain service by using the option message pair, *ServiceDetailRes/Res*. The EVCC will send just the ID of a certain service over the *ServiceDetialReq* and the SECC will respond with the details of service over a *ServiceDetailRes* message. As we mentioned, the *ServiceDetailReq/Res* pair will not be used unless the EVCC triggers it or in other words, the EV is interested in using added value services. It is worth noting here that the added value services market is expected to grow as the acceptance and use of ISO15118 roles out.

State E (0V): Problem with the grid or no grid connection
State F (-12V): Charging station not available.

Next, the optional message **ServiceDetailReq** can be sent by the EVCC if it wants to use one of the offered services in addition to the mandatory charge service. Within the ServiceDetailRes, the SECC also specifies the available payment services so that the EVCC can select the service it wants to use by sending the next message pair, the ***PaymentServiceSelectionReq/Res***. If Plug & Charge should be used, the payment service needs to be "contract". Also, if the EVCC has no valid contract certificate installed, it can inform the SECC via the *PaymentServiceSelectionReq* that it needs to install a certificate. If so, the optional message pair ***CertificateInstallationReq/Res*** is exchanged. Within this process, there's also communication with the CPO backend and the certificate authority so that in the end all data needed to get a valid contract certificate can be provided to the EVCC by the *CertificateInstallationRes*.

Within the next message pair, the *PaymentDetailsReq/Res*, the contract certificate is transferred so that it can be verified by the CPO Backend and the contracting authority. In response, the EVCC gets a GenChallenge which it sends back, using the *AuthorizationReq* to prove its authenticity. The SECC checks that the GenChallenge is the same it sent before and that the signature is valid and if so, sends back a confirmation via AuthorizationRes. If the authentication process takes too long, the Message can be sent again so that there is a loop until authentication was finished. The next message pair, the ***AuthorizationReq/Res*** makes sure that the CPO backend has the EMAID and can use it as needed. Now it is time that the EVCC informs the SECC about the technical charging parameters such as maximum voltage and maximum current. The SECC processes the information and while doing so, repeatedly sends ChargeParameterDiscoveryRes with the processing parameter set to ongoing. After the processing is done, the SECC responds with one final *ChargeParameterDiscoveryRes* in which charging parameters and restrictions from the charging station side are included. With this, the EV also changes the CP State to State C. Finally, with the *CableCheckReq/Res*, the vehicle triggers a test of the high-voltage system isolation.

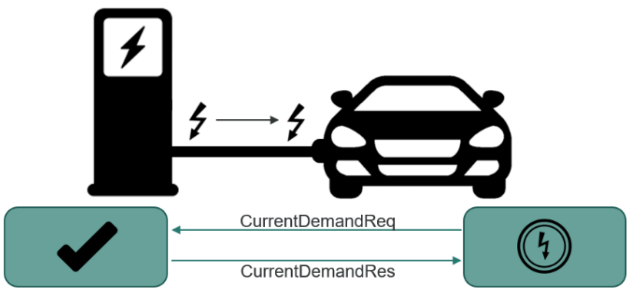


Figure 7: CurrentDemandReq/Res

After this check, the EVCC sends a ***PreChargeReq*** which contains the requested DC and voltage. The power unit then adapts the output voltage and informs the EV via PreChargeRes. This can also include a loop. Next, the charging is started by the ***PowerDeliveryReq/Res*** message pair which includes the charging profile according to the previous exchanged limitations. Within this message pair, the CPO backend will also be informed about the start of the charging session and meter values before the charging session will be obtained. During the charging process, there is a loop of exchanged ***CurrentDemandReq/Res*** message by which, the EV informs the charging station about the instantaneous current requested. The response message triggers the EVCC to repeatedly send updated CurrentDemandReq messages. If the SECC also wants the EVCC to sign the meter info, it can indicate that by setting the respective parameter of the CurrentDemandRes to True. If so, the ***MeteringReceiptReq/Res*** message pair is used to exchange the signed metering information.

After this, the charging process is again continued by the CurrentDemandReq/Res loop as described before. Thereby, the EVCC or the SECC can use the respective message to request a renegotiation. This can be necessary if for example parameters within the electricity grid change. If that is the case, the CP State is changed back to B and new *ChargeParameterDiscoveryReq/Res*, *CableCheckReq/Res*, *PreChargeReq/Res* and *PowerDeliveryReq/Res* message pairs must be resent until the CP State can be changed back to C and the charging can be continued by again sending CurrentDemandReq/Res messages.

If the EVCC wants to stop the charging process, one final *PowerDeliveryReq* message is sent with the *ChargeProgress* parameter indicating that a stop is wanted. The SECC informs the CPO Backend, obtains the final meter data, and responds with the final *PowerDeliveryRes* message which also marks the point when the CP State changes back to status B. The EVCC now can also send the optional *WeldingDetectionReq* which triggers a test to determine if one of the EV's contactors has welded. After the test, the SECC responds with the *WeldingDetectionRes* which includes the results of the test. After that, the last message pair is exchanged, the *SessionStopReq/res* pair which completely ends the charging session. The user can now unplug the vehicle, the CP Status changes back to A, the CPO backend gets notified and the charging station is available again.

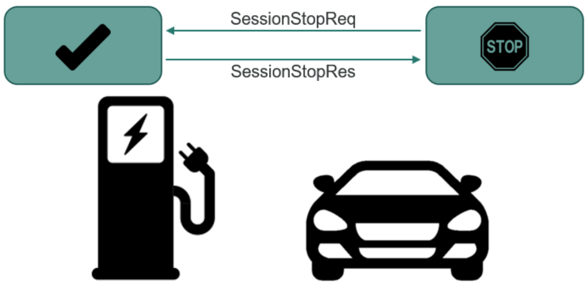


Figure 8: SessionStopReq/Res

Implementation of Plug&charge APIs

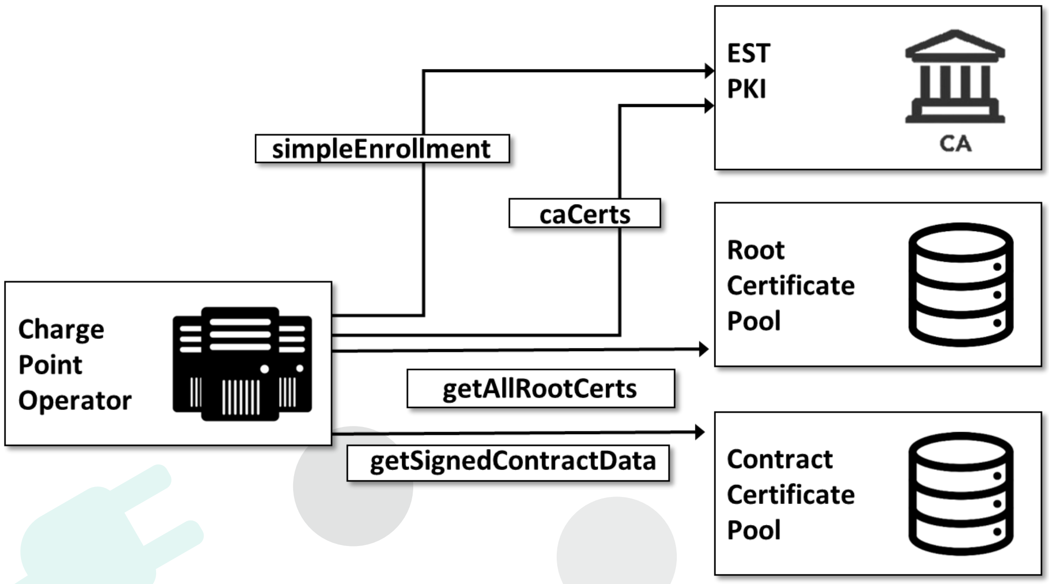


Figure 9: The interfaces for the CPO role with the relevant certificate pools.

The CPO must communicate with the KPI or Plug&Charge ecosystem during the authentication process to request the installation or update of the certificate. This type of communication is done through a set of APIs. **A CPO that wants to integrate Plug&Charge functionality must integrate these APIs into its system**. Figure *9* visualizes the interaction between the CPO backend and the Plug&Charge ecosystem.

3.2 Plug&Charge implementation – OEM

In the following, we provide insights into the requirements for the OEM to implement Plug&Charge. Prerequisites. In the following, we are listing the main requirements from the OEM to enable their vehicle with Plug&Charge based on ISO 15118.

Security measures

Similar to the CPO, OEMs have two security measures that need to be met by the OEM side. The first is the hardware and the second is the software. The hardware part consists of installing the hardware security module (HSM) in the vehicle to secure the storage and use of the private key and digital certificates. The software security part on the other hand is responsible for securing the communication and data flow from the EV to the charging point and from the charging point to the CPO backend. The software security measures can be achieved by implementing the TLS handshake for both communications.

The ISO 15118 protocol is an essential requirement for the OEM as well. In the OEM role, we will be referring to this section to avoid repetition and just focus on the specific requirements that are meant to be special for the OEM role.

Implementation of Plug&charge APIs

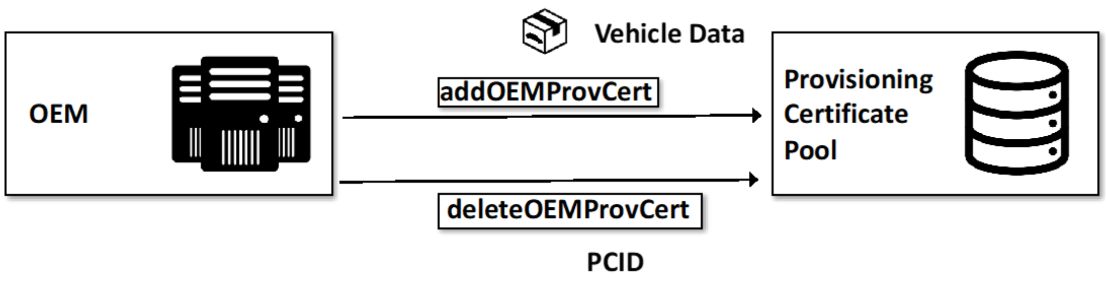


Figure 10: The interfaces for the OEM role with the relevant certificate pool.

OEMs only need an interface to one certificate pool, the PCP. This is essentially used to store or provoke provisioning certificates of the vehicles. This communication is made possible by two interfaces as shown in Figure 10. The first is for adding a provisioning certificate to the PCP pool and the second is to provoke an existing certificate.

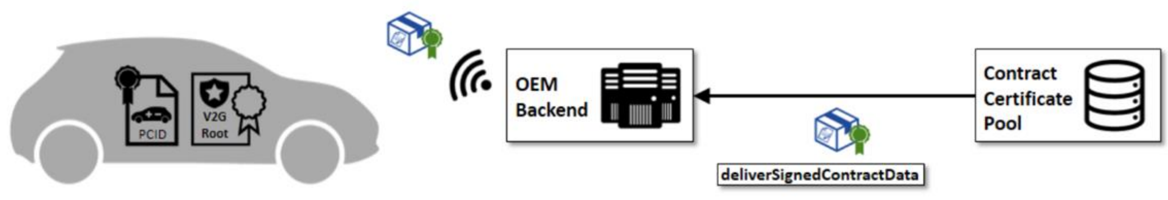


Figure 11: Possible ways for the OEMs to communicate with their vehicle fleets.

OEMs have the option to communicate with their vehicles directly or via the plug&charge ecosystem. Direct communication, as mentioned before can be done with a physical connection, which requires a lot of effort to get access to the user's vehicles. Another way to do that is to have to manage the certificates of the vehicles using telematics (see Figure 11). The latter can be more convenient for the user and more efficient for the OEM.

3.3 Plug&Charge implementation – MO

Mobility providers are not involved in the manufacture of the vehicle or the charging station. They also do not operate the charging infrastructures. For these reasons, they are not required to implement OCCP, ISO15118, or other protocols. This is true if they have not chosen to operate their own PKI. An MO can run its own KPI, in which case it must implement the PKI and the composing protocols. Otherwise, MOs can implement the Plug&Charge in very simple steps. In the case of MO, only two conditions are needed. We summarize them in the sections below.

Security measures for MO

As with CPOs, MOs also need to undertake some security measures. This time, the measures are not related to the vehicle or the charging point. The focus for this kind of security is only the communication with the Plug&Charge ecosystem pools and the PKI. This is achieved by implementing a TLS handshake for the way of communication with the ecosystem.

Implementation of Plug&charge APIs for MO

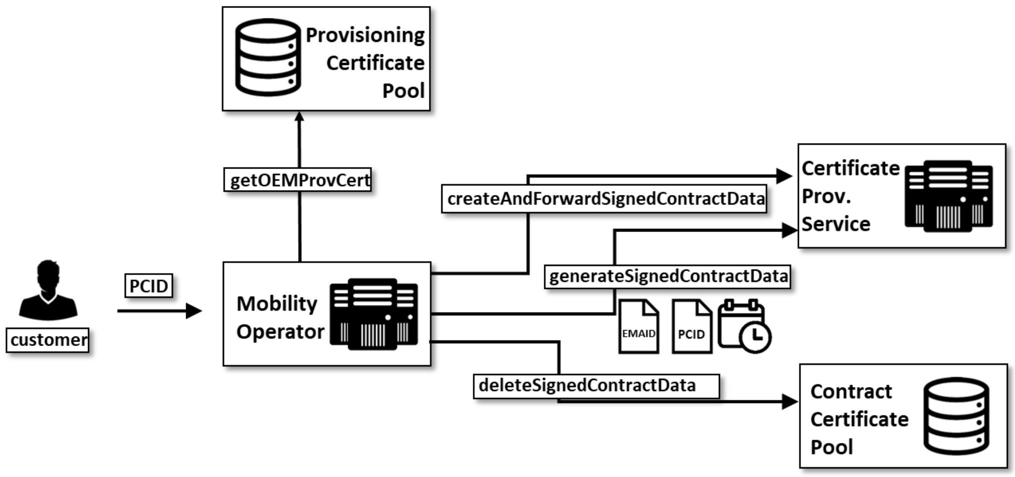

Provisioning Certificate Pool
getOEMProvCert
createAndForwardSignedContractData
Certificate Prov. Service
PCID
customer
Mobility Operator
generateSignedContractData
EMAID
PCID
deleteSignedContractData
Contract Certificate Pool

Figure 12: The interfaces for the MO role with the relevant certificate pools.

Mobility Operators need to communicate with the KPI or the Plug&Charge ecosystem during the user contracting process to request a provisioning certificate for a specific vehicle or to store or manage contract certificates in the Plug&Charge ecosystem. This type of communication is handled through a set of APIs. A MO that wants to integrate Plug&Charge functionality must integrate these APIs into its system. Figure 12 visualizes the interaction between the MO backend and the Plug&Charge ecosystem.

3.4 Hubject's ISO 15118 V2G PKI

This section is not intended to describe the Plug&Charge ecosystem from a technical perspective which was detailed above, but to provide practical guidance for onboarding as a CPO. In addition to this, the interface description gives a detailed technical description of the system.

This document serves as a checklist and a guidance Document for CPOs to be onboarded to the Hubject ISO 15118 Plug&Charge Ecosystem as well as the V2G Public Key Infrastructure. These systems were created and maintained by Hubject following ISO 15118:2014 and the VDE Application Guide VDE-ARE- 2802-100-1. Access to these systems is needed to enable Contract-Certificate Installation and successful authentication via Plug&Charge by the EVSEs and the Charge Point Management System of the CPO.

The Hubject Plug&Charge Ecosystem enables EV OEMs, Mobility Operators, and Charge Point Operators to use the ISO 15118 standard for the needed Plug&Charge functionalities. The Ecosystem provides a V2G root CA for the managing of the ISO 15118 certificates, pools for publishing, and signing service for trusted exchange of certificates. The root certificate of the Hubject V2G Root CA is the trust anchor for all participants of ISO 15118, which is published in the Root Certificate Pool of the Plug&Charge Hubject Ecosystem.

The interfaces and technical implementation of the Ecosystem are fully compatible with ISO 15118-2:2014 and the VDE Application Guide VDE-AR-E 2802-100-1 ("VDE Anwendungsregel"), to make the platform interoperable for all relying parties. In the implementation, the international best practices are considered for the security of communication and protection of data.

Onboarding process

The phases of onboarding are not fixed and depend on the respective speed of the partner's implementation. The following steps are intended to show how and in which order Hubject can provide support. OAuth (Open Authorization) is an open protocol that provides a standardized, secure API authorization. In the Hubject Plug&Charge system, client authorization is operated with OAuth 2.0. For more information about the authentication, please see the authentication section in the interface description.

General procedures

Here we give the client a step-by-step process for the entire onboarding process. It shows what the partner should do in a nutshell to complete the Plug&Charge integration These steps are summarized below. Also, *Figure 13* gives an illustration of the above-mentioned steps.

1) Signing a Non-Disclosure Agreement (NDA)
2) This step is important to protect confidential information, documents, and any other digital files delivered by Hubject to the partner.
3) Interface Description Handover

This gives detailed technical information about the Hubject's PKI ecosystem. This can be used by the partner during the implementation and also after going to prod mode as a reference.

4) Certificate Policy Handover
5) Sending the QA Credentials

The QA credentials are transferred in the form of a Postman Collection. The Postman Collection contains the URLs of all relevant APIs and the OAuth access data. Handover of the PnC QA Account inside the Postman Collection

6) Signing user Agreements
7) Reprod Testing on QA

See 4.1 for all Testcases and add into the List of 4.1 the following

8) Signing an EVSE Leaf Cert by creating a CSR and using simple enrollment of the Hubject EST Interface (for the CPO)
9) Obtaining the corresponding Sub-Certificate Chain via caCerts
10) Obtaining all Root certificates via getAllRootCerts for trust purposes
11) Getting a Certificate Installation request via getSignedContractData
12) IT Security audit

The audit serves as a mutual trust on certain security standards in the Plug&Charge ecosystem. Questions are asked based on the requirements and other contents of the Hubject Certificate Policy. The partner must answer the questions by providing documents that described the security process in detail. Often, several iterations are necessary for clarification.

13) Change of Rights to Prod

14) Basic Prod Testing (optional)

    a) Testing of connection

    b) End-to-End Testing with an EV simulator

➢ Go-live on Prod

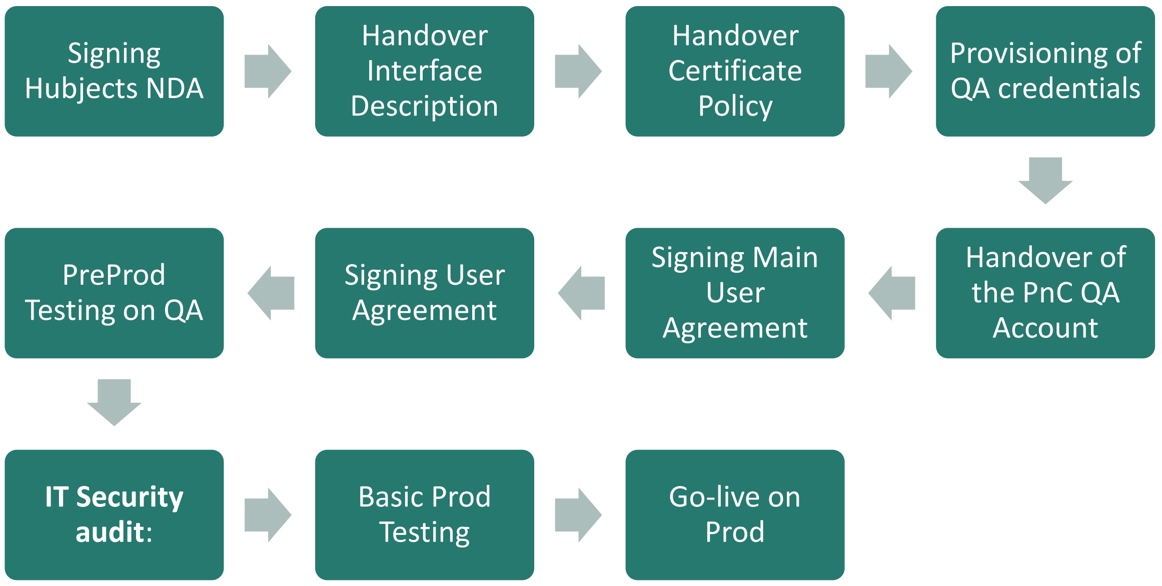

Signing Hubjects NDA
Handover Interface Description
Handover Certificate Policy
Provisioning of QA credentials
Handover of the PnC QA Account
Signing Main User Agreement
Signing User Agreement
PreProd Testing on QA
IT Security audit:
Basic Prod Testing
Go-live on Prod

Figure 13: Process for the Plug&Charge process.

3.4.1 Processes relevant to the CPO role

Preconditions for the CPO role

The preconditions do not have to be met before the start of QA testing but can be implemented during the ongoing project. Hubject can provide consulting support here. The preconditions below are important to consider and review within your organization before starting a Plug&Charge implementation project with Hubject. The below statements are primarily indicative of the most crucial processes related to certificate handling with the EV and the connected CPO backend (CPMS), based on the international norm ISO 15118, the VDE Application Guide and OCPP.

Manage the exchange of ISO 15118 certificates between CPMS and EVSE

One of the following options need to be fulfilled:

- CPMS Operates OCPP 1.6 with Plug&Charge as described by OCA
- Data transfer mechanism used for Messages from OCPP 2.x
- CPMS Operates OCPP 2.0.1
- CPMS must be able to install ISO 15118 certificates (such as the EVSE Leaf Certificate and V2G-/ or MO-Roots) within charging points that are connected to the CPMS, respectively Charging-Network.

Store EVSE Leaf certificate securely in the EVSE/SECC

Automatic and (semi-)manual creation of a CSR

- See OCPP TriggerMessage.req
- EVSE and or CPMS may be responsible for the recreation of the Leaf Certificate

Secure Storage of the Private Key in the SECC e.g.:

- Discrete HSM

The Security Module may be a discrete HSM, i.e., a separate chip dedicated to the security functionality with an external communication interface used by the EVCC and SECC to communicate with the Security Module. An example can be Thales chop Keystore

- Integrated Implementation

The Security Module may be implemented with specialized hardware integrated into the application processor used by the EVCC and SECC, including an internal communication interface.
An example of that is the nChipher HSM

- Firmware Implementation

The Security Module may be implemented by software running in a Trusted Environment offered by the application processor used by the EVCC and SECC, including a secure Application Programming Interface (API), which offers a controlled communication across the boundary of the trusted Environment. Again, Java Keystore can be used for that.

The signing of a CSR for an EVSE Leaf Certificate from a CPO Sub2 CA

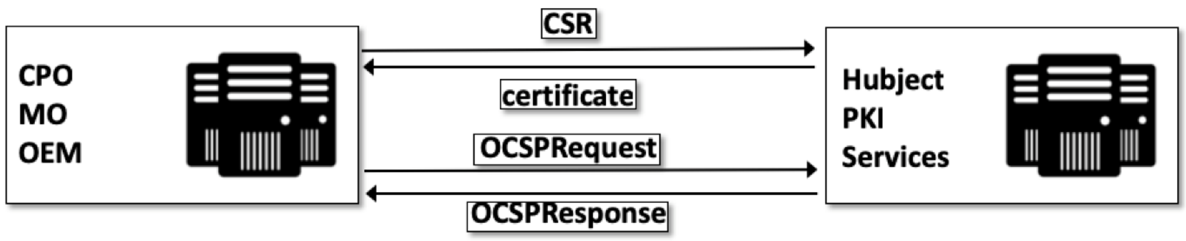
This process describes the signing of a CPO CSR with the Hubject CPO Sub 2 CA, which is performed (automatically) by the CPMS of the CPO.

- ➢ Creation of the public/private key pair
- ➢ Creation of the CSR
- ➢ Sending the CSR to Hubjects EST Interface using simple enrollment (RFC 7030)
- ➢ Obtain the signed public part of the Certificate
- ➢ Retrieve the chain certificate via caCerts (RFC 7030)
- ➢ Sending the Certificate and the belonging Chain to the EVSE/SECC

Obtaining an EVSE Leaf Certificate and its corresponding Sub-Certificate Chain

First, before going onto the process that describes how the CPMS can get access to and install the leaf and chain certificates, we need to make sure that the CPMS is capable of installing ISO 15118 certificates (such as the EVSE Leaf Certificate and V2G-/ or MO-Roots) in charging points. In another word, this process is only possible if the CPMS and the EVSE are ISO15118 compatible. The concrete requirements from the CPMS are: 1) CPMS should respond to a request from the charging point to update ISO 15118 certificates; 2) CPMS should be able to forward a *CertificateInstallationReq* from the charging point to the Contract Certificate Pool (CCP) of Hubjects Ecosystem using the *getSignedContractData*, and 3) CPMS also need to be able to recognize whether a charging point connected to the charging network is ISO 15118 compatible or not. If all these requests are fulfilled, then, all processes related to the leaf certificates are ready to be carried out.

Let's illustrate the singing process using the EVSE Leaf certificate. The process starts with the charging point creates a Certificate Signing Request (CSR). The CSR is sent to the CPMS over an OCPP message (again, version 1.6+ or higher). The CPMS then forwards the CSR to Hubject PKI services via Enrollment over Secure Transport (EST) as defined in RFC 7030. The Hubject PKI sings the CSR and sends it back to the CPMS. Untill now, the singed CSR is sorted in the CPMS.



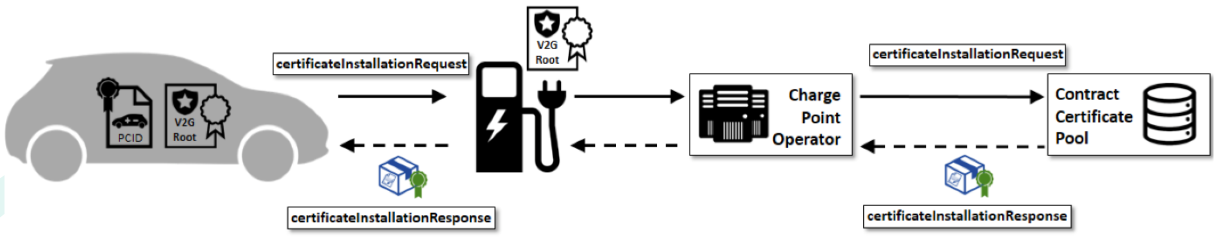
In the next step, the CPMS requests the Certificate Chain (caCerts) of the EVSE Leaf Certificate via an EST. This also goes to Hubjec KPI with a similar process to the process used for signing the CSR. The Hubject PKI sends back the caCers to the CPMS. The CPMS then stores the caCers together with the corresponding EVSE Leaf Certificate. After that, the CPMS installs both, the EVSE Leaf Certificate and its chain in the charging points.

Obtaining all Root certificates

As we explained before, the ISO 15118 -2 is designed to function with multiple certificate authorities with multiple CA roots. The Hubject PKI allows for storing several CARroots in its Rool Certificate Pool (RCP). All certificates stored in this RCP pool can be accessed via the *getAllRootCerts* API. This API is used to get all available Roots from the RCP to install all MO Roots in the EVSE. As a reminder here, the RCP is used for communication between the Root Certificate Pool and the various Certificate Authorities of ISO 15118 participants (V2G, OEM, MO). The RCP also provides root certificates to their actors that they need the root certificates. The stored root certificates are checked regularly with automated processes. The aim of this is to check is to make sure that the existing roots certificates are valid and delete all expired certificates. The storage of root certificates is executed manually by Hubject administrators.

Retrieving Contract Certificate

An EV needs a contract to be installed in this EVSE to be able to charge via ISO15118. This contract can be installed or updated via ISO15118 message. These message pairs are the *CertitificateInstalltionReq/Res* and *CertificatUpdateReq/Res*. Let's focus here on the installation request as an example to illustrate the process. The *CertificateInstalltionReq* is sent from the EVCC to the SECC. Then the SECC takes the request and forwards it over an OCCP message – *Get15118Certificate*. This message is then forwarded to the Hubject PKI over the API called get*SignedContractData*.


PCID
V2G
Root
certificateInstallationRequest
certificateInstallationResponse
V2G
Root
Charge
Point
Operator
certificateInstallationRequest
certificateInstallationResponse
Contract
Certificate
Pool

To conclude, for retrieving certificates, the CPO needs to implement the APIs. The first is the *certificateInstalationRequest* and the second one is the *getSignedContractData.*

Quality Assurance (QA) Testing

In this phase, we aim at testing all possible edge cases that are necessary for the preparation for productive Plug&Charge use. The following test cases are illustrated for this purpose. Hubject will provide you with the necessary test data to execute these tests. After completing these test units, the client is ready to go live. This means the Plug&Charge functionality is fully tested is can be shipped to the ends users.

Table 2: The test use cases employed for the QA testing for the CPO case.

Use Case	ID	Precondition	Test Data	Rest URL	Description	Expected Behavior
Contract Installation	CPO3	Vehicle not available	installation request (Hubject)	v1/ccp/signedContractData	install contract data with an unknown vehicle	4xx not possible - PCID not PCP
	CPO6	Contract valid	installation request (Hubject)	v1/ccp/signedContractData	install contract data	possible http 200
Contract Authorization (Optional)	CPO4	Contract expired	installation request (Hubject)	v1/ccp/signedContractData	install non valid contract data	authorization not possible
	CPO5	Contract revoked	installation request (Hubject)	v1/ccp/signedContractData	install non valid contract data	authorization not possible
EVSE Certificate Creation	CPO7	Valid CSR for EVCC	CSR by a customer from charging station	est/cpo/simpleenroll	Generation of EVSE Leaf Certificates	http 200: Successful EVSE Leaf Cert as pkcs7
	CPO7	Invalid CSR for EVCC	Invalid CSR by customer from charging station (fake base64 code)	est/cpo/simpleenroll	Generation of EVSE Leaf Certificates	http 500: Could not verify PKCS10 signature: Signature verification failure
	CPO8	-	Get CA Certificate chain and root CA	est/cpo/cacerts	Generation of EVSE Leaf Certificates	http 200 Successful EVSE CA Chain
GET MO Roots	CPO9	-	Get all roots from Root pool	v1/root/rootCerts	Get all roots from Root pool	http 200: get RootCertificateCollection
Access	CPO1	Token valid	CSR customer, Oauth Access Token and CPO-rights	v1/ccp/signedContractData	Test call if general access is possible	Successful response http 2xx
	CPO2	Token expired	CSR customer, Oauth Access Token expired	v1/ccp/signedContractData	Test call if general access is impossible	401 unauthorized Jwt is expired
	CPO2.1	Rights invalid	CSR customer, Oauth Access Token, and improper rights	v1/ccp/signedContractData	Test call if general access is impossible	403 Forbidden RBAC: access denied

3.4.2 Processes relevant to the OEM role

The business process is very similar to the one described in the previous section on the CPO role excluding the steps related to the charging point. It worth noting here that many OEMs may prefer to have their own PKI. In this case, they will create and sign their leaf certificate for their electric vehicles. The Hubject role will be limited to 1) singing the OEM sub1 CA (as in Figure 2), and 2) providing the security audit on their system to make sure that they are following the security guidelines of Hubject.

The relevant Interfaces

An OEM must interact with the Hubject ecosystem in two distinct processes. The first is during the production of the vehicle. This step is necessary to create the PCID of the vehicle and publish the provisioning certificate to Hubject's Provisioning Certificate Pool (PCP). An OEM also needs to connect to Hubject's Root Certificate Pool (RCP) to obtain at least one root CA to be installed in the vehicle. Once the vehicle has the root CA, the provisioning certificate, and the vehicle's private key (which is

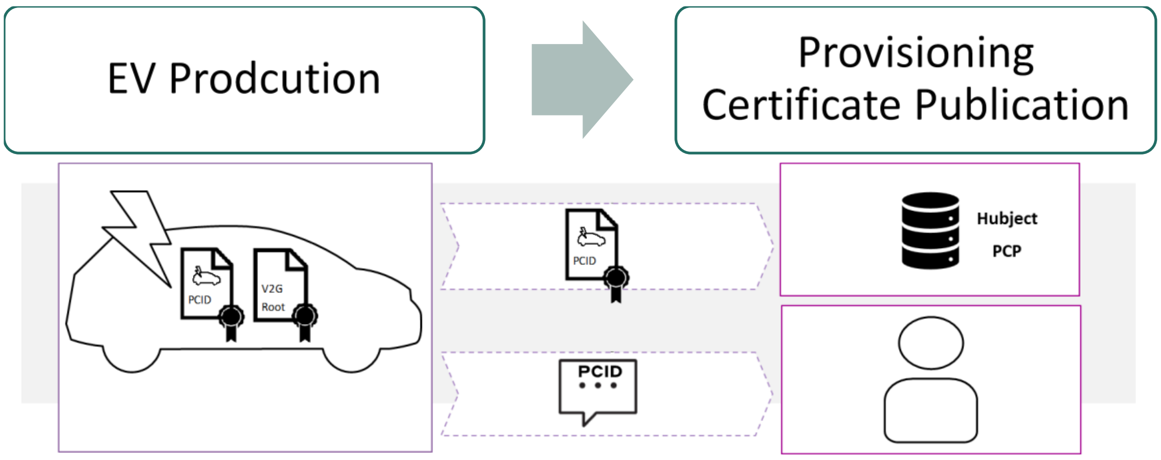


Figure 14: The OEM interface to PCP of Hubject.

created by the OEM), the EV is ready and the OEM's role is complete, at least for this phase. As we explained earlier, the OEM may have another role, which is to install the contract certificate into the EV by communicating directly with the vehicle via the telematics link. For an OEM to access the Hubject ecosystem, it needs to implement two sets of interfaces (or APIs). The first deals with the root CAs and the other are used to manage the vehicles' PCIDs. The latter can be used to publish a PCID for a new vehicle (AddOEMProvCer) or delete an existing provisioning certificate (DeleteOEMProvCer). The former can be used to access all root certificates available in the RCP. There is only one API used for this purpose, GetAllRootCertificates.

Quality Assurance (QA) Testing

In this phase, we aim at testing all possible edge cases that are necessary for the preparation for productive Plug&Charge use. The following test cases are illustrated for this purpose. Hubject will provide you with the necessary test data to execute these tests. After completing these test units, the client is ready to go live. This means the Plug&Charge functionality is fully tested is can be shipped to the ends users.

Table 3: The test use cases employed for the QA testing for the OEM case.

Use Case	Test ID	Precondition	Test Data	Rest URL	Description	Expected Behavior
Access	OEM1	Token valid	OAuth Access Token and OEM-rights		Test call if general access is possible	Successful response http 2xx
	OEM2	Token expired	OAuth Access Token expired		Test call if general access is impossible	401 unauthorized Jwt is expired
		Rights invalid	OAuth Access Token and improper rights		Test call if general access is impossible	403 Forbidden RBAC: access denied
PCP	OEM3	OEM root in pool			addOEM successful	201 created
	OEM4	OEM root not in the pool			addOEM not successful	404 Not Found
	OEM5	Invalid Chain (SubCA 1 or 2 revoked)			addOEM not successful	400 Bad Request
	OEM6	Invalid chain (SubCA 1 or 2 invalid)			addOEM not successful	
	OEM7	Provisioning Certificate revoked			addOEM not successful	
	OEM8	Provisioning Certificate expired			addOEM not successful	
	OEM9	PCID wrong format			addOEM not successful	
	OEM10	PCID with not customer-related PCID			addOEM not successful	
	OEM11	Prov cert in pool			Delete Prov Cert successful	
	OEM12	Prov Cert not in the pool			Delete Prov Cert not successful	

	OEM13	PCID wrong format			Delete Prov Cert not successful	
	OEM14	PCID with not customer-related PCID			Delete Prov Cert not successful	
	OEM15	Prov cert in pool			Get OEM successful	
	OEM16	Prov Cert not in the pool			Get OEM, not successful	
	OEM17	PCID wrong format			Get OEM, not successful	
	OEM18	PCID with not customer-related PCID			Get OEM, not successful	
CCP	OEM20	valid contract cert	installation request (Hubject)	v1/ccp/signedContractData	install non valid contract data	authorization not possible
	OEM21	expired contract cert	installation request (Hubject)	v1/ccp/signedContractData	install non valid contract data	authorization not possible
	OEM22	expires				
	OEM24	invalid emaid				
	OEM25	valid update				
	OEM26	wrong emaid				

3.4.3 Processes relevant to the MO role

The business process that a MO must follow is very similar to the OEM role process. In summary, a MO has two options. The first option is if the MO runs its own PKI, the process will first start with the general process for all roles. Then the MO must send a CSR for its sub1 CA. The MO will then use that to sign the certificate contracts for the end user before publishing them to Hubject's CCP pool. The second option is if an MO chooses to use Hubject's PKI to sign the contracts it creates. In this case, the MO will use a sub-CA of the V2G root to sign the contract certificates. The number of interfaces required for these processes varies depending on the MO's decision regarding PKI.

The phases of onboarding are not fixed and depend on the respective speed of the partner's implementation. The following steps are intended to show how and in what order Hubject can provide support. These phases can be summarized as follows.

1) Preliminary phase: signing of the Hubject Plug&Charge agreement in the form of a product or project agreement.

2) provision of QA access data: In the first step, the QA credentials are transferred in the form of a Postman Collection. The Postman Collection contains the URLs of all relevant APIs and the OAuth credentials. The organization, including the WMIs of the client organization, must be communicated upfront. The details of the OAuth authorization and relevant APIs can be found in the interface description.

3) Testing on QA: Testing all system-relevant functions on the QA Stage is essential for the sites. A detailed list of all test cases can be found in Section 5 and the relevant test data will be provided by Hubject.

4) IT Security Audit: The audit is used to rely on certain security standards in the Plug&Charge ecosystem. Questions are asked based on the requirements and other contents of the Hubject Certificate Policy.

5) prod credentials (and signing of OEM SubCA): after the audit and security audit are completed and evaluated, the credentials are provided for Plug&Charge productive use. If the OEM uses its own PKI, the process of signing the SubCA is initiated by sending the Certificate Signing Request (CSR).

The relevant Interfaces

In the following, we will briefly discuss the APIs used for the MO case. The relevant APIs used by a MO to interface with the Hubject ecosystem vary depending on whether the MO runs its own PKI or not. We will describe the general case here. After a MO completes a contract for a user or EV, it must publish it to Hubject's CCP. To do this, the MO can ask Hubject to generate the contract for it using the GenerateSignedContractData API or generate the contract and forward it to the Provisioning Service and then to the contract pool using the CreateAndForwardSignedContractData. Or if the MO runs its own PKI, then the MO uses the ForwardSignedContractData API. Another API to publish the contract certificate to the CCP is the addSignedContractData. Finally, however, the MO must also look for this particular user with its PCID to have a provisioning certificate published to the PCP. For this purpose, the MO can use the lookupvehicle API. The MO can also perform some operations on the contracts. This operation can be the deletion of an existing contract for one or more of its consumers. This can be done using the DeleteSignedContractDate. Shows the processes and interfaces involving the MO and all relevant certificate pools from the Hubject ecosystem.

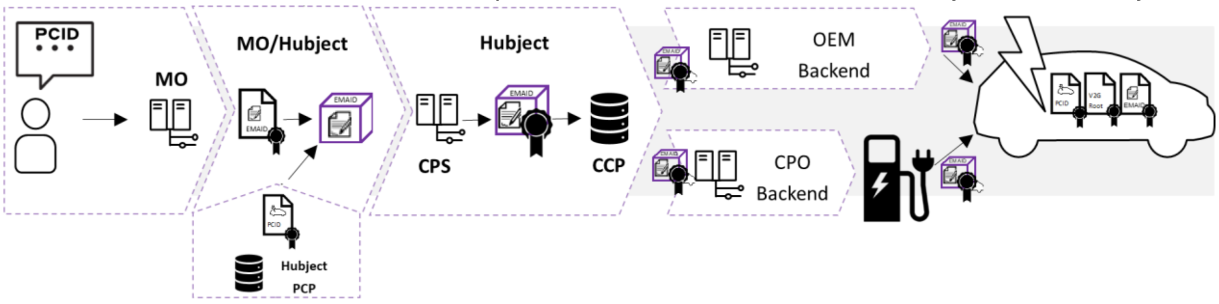


Figure 15: the process and the interfaces used by the MO role.

Quality Assurance (QA) Testing

We aim at testing all possible edge cases that are necessary for the preparation for productive Plug&Charge use. The following test cases are illustrated for this purpose. Hubject will provide you with the necessary test data to execute these tests. After completing these test units, the client is ready to go live. This means the Plug&Charge functionality is fully tested can be shipped to the ends users.

Table 4: The test use cases employed for the QA testing for the MO case.

Use Case	Test ID	Precondition	Test Data	Rest URL	Description	Expected Behavior
Access	MO1	Token valid	CSR customer, Oauth Access Token and CPO-rights	v1/oem/provCerts/{PCID}	Test call if general access is possible	Successful response http 2xx
	MO2	Token expired	CSR customer, Oauth Access Token expired	v1/oem/provCerts/{PCID}	Test call if general access is impossible	401 unauthorized Jwt is expired
	MO2.1	Rights invalid	CSR customer, Oauth Access Token, and improper rights	v1/oem/provCerts/{PCID}	Test call if general access is impossible	403 Forbidden RBAC: access denied
Sign Contract	MO3a (with PKI)	Vehicle in PCP	PCID by Hubject	v1/oem/provCerts/{PCID}	get Prov Cert from PCP for DH-Public Key	http 200 and OEM Prov cert incl. Chain
	MO4a (with PKI)	Vehicle not in PCP	PCID by Hubject	v1/oem/provCerts/{PCID}	get Prov Cert from PCP for DH-Public Key	http 404: "E0200": "The OEM provisioning certificate with the given PCID is not found.",
	MO5a (with PKI)	Valid Contract Data and MO Root Cert in Hubject Root Pool	Valid Contract Data by MO	v1/cps/forward/signedContractData	Sign and store Contract Data from MO in CCP	http 201
	MO6a (with PKI)	invalid Contract Data - empty dhPublic key	invalid data by MO	v1/cps/forward/signedContractData	try to sign and store contract data from MO in CCP	http 400
Generate and sign contract data	MO3b (hubject PKI)	Valid EMAID and Valid time frame, PCID in PCP	EMAID and time by MO	v1/cps/generateSignedContractData	create and sign contract data with Hubject PKI	http 201
	M4b (hubject PKI)	Vehicle not in PCP	PCID by Hubject, EMAID by MO	v1/cps/generateSignedContractData	try to create and sign contract data with Hubject PKI	http 404: E0200": "The OEM provisioning certificate with the given PCID is not found.",

	MO5b (hubject PKI)	invalid EMAID	EMAID and time by MO	v1/cps/generateSignedContractData		http 400: "E0251": "The given emaid does not match the specification.",
	MO6b (hubject PKI)	invalid time frame (contract longer than 2 years)		v1/cps/generateSignedContractData		htttp 400: "Exxx": "Invalid contract duration.",

3.4.4 Certificate Profiles of different roles

The certificate profiles as listed below are based on the ISO 15118 certificate profiles but reflect the Hubject stipulated requirements and validity times. The certificate profiles for the CPO, MO, and OEM Sub CAs are defined as follows.

Table 5: The certificate profile for the CPO role.

ISO 15118-2 Certificate Profiles	Cluster:	**Charge Point Operator (CPO)**			**OCSP**
	Name:	CPO Sub-CA 1	CPO Sub-CA 2	SECC Leaf Cert	OCSP Cert
	Typ:	Sub	Sub	Leaf	Leaf
tbsCertificate	Version	2 (X.509v3)	2 (X.509v3)	2 (X.509v3)	2 (X.509v3)
	SerialNumber	Integer	Integer	Integer	Integer
	Signature	ecdsa-with-SHA256	ecdsa-with-SHA256	ecdsa-with-SHA256	ecdsa-with-SHA256
Issuer	Country	(x)	(x)	(x)	(x)
	Organization	x	x	x	x
	Organization Unit	(x)	(x)	(x)	(x)
	CommonName	x	x	x	x
	Domain Component	"V2G"	(x)	"CPO"	(x)
Validity		40 years	1-2 years	2-3 month	up to 1 year
Subject	Country	(x)	(x)	x	-
	Organization	x	x	x	x
	Organization Unit	(x)	(x)	(x)	(x)
	Common Name	x	x	SECCID	x
	Domain Component	(x)	(x)	(x)	(x)
SubjectPublic KeyInfo	Public Key	x	x	x	x
	Cryptographic Algorithm	id-ecPublicKey	id-ecPublicKey	id-ecPublicKey	id-ecPublicKey
	Parameters	ECParameters (namedCurve secp256r1)	ECParameters (namedCurve secp256r1)	ECParameters (namedCurve secp256r1)	ECParameters (namedCurve secp256r1)
Extensions	AuthorityKeyIdentifier	(x)	(x)	(x)	(x)
	SubjectKeyIdentifier	(x) / nc	(x) / nc	(x) / nc	(x) / nc
	KeyUsage	c	c	c	c
	digitalSignature	0/1	0/1	1	0/1
	nonRepudiation (contentCommitment)	0/1	0/1	0/1	0/1
	keyEncipherment	0/1	0/1	0/1	0/1
	dataEncipherment	0	0	0	0
	keyAgreement	0/1	0/1	1	0/1
	keyCertSign	1	1	0	0
	cRLSign	1	1	0	0
	encipherOnly	0	0	0	0
	decipherOnly	0	0	0	0
	ExtendedKeyUsage	-	-	-	1.3.6.1.5.5.7.3.9 - OCSPSigning
	CertificatePolicies	-	-	-	-
	BasicConstraints	c	c	c	c
	CA	true	true	false	false
	PathLength	1	0	-	-
	CRLDistributionPoint	(x) / nc	(x) / nc	(x) / nc	(x) / nc

	s				
	Authority Information Access (OCSP)	(x) / nc id-ad-ocsp / location of the OCSP responder	(x) / nc id-ad-ocsp / location of the OCSP responder	(x) / nc id-ad-ocsp / location of the OCSP responder	(x) / nc id-ad-ocsp / location of the OCSP responder
	Cryptographic Algorithm	ecdsa-with-SHA256	ecdsa-with-SHA256	ecdsa-with-SHA256	ecdsa-with-SHA256
Signature Value		Octect-String	Octect-String	Octect-String	Octect-String

Table 6: The certificate profile for the MO role.

Hubject PKI Certificate Profiles	Cluster:	**Mobility Operator**			
	Name:	V2G Root CA	MO Sub-CA 1	MO Sub-CA 2	Contract Cert
	Typ:	Root	Sub	Sub/Leaf	Leaf
tbsCertificate	Version	2 (X.509v3)	2 (X.509v3)	2 (X.509v3)	2 (X.509v3)
	SerialNumber	Integer	Integer	Integer	Integer
	Signature	ecdsa-with-SHA256	ecdsa-with-SHA256	ecdsa-with-SHA256	ecdsa-with-SHA256
Issuer	Country	(x)	(x)	(x)	(x)
	Organization	x	x	x	x
	Organization Unit	(x)	(x)	(x)	(x)
	Common Name	x	x	x	x
	Domain Component	(x)	(x)	(x)	(x)
Validity		40 years	20 years	10 years	1 day - 2 years
Subject	Country	(x)	(x)	(x)	-
	Organization	x	x	x	x
	Organization Unit	(x)	(x)	(x)	(x)
	Common Name	x	x	x	EMAID
	Domain Component	"V2G"	(x)	(x)	(x)
SubjectPublic KeyInfo	Public Key	x	X	x	x
	Cryptographic Algorithm	id-ecPublicKey	id-ecPublicKey	id-ecPublicKey	id-ecPublicKey
	Parameters	ECParameters (namedCurve secp256r1)	ECParameters (namedCurve secp256r1)	ECParameters (namedCurve secp256r1)	ECParameters (namedCurve secp256r1)
Extensions	AuthorityKeyIdentifier	(x)	(x)	(x)	(x)
	SubjectKeyIdentifier	(x) / nc	(x) / nc	(x) / nc	(x) / nc
	KeyUsage	c	c	c	c
	digitalSignature	0/1	0/1	1	1
	nonRepudiation (contentCommitment)	0/1	0/1	1	1
	keyEncipherment	0/1	0/1	0/1	1
	dataEncipherment	0	0	0	0
	keyAgreement	0/1	0/1	0/1	1
	keyCertSign	1	1	1	0
	cRLSign	1	1	1	0
	encipherOnly	0	0	0	0
	decipherOnly	0	0	0	0
	ExtendedKeyUsage	-	-	-	-
	CertificatePolicies	(x) / nc	-	-	-
	BasicConstraints	c	c	c	c
	CA	true	true	true	false
	PathLength	-	1	0	-
	CRLDistributionPoints	(x) / nc	(x) / nc	(x) / nc	(x) / nc

	Authority Information Access (OCSP)	(x) / nc id-ad-ocsp / location of the OCSP responder	(x) / nc id-ad-ocsp / location of the OCSP responder	(x) / nc id-ad-ocsp / location of the OCSP responder	(x) / nc id-ad-ocsp / location of the OCSP responder
	Cryptographic Algorithm	ecdsa-with-SHA256	ecdsa-with-SHA256	ecdsa-with-SHA256	ecdsa-with-SHA256
Signature Value		Octect-String	Octect-String	Octect-String	Octect-String

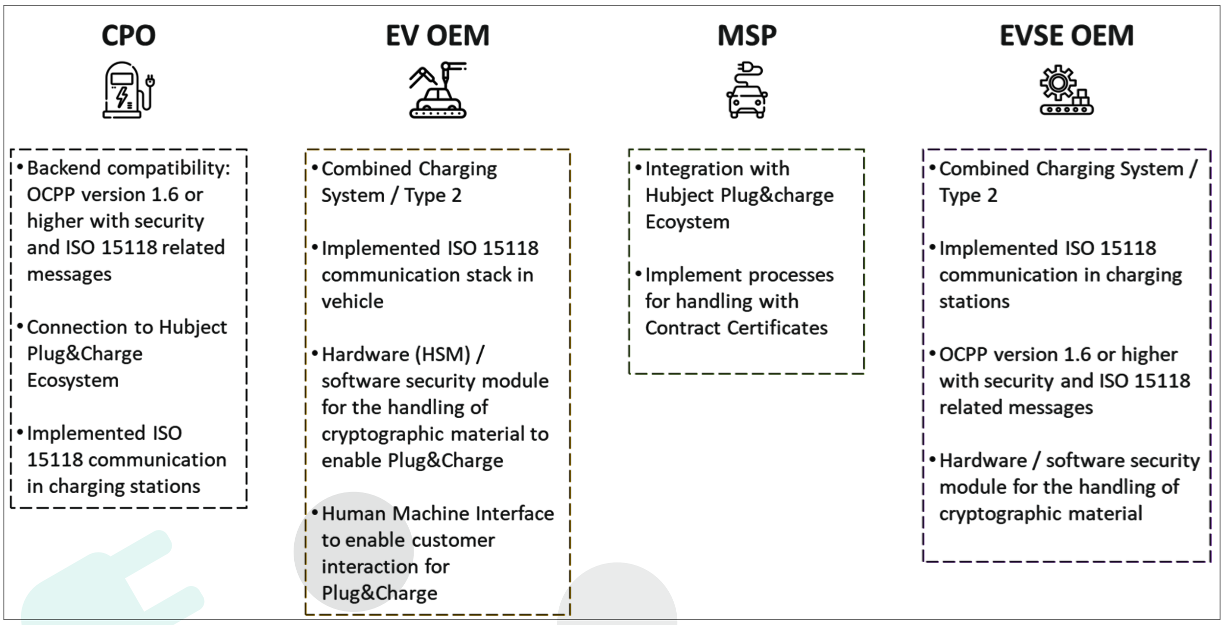
Table 7: The certificate profile for the OEM role.

Hubject PKI Certificate Profiles	Cluster:	EV Manufacturer: OEM Provisioning			
	Name:	V2G Root CA	OEM Sub-CA 1	OEM Sub-CA 2	OEM Prov. Cert.
	Typ:	Root	Sub	Sub	Leafs
tbsCertificate	Version	2 (X.509v3)	2 (X.509v3)	2 (X.509v3)	2 (X.509v3)
	SerialNumber	Integer	Integer	Integer	Integer
	Signature	ecdsa-with-SHA256	ecdsa-with-SHA256	ecdsa-with-SHA256	ecdsa-with-SHA256
Issuer	Country	(x)	(x)	(x)	(x)
	Organization	x	x	x	x
	Organization Unit	(x)	(x)	(x)	(x)
	Common Name	x	x	x	x
	Domain Component	(x)	(x)	(x)	(x)
Validity		40 years	20 years	10 years	Max. 10 years
Subject	Country	(x)	(x)	(x)	-
	Organization	x	x	x	x
	Organization Unit	(x)	(x)	(x)	(x)
	Common Name	x	x	x	PCID
	Domain Component	(x)	(x)	(x)	(x)
SubjectPublic KeyInfo	Public Key	x	x	x	x
	Cryptographic Algorithm	id-ecPublicKey	id-ecPublicKey	id-ecPublicKey	id-ecPublicKey
	Parameters	ECParameters (namedCurve secp256r1)	ECParameters (namedCurve secp256r1)	ECParameters (namedCurve secp256r1)	ECParameters (namedCurve secp256r1)
Extensions	AuthorityKeyIdentifier	(x)	(x)	(x)	(x)
	SubjectKeyIdentifier	(x) / nc	(x) / nc	(x) / nc	(x) / nc
	KeyUsage	c	c	c	c
	digitalSignature	0/1	0/1	0/1	1
	nonRepudiation (contentCommitment)	0/1	0/1	0/1	0/1
	keyEncipherment	0/1	0/1	0/1	0/1
	dataEncipherment	0	0	0	0
	keyAgreement	0/1	0/1	0/1	1
	keyCertSign	1	1	1	0
	cRLSign	0/1	1	1	0
	encipherOnly	0	0	0	0
	decipherOnly	0	0	0	0
	ExtendedKeyUsage	-	-	-	-
	CertificatePolicies	(x) / nc	-	-	-

	BasicConstraints	c	c	c	c
	CA	true	true	true	false
	PathLength	-	1	0	-
	CRLDistributionPoints	(x) / nc	(x) / nc	(x) / nc	(x) / nc
	Authority Information Access (OCSP)	(x) / nc id-ad-ocsp / location of the OCSP responder	(x) / nc id-ad-ocsp / location of the OCSP responder	(x) / nc id-ad-ocsp / location of the OCSP responder	(x) / nc id-ad-ocsp / location of the OCSP responder
	Cryptographic Algorithm	ecdsa-with-SHA256	ecdsa-with-SHA256	ecdsa-with-SHA256	ecdsa-with-SHA256
Signature Value		Octect-String	Octect-String	Octect-String	Octect-String

3.5 Summary of the requirements

We can summarize here the part of the requirements in the following table. One more actor is added, which is not strictly active in the Plug&Charge ecosystem but is important to complete the chain that allows ISO15118 to work. This actor is the manufacturer of the EVSE or the EVSE OEM. The main requirement for the EVSE OEM is the hardware security module that makes the access and storage of the private key secure. The EVSE OEM must also implement the ISO15118 communication stack and provide an ICCS Type 2 connection port. The ISO 15118 communication can either be performed by the EVSE OEM during the manufacturing process. Or the CPO is responsible for updating their charge points with the ISO 15118 requirements if they do not have them.



CPO

- Backend compatibility: OCPP version 1.6 or higher with security and ISO 15118 related messages
- Connection to Hubject Plug&Charge Ecosystem
- Implemented ISO 15118 communication in charging stations

EV OEM

- Combined Charging System / Type 2
- Implemented ISO 15118 communication stack in vehicle
- Hardware (HSM) / software security module for the handling of cryptographic material to enable Plug&Charge
- Human Machine Interface to enable customer interaction for Plug&Charge

MSP

- Integration with Hubject Plug&charge Ecoystem
- Implement processes for handling with Contract Certificates

EVSE OEM

- Combined Charging System / Type 2
- Implemented ISO 15118 communication in charging stations
- OCPP version 1.6 or higher with security and ISO 15118 related messages
- Hardware / software security module for the handling of cryptographic material

Figure 16: A summary of the requirements for each e-mobility actor in the Plug&Charge ecosystem.

On-line trainning material

The support provided to Task 4.1 participants in implementing the PnC function based on ISO 15118 has taken many forms. These include workshops, email communications, weekly support sessions and messaging channels. Some of these activities have been recoded and made available to all participants to use as reference when working on the project. We plan to make some of these recordings publicly available after going through a review process and obtaining the necessary permissions from relevant parties. This would not only help the project participants to learn more about PnC and ISO 15118, but also the entire mobility market.

The training material is available in the project website (https://echarge4drivers.eu/library/) for further and/or more updated information.

4. CONCLUSIONS

This task is intended to achieve three goals. The first objective is to develop a guide for implementing plug-and-charge technology. The second is to provide task members with the support and know-how needed for the implementation. The third goal is to have all members of the task implement and integrate the ISO15118 Plug&Charge feature into their backends.

Training and support provided by Hubject to eC4D members ranged from workshops, customized sessions, jour fixes, support lines, group chats, and emails. We also provided recorded sessions and various support materials to assist task members with implementation.

This deliverable serves the second goal. The training materials in the form of recorded training sessions, along with this document should meet the requirements of the second goal. However, we have found through communication with our partners over the past 12 months that despite support and training, many members are struggling with ISO 15118 implementation and progress is slower than planned and expected. For this reason, we have initiated a parallel task force of several project partners to develop an open-source Python-based implementation for ISO 15118 to be made available to all. The open-source ISO15118 can be delivered as a result of this task yet not necessarily tied to its schedule.

The third objective of this task focuses on the implementation of the Plug&Charge feature. This task can be divided into two sub-tasks. The implementation of the interfaces with the Hubject Plug&Charge ecosystem and the implementations of the ISO15118 communication stack. Each partner was provided with the means to test the Plug&Charge interfaces through interactive technical training sessions.

REFERENCES

15118-2:2014, ISO. (2014). *Road vehicles — Vehicle-to-Grid Communication Interface — Part 2: Network and application protocol requirements.* ISO.

Fraunhofer SIT. (2019). *System Security Mechanisms for Electric Vehicles and Charge Points Supporting ISO 15118.* Fraunhofer SIT.

Open Charge Alliance. (2020). *Using ISO 15118 Plug & Charge with OCPP 1.6.* PCA.

Open Charge Alliance. (2021). *Openchargealliance.org*. Retrieved 2021, from https://www.openchargealliance.org/protocols/ocpp-201/

Oppliger, R. (2016). *SSL and TLS: Theory and Practice.* Artech House.

VDE. (2019). *Handling of certificates for electric vehicles, charging infrastructure and backend systems within the framework of ISO 15118.*