ETSI GR PDL 019 V1.1.1 (2023-05)

GROUP REPORT

PDL Services for Decentralized Identity and Trust Management

Disclaimer

The present document has been produced and approved by the Permissioned Distributed Ledger (PDL) ETSI Industry Specification Group (ISG) and represents the views of those members who participated in this ISG.
It does not necessarily represent the views of the entire ETSI membership.

Reference

DGR/PDL-0019_Trust_Management

Keywords

decentralized identifier, keyword, PDL

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - APE 7112B
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° w061004871

Important notice

The present document can be downloaded from:
https://www.etsi.org/standards-search

The present document may be made available in electronic versions and/or in print. The content of any electronic and/or print versions of the present document shall not be modified without the prior written authorization of ETSI. In case of any existing or perceived difference in contents between such versions and/or in print, the prevailing version of an ETSI deliverable is the one made publicly available in PDF format at www.etsi.org/deliver.

Users of the present document should be aware that the document may be subject to revision or change of status. Information on the current status of this and other ETSI documents is available at https://portal.etsi.org/TB/ETSIDeliverableStatus.aspx

If you find errors in the present document, please send your comment to one of the following services:
https://portal.etsi.org/People/CommiteeSupportStaff.aspx

If you find a security vulnerability in the present document, please report it through our Coordinated Vulnerability Disclosure Program:
https://www.etsi.org/standards/coordinated-vulnerability-disclosure

Notice of disclaimer & limitation of liability

The information provided in the present deliverable is directed solely to professionals who have the appropriate degree of experience to understand and interpret its content in accordance with generally accepted engineering or other professional standard and applicable regulations.
No recommendation as to products and services or vendors is made or should be implied.
No representation or warranty is made that this deliverable is technically accurate or sufficient or conforms to any law and/or governmental rule and/or regulation and further, no representation or warranty is made of merchantability or fitness for any particular purpose or against infringement of intellectual property rights.
In no event shall ETSI be held liable for loss of profits or any other incidental or consequential damages.

Any software contained in this deliverable is provided "AS IS" with no warranties, express or implied, including but not limited to, the warranties of merchantability, fitness for a particular purpose and non-infringement of intellectual property rights and ETSI shall not be held liable in any event for any damages whatsoever (including, without limitation, damages for loss of profits, business interruption, loss of information, or any other pecuniary loss) arising out of or related to the use of or inability to use the software.

Copyright Notification

Contents

Intellectual Property Rights

Essential patents

IPRs essential or potentially essential to normative deliverables may have been declared to ETSI. The declarations pertaining to these essential IPRs, if any, are publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (https://ipr.etsi.org/).

Pursuant to the ETSI Directives including the ETSI IPR Policy, no investigation regarding the essentiality of IPRs, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Trademarks

The present document may include trademarks and/or tradenames which are asserted and/or registered by their owners. ETSI claims no ownership of these except for any which are indicated as being the property of ETSI, and conveys no right to use or reproduce any trademark and/or tradename. Mention of those trademarks in the present document does not constitute an endorsement by ETSI of products, services or organizations associated with those trademarks.

DECT™, **PLUGTESTS™**, **UMTS™** and the ETSI logo are trademarks of ETSI registered for the benefit of its Members. **3GPP™** and **LTE™** are trademarks of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners. **oneM2M™** logo is a trademark of ETSI registered for the benefit of its Members and of the oneM2M Partners. **GSM**® and the GSM logo are trademarks registered and owned by the GSM Association.

Foreword

This Group Report (GR) has been produced by ETSI Industry Specification Group (ISG) Permissioned Distributed Ledger (PDL).

Modal verbs terminology

In the present document "**should**", "**should not**", "**may**", "**need not**", "**will**", "**will not**", "**can**" and "**cannot**" are to be interpreted as described in clause 3.2 of the ETSI Drafting Rules (Verbal forms for the expression of provisions).

"**must**" and "**must not**" are **NOT** allowed in ETSI deliverables except when used in direct citation.

Executive summary

The present document presents the potential security and privacy benefits of decentralized identification that can benefit various public and private services. Further the present document also discusses a set of PDL services that can together enable a PDL based Identity and Trust Management framework.

Introduction

The study analyses and presents the overview of decentralized identification approaches and trust data management methodologies that can benefit different set of services (which involves electronic transactions) taking into account various factors such as the requirement of the service(s), privacy requirements, security requirements and type of involved stakeholders, etc. The decentralized identification method links various essential and limited set of attributes (specific to the end-user(s) or device) as required for any specific service that need to be shared with the service provider(s) or verifier(s) in order to authenticate end-user/device to offer a specific service. The study also discusses various use case(s) that can rely on the method of decentralized identification and further the study presents the method(s) to efficiently realize a PDL based decentralized identification and trust management framework and service(s).

1 Scope

The present document studies and analyses required PDL framework services related to the following aspects such as:

- Various Decentralized identification methods, benefits, security, and privacy considerations:
    - overview of related activities and initiatives.
- PDL based Decentralized identification and trust service management framework:
    - includes concept to build trust, binding limited attributes, trust service(s) co-operation, data management, secure data sharing and verification;
    - governance of various stakeholders participating in the framework.
- Co-operation with APIs related to public services (e.g. eIDAS framework and EBSI services) and private services.

2 References

2.1 Normative references

Normative references are not applicable in the present document.

2.2 Informative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

[i.1] ENISA press release on: "Beware of Digital ID attacks: your face can be spoofed!", January 20, 2022.

[i.2] ENISA publications on: "Remote ID Proofing", March 11, 2021.

[i.3] W3C, Decentralized Identifiers (DIDs) v1.0: "Core architecture, data model, and representations", August 03, 2021.

[i.4] NIST IR 8413: "Status Report on the Third Round of the NIST Post-Quantum Cryptography Standardization Process", July 2022.

[i.5] EIDAS: "Supported Self-Sovereign Identity", May 2019.

[i.6] ENISA: "eIDAS Compliant eID Solutions", March 2020.

[i.7] ENISA: "Digital Identity, Leveraging the Self-Sovereign Identity (SSI) Concept to Build Trust", January 2022.

[i.8] GSMA: "Blockchain for Development: Emerging Opportunities for Mobile, Identity and Aid", 2017.

[i.9] GSMA: "Connecting through a secure digital identity with Mobile Connect".

[i.10] ETSI GS PDL 012 (V1.1.1): "Permissioned Distributed Ledger (PDL); ReferenceArchitecture".

[i.11] ETSI GR PDL 003 (V1.1.1): "Permissioned Distributed Ledger (PDL); Application Scenarios". .

[i.12] ETSI GR PDL 004 (V1.1.1): "Permissioned Distributed Ledgers (PDL); Smart Contracts; System Architecture and Functional Specification".

[i.13] ETSI GR PDL 010 (V1.1.1): "PDL Operations in Offline Mode".

[i.14] ETSI GR DPL 018 (V1.1.1): "Redactable Distributed Ledgers".

[i.15] "What Do Web3, Decentralized Identity, And Reese Witherspoon Have In Common?".

3 Definition of terms, symbols and abbreviations

3.1 Terms

Void.

3.2 Symbols

Void.

3.3 Abbreviations

For the purposes of the present document, the following abbreviations apply:

API	Application Programming Interface
DID	Decentralized IDentifier
DLT	Distributed Ledger Technology
EBSI	European Blockchain Services Infrastructure
eID	electronic Identification
eIDAS	electronic IDentification, Authentication and trust Services
GDPR	General Data Protection Regulation
ID	Identifier
IoT	Internet of Things
KYC	Know Your Customer
L-RMS	Ledger role-based Registration Management Service
RFID	Radio Frequency Identification
SIM	Subscriber Identity Module
SLA	Service Level Agreement
SSI	Self Sovereign Identity
URI	Uniform Resource Identifiers
URL	Uniform Resource Locator
VC	Verifiable Credentials
W3C	World Wide Web Consortium
ZKP	Zero KnowledgeProof

4 Overview of Decentralized Identification and Trust Management

4.1 Need for Decentralized Identification

With the evolution of technologies, business and advanced services, a seamless, user friendly, trusted and privacy preserved identity management system is required. The traditional centralized identity management that serves as a promising candidate for decades, may fall short to meet the demands of emerging advanced services (e.g. on-demand identity creation, binding trust, trust verification, service specific limited information sharing, improved user control over identity and identity related data, etc.). Decentralized IDentifiers (DIDs) are expected to become the next generation digital identities as they can be generated seamlessly, decoupled from formal identities (e.g. passport number, university ID, national ID, any service subscription identifier, etc.) and the end-user can have full control over the DID (i.e. generation, binding of any data as attributes, deletion, etc.). Any number of DIDs can be generated and used based on the user and service requirements (i.e. for any number of services) independent of any specific identity provider or third party, building trust and authenticating with service providers.

Individuals and organizations use globally unique identifiers in a wide variety of contexts. Examples thereof could be:

- communications addresses (telephone numbers, email addresses, usernames on social media);
- IDentification (ID) numbers (for passports, drivers licenses, tax IDs, health insurance);
- product identifiers (serial numbers, barcodes, Radio-Frequency IDentification (RFIDs));
- Uniform Resource Identifiers (URIs) used for resources on the Web.

Each web page that is viewed in a browser has a globally unique Uniform Resource Locator) (URL).

Similarly, DIDs can be used as a reference to the subject to be identified (e.g. user/entity) facilitating the identification, verification, and related authentication process. Such reference could be, for example, a URL directing to a document which provides sufficient data for identification purposes.

The vast majority of these globally unique identifiers are not under the control of the object being identified. In a centralized identity management environment the identifiers are issued by external authorities that define and control what objects they identify to and the validity of such identifiers. They are useful in certain contexts and recognized by certain bodies. However, they are not suitable for some contexts and not recognized by all (e.g. a solicitor's license issued by a certain country may not be accepted or recognized by another country and its carrier may not be able to practice law in that other country). Such identifiers may be revoked or deemed invalid in the event that the issuer suffers a technical failure and is unable to confirm validity on-demand. Identifiers might unnecessarily reveal personal information that is not required for identification. In many cases, identifiers are prone to fraudulent replication and assertation by malicious third-parties, a process commonly known as "identity theft".

The DIDs discussed in this study represent a new type of globally unique identifiers, where associated data can be tailored according to the object's privacy and service requirements. This allows individuals and organizations to generate their own identifiers using systems they trust. These new identifiers allow the identity holders (entities or users) to prove ownership and control by authenticating using cryptographic proofs such as digital signatures.

Since the generation and assertion of DID can be controlled by the object or related organization, each object can have as many DIDs as necessary to maintain their desired separation of identities, personas, and interactions specific to different public and private services respectively. The use of these identifiers can be scoped appropriately to different contexts as required by the service(s). DIDs support interactions with other people, institutions, or systems that require entities to identify themselves, or things they control, while providing control over how much personal or private data should be revealed, all without depending on a central authority to guarantee the continued existence of the identifier.

4.2 General Identity Security Risks

Identity security should be a comprehensive approach that needs to protect any type of identity that may belong to an object (person, entity or device). Such an approach should detect and prevent identity-driven breaches with specific consideration to scenarios where skilled adversaries might manage to circumvent endpoint security measures. The majority of modern day breaches are identity driven, where attackers circumvent traditional security measures by sniffing or directly leveraging compromised credentials. Such breaches may result in data theft, illegitimate access, lateral movements, and more catastrophic scenarios. Identity-driven attacks are often extremely hard to detect i.e. if a valid user's credentials have been compromised and an adversary attempts to masquerade as a valid user, it is often very difficult to differentiate between the user's typical behaviour and the hacker's behaviour using traditional security measures. This clause describes several identity related threats that should be taken into account when considering an identity security approach:

1) Data leakage

   Identifier(s) which can directly identify an identity holder (e.g. a bank account owner) may contain meaningful information about the identity holder that can be exploited to extract meaningful information about the identity holder (e.g. username, subscription number, telephone number, etc.). In such a case, access to such identifiers allows attackers with malicious intentions to collect sensitive information about the user (e.g. user behaviour pattern, bank account details, passwords, etc.).

2) Replay

   Attackers with malicious intension can attempt to eavesdrop on a communication medium, record the identifier and related messages and later replay the recorded content to impersonate the authentic user in order to gain access to the service or to misdirect the receiver/relying party.

3) Identity holder Tracking

   When attackers are able to track identifiers, even where such tracking does not reveal the identity of the identity holder, they may monitor and track the activities of the identity holder which may cause serious impacts to the identity holder's privacy and safety. Through cross-referencing information from other sources the actual identity may be discovered.

4) Spear phishing

   Attackers knowing the identifier(s) which directly identify or address the identity holder can target the user or the organization related to the identifier to extract more sensitive information (such as passwords, credit card details, etc.). Such phishing will be masqueraded as a genuine request for information which the user may be tempted to trust and thus provide said information. For example, an email or a text message will be crafted as a genuine message to set trap for the identified user/organization to increase the probability of attack success rate. Spear phishing is also known as *credential interception*.

5) Credential stuffing

   Attackers can use automated scripts to use known compromised credentials obtained from other compromised service(s). This attack success rate is relatively high, as the majority of users reuse their credentials for multiple accounts or services.

6) Password spray or guessing

   Automated scripts can be used to compromise user accounts or services by guessing random passwords related to the identifiers or username. This method is also known as *birthday attack* (representing users' tendency to use their birthday as a password) or *brute force attacks*. A counter measure to brute force attack would be to block access to an account after a certain number of attempts with wrong passwords and alerting the user and administrators of the event. Other approaches would be a temporary block that is automatically lifted after a certain pause. Attackers may exploit such temporary blocks with a "low-and-slow" approach, to avoid detection.

7) Flooding

This exploit may not reveal the identity of users but may attempt resource exhaustion over the authentication system and prohibits use of the attacked system by flooding the identification service with a higher volume of (fraudulent) requests than it can process, thus disabling valid users from being identified and restricting their access to the respective systems. The system which utilizes authentication methods that involves multiple round trips of authentication message exchanges between the end device and authenticator to verify the identity are prone to this attack.

8) Spoofing

Remote identity proofing is a popular method to collect and use biometric evidence (e.g. fingerprint, facial recognition) to gain access to applications handling certain personal information (e.g. credit history, personal demographic information, health information). A person with malicious intension can attempt to masquerade or impersonate legitimate users by spoofing the human face using methods such as 3D mask, deep fake attacks, etc., [i.1] and [i.2].

9) Lack of flexibility with identifiers

The traditional identification methods as well as the services which rely on such identification methods, are inflexible when it comes to switching to a new identifier. It is often impossible to retain or transfer access to a service to the same user when such user has changed its identity or has switched to a new, more secure, identity service. As a result, identity holders will tend to retain old static, insecure, identifiers that are at higher risk to be compromised.

10) Lack of identity holder related data exposure control

During onboarding to any new service, the user may need to establish initial trust with the service provider either directly or via a third party. This would be a prerequisite to gain subscription to such service and would allow the exchange of subscription specific credentials (e.g. subscription identifier, cryptographic materials, etc.). Such trust is also required to access the actual service (e.g. to activate communication service, opening bank accounts, property/vehicle rental service, etc.). To establish the initial trust, the user would typically need to provide sensitive identity related documents (e.g. passport, driving licence, national identity card, etc.). The service provider may need to rely on third parties to verify the validity and authenticity of such documents with government and institutional databases. In the event of identity cloning (i.e. identity document copying, hijacking, forgery) the service provider's reputation will be impacted and the user/customer's safety and security will be put at risk. Most service providers do not need access to each and every detail in such identity related documents. For example, access to age restricted services would require date of birth information, while the supporting document may also include the nationality and address of the user which are not needed for that purpose. The ability to control the level of details and to select what details are exposed or kept hidden would reduce the risk of data leakage and identity theft. Lack of sufficient data exposure control will lead to unnecessary user data sharing and availability in the digital network space, which if collected and available in the hands of any attackers will give way for more serious privacy and security threats specific to the identity holder.

The threats discussed in this clause are presented with the relevant security properties which can be impacted along with the respective consequences in the following Table 4.2-1.

Table 4.2-1: Threats and assessment overview

Threat	Properties violated	Consequence(s)
Data leakage	Privacy	User data extraction Tracking Targeted attacks Simplifies attack complexity
Replay	Non-repudiation Authentication	Unauthorized service access Illegitimate access
Identity holder Tracking	Privacy	Tracking of user (e.g. user service access pattern, location tracking, etc.)
Spear phishing	Privacy, data security	Targeted attacks to infiltrate and extract more information (e.g. data or device hijack)
Credential stuffing	Access Control, Authentication	Unauthorized service access Illegitimate access
Password spray or guessing	Access Control, Authentication	Unauthorized service access Illegitimate access
Flooding	Authentication	Denial of service or distributed denial of service
Spoofing	Authentication and Authorization	Impersonation/Masquerading, and illegitimate access of service and data
Lack of flexibility with identifiers	User access control, User account preferences	Vulnerability of user identifiers and accounts
Lack of identity holder related data exposure control	User consent	Sensitive data being exposed to parties (e.g. service provider or intermediaries) leading to misuse of data

4.3 Properties of Decentralized Identity (DID)

Trust in the identity of the subject or object (i.e. a natural or legal person, entity, etc.) has become the cornerstone of all digital services and activities. Therefore, all form of decentralized identities (including, but not limited to W3C DIDs [i.4]) considers the following set of properties to meet the security, privacy and flexibility requirements:

1) Decentralized management: Single point of failure will be prevented with adoption of decentralized identity management. Any digital service specific identification and authentication of an identity holder (i.e. user) can be facilitated with a decentralized platform that enables globally unique digital identifier (i.e. with no possibility of duplication) registration, management and control of associated cryptographic verification data, service information, etc.

2) Identity Control: The identity holder (i.e. a user or entity), should be given the control to manage (e.g. create, re-fresh, re-use, revoke) their digital identity (which is in a DID form), without being assigned, or provided (e.g. sold or rented) by any external party.

3) Proof-driven: The DID should provide cryptographic proofs to validate the identifier and the corresponding identity holder's request (e.g. service request). This in turn enables the relying party (e.g. any service provider) to verify if the claimed entity is the genuine identity holder or the controller.

4) Recoverable: DIDs should be recoverable even if the wallet is stolen or if any of the associated document gets destroyed (e.g. due to any natural disaster or theft as artifacts can be stolen). A genuine identity holder should be able to reassert the identification information to recover the DIDs as required.

5) Minimal end-user involvement: The verification of DID should be solely based on the identification framework and the corresponding trust binding information (i.e. associated for the managed identity holder related verification information). Identifier and authentication need not involve issuer of the identifier in the DID verification process.

6) Sufficient cryptographic future proof and resilience: The decentralized identification framework should facilitate, to use DIDs with most recent technologies as and when it evolves. Current cryptographic techniques (e.g. asymmetric cryptography which involves public and private key pairs) are known to be susceptible to quantum computational attacks. Future proof cryptographic methods such as defined by NIST IR 8413 [i.4] if adopted can enable DID usage with quantum safe cryptography.

7) Privacy by design: The DID by itself should not be linkable to the actual identity holder related information in any form by anyone except who has the authorization (e.g. respective service provider or a regulatory body under judiciary request) to the associated identification related information (e.g. identification verification and authentication specific data.

8) Selective disclosure: The identity holder should be able to control the privacy of information, by binding minimal, selective and controlled disclosure of attributes or other data related to the DID verification.

9) Replay resistance: Even if the DID is cached through interception by any attacker, the DID should be replay protected to prevent illegitimate access and flooding attacks.

10) Delegation of control: The controller of the identifier (i.e. an identity holder) should be able to delegate the controller role to another entity or organization if required (e.g. can include a use case where an operator need to control and manage the devices in the factory floor; another use case includes, an employer, would like to manage the identities related to the employees, etc.).

11) Portability: The DID based identification framework should be system independent as well as network independent and enable entities to use their digital identifiers with any system that supports DIDs, DID methods and interactions with distributed ledger technology.

4.4 Overview of various forms of Decentralized Identifiers and related initiatives

This clause describes various forms of decentralized identities including Self Sovereign Identity (SSI) that can be used for decentralized identification purpose based on the different business case and the public/private service needs. Further this clause also presents an overview of various standards initiatives related to decentralized identities, which mainly focus on the identity framework, schemas, data models, protocols, APIs, open-source code and so on.

Table 4.4-1

Few key DID related Initiatives	Features and Characteristics
W3C DID [i.3]	According to the W3C standard, DID is an URI composed of three parts: the scheme did, a method identifier, and a unique, method-specific identifier specified by the DID method. DIDs are resolvable to DID documents (which provides information on the verification methods, cryptographic keys and services relevant to the interactions with the DID holder/subject). The subject of a DID is the entity identified by the DID, where the subject of a DID can be any such as a person, group, organization, thing, or concept. The DID subject can also be the DID controller. The W3C covers various activities (standards and implementation) related to DID, Verifiable Credentials (VC) data model, DID resolution, APIs for Issuers, APIs for Verifiers, linked data vocabulary (i.e. for asserting VCs related to DID holder data i.e. residency and citizenship data such as name, country of citizenship, birthday and other required attributes to determine the status of the DID holder's citizenship), and APIs for credential handling.
eIDAS Digital Identity/electronic ID (eID) [i.5] and [i.6]	Blockchain-based eID solutions can be used as electronic identification means, once the identity information is proofed using a third party eID solution. In this case, the trustworthiness of the identity information or verifiable claim a user can share is inherited from the authority that proofed that piece of information. It may appear inconsistent to rely on trusted third parties to proof identities to be used in a decentralized system, but blockchain-based eID solutions offer a standard independent cross-platform technology that a trusted third parties could offer and manage. Identity proofing can be performed when the identity information is first inserted by the owner, or later when the user wants to verify it against a certain level of assurance requested by signature a service provider. Under the eIDAS framework, digital identity is asserted (i.e. identity proofing) in two different ways to link the DID to the actual identity data of the DID owner, depending on how this digital identity is used: i) By means of an authentication done with a notified electronic identification (eID) scheme, when identification is required to access online services. ii) By means of the production of an electronic or an electronic seal, when the identity of signer/sealer needs to be associated to the content signed or sealed. This is done in practice by using electronic certificates issued by trust service providers.

4.5 Benefits of Decentralized IDentity (DID)

The various properties of DIDs itself bring in significant benefits to the DID holders and the relying parties who utilizes a DID based identification and trust management framework. In addition, few of the key benefits of DID based identification and authentication includes:

- Zero Knowledge Proof (ZKP), where a proof uses special cryptography to support selective Disclosure of information (for an identity holder) about a set of Claims from a set of Credentials. A ZKP provides cryptographic proof about some or all of the data in a set of Credentials without revealing the actual data or any additional information, including the identity of the holder (i.e. who is the identity prover).
- Controlled Transparency while disclosing necessary user/identity holder data, can be achieved if the DID based identification framework is implemented with a permissioned ledger, as only registered participants with significant access control will be allowed to request and receive identity holder's data specific to the required service.
- Pseudonymization is a direct benefit of DID. More suitable DID registries and DID methods usage can guarantee pseudonymization , where it will allow an identity holder to manage as many pseudonyms as desired for more than one service, so that a pseudonym identity holder can interact with various services securely. This enables authentication without revealing more data. Pseudonymity is also one of the main advantages of DID documents and verifiable presentations over the traditional X.509 for electronic identification.

5 Trust Management Model for decentralized identification and data handling

5.1 An overview of identification and related data handling trust management model

Basic architectural elements and functionalities that forms a decentralized identification and trust management model is shown in Figure 5.1-1 below. This clause describes the various participants, their roles, and essential operations that are involved in the trust management model to enable a DID based identification and authentication considering [i.3], [i.5] and [i.7].

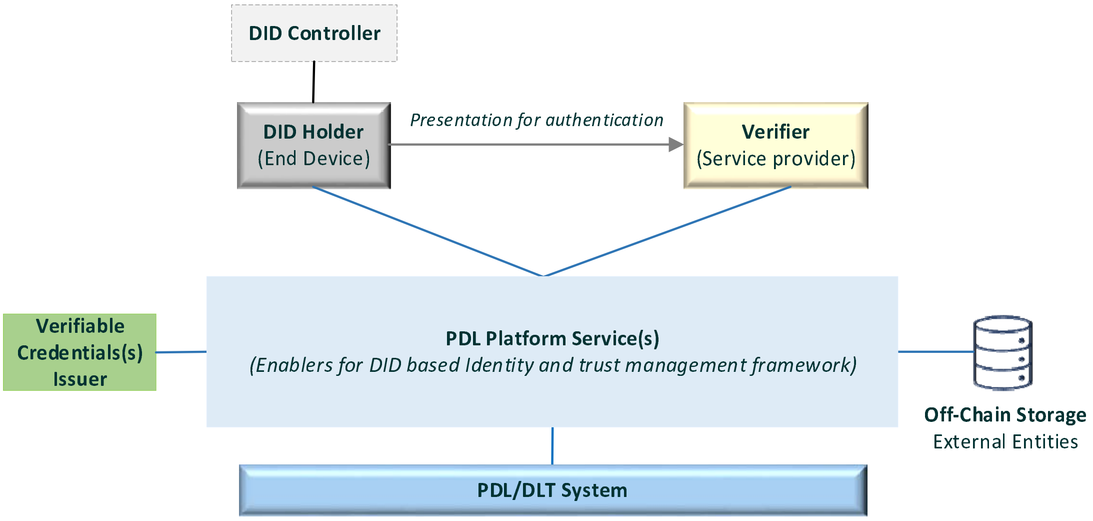

Figure 5.1-1: Basic trust management model for decentralized identification

DID holder related aspects:

- ***Decentralized IDentifier (DID):*** DIDs are a new type of identifier(s) for verifiable, "self-sovereign" digital identity. DIDs are fully under the control of the DID holder (i.e. subject), independent from any centralized registry, identity provider, or certificate authority. DIDs can be URLs/URIs that relate a DID subject to means to enable trustable interactions with that subject. DID refers to any subject (e.g. a person, organization, thing, data model, abstract entity, etc.) as determined by the controller of the DID. A DID may be considered as a form of pseudonym as used in eIDAS as it is not directly linked to a formal identifier of the natural or legal person.

- ***DID Document:*** DIDs resolve to DID Documents, i.e. a set of simple documents that contains information associated to a DID and describes how to use that specific DID. Each DID Document may contain at least three information such as proof purposes, verification methods (such as cryptographic public keys), and service endpoints (can also indicate services relevant to interactions with the DID holder). Proof purposes are combined with verification methods to provide mechanisms for proving various aspects (i.e. related to identification, authentication and authorization). For example, a DID Document can specify that a particular verification method, such as a cryptographic public key or pseudonymous biometric protocol, can be used to verify a proof that was created for the purpose of authentication. Service endpoints enable trusted interactions with the DID controller. A DID document may be signed by a DID holder (being the DID controller) or a different DID controller (e.g. In an organization an employee can be the DID holder and the employer/or any entity from the employer side can be the DID Controller. In another case an IoT object can be the DID holder and an operator's device at the factory floor who controls the IoT object can be the DID controller).

- ***Applications at end-device:*** Application (e.g. a client application or wallet) used by the ID holder to generate, manage, store, or use private and public key pairs. The sensitive information (such as cryptographic materials) may need to be protected by the "secure element" within the device or wallet. The use of the cryptographic keys is restricted to the DID holder.

DID Controller related aspects:

- The controller of a DID is the entity (person, organization, or autonomous software) that has the capability as defined by a DID method and indicated in the DID document to make any changes to a DID document. A DID holder can be the DID controller or a DID controller can be a different entity as authorized by the DID holder. DID controller actually have the proof of possession or control of the holder's private key and will be responsible for issuance of a unique and anonymous DID to the holder.

VC Issuer related aspects:

- ***VC Issuer** is a* role an entity (e.g. a trust entity or a trust service provider) can perform by asserting claims about one or more subjects, creating a verifiable credential from these claims, and transmitting the verifiable credential to a holder. Trust on the issuer is established either by trusting the issuer's DID (e.g. out-of-band, bilateral relationship, trusted lists) or by any other means. The third party can then use the presented cryptographically protected proof to verify the ownership and trustworthiness of the claims about the subject.

- ***Verifiable Credentials (VC)*** includes a set of one or more claims made by an issuer for the DID holder (i.e. subject). A verifiable credential is a tamper-evident credential that has authorship that can be cryptographically verified. As DIDs are just an identifier, they do not provide information about the subject itself. In practice, DIDs are used in combination with VC to support digital interactions in which information about the subject will be shared with third parties, by proving to those third parties that the DID subject has ownership of certain attestations or attributes. This proof is based on the cryptographic link between the VC, the DID subject the VC is about, and the issuer of the VC, which can be the own DID subject (self-asserted claims), or a trusted entity.

DID related data storage aspects:

- ***Ledgers (A DLT System):*** The PDL services can facilitate for the repository of DID related data such as DID documents, verifiable credentials, etc. The ledgers which store the DID related data should be considered as a form of Secure Area (e.g. SA-Application). The storage of DID can be supported through use of an agent service (such as PDL platform service if a distributed ledger is implemented for the storage) to remotely access the data from the user's device and controlled through multiple authentication and authorization factors.

- ***DID Registry for DID Resolver function:*** DIDs can be resolvable to their corresponding DID documents, where the DIDs are typically recorded on an underlying system or network of some kind. Regardless of the specific technology used, a DID Resolver function can be offered by any system that supports recording DIDs and returning data necessary to produce DID documents. The DID registry can be based on a distributed ledger (e.g. permissioned ones).

- ***VC Registry:*** To enable usage of verifiable credentials, the system that implements VC registry may perform mediation service for the creation and verification of identifiers, keys, and other relevant data, such as verifiable credential schemas, revocation registries, issuer public keys, and so on. Some configurations might require correlation of identifiers for subjects. Some registries, such as ones for identifiers and public keys, might just act as namespaces for identifiers.

- ***Off-chain Storage:*** The privacy sensitive data associated to the DID can be stored and managed in the off-chain or using any local/external authorized storage space.

DID Verifier related aspects:

- DID Verifier is a role that any service provider or application server would perform to identify and authenticate the DID holder using the trust management framework.

Identification and Authentication related aspects:

- The DID holder presents the data derived from one or more verifiable credentials, issued by one or more VC issuers, with a specific verifier to request and receive specific service of interest to the DID holder. A verifiable presentation is a tamper resistant/evident presentation encoded (with cryptographic methods) in such a way that authorship of the data can be trusted after a process of cryptographic verification. The DID holder authentication is facilitated with protocol exchanges between the DID holder, DID verifier and the trust framework to verify the DID and validate the VCs (as part of authentication) to check if that can be sufficient to provide a requested service (i.e. resource access) for the DID holder.

5.2 Threat Model and key issue analysis

The clause describes the various issues that need to be handled in a PDL based Trust management framework for decentralized identification, related data handling, and operational management.

DID Operation Participant(s) Management:

- The DID based identification and authentication involves several actors such as an Identity holder, Identity controller (i.e. which can be another entity than the subject itself in dependent cases), VC Issuer, and ID verifier to be part of a PDL based trust management framework to perform various purpose such as DID creation, data management, VC provision, DID verification and authentication respectively. If any of the actor deviates from their role of action, (due to an error or malicious intensions), it largely impacts the credibility of various process involved by the other actors. For example, if an end-user device which gains registration to the basic PDL platform services, escalates its access to impersonate like a VC issuer, then any illegitimate operations (e.g. storing of corrupted data) over the managed VCs/DID documents can lead to denial of service for genuine end-devices.

- Each participant in a trust management framework should be registered based on their role to allow only operations specific to the roles and to prevent illegitimate role switching.

Data management (for DIDs, DID documents, VCs):

- The sensitivity of any data depends on the type of the data and the level of information that is exposed. Even if multiple data are collectively required to perform an operation (e.g. identification and authentication), the sensitivity of any data may not remain the same for the entire set of data, so storage and management of all data sets related to an operation in a same ledger or data store can impact applying selective security measures for data that needs more confidentiality and access control. For the case of DID based identification and authentication, there are multiple set of data that are involved and required to be managed using a PDL platform, where the different data includes, DIDs, DID document storage information, actual DID documents, VCs, and any other data as required by the trust management framework governance.

- Based on the sensitivity of different DID related data, the PDL based trust management framework should facilitate required storage, access control and data management (such as create, update, revoke/delete).

DID verification and data sharing:

- The DID verifier (e.g. any service provider) can utilize the trust management framework, to query and access data (e.g. VCs or data) to perform DID verification and DID holder authentication. In such case, the trust management framework which manages all data related to the ID holder (i.e. end-device or end-user) is required to share selective data specific to the service as configured by the ID holder, DID controller (if any) and/or VC Issuers. As a claim will be dependent on a credentials associated to it, if any of the VCs associated to the end-users have limited validity period (e.g. a resident claim which was derived based on a passport that expires at some point of time), then the dependent claim should have the same validity, where the trust management framework need to manage the DID related data validity considering all the involved dependency factors (e.g. implicit data validity, explicit data revocation, etc.). The trust management framework should also be resistant and resilient to any masquerade attack (i.e. an entity having access to the basic PDL platform service claiming as a DID verifier to access the end-device related data, should be sufficiently verified before exposing any end-user data to prevent masquerade attacks). Further the trust management framework may need to manage one or more DIDs and related data for any single ID holder related to different services, in this case lack of sufficient control over DID specific DID document exposure and service specific VC related selective data exposure can lead to privacy violations (e.g. for a single ID holder, if two service providers need to offer different services Travel-X and Sport-Y, then the service provider who offers Travel-X related service should be exposed with VCs specific to the Travel-X related DID associated to the ID holder and no Sport-Y service related VCs or any data should be exposed.

6 Opportunities, Use Cases and scenarios of DID usage

6.1 Introduction to opportunities, use cases and scenarios

Sharing an identity of an end-user or device always matters, e.g. when a person needs to open a back account, while renting an infrastructure/vehicle, subscribing to a new service (such as telecom service), onboarding/Contracting to jobs, etc. Considering the sensitivity (i.e. privacy and confidentiality) of the data exposed during the identity verification process and related operational cost, any digital/electronic service related use-case can utilize a PDL based trust management framework to establish an initial trust between the service provider(s) and the service consumer(s) (i.e. end-devices) and to enable digital/electronic identification and authentication for digital service provision. Any end-device which need to consume a digital service can then simply use a configured application (e.g. a browser or mobile wallet application) to request any service with digital identity verification to prove the identity or any entitlements quickly and reliably. The PDL based trust management framework can be leveraged to store electronic forms of identification and other official documents (driver's license, prescriptions, diplomas etc) safely, which such information can be provided by trusted sources. The users can be able to decide how much data they want to share for a specific purpose related to different services. Few example use-case(s) which can utilize PDL-based trust management framework to set up the initial trust can include (but not limited to) the following:

1) On-boarding a customer to a digital service.

2) Signing a business contract.

3) Admission to educational institutions.

4) Opening a bank account.

5) Filing a tax return.

6) Applying for a university education.

7) Age restricted services.

8) Renting a vehicle.

9) Checking in to a hotel.

10) Requesting access to a public record such as birth certificate, medical record or land registry, etc.

6.2 Use case 1: Web3

Web3 or Web 3.0 refers to the next generation of the internet, where the internet users can take control of their data enabled with decentralization, blockchain and zero knowledge proof. Earlier in the Web 1.0 the internet users were the content consumers, where the internet was developed without a native identity layer for the service consumers. In Web 2.0, the internet users performed both content consumption and content production using their respective social media accounts. The concept of digital identity was relegated to websites and applications, where users do not actually own their online identity. Instead, the internet and application service users rent their identity from companies and centralized entities, where their digital identity is prone to the risk of being hacked, manipulated, censored, or simply lost. Where the siloed approach may have been appropriate for the early days of the internet, but with the billions of people now online and large number of subscriptions to multitude of different services, managing different identity, credentials, data, and its security as well as privacy handling becomes more challenging (resulting in inferior user experience), also its drawbacks and limitations are becoming more apparent. Usernames and passwords continue to be the dominant paradigm, despite being repeatedly demonstrated to be an insecure model. In this evolution, ownership and control of identity becomes the core of web 3.0, where each service consumer or service producer (e.g. a person) should have complete control over who has access to their data. The users need to be empowered to provide access grant, grant modification, or revoke access at any time, as well as have a unified view of all the data they share. Further there requires a need to identify and verify any device that's connected to the Internet (e.g. by utilizing blockchain). Blockchain's ability to security identity, build trust, automate, and keep transactions accountable via a secure shared ledger and smart contracts has the potential to enable new and interesting use cases over web3, like an autonomous vehicle that can authenticate its driver [i.15].

One of the key opportunity Web3 presents in the identity space is the ability to interact with a user's blockchain data which presents two main benefits such as:

i) enriching user profiles; and

ii) streamlining the login process with decentralized identification and verifiable credentials authentication (e.g. with federated logins using storage wallets).

Organizations can collect blockchain data from users as they interact with their applications, storing it in a unique portable user profile that any organization can use. Once sufficient data is committed to the chain, organizations can infer individual user preferences or make fundamental changes to their applications based on broader user behaviour and preferences. While this is possible in the Web2 world, the decentralized approach removes the need for siloes. Within Web3, the data control belongs to the person who is the address holder on the blockchain.

6.3 Use case 2: Telecom Service

Identity management plays an important role, when comes to subscriber enrolment for various type of network service access such long-term service, temporary/short-term service, on-demand localized service (e.g. offering service in a stadium, large trade event), etc. Across more than 140 countries, mobile network operators are subject to mandatory SIM registration obligations such as verification of customer's recognized identity credentials to do SIM card/e-SIM activation. Know-Your-Customer (KYC) regulations need customer's identity related document to be verified (e.g. such as any of passport, driving license, government issued identity card, etc.). The KYC process can be bit expensive, time-consuming, and sometimes challenging for service providers, when the operator requires to validate the customers identity credentials against any government database and each validation request incurs a fee. Meanwhile KYC process also involves other costs associated to agent commissioning, back-office procedures, and database management (e.g. these can be part of operational cost). Any incident of identity fraud or misuse of identity documents can lead to heavy penalty and impact to reputation [i.8]. In this regard, there is an opportunity for the network operators, policy makers, service providers, vendors and other partners to develop and adopt a more suitable alternative KYC process which can depend on DLT/PDL platform to facilitate building the initial trust between the subscribers and the service providers, i.e. PDL can allow users to establish their own trusted DID/SSI and manage the identity related credentials in a secure and privacy protected manner, which can lower the cost of new subscriber enrolment, costs associated with traditional KYC/SIM registration process and can adhere to General Data Protection Regulation (GDPR). The PDL system based decentralized identification has very wide benefits such as authenticity, privacy enabled with data integrity proof, provenance, blinding (i.e. to be anonymous), ZKP (i.e. service specific limited and required information disclosure), resiliency, and helps to establish trust in a trustless digital world. Moreover, based on the service requirement, the PDL system-based identity management service can allow user to control and take ownership of the identity attributes and ensures immutability of digital identity related operations. A more suitable identity service example can be GSMA Mobile Connect [i.9].

7 Architectural functionalities and considerations for Decentralized Identification and Trust management framework

7.1 Introduction

The basic trust management model for decentralized identification described in clause 5.1 can consider the threat model and key issues discussed in clause 5.2 to leverage the core features of distributed ledger enabled identification and trust management system to enable resilient access control and data management with the functionalities and services listed below. The functionalities to be considered for the decentralized identification and trust management listed below are described in detail in clause 7.2:

- Role-based registration management service;
- DID Operational participants Registry service;
- DID Registry/DID Resolver service;
- DID Document Registry service;
- VC Data Registry service; and
- DID Verification management service.

7.2 DID framework and functionalities

7.2.1 General discussion of the DID system

A Decentralized Identification and Trust management framework can utilize the PDL services described in the PDL reference architecture [i.10] and the following DID management and operation specific PDL services as shown in Figure 7.2.1-1.

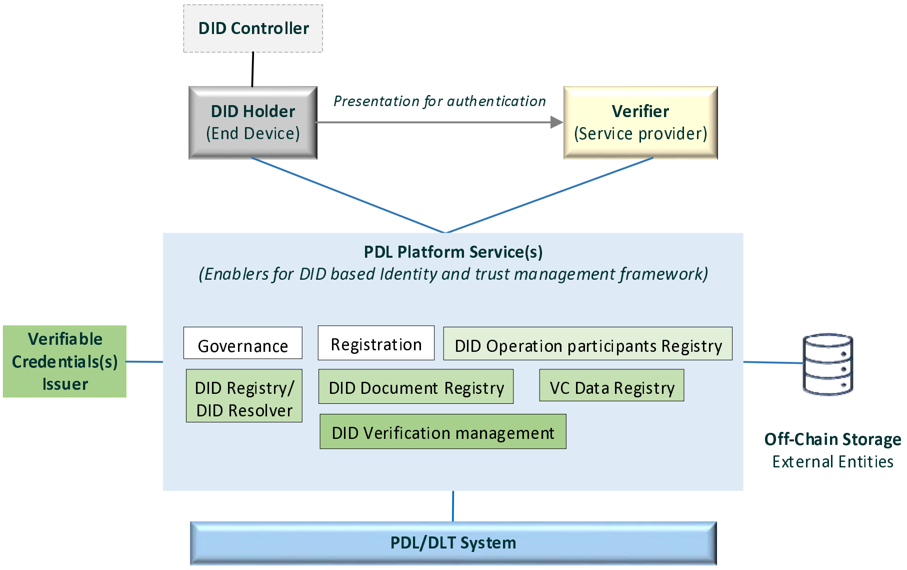

Figure 7.2.1-1: PDL platform-based Identity and Trust management model for decentralized identification

7.2.2 Role-based registration management service

The role-based registration management service considers the following different roles/actors/participants to be involved in the identity and trust management framework, and it provides registration service (along with authorization) specific to the corresponding roles of the actor in the PDL platform. The service operation can involve registration, revocation or de-registration respectively:

- identity holder;
- identity controller;
- VC Issuer;
- ID Verifier; and
- any other role as needed for a service (e.g. any participant/stakeholder to be involved in the identity and trust management framework).

7.2.3 DID Operational participants Registry service

The DID Operation(al) participants registry service records and keeps track of the registered and de-registered identity and trust management framework participants in the PDL platform based on instructions from the Role based registration management service.

7.2.4 DID Registry/DID Resolver service

The DID Registry/DID Resolver service stores and keeps track of the DID(s) and its associated DID document location information (e.g. address) to enable DID document fetching and verification by the authorized services and entities.

7.2.5 DID Document Registry service

The DID Document Registry service can store and manage the DID documents associated to the DID to facilitate DID verification. Whereas each DID Document can contain at least three things: proof purposes, service specific information for which the DID document can be used, verification methods, and service endpoints. Proof purposes are combined with verification methods to provide mechanisms for proving things. For example, a DID Document can specify that a particular verification method, such as a cryptographic public key or pseudonymous biometric protocol, can be used to verify a proof that was created for the purpose of authentication. Service endpoints enable trusted interactions with the DID controller as well as authorized verifier. The service operation can involve Create/store, Update, Delete/Revoke DID documents respectively.

7.2.6 VC Data Registry service

The VC Registry service can store and manage the VCs associated to the DID to facilitate VC based DID verification and validation related to service request. The service operation can involve Create/store, Update, Delete/Revoke VCs respectively.

7.2.7 DID Verification management service

The DID verification service can be a composite service that uses DID registry service/DID resolver service, DID document registry service and DID operation(al) participant registry service to fetch necessary data related to verification of DID (i.e. authentication of the subject identified by the DID), and exposure of selective data to the verifier to enable authorization verification of subject to respective service(s).

8 PDL services for Decentralized Identification and Trust Management

8.1 Introduction

This clause describes how the DID management and operation specific PDL services can be used to facilitate decentralized Identification and trust management.

8.2 Role based Registration management

8.2.1 Registration of DID Operation participants to a PDL platform

The role-based registration procedure involving a role-based registration management service and DID Operation(al) participants registry service is shown in Figure 8.2.1-1 (e.g. for registration related to DID holder, DID Controller, VC Issuer, ID Verifier, etc.) and the respective steps are described below in clauses 8.2.2 to 8.2.5 respectively.

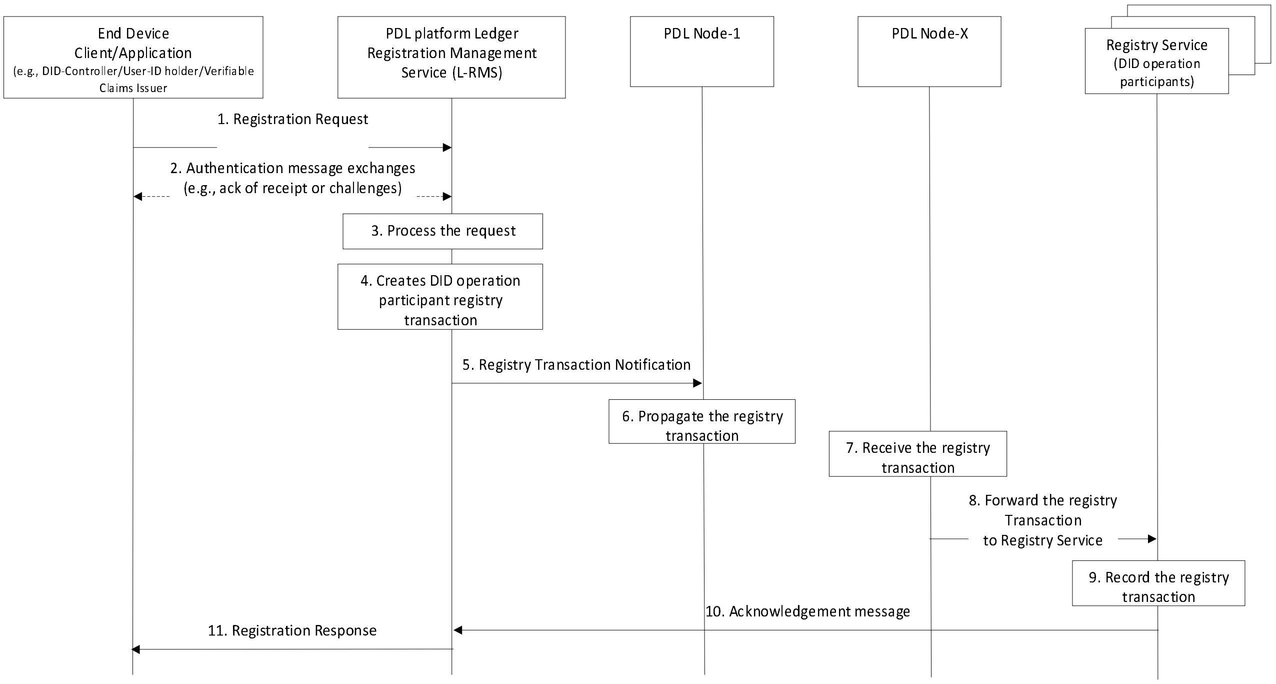

Figure 8.2.1-1: Role based registration procedure

8.2.2 DID holder registration

Precondition: The end-device i.e. ID holder may have registered to the PDL platform as a general user (e.g. using ETSI GS PDL 012 [i.10]) of the PDL platform, in which case the ID holder may have a source identity:

1) The ID holder sends to the PDL platform ledger role-based registration management service (L-RMS) a registration request, which can include a source identity, service type information (i.e. as DID service, to indicate that the registration is related to the DID end-device to act as the DID based ID holder in the identity management framework), access role (indicates that the Id holder role is requested) and the DID (i.e. a digital identity based on DID or SSI generated for the ID holder either by the subject or by the service provider and provisioned to the ID holder).

2) The L-RMS can initiate and perform mutual authentication (e.g. based on local policy) with the ID holder based on any preconfigured credentials (e.g. certificates or public-private key pair or any secret key) or any local policies.

3) On a successful mutual authentication, the L-RMS process the registration request.

4) The L-RMS determines to register the ID holder and it sets a registration ID for the ID holder. Further it creates a Registry transaction notification message which includes the L-RMS ID, target Registry service information (i.e. such as registry service name, ID or address related to the DID Operation(al) participant registry), Source Identity, Service type information (DID service), Registration ID, DID, Authorized access role (set as ID holder), Authorization code, and Lifetime (for the validity of the registration). Further the message can be transformed into a transaction (i.e. DID Operation(al) participant registry transaction) to add the new participant to the registry (i.e. called as DID Operation(al) participant registry).

5) The L-RMS sends to the configured PDL node a DID Operation(al) participant registry transaction (which includes the Registry transaction notification message).

6) PDL Node-1 propagates the received transaction through the target PDL network.

7) PDL Node-X (e.g. any PDL Node-2) receives the transaction from the target PDL network as the result of transaction propagation.

8) After the transaction is validated and it is successfully stored to the ledger (e.g. as a result of PDL consensus process in a ledger related to the registry service associated to the DID Operation(al) participant registry). Also, the PDL Node-2 forwards the transaction to the registry service based on the target Registry service information. The registry service transforms the transaction into message to recover the message (i.e. DID Operation(al) participant registry transaction as a registry transaction notification message).

9) The registry service can store the DID Operation(al) participant registry transaction received as part of the Registry transaction notification message based on local policies, e.g. in a local storage/off-chain/ledger.

10) The registry service can send to L-RMS, an acknowledgement message with the L-RMS ID, Source ID, Registration ID, Registry service type/ID, and the result as Success.

11) The L-RMS sends to the end device, a Registration Response with L-RMS ID, Service type information (DID service), Registration ID, Authorized access role, Authorization information (e.g. code/token), and Lifetime.

Based on step 1-2, if the L-RMS determines not to register the end device or application, it sends a Registration Response with failure.

8.2.3 Adaptations for the Identity (i.e. DID) Controller Registration

The message flow shown in Figure 8.2.1-1 can be used for the Identity controller registration to the Identity and trust management framework of the PDL platform, where the steps description can be applicable with the following adaption specific to the Identity controller (instead of the ID holder/subject).

Step 1: The DID controller sends to the L-RMS a registration request, with the required access role set as, "ID/DID Controller", Source Identity of the DID controller and the ID holder's source ID along with the other information described for step 1 above.

Steps 2 to 3: Same as above.

Step 4: The L-RMS determines to register the DID controller and it sets a registration ID for the DID controller. Further it creates Registry transaction notification message which includes the L-RMS ID, target Registry service information (i.e. such as registry service name, ID or address related to the DID Operation(al) participant registry), Source Identity of the DID controller, the ID holder's source ID, Service type information (DID service), Registration ID, DID, Authorized access role (set as DID controller), Authorization code, and Lifetime (for the validity of the registration). Further the message can be transformed into a transaction (i.e. DID Operation(al) participant registry transaction) to add the new participant to the registry (i.e. called as DID Operation(al) participant registry).

Steps 5 to 11: Same as above.

8.2.4 Adaptations for the VC Issuer Registration

The message flow shown in Figure 8.2.1-1 can be used for the VC Issuer registration to the Identity and trust management framework of the PDL platform, where the steps descriptions can be applicable with the following adaption specific to the VC Issuer (instead of the ID holder/subject).

Step 1: The VC Issuer sends to the L-RMS a registration request where the required access role is set as, "VC Issuer", and Source Identity of the VC Issuer and the ID holder's source ID is also included with the other information as described for step 1 above.

Steps 2 to 3: Same as above.

Step 4: The L-RMS determines to register the VC Issuer and it sets a registration ID for the VC Issuer. Further it creates Registry transaction notification message which includes the L-RMS ID, target Registry service information (i.e. such as registry service name, ID or address related to the DID Operation(al) participant registry), Source Identity of the VC Issuer, the ID holder's source ID, Service type information (DID service), Registration ID, DID, Authorized access role (set as VC Issuer), Authorization code, and Lifetime (for the validity of the registration). Further the message can be transformed into a transaction (i.e. DID Operation(al) participant registry transaction) to add the new participant to the registry (i.e. called as DID Operation(al) participant registry).

Steps 5 to 11: Same as above.

8.2.5 Adaptations for the Identity (i.e. DID) Verifier Registration

The message flow shown in Figure 8.2.1-1 can be used for the DID Verifier registration to the Identity and trust management framework of the PDL platform, where the steps description can be applicable with the following adaption specific to the DID Verifier (instead of the ID holder/subject).

Step 1: The DID Verifier sends to the L-RMS a registration request where the required access role is set as, "DID Verifier", and Source Identity of the DID Verifier is also included with the other information as described for step 1 above.

Steps 2 to 3: Same as above.

Step 4: The L-RMS determines to register the DID Verifier and so it sets a registration ID for the DID Verifier. Further it creates Registry transaction notification message which includes the L-RMS ID, target Registry service information (i.e. such as registry service name, ID or address related to the DID Operation(al) participant registry), Source Identity of the DID Verifier, Service type information (DID service), Registration ID, DID, Authorized access role (set as DID Verifier), Authorization code, and Lifetime (for the validity of the registration). Further the message can be transformed into a transaction (i.e. DID Operation(al) participant registry transaction) to add the new participant to the registry (i.e. called as DID Operation(al) participant registry).

Steps 5 to 11: Same as above.

NOTE 1: The L-RMS can accept the required access role provided by the end-device/client/applicant (in step 1) based on the authentication results (e.g. based on end-user's information in the certificate or any SLA agreement which is outside the scope of the present document). So, based on the local policies, authentication credentials evaluation and authentication result the L-RMS can determine to agree or deny a required access role requested by the end-device/application.

NOTE 2: A smart contracts can be used by the registry services to keep track of lifetime related expirations, linking of all DID related entries, etc.

8.3 Deregistration of DID Operation participants from a PDL platform

The revocation of any registration of any participant to a PDL platform-based identity and trust management system can be defined as deregistration. This clause describes how an end-user client/application (i.e. related to different registered participants such as Identity Holder, Identity Controller, VC Issuer, ID Verifier) can be deregistered from the PDL platform's Identity and trust management system as shown in Figure 8.3-1. The deregistration is invoked when one or more of the following various cases are identified:

i) the registry service identifies that a role-based registration has expired based on the lifetime of the registration e.g. registration is valid for, say, a month by default, and therefore the registration expires automatically when a month is lapsed;

ii) similar to option i), but the registration expiry is identified by a smart contract designed for the same purpose;

iii) similar to option i) and ii), but the registration expiry is identified by the governance;

iv) the governance identifies that the registered participant has violated any operations based on local policy;

v) the L-RMS identifies that the registered participant has violated any operations or misbehaves based on local policy.

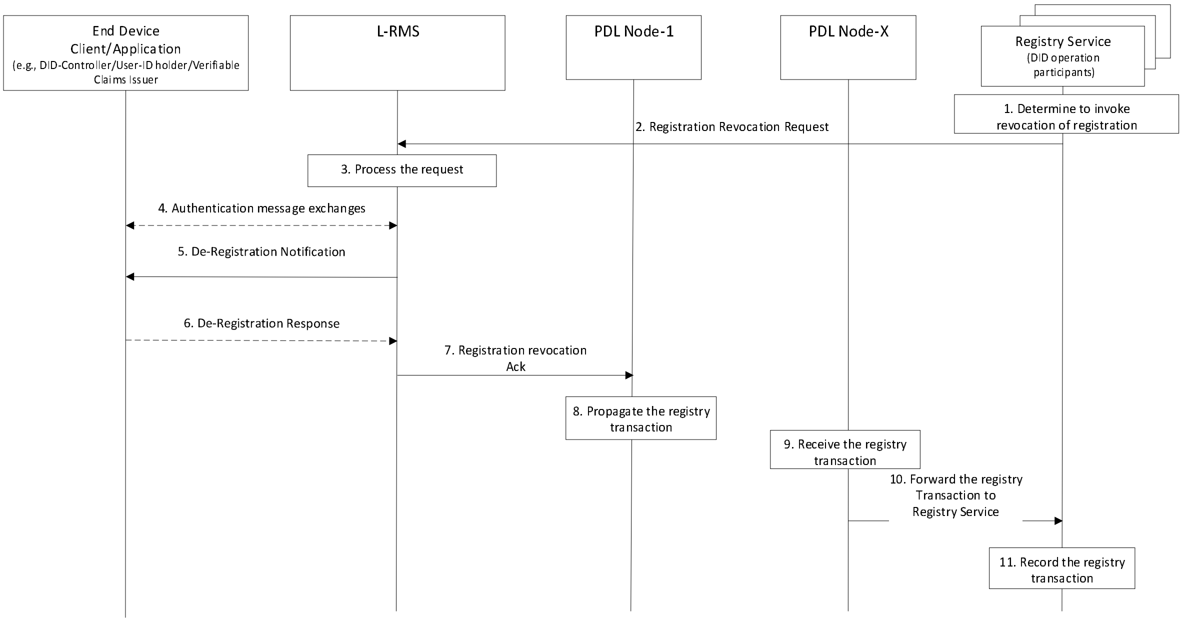
End Device Client/Application (e.g., DID-Controller/User-ID holder/Verifiable Claims Issuer
L-RMS
PDL Node-1
PDL Node-X
Registry Service (DID operation participants)
1. Determine to invoke revocation of registration
2. Registration Revocation Request
3. Process the request
4. Authentication message exchanges
5. De-Registration Notification
6. De-Registration Response
7. Registration revocation Ack
8. Propagate the registry transaction
9. Receive the registry transaction
10. Forward the registry Transaction to Registry Service
11. Record the registry transaction

Figure 8.3-1: De-registration procedure (Option 1 L-RMS is not a PDL node, where PDL node X acts as a proxy)

The message flow shown in Figure 8.3-1 is described below:

Step 1: The DID operation participants registry service can determine to revoke the registration for the registered participants.

Step 2: The Registry service sends to the L-RMS the Registration revocation request based on the L-RMS ID associated for the Registration ID related to the DID Operation(al) participant registry information.

The registration revocation request can include L-RMS ID, Registry service information, Registration ID, authorized access role and a cause value related to the registration revocation (e.g. Lifetime expiry, error, any operational violation reason code).

Step 3: The L-RMS on receiving the registration revocation request for any registered participant associated with its L-RMS ID, the L-RMS can process and determine to invoke the registration revocation.

Step 4: Based on local policy the L-RMS can perform mutual authentication with the end-device to authenticate and set up a secure connection.

Step 5: The L-RMS can send to the end-device, a de-registration notification message which can include the Service type information (DID service), Registration ID, and Cause value.

Step 6: The end-device can send to the L-RMS, the de-registration response message with Registration ID and successful registration revocation acknowledgement indication. Further the end-device can delete all information associated to the registration such as registration ID, and authorization information.

Step 7: The L-RMS can create a registration revocation acknowledgement message and coverts it to a transaction (i.e. registration revocation ack notification) and sends to the PDL Node-1 based on the local configuration. The registration revocation ack notification can include the information related to the Registry transaction such as Registry service information (i.e. type/ID), L-RMS ID, Source Identity, Service type information (DID service), Registration ID, and Revoked Indication (e.g. revocation successful indication).

If the L-RMS does not receive any de-registration response from the end-device in step 6, based on local policy (e.g. after a preconfigured waiting time), it can perform step 7.

Step 8: PDL Node-1 propagates the received transaction through the target PDL network.

Step 9: PDL Node-X receives the transaction from the target PDL network as the result of transaction propagation.

Step 10: After the transaction is validated, it is successfully stored to the ledger (e.g. as a result of PDL consensus process in a ledger related to the registry service) associated to the DID Operation(al) participant registry based on the target registry service type information. Also, the PDL Node-X forwards the transaction to the registry service and the registry service transforms the transaction into message to recover the message (i.e. registration revocation acknowledgement message).

Step 11: The registry service can store the registration revocation ack notification based on local policies in the local storage/off-chain/ledger.

NOTE 1: Irrespective of the roles, any end-device/application related to an ID holder/Identity Controller/VC Issuer/ID Verifier associated registration can be revoked using the procedure described in the Figure 8.3-1 by providing the corresponding access role information in step 2.

NOTE 2: Smart contracts can be configured to link and maintain the registration status (such as successful registration and respective revocations) related to a registration ID. Further the smart contracts can be used by the registry services to keep track of lifetime related expirations, linking of all DID related entries, etc.

Another option of deregistration procedure with L-RMS as a PDL Node:

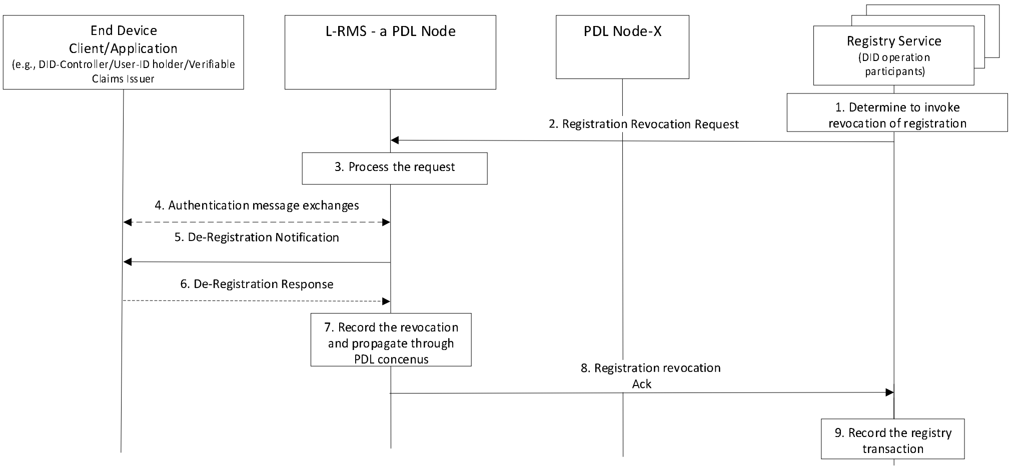
End Device
Client/Application
(e.g., DID-Controller/User-ID holder/Verifiable Claims Issuer
L-RMS - a PDL Node
PDL Node-X
Registry Service
(DID operation participants)
1. Determine to invoke revocation of registration
2. Registration Revocation Request
3. Process the request
4. Authentication message exchanges
5. De-Registration Notification
6. De-Registration Response
7. Record the revocation and propagate through PDL concenus
8. Registration revocation Ack
9. Record the registry transaction

Figure 8.3-2: De-registration procedure (Option 2 L-RMS is a PDL node by itself)

An alternative option for deregistration is shown in Figure 8.3-2 where the L-RMS is a PDL node by itself and the L-RMS can propagate the PDL transaction (related to the Registration revocation acknowledgement notification) to the PDL network by itself (as in step 7). Rest of the message flow description for steps 1 to 6 is same as Figure 8.3-1. Further the description of steps 10 and 11 of Figure 8.3-1 can be applied the steps 8 and 9 of Figure 8.3-2 respectively.

8.4 DID and DID documents management in PDL platform

8.4.1 General Procedure

1) This clause describes the process to manage the DID and DID documents in a PDL platform as shown in Figure 8.4.1-1. The management of DID and the related DID documents involves various operations such as listed below:

   - Storage of DID and DID Documents (i.e. on request from the DID holder or DID controller).
   - Update of DID and DID Documents (i.e. on request from the DID holder or DID controller).
   - Deletion/Revocation of DID and DID Documents (i.e. on request from the DID holder/DID controller/Ledger-registration management service in the PDL platform).

The general procedure to manage the DID and DID documents in a PDL platform is shown in Figure 8.4.1-1.

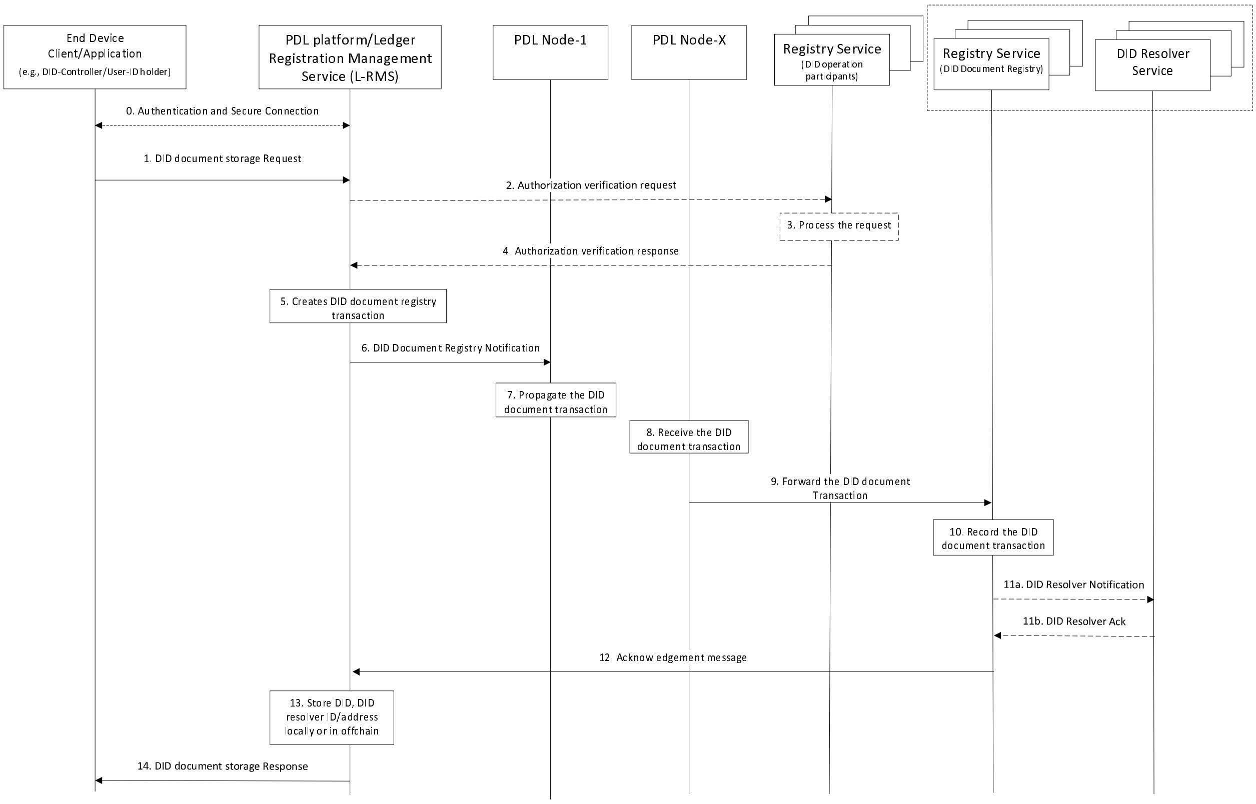
End Device Client/Application (e.g., DID-Controller/User-ID holder)
PDL platform/Ledger Registration Management Service (L-RMS)
PDL Node-1
PDL Node-X
Registry Service (DID operation participants)
Registry Service (DID Document Registry)
DID Resolver Service
0. Authentication and Secure Connection
1. DID document storage Request
2. Authorization verification request
3. Process the request
4. Authorization verification response
5. Creates DID document registry transaction
6. DID Document Registry Notification
7. Propagate the DID document transaction
8. Receive the DID document transaction
9. Forward the DID document Transaction
10. Record the DID document transaction
11a. DID Resolver Notification
11b. DID Resolver Ack
12. Acknowledgement message
13. Store DID, DID resolver ID/address locally or in offchain
14. DID document storage Response

Figure 8.4.1-1: DID and DID Document storage management in PDL platform

The steps shown is Figure 8.4.1-1 is described as follows:

Step 0: If a secure connection exists, the ID holder can send step 1. Else the ID holder and the L-RMS performs mutual authentication and sets up a secure connection before sending step 1. i.e. the authentication can be based on Security Platform Services defined in ETSI GS PDL 012 [i.10].

Step 1: The end device client/application can send to the L-RMS, a DID document storage request which can include Source Identity (i.e. a PDL user ID), Registration ID, service type information (i.e. it indicates a DID service), access role (i.e. indicated as ID/DID holder, which can be the subject/end-user), authorization information (i.e. a code or token received as authorization information during a successful registration), the DID document(s), request type (set as store/create).

The DID document(s) can include information such as the DID, DID controller ID (if applicable/exists), verification method(s), cryptographic public key, service type/info, verifiable claims, URI related to claims, etc.

Step 2: The L-RMS sends to the Registry service (related to the DID Operation(al) participants) based on the local configuration and policies, an authorization verification request message, which can include Registration ID, Source ID, Access role and authorization information.

Step 3: The Registry service verifies the authorization information related to the Registration ID and the access role by querying the respective ledger (for a related transaction history/records) or by checking an offline/local storage to check if the authorization information and registration ID matches with any of the records related to the registered participant. The Registry service also checks if the access role of the participant is correct based on the records.

Step 4: If the verification of the registration ID, access role and authorization information are successful, then the Registry service sends to the L-RMS, an authorization verification response message, which can include the registration ID, source ID and result as "successful".

Alternatively, if the verification of the registration ID, access role and authorization information do not match with the records, or if the registered access role is different from the one received in step 2, then the Registry service sends to the L-RMS, an authorization verification response message, which can include the registration ID, source ID, and result as "failure".

Step 5: The L-RMS if finds that the authorization verification is successful, then it generates a DID document registry notification message which can include the target Registry service type/ID, Create Indication*, L-RMS ID, Source Identity, Service type information (DID service), Registration ID, authorized access role, Lifetime, DID, and DID Document(s).

Further the message can be transformed into a DID document transaction to store the DID documents to the corresponding registry (i.e. called as DID document registry) indicated by the target Registry service type/ID.

Alternatively, for failure case described in step 4, then steps 5 to 13 are skipped, and step 14 is executed related to the failure case.

Step 6: The L-RMS sends to the configured PDL node-1, the DID document transaction (which includes the DID document registry notification message).

Step 7: PDL Node-1 propagates the received DID document transaction through the target PDL network.

Step 8: PDL Node-X (e.g. any PDL Node-2) receives the DID document transaction from the target PDL network as the result of transaction propagation.

Step 9: After the DID document transaction is validated, it is successfully stored to the ledger (e.g. as a result of PDL consensus process in a ledger related to the registry service associated to the DID Document (transaction) based on the target registry service type information). Also, the PDL Node-2 forwards the DID document transaction to the registry service (i.e. DID Document registry) based on the target Registry service information. The registry service transforms the transaction into message to recover the message (i.e. DID Document registry transaction as a DID document registry notification message).

Step 10: The DID document registry service can store the DID Document information received as part of the DID document Registry notification message in a local storage/off-chain/ledger.

Step 11a: The DID registry can send to a DID registry/DID resolver service, a notification message (i.e. a DID resolver notification message) which can include DID, request type (Create/store Indication)*, and DID Document Registry ID/address.

Step 11b: The DID resolver services stores (i.e. as a record) the DID and the DID Document Registry ID/address. Further the DID resolver service sends to the DID Document registry service, a DID resolver acknowledgement (Ack) message which can include DID and a success indication.

Step 12: The DID Document registry service sends to the L-RMS, an acknowledgement message which can include the L-RMS ID (received in step 9 as part of the transaction/message related to the DID document registry notification), Source Identity, Registration ID, Registry service type/ID, DID Document Registry address, Success, DID resolver registry service ID/address. Based on the implementation, the DID resolver ID can be an address of the DID resolver service.

Step 13: The L-RMS can store the DID, resolved ID locally or in off-chain.

Step 14: The L-RMS sends to the end device client/application, a DID document storage response message, which can include the DID resolver ID and Success.

NOTE 1: The end device client/application while requesting service from any service provider, it can provide the DID resolver ID to enable the service provider to request the DID resolver for respective authentication of the end-device, which is outside the scope of the present document. Alternatively, for the failure case described in steps 4 and 5, the L-RMS sends to the end device client/application, a DID document storage response message, which can include the failure indication with a suitable cause value (i.e. such as violation code/authorization failure/authentication failure, etc.

NOTE 2: Based on different implementation, in Figure 8.4.1-1, alternatively the L-RMS can be a PDL node by itself and the L-RMS can propagate the PDL transaction (related to the DID document related data storage management notification) to the PDL network by itself.

8.4.2 Adaptations for DID controller performing DID document storage for a DID holder

Figure 8.4.1-1, steps 1 to 14 can be applied with an additional adaptation that step 1, step 3, steps 4, 5, 6, 7, 8, 9, 10, 11, 12 also includes source Identity corresponding to the DID holder (i.e. subject) whose DID is being controlled by the DID controller in addition to the DID Controller ID. Further, the access role information specific to the DID controller need to be used for the storage procedure.

8.4.3 Adaptations for Update of DID and DID Documents (i.e. on request from the DID holder/DID controller)

Step 1: The end device client/application can send to the L-RMS, a DID document storage request which can include Source Identity (i.e. a PDL user ID), Registration ID, service type information (i.e. it indicates a DID service), access role (i.e. indicated as ID/DID holder, which can be the subject/end-user), authorization information (i.e. a code or token received as authorization information during a successful registration), the DID document(s) and the request type set as DID document(s) update indication.

Steps 2 to 5: Same as Figure 8.4.1-1 description.

Steps 6 to 7: DID document registry notification message (as well as the related transaction) includes, the request type set as "the DID document(s) update indication" (instead of create/store indication) in addition to the other information described for Figure 8.4.1-1.

Steps 8 to 10: Same as Figure 8.4.1-1 description.

Step 11a: The DID registry can send to a DID registry/DID resolver service, a notification message (i.e. a DID resolver notification message) which can include DID, the request type set as "the DID document(s) update indication" and DID Document Registry ID/address.

Step 11b: The DID resolver services updates the DID related DID Document Registry ID/address. Further the DID resolver service sends to the DID Document registry service, a DID resolver acknowledgement (Ack) message which can include DID and a success indication.

Steps 12 to 14: Same as Figure 8.4.1-1 description.

8.4.4 Adaptations for Deletion/Revocation of DID and DID Documents (i.e. on request from the DID holder/DID controller)

Step 1: The end device client/application can send to the L-RMS, a DID document storage request which can include Source Identity (i.e. a PDL user ID), Registration ID, service type information (i.e. it indicates a DID service), access role (i.e. indicated as ID/DID holder, which can be the subject/end-user), authorization information (i.e. a code or token received as authorization information during a successful registration), the DID and the request type set as DID document(s) revoke indication.

Steps 2 to 5: Same as Figure 8.4.1-1 description (but with DID instead of DID documents).

Steps 6 to 7: DID document registry notification message (as well as the related transaction) includes, the request type set as "the DID document(s) revoke indication" (instead of create/store indication) in addition to the other information described for Figure 8.4.1-1 (but with DID instead of DID documents).

Steps 8 to 10: Same as Figure 8.4.1-1 description.

Step 11a: The DID registry can send to a DID registry/DID resolver service, a notification message (i.e. a DID resolver notification message) which can include DID, the request type set as "the DID document(s) revoke indication" and DID Document Registry ID/address.

Step 11b: The DID resolver services revokes the DID related DID Document Registry ID/address. Further the DID resolver service sends to the DID Document registry service, a DID resolver acknowledgement (Ack) message which can include DID and a success indication.

Step 12: Same as Figure 8.4.1-1 description.

Step 13 Not needed.

Step 14: The L-RMS sends to the end device client/application, a DID document storage response message, which can include the Success indication.

8.5 Verifiable Credentials management in PDL platform

8.5.1 General Procedure

This clause describes the process to manage the Verifiable Credentials (VCs) in a PDL platform as shown in Figure 8.5.1-1. The VCs are described in clause 5.1 of the present document. The management of VCs involves various operations such as listed below:

- Storage of VC (i.e. on request from the DID holder/DID controller/VC Issuer)
- Update of VC (i.e. on request from the DID holder/DID controller/VC Issuer)
- Deletion/Revocation of VC (i.e. on request from the DID holder/DID controller/VC Issuer)

The general procedure to manage the VC storage in a PDL platform is shown in Figure 8.5.1-1.

The VC storage to a registry service in a PDL platform can be similar to the DID document storage management process described in clause 8.4.1, but in the case of VC storage, in addition to the respective DID holder/DID controller, it is very likely that an associated VC Issuer (e.g. any trust service provider can issue VC for the DID holder.

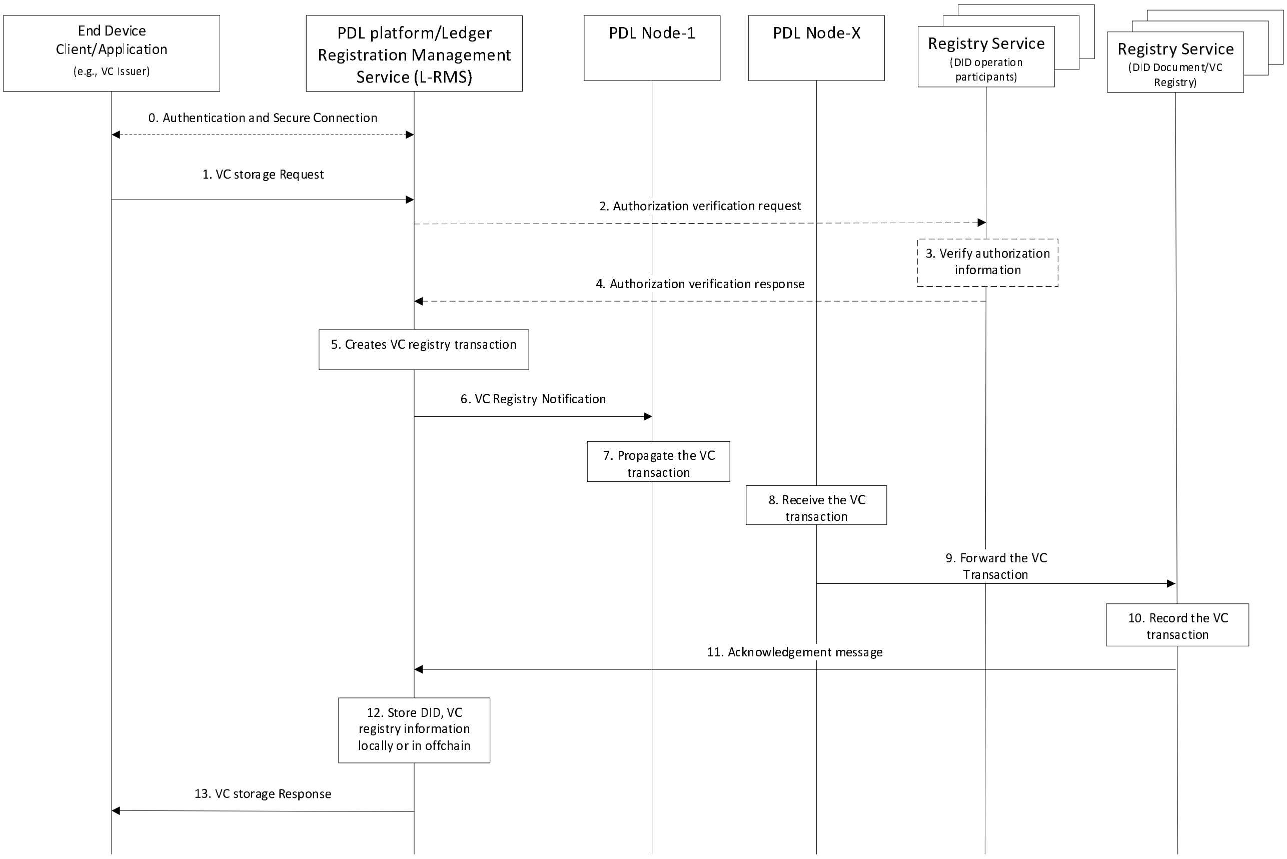
End Device Client/Application (e.g., VC Issuer)
PDL platform/Ledger Registration Management Service (L-RMS)
PDL Node-1
PDL Node-X
Registry Service (DID operation participants)
Registry Service (DID Document/VC Registry)
0. Authentication and Secure Connection
1. VC storage Request
2. Authorization verification request
3. Verify authorization information
4. Authorization verification response
5. Creates VC registry transaction
6. VC Registry Notification
7. Propagate the VC transaction
8. Receive the VC transaction
9. Forward the VC Transaction
10. Record the VC transaction
11. Acknowledgement message
12. Store DID, VC registry information locally or in offchain
13. VC storage Response

Figure 8.5.1-1: VC storage management in PDL platform

The steps shown is Figure 8.5.1-1 is described as follows:

Step 0: If a secure connection exists, the VC Issuer can send step 1. Else the VC Issuer and the L-RMS performs mutual authentication and sets up a secure connection before sending step 1. i.e. the authentication can be based on Security Platform Services defined in ETSI GS PDL 012. The VC Issuer can be a DID holder/DID Controller/Trust service provider of the DID holder.

Step 1: The end device client/application can send to the L-RMS, a VC storage request which can include Source Identity (i.e. a PDL user ID related to the VC Issuer), Registration ID (of the VC Issuer, service type information (i.e. it indicates a DID service), access role (i.e. indicated as VC Issuer, which can be the subject/end-user or a trust service provider), authorization information (i.e. a code or token received as authorization information during a successful registration), DID, VCs, and request type (set as store/create).

Step 2: The L-RMS sends to the Registry service (related to the DID Operation(al) participants) based on the local configuration and policies, an authorization verification request message, which can include Registration ID, Source ID, Access role and authorization information.

Step 3: The Registry service verifies the authorization information related to the Registration ID and the access role by querying the respective ledger (for a related transaction history/records) or by checking an offline/local storage to check if the authorization information and registration ID matches with any of the records related to the registered participant. The Registry service also checks if the access role of the participant is correct based on the records.

Step 4: If the verification of the registration ID, access role and authorization information are successful, then the Registry service sends to the L-RMS, an authorization verification response message, which can include the registration ID (of the VC Issuer), source ID and result as "successful".

Alternatively, if the verification of the registration ID, access role and authorization information do not match with the records, or if the registered access role is different from the one received in step 2, then the Registry service sends to the L-RMS, an authorization verification response message, which can include the registration ID, source ID, and result as "failure".

Step 5: The L-RMS if finds that the authorization verification is successful, then it generates a VC registry notification message which can include the target Registry service type/ID, Create Indication*, L-RMS ID, Source Identity, Service type information (DID service), Registration ID, authorized access role, Lifetime, DID, and VC(s).

Further the message can be transformed into a VC to store the VCs along with the respective DID to the corresponding registry (i.e. called as VC registry) indicated by the target Registry service type/ID.

NOTE 1: Based on implementation a DID document registry can also be used to storage a VC or a different VC registry can be managed to store and handle VCs for the DID(s). In case separate registries are maintained for the DID documents and VCs corresponding to a DID, then a smart contract can be implemented to keep track of the DID related DID documents and VC records in different registries.

Alternatively, for failure case described in step 4, then steps 5 to 12 are skipped and step 13 is executed related to the failure case.

Step 6: The L-RMS sends to the configured PDL node-1, the VC transaction (which includes the VC registry notification message).

Step 7: PDL Node-1 propagates the received VC transaction through the target PDL network.

Step 8: PDL Node-X (e.g. any PDL Node-2) receives the VC transaction from the target PDL network as the result of transaction propagation.

Step 9: After the VC transaction is validated, it is successfully stored to the ledger (e.g. as a result of PDL consensus process in a ledger related to the registry service associated to the VC (transaction) based on the target registry service type information). Also, the PDL Node-2 forwards the VC transaction to the registry service (i.e. VC registry) based on the target Registry service information. The registry service transforms the transaction into message to recover the message (i.e. VC registry transaction as a VC registry notification message).

Step 10: The VC registry service can store the VC transaction/information (i.e. DID and VC) received as part of the VC Registry notification message (or vice versa) in a local storage/off-chain/ledger.

Step 11: The VC registry service sends to the L-RMS, an acknowledgement message which can include the L-RMS ID (received in step 10 as part of the transaction/message related to the VC registry notification), Source Identity, Registration ID, Registry service type/ID, VC Registry address, and Success indication.

Step 12: The L-RMS can store and maintain the DID with related VC registry address/ID information locally or in off-chain.

Step 13: The L-RMS sends to the end device client/application, a VC storage response message, which can include the DID and Success indication.

Alternatively, for the failure case described in steps 4 and 5, the L-RMS sends to the end device client/application, a VC storage response message, which can include the failure indication with a suitable cause value (i.e. such as violation code/authorization failure/authentication failure, etc.

NOTE 2: Based on different implementation, in Figure 8.5.1-1, alternatively the L-RMS can be a PDL node by itself and the L-RMS can propagate the PDL transaction (related to the DID document related data storage management notification) to the PDL network by itself.

8.5.2 Adaptations for DID holder/DID controller performing VC storage for an DID holder/subject

Figure 8.5.1-1, steps 1 to 13 can be applied with an additional adaptation that step 1, step 3, steps 4, 5, 6, 7, 8, 9, 10, 11, 12 also includes source Identity corresponding to the DID holder (i.e. subject) whose DID is being controlled by the DID controller in addition to the DID Controller ID (if it exists). Further, the access role information specific to the DID holder or DID controller need to be used for the storage procedure respectively based on the type of involvement.

8.5.3 Adaptations for update of VCs (i.e. on request from the DID holder/DID controller/VC Issuer)

Step 1: The end device client/application can send to the L-RMS, a VC storage request which can include Source Identity (i.e. a PDL user ID related to the VC Issuer), Registration ID (of the VC Issuer, service type information (i.e. it indicates a DID service), access role (i.e. indicated as VC Issuer, which can be the subject/end-user or a trust service provider), authorization information (i.e. a code or token received as authorization information during a successful registration), DID, VCs, and request type set as VC(s) update indication.

NOTE: The VC issuer can be a DID holder/DID Controller (e.g. in case of self-asserted claims/VCs) (or) a different entity like Trust service provider related to the DID holder. In case the DID holder/DID Controller, takes the role of VC issuer, the access role can be indicated as ID holder/DID controller respectively. If the VC Issuer is a different entity like Trust service provider, then the access role can be indicated as VC Issuer.

Steps 2 to 5: Same as Figure 8.5.1-1 description.

Steps 6 to 7: VC registry notification message (as well as the related transaction) includes, "the request type set as VC(s) update indication" (instead of create/store indication) in addition to the other information described in Figure 8.5.1-1.

Steps 8 to 13: Same as Figure 8.5.1-1 description.

8.5.4 Adaptations for Deletion/Revocation of VC (i.e. on request from the DID holder/DID controller/VC Issuer)

Step 1: The end device client/application can send to the L-RMS, a VC storage request which can include Source Identity (i.e. a PDL user ID related to the VC Issuer), Registration ID (of the VC Issuer, service type information (i.e. it indicates a DID service), access role (i.e. indicated as VC Issuer, which can be the subject/end-user or a trust service provider), authorization information (i.e. a code or token received as authorization information during a successful registration), DID, VCs, and request type set as VC(s) revoke indication.

NOTE 1: The VC issuer can be a DID holder/DID Controller (e.g. in case of self-asserted claims/VCs) (or) a different entity like Trust service provider related to the DID holder. In case the DID holder/DID Controller, takes the role of VC issuer, the access role can be indicated as ID holder/DID controller respectively. If the VC Issuer is a different entity like Trust service provider, then the access role can be indicated as VC Issuer.

Steps 2 to 5: Same as Figure 8.5.1-1 description.

Steps 6 to 7: VC registry notification message (as well as the related transaction) includes, "the request type set as VC(s) revoke indication" (instead of create/store indication) in addition to the other information described in Figure 8.5.1-1.

Steps 8 to 13: Same as Figure 8.5.1-1 description.

NOTE 2: The smart contracts can be used by the registry services described in the VC storage management procedure to keep track of VC storage, update, and revoke related operations associated to the DID by linking all the DID related entries.

8.6 DID Verification management

This clause describes how a verifier (i.e. a DID verifier e.g. any service provider) can utilize the PDL service(s) to verify a DID related to an end-device/user (i.e. DID holder) to authenticate the end-device (which requests a service that is offered by the service provider) utilizing the ledger-based Identity and trust management framework shown in clause 7.2 of the present document. The ledger-based identity and trust management framework can offer a DID verification service to enable the DID verification and DID holder authentication as shown in Figure 8.6-1.

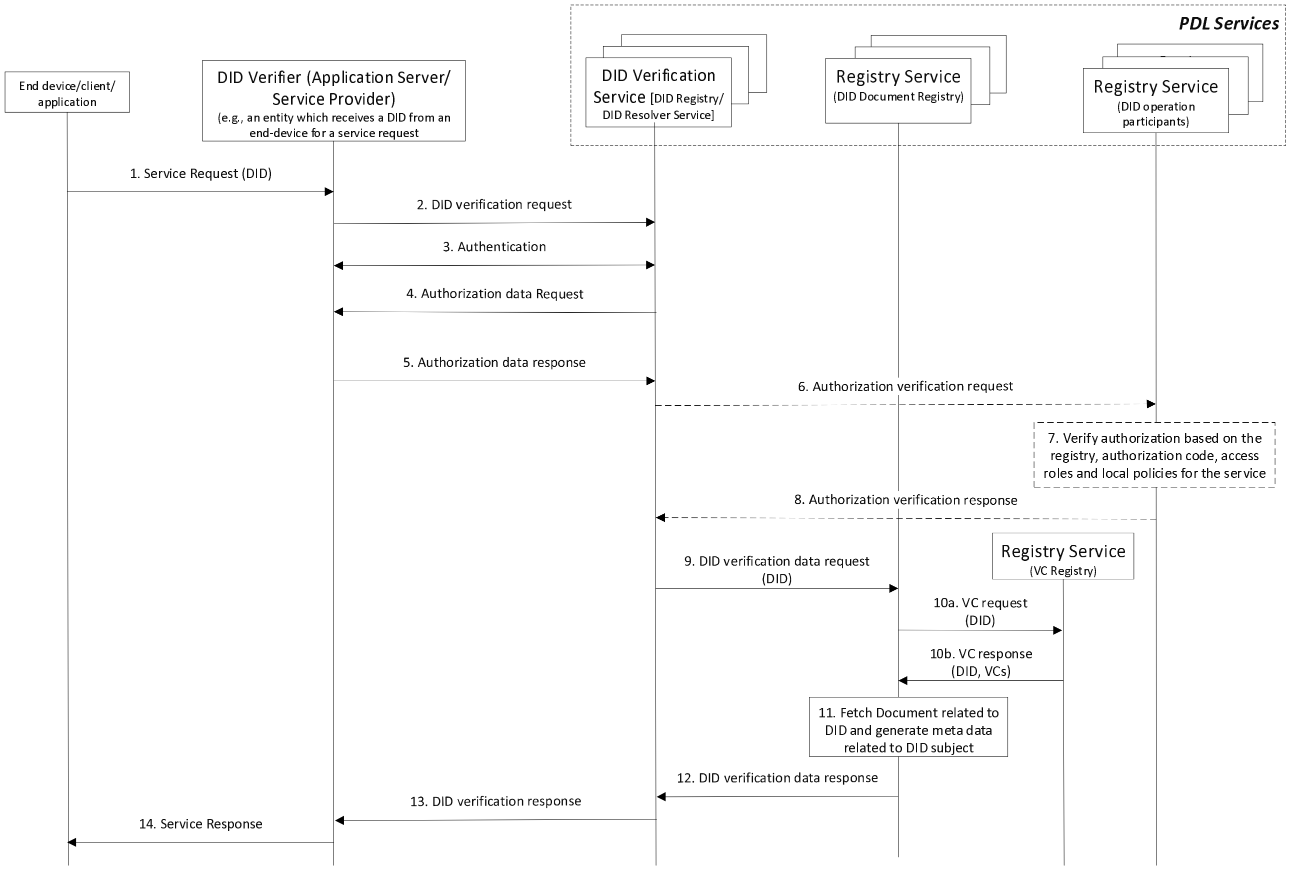
PDL Services
End device/client/ application
DID Verifier (Application Server/ Service Provider)
(e.g., an entity which receives a DID from an end-device for a service request
DID Verification Service [DID Registry/ DID Resolver Service]
Registry Service
(DID Document Registry)
Registry Service
(DID operation participants)
1. Service Request (DID)
2. DID verification request
3. Authentication
4. Authorization data Request
5. Authorization data response
6. Authorization verification request
7. Verify authorization based on the registry, authorization code, access roles and local policies for the service
8. Authorization verification response
Registry Service
(VC Registry)
9. DID verification data request (DID)
10a. VC request (DID)
10b. VC response (DID, VCs)
11. Fetch Document related to DID and generate meta data related to DID subject
12. DID verification data response
13. DID verification response
14. Service Response

Figure 8.6-1: DID verification and DID holder authentication using PDL based DID verification service

The steps shown in Figure 8.6-1 is described as follows.

As a precondition, the DID holder and the DID Verifier are registered to the PDL DID and Trust Management framework based on clause 8 of the present document to perform various PDL services as required.

Step 1: The end-device/client/application (i.e. a DID holder) sends to the DID Verifier (e.g. an application server/service provider application function, which can be an entity that receives a DID from an end-device in a service request (or any access request) for a service provision.

NOTE 1: The service request message that is being sent between the end-device and the DID verifier can be over any interface which is outside the scope of the present document. Steps 1 and 14 are operations which happens external to the PDL framework, and it is outside the scope of the present document. Only for illustrative purpose steps 1 and 14 are described here.

Step 2: The DID verifier can determine the DID Verification Service to be used based on the DID (e.g. realm information or information in the DID).

The DID verifier sends a DID verification request message to the respective DID verification service (in the ledger-based identity and trust management framework). The DID verification request message includes the DID verifier's source ID, DID, target DID service type (i.e. the type of service for which the DID is being associated to the DID holder and being verified by the DID verifier).

Step 3: If there is no secure connection exists between the DID Verifier and the DID verification service, the DID Verifier and the DID verification service can perform mutual authentication and sets up a secure connection i.e. the authentication can be based on Security Platform Services defined in ETSI GS PDL 012.

Step 4: The DID Verification service can send to the DID Verifier, an Authorization data request message with source identity.

Step 5: The DID Verifier can send to the DID Verification service, an Authorization data response message which includes its Registration ID and its corresponding authorization information ((e.g. can be an authorization code or token).

NOTE 2: The registration ID and the corresponding authorization information are the ones received by the DID verifier during its successful role-based registration with a L-RMS to utilize various PDL services).

Step 6: The DID Verification service sends to the DID Operation(al) participants Registry service, an Authorization verification request message, which can include the registration ID (of the DID verifier), Source identity (of the DID verifier), authorization information (e.g. can be an authorization code or token), access role "set as DID verifier", and the service type information (i.e. received in step 2).

Step 7: The Registry service verifies the authorization information related to the Registration ID and the access role by querying the respective ledger (for a related transaction record) or by checking an offline/local storage to check if the authorization information and registration ID matches with any of the records related to the registered participant.

Step 8: If the verification of the registration ID, access role and authorization information are successful, then the Registry service sends to the DID Verification Service, an authorization verification response message with the registration ID (of the Verifier) and result as "successful".

Alternatively, if the verification of the registration ID, access role and authorization information do not match with the records, then the Registry service sends to the DID Verification Service, an authorization verification response message, with the registration ID, and result as "failure". Further directly step 13 is performed related to the failure case.

Step 9: The DID Verification services invokes the DID resolver service (i.e. co-located) with the DID verification service and fetches the corresponding DID-related DID document registry service ID/address information. The DID verification service sends to the DID Document Registry service, a DID verification data request, which can include a DID.

Step 10a: The DID Document registry service checks if the DID document is available (e.g. in a ledger/chain) for the DID. Further if the DID document is available, the DID document service based on local configuration finds also the VC registry service ID/address and sends to the VC registry service, a VC request message, with the DID.

Step 10b: The VC Registry service fetches the VCs associated to the DID from the respective chain/ledger and sends to the DID document registry service, a VC response message which includes the DID and the associated VC(s).

Step 11: The DID Document Registry service fetches the DID document(s) related to the DID and generates the metadata from the VCs to enable the verifier to authenticate and authorize the DID as required for the service provision.

Step 12: The DID Document Registry service then sends to the DID Verification service, a DID verification data response, which includes the DID, DID documents, and the metadata (based on the VCs i.e. claims asserted related to the DID holder/subject respective to the service).

Step 13: The DID Verification service, can send to the DID Verifier, a DID Verification response message, which can include result (with successful indication), DID document, and metadata (related to VCs).

Alternatively, for the failure case operations, the DID Verification service sends to the DID verifier, a DID verification response message with result set as "failure indication", and cause information (i.e. such as violation code/authorization failure/authentication failure respectively).

Step 14: The DID Verifier, can use the DID documents to verify (e.g. integrity check or authenticate) the DID and authenticate the DID subject. Further the DID verifier can also use the metadata (based on VCs) associated to the DID subject to authenticate/authorize the DID subject specific to the requested service provision. If the verification of the DID, authentication of the DID subject and the VCs meets the service requirement criteria (e.g. the metadata based on the VCs can enable to authenticate the subject based on the service specific criteria which are asserted by the claims of the VCs linked to the documents such as passport, driving license, any government issued ID card, college/degree certificate, etc. For example, the DID holder should be of age above 15 to consume a service (or) the DID holder should belong to a location to consume a service, the DID holder should belong to a country or university or company to consume a service (or) the DID holder should hold a valid driving license to consume a service, etc.), then the DID verifier sends to the end-device (i.e. DID holder), a Service response message, which can include a successful result. Following which the DID holder will be provided with the requested service. A key associated from the DID document can be used to set up an initial secure communication between the DID holder and the DID Verifier.

Alternatively, for the failure case operation, the DID Verifier can deny the service request by sending to the end-device (i.e. DID holder), a Service response message with the result as failure and the cause information.

9 Governance of various participants in Decentralized Identification framework

PDL governance in general ensures the proper monitoring and execution of the PDL ETSI GR PDL 003 [i.11], ETSI GR PDL 004 [i.12], ETSI GR PDL 010 [i.13]. In particular to a PDL based decentralized identification framework, a resilient governance model is more important for the overall trustworthiness of the data (i.e. DIDs, DID documents, VCs) managed over the PDL framework to ensure the trustworthiness of data towards all the relying parties such as the DID holder/controller, VC Issuer, and DID Verifier. Governance can include a governing body to formulate set of rules to assure the overall operations of the DID service providers [i.7]. The Governance can oversee the operations of the DID service providers either by itself or by involving independent assessment body as needed. The security of the DID depends on the application or wallet that facilitates DID generation and usage of DID at the end user side (e.g. at DID holder/Controller device), so the application client/wallet software, environment, cryptographic algorithms (used in key generation), functions (used for DID generation), influences both security and privacy of the user data managed over PDL. Even though the application client/wallet associated information is under the control of the end-user/device, the security of the application client/wallet needs certification/authorization against a specific-criteria set by the governance to assure the security of the DID generation and usage environment (i.e. application clients/wallets). Governance can enable periodic audit to verify various DID holder/controller related aspects such as uniqueness of the DID(s), cryptographic binding of the DID holder credential to the DID Documents, integrity of DID documents to assure that no parties other than DID holders/Controller can modify the DID documents, etc.

The issuance and usage of VCs plays a vital role in the initial trust establishment between any DID holder and the DID Verifier (e.g. service provider), so the governance of the VC issuance and VC usage is very crucial. The governance of VC issuance and usage should ensure various aspects such as:

i) if the DID related to the subject of the VC belongs to an identifiable entity (e.g. device/person);

ii) if the credentials asserted by the VC issuer which is part of the VC (e.g. claims) belongs to the identified entity;

iii) if the issued VC is managed securely in a way that modification is not possible;

iv) if a DID holder information associated to the VC evolves and if needed a new VC is issued and managed;

v) the VC that is no longer valid is revoked;

vi) the VC is issued by an entity that belongs to the list of authorized VC issuers, etc.

The governance should ensure that the DID, DID documents and VC storage management should be independent of the applications/wallets involved in the DID generation and VC issuance process. The DID associated DID documents, VCs status and validity information need to be governed based on the set of agreed rules as the DID holder, and VC Issuer depends on the respective registry(ies) to provide the relying parties (e.g. DID Verifier) with the latest state information of the DID document and VC(s).

10 Security and Privacy Considerations

The overall security and privacy of the decentralized identification process depends on two main aspects which are broadly classified as:

i) the native properties of the decentralized Identifier; and

ii) the decentralized identification process enablers (i.e. various PDL services which facilitates Role based registration management of different DID operational participants, DID verification management), different DID related data handling with registries (i.e. for DIDs, DID documents, VCs, etc.), and operational management (i.e. overall governance).

The present document covers the native properties of DID in clause 4.3 and the decentralized identification process enablers (i.e. core PDL services that facilitates secure and privacy protected decentralized identification) are summarized in clause 7.2. Currently the DID once generated, it will not be changed until its life-time, so there can be possibilities of replay to attempt flooding attacks. Even though the actual cryptographic algorithms or functions that are used in a DID generation is not within the scope of the present document, as a best practise it is suggested that a DID even if captured and replayed, to enable replay protection, there can be means which facilitates static and dynamic parts in the DID format. For example, a dynamic part can include timestamps or related information (e.g. signing) to ensure freshness and replay protection on top of privacy feature. For DID based data management related to the DID documents, and VCs the operations like update, revocation are covered in the present document with relative transaction being added to the ledgers but, based on different implementations, redactable ledgers can also be considered for updates and revocation kind of operations [i.14].

11 Recommendations

It is recommended to consider the additional PDL services described in clause 7.2 to enable a PDL based DID framework which can facilitate decentralized identification, authentication, and related offering of services.

History

Document history		
V1.1.1	May 2023	Publication