# Identity, Credential and Access Management Document - 24.07 Release

| | |
|---|---|
| Description | Gaia-X specification to build trusted decentralised digital ecosystems. |
| Repository | https://gitlab.com/gaia-x/technical-committee/identity-credentials-and-access-management-working-group/icam |
| Author(s) | Gaia-X European Association for Data and Cloud AISBL |
| Copyright(s) | ©2024 Gaia-X European Association for Data and Cloud AISBL |

# Table of Contents

# 1 Identity, Credential and Access Management Document

## 1.1 Publisher

Gaia-X European Association for Data and Cloud AISBL
Avenue des Arts 6-9
1210 Brussels
www.gaia-x.eu

## 1.2 Authors

Gaia-X European Association for Data and Cloud AISBL

## 1.3 Contact

https://gaia-x.eu/contact/

## 1.4 Other format

For convenience a PDF version of this document is generated here.

## 1.5 Copyright notice

# 2 Introduction and Scope of the Document

This document covers the Gaia-X Identity, Credential and Access Management(ICAM) specifications.

The scope of the document is to describe the components for “Authorization & Authentication” which shall deliver core functionalities for authorization, access management and authentication as well as services around it to Gaia-X Participants, with the purpose to join the trustful environment of the Gaia-X Ecosystem.

Note: this document does not describe how to replace an already established IAM System within a Gaia-X participant environment or how to operate it within the Gaia-X participant environment.

# 3 Credential Format

This section extends the information that the following documents provide about **Gaia-X Credentials**:

- the Gaia-X Architecture Document, which
- defines the Conceptual Model of entities that can have Gaia-X Credentials (**ServiceOffering**s, their **Provider**s, and the **Resource**s they aggregate), and
- introduces Gaia-X Credentials and their lifecycle on a high level
- the Gaia-X Compliance Document, which defines the Gaia-X conformity assessment schemes and the requirements for the respective Trust Anchors.
- the Gaia-X Ontology containing the models to automate the Gaia-X Compliance.

## 3.1 Gaia-X Credential Format

A **Gaia-X Credential** is a Verifiable Credential (VC) following W3C Verifiable Credential Data Model 2.0 using the Gaia-X Ontology which is available via the Gaia-X Registry. A holder can put several Gaia-X credentials together to build a Verifiable Presentation (VP).

A Verifiable Presentation contains one or more Verifiable Credentials with individual disclosed claims and packaged in such a way that the authorship of the data is verifiable. It SHOULD be extremely short-lived, and bound to a challenge provided by a verifier. Each Verifiable Credential that might have been issued by multiple issuers contains signed **claims** about one or more *subjects*.

This section extends the W3C Verifiable Credentials Data Model v2.0 to specify how it shall be applied in the scope of Gaia-X.

### 3.1.1 Gaia-X Credential Example

This section extends the W3C Verifiable Credentials Data Model v2.0 to specify how it shall be applied in the scope of Gaia-X.

The following listing shows an example of a Gaia-X Credential document before becoming a Verifiable Credential.

### Example Verifiable Credential

**document.json**

```json
{
  "@context": [
    "https://www.w3.org/ns/credentials/v2",
    "https://w3id.org/gaia-x/development#"
  ],
  "@type": [
    "VerifiableCredential",
    "LegalPerson"
  ],
  "@id": "https://example.org/legal-participant/68a5bbea9518e7e2ac1cc75bcc8819a7edd5c4711e073ffa4bb260034dc6423c/data.json",
  "issuer": "did:web:example.org",
  "validFrom": "2024-01-01T12:26:22.601516+00:00",
  "validUntil": "2024-04-01T12:26:22.601516+00:00",
  "credentialSubject": {
    "id": "https://example.org/legal-participant-json/68a5bbea9518e7e2ac1cc75bcc8819a7edd5c4711e073ffa4bb260034dc6423c/data.json",
    "type": "gx:LegalPerson",
    "gx:legalName": "Example Org",
    "gx:legalRegistrationNumber": {
      "id": "https://example.org/gaiax-legal-registration-number/68a5bbea9518e7e2ac1cc75bcc8819a7edd5c4711e073ffa4bb260034dc6423c/data.json"
    },
    "gx:headquarterAddress": {
      "gx:countrySubdivisionCode": "FR-75"
    },
    "gx:legalAddress": {
      "gx:countrySubdivisionCode": "FR-75"
    }
  }
}
```

Once it has been encoded into a Verifiable Credential using the [VC-JWT specification](https://www.w3.org/TR/vc-jose-cose/#securing-json-ld-verifiable-credentials-with-jose) it will become the following Verifiable Credential.

```
eyJhbGciOiJQUzI1NiIsInR5cCI6InZjK2xkK2pzb24rand0IiwiY3R5IjoidmMrbGQranNvbiIsImtpZCI6ImRpZDp3ZWI6ZXhhbXBsZS5vcmcjSldLMjAyMC1SU0EifQ.eyJAY29udGV4dCI6WyJodHRwczovL3
ZoPAKQJLKK1gWHsMh5Ge1I99vhZZ61vsGBfjLO0gFhLBwpriLMW7YkJnKD4QoTv-
RxBX3JCakUCE_vkSceUOeRUfJKfEEfbyAAMjBnRZsbeH7xt5MLrs482TxYx2HhSdNkxVZU4UHK0hGSauoGfZrHV5e7XT4N2q4vXIRfN3iihYbw4-
27sSDgNwOkuY34lWwRZSQsP3PoBneJcH0KDvEPgKvOt8V9ZM78wbyH9NIae8qAEKwVNF61cs3XQx6-0bqI6h0n9I4C93ShXxrqmjgTA
```

This VC-JWT can be analyzed and verified with tools such as [JWT.io's debugger](https://jwt.io/#debugger-io?token=eyJhbGciOiJQUzI1NiIsInR5cCI6InZjK2xkK2pzb24rand0IiwiY3R5IjoidmMrbGQranNvbiIsImtpZCI6ImRpZDp3ZWI6ZXhhbXBsZS5vcmcjSldLMjAyMC1SU0EifQ.eyJAY29udG ZoPAKQJLKK1gWHsMh5Ge1I99vhZZ61vsGBfjLO0gFhLBwpriLMW7YkJnKD4QoTv-RxBX3JCakUCE_vkSceUOeRUfJKfEEfbyAAMjBnRZsbeH7xt5MLrs482TxYx2HhSdNkxVZU4UHK0hGSauoGfZrHV5e7XT4N2q4vXIRfN3iihYbw4-27sSDgNwOkuY34lWwRZSQsP3PoBneJcH0KDvEPgKvOt8V9ZM78wbyH9NIae8qAEKwVNF61cs3XQx6-0bqI6h0n9I4C93ShXxrqmjgTA). The following private and public key have been used to sign this VC-JWT example and **all the examples in the upcoming chapters**. This key pair will have the following ID : `did:web:example.org#JWK-RSA`. You will notice it in the `kid` header claim of the example JWT below.

```
-----BEGIN PRIVATE KEY-----
MIIEvwIBADANBgkqhkiG9w0BAQEFAASCBKkwggSlAgEAAoIBAQC7VJTUt9Us8cKj
MzEfYyjiWA4R4/M2bS1GB4t7NXp98C3SC6dVMvDuictGeurT8jNbvJZHtCSuYEvu
NMoSfm76oqFvAp8Gy0iz5sxjZmSnXyCdPEovGhLa0VzMaQ8s+CLOyS56YyCFGeJZ
qgtzJ6GR3eqoYSW9b9UMvkBpZODSctWSNGj3P7jRFDO5VoTwCQAWbFnOjDfH5Ulg
p2PKSQnSJP3AJLQNFNe7br1XbrhV//eO+t51mIpGSDCUv3E0DDFcWDTH9cXDTTlR
ZVEiR2BwpZOOkE/Z0/BVnhZYL71oZV34bKfWjQIt6V/isSMahdsAASACp4ZTGtwi
VuNd9tybAgMBAAECggEBAKTmjaS6tkK8BlPXClTQ2vpz/N6uxDeS35mXpqasqskV
laAidgg/sWqpjXDbXr93otIMLlWsM+X0CqMDgSXKejLS2jx4GDjI1ZTXg++0AMJ8
sJ74pWzVDOfmCEQ/7wXs3+cbnXhKriO8Z036q92Qc1+N87SI38nkGa0ABH9CN83H
mQqt4fB7UdHzuIRe/me2PGhIq5ZBzj6h3BpoPGzEP+x3l9YmK8t/1cN0pqI+dQwY
dgfGjackLu/2qH80MCF7IyQaseZUOJyKrCLtSD/Iixv/hzDEUPfOCjFDgTpzf3cw
ta8+oE4wHCo1iI1/4TlPkwmXx4qSXtmw4aQPz7IDQvECgYEA8KNThCO2gsC2I9PQ
DM/8Cw0O983WCDY+oi+7JPiNAJwv5DYBqEZB1QYdj06YD16XlC/HAZMsMku1na2T
N0driwenQQWzoev3g2S7gRDoS/FCJSI3jJ+kjgtaA7Qmzlgk1TxODN+G1H91HW7t
0l7VnL27IWyYo2qRRK3jzxqUiPUCgYEAx0oQs2reBQGMVZnApD1jeq7n4MvNLcPv
t8b/eU9iUv6Y4Mj0Suo/AU8lYZXm8ubbqAlwz2VSVunD2tOplHyMUrtCtObAfVDU
AhCndKaA9gApgfb3xw1IKbuQ1u4IF1FJl3VtumfQn//LiH1B3rXhcdyo3/vIttEk
48RakUKClU8CgYEAzV7W3COOlDDcQd935DdtKBFRAPRPAlspQUnzMi5eSHMD/ISL
DY5IiQHbIH83D4bvXq0X7qQoSBSNP7Dvv3HYuqMhf0DaegrlBuJllFVVq9qPVRnK
xt1Il2HgxOBvbhOT+9in1BzA+YJ99UzC85O0Qz06A+CmtHEy4aZ2kj5hHjECgYEA
mNS4+A8Fkss8Js1RieK2LniBxMgmYml3pfVLKGnzmng7H2+cwPLhPIzIuwytXywh
2bzbsYEfYx3EoEVgMEpPhoarQnYPukrJO4gwE2o5Te6T5mJSZGlQJQj9q4ZB2Dfz
et6INsK0oG8XVGXSpQvQh3RUYekCZQkBBFcpqWpbIEsCgYAnM3DQf3FJoSnXaMhr
VBIovic5l0xFkEHskAjFTevO86Fsz1C2aSeRKSqGFoOQ0tmJzBEs1R6KqnHInicD
TQrKhArgLXX4v3CddjfTRJkFWDbE/CkvKZNOrcf1nhaGCPspRJj2KUkj1Fhl9Cnc
dn/RsYEONbwQSjIfMPkvxF+8HQ==
-----END PRIVATE KEY-----

-----BEGIN PUBLIC KEY-----
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAu1SU1LfVLPHCozMxH2Mo
4lgOEePzNm0tRgeLezV6ffAt0gunVTLw7onLRnrq0/IzW7yWR7QkrmBL7jTKEn5u
+qKhbwKfBstIs+bMY2Zkp18gnTxKLxoS2tFczGkPLPgizskuemMghRniWaoLcyeh
kd3qqGElvW/VDL5AaWTg0nLVkjRo9z+40RQzuVaE8AkAFmxZzow3x+VJYKdjykkJ
0iT9wCS0DRTXu269V264Vf/3jvredZiKRkgwlL9xNAwxXFg0x/XFw005UWVRIkdg
cKWTjpBP2dPwVZ4WWC+9aGVd+Gyn1o0CLelf4rEjGoXbAAEgAqeGUxrcIlbjXfbc
mwIDAQAB
-----END PUBLIC KEY-----
```

## 3.2 Digital Signature Standard

This document follows the FIPS 186-5 standard on Digital Signature Standard (DSS).

## 3.3 Decentralized Identifiers

This section extends the W3C Decentralized Identifiers to specify how it shall be applied in the scope of Gaia-X.

### 3.3.1 Verification Methods

To ensure a Gaia-X Credential's integrity and authenticity, its claims MUST be cryptographically signed by the Participant that is issuing them.
This is done to avoid tampering and to technically allow to check the origin of the claims.

The supported verification methods are described below.

#### 3.3.1.1 JSON Web Key

This section extends the specification from the W3C JSON Web Key.

A Verifiable Credential is *Gaia-X Conformant* if:

- it is signed by a trusted issuer present in the Gaia-X Registry
- its issuer has a verifiable identity coming from one of the Trust Anchors
- it complies with the Gaia-X Ontology Shacl Shapes
- it uses the enveloping proof encoding specified by this document

Without a means to link the issuer's verification method to a Gaia-X Trust Anchor, the Gaia-X Compliance verification will fail.

To be able to assess the chain of trust, the `publicKeyJwk` property MUST include either the RFC7517 `x5c` (X.509 Certificate Chain) parameter or RFC7517 `x5u` (X.509 URL) parameter.
The `x5u` parameter should be resolvable to a `X509` `.crt`, `.pem`, `.der` or `.p7b` file which contains a valid Gaia-X Trust Anchor eligible for the signed claims.

To ensure the correct cryptographic tools are used with the public key, the `alg` property MUST be specified, the value must comply with the JSON Web Algorithms RFC7518 `alg`

Example of verificationMethod

verifiable_method.json

```
{
 "@context": [
  "https://www.w3.org/ns/did/v1",
  "https://w3id.org/security/jwk/v1"
 ],
 "id": "did:web:example.org",
 "verificationMethod": [
  {
   "id": "did:web:example.org#JWK-RSA",
   "type": "JsonWebKey",
   "controller": "did:web:example.org",
   "publicKeyJwk": {
    "kty": "RSA",
    "n": "0oxjNiK1D5lcowRFjpzY8AY8DwkH5I4mXnz9f4ILcFIV8HG3EyWnAkYgf5EJO91P7t4NHESxuNvSXSSYe5UBizWXNfmKClX2I3g0-6Iw0amxtdgrAmX-
HxOprdxfBMt1xRwf6B4M9CQzGBIDAMW5B8-
zJsbnGPIz0iWJ2qdFvtPtD3He3ds7azrcLmEaQLqg2yw7Fw5xwmSRodYasXIOhk1Wg4lqiJp2bG9JBaWdJW7Q2kee39UxnAXCQKmflkQuPAALxj5C-5436n0-
-64Xd6JH6QeaKgyQPGLEEVwvNibiZD8PcbXqyCDCNBD7DmTOTLfjz03qH5qjYvpuo5K6Aw",
    "e": "AQAB",
    "kid": "q44a8UEJIUNs43nWyLSCxNQXQRB40ccilgRqsZ7n0Og",
    "alg": "PS256",
    "x5u": "https://example.org/.well-known/chain.pem"
   }
  }
 ],
 "authentication": [
  "did:web:example.org#JWK-RSA"
 ],
 "assertionMethod": [
  "did:web:example.org#JWK-RSA"
 ]
}
```

## 3.4 Use of Identifiers in Gaia-X Credentials

Each of the following MUST have a different identifier:

- a Verifiable Presentation
- a Verifiable Credential inside a Verifiable Presentation
- the subject of a Verifiable Credential, i.e., the Conceptual Model entity that claims are made about.

Gaia-X Credentials MAY reference other Gaia-X Credentials. Consider, for example, a *ServiceOffering* that:

- is provided by a *Provider*,
- is a composition of other *ServiceOffering*s, or
- is an aggregation of *Resources*.

## 3.5 Verifiable Credential and Verifiable Presentation

This section extends the W3C Verifiable Credentials Data Model v2.0 to specify how it shall be applied in the scope of Gaia-X.

### 3.5.1 Namespace Bindings and Contexts

On the level of the Verifiable Presentation and the Verifiable Credentials contained in the Verifiable Presentation, a Gaia-X Credential MUST adhere to the vocabulary of the Verifiable Credentials Data Model, i.e., use terms from the `https://www.w3.org/2018/credentials#` namespace.

To enable human authors of Gaia-X Credentials to write down these terms conveniently, they MAY, by using the `@context` keyword on the level of the Verifiable Presentation, e.g.:

- reference the JSON-LD context provided by the Verifiable Credentials Data Model (https://www.w3.org/ns/credentials/v2) like in the initial example listing, or
- define their own context, which
- defines the above namespace as the default vocabulary using the `@vocab` keyword, or
- maps the above namespace to a designated prefix, e.g., `"cred"`.

Similarly, the claims about any credential subject MUST adhere to the vocabulary of the Gaia-X Credential Schemas published in the Gaia-X Registry, or to Federation-specific specializations of this vocabulary.

### 3.5.2 Identifiers

The `@id` MUST be present and unique for a given `issuer`.

The `@id` keyword is aliased to `id`. Consequently we MAY also use this alias.

It is up to the `issuer` to decide if the `@id` is a resolvable URL or not.

### 3.5.3 Integrity of Related Resources

In order to enable reference to objects - Verifiable Credentials or credential subject - which are not under control of the same issuers, it is RECOMMENDED to specify an `@sri` Subresource Integrity attribute to enable the verification of the integrity of the referenced object.

The `sri` attribute is computed by taking the hash of the referenced normalized JSON object.
The JSON object is normalized following the JSON Canonicalization Scheme (JCS) defined in the RFC 8785.

**Example of `sri` attribute**

**SRI attribute**

```
{
  "@id": "https://example.com/ABC",
  "sri": "sha256-b9a822666c3569a8ae80c897a1984f68bbdffa1f8141cacdb3f168b1c0b9aa36"
}
```

### 3.5.4 Types

The `@type` property MUST be present in Verifiable Presentation, Verifiable Credentials, and Credentials. The expected values for the first `@type` property are:

- `"VerifiablePresentation"` for a Verifiable Presentation
- `"EnvelopedVerifiablePresentation"` for an Enveloped Verifiable Presentation encoded as a VC-JWT
- `"VerifiableCredential"` for a Verifiable Credential
- `"EnvelopedVerifiableCredential"` for an Enveloped Verifiable Credential encoded as a VC-JWT

This `@type` can be followed with one or more credential related types ( ie. `@type: ['Verifiable Credential', 'LegalPerson']`).

The `@type` keyword is aliased to `type`. Consequently, we MAY also use this alias.

The expected values for the `@type` property of a credential subject are given by the taxonomy of classes defined in the Gaia-X Trust Framework, having the superclasses `Participant`, `ServiceOffering` and `Resource`.

A Federation MAY define additional subclasses of these by further shapes hosted in its Catalogue(s). In the future, Gaia-X and federations MAY also define additional, more specific credential types.

#### 3.5.4.1 Schema Validation

A Schema for Gaia-X Credentials, to be used as the vocabulary of the claims about credential subjects, MUST be available in the form of SHACL shapes (cf. the W3C Shapes Constraint Language SHACL) in the Gaia-X Registry or in the Catalogue of a Federation.
At any point where Gaia-X Credentials are created or received, a certain set of SHACL shapes is known, which forms a *shapes graph*. A Gaia-X Credential forms a *data graph*. For compliance with Gaia-X and/or a specific Federation, this *data graph* MUST be validated against the given *shapes graph* according to the SHACL specification.

### 3.5.5 Issuers

The `issuer` property MUST be present in Verifiable Credential and Verifiable Presentation. The value of the `issuer` property must be a resolvable URI.

The supported schemes for `issuer`'s URI are:

- `https`
- `did`. The supported DID methods are:
- `web`

### 3.5.6 validFrom

The `validFrom` property is MANDATORY for Verifiable Credential and Verifiable Presentation.

### 3.5.7 validUntil

The `validUntil` property is RECOMMENDED for Verifiable Credential and Verifiable Presentation.

## 3.5.8 Verifiable Credential

Verifiable Credentials are encoded as Json Web Tokens as described in the VC-JWT specification. This type of proofing is an enveloping proof.

A JWT consists in a header, a payload and a signature each element being separated by a dot ( `.` ).

**Example Verifiable Credential**

If we use the following credential:

```
{
 "@context": [
  "https://www.w3.org/2018/credentials/v2",
  "https://w3id.org/gaia-x/development#"
 ],
 "@type": [
  "VerifiableCredential",
  "LegalParticipant"
 ],
 "@id": "https://example.org/legal-participant/68a5bbea9518e7e2ac1cc75bcc8819a7edd5c4711e073ffa4bb260034dc6423c/data.json",
 "issuer": "did:web:example.org",
 "validFrom": "2024-04-01T12:26:22.601516+00:00",
 "validUntil": "2024-01-01T12:26:22.601516+00:00",
 "credentialSubject": {
  "id": "https://example.org/legal-participant-json/68a5bbea9518e7e2ac1cc75bcc8819a7edd5c4711e073ffa4bb260034dc6423c/data.json",
  "type": "gx:LegalPerson",
  "gx:legalName": "Example Org",
  "gx:legalRegistrationNumber": {
   "id": "https://example.org/gaiax-legal-registration-number/68a5bbea9518e7e2ac1cc75bcc8819a7edd5c4711e073ffa4bb260034dc6423c/data.json"
  },
  "gx:headquarterAddress": {
   "gx:countrySubdivisionCode": "FR-75"
  },
  "gx:legalAddress": {
   "gx:countrySubdivisionCode": "FR-75"
  }
 }
}
```

The VC-JWT representation as a Verifiable Credential would be:

```
eyJhbGciOiJQUzI1NiIsInR5cCI6InZjK2xkK2pzb24rand0IiwiY3R5IjoidmMrbGQranNvbiIsImtpZCI6ImRpZDp3ZWI6ZXhhbXBsZS5vcmcjSldLMjAyMC1SU0EifQ.eyJAY29udGV4dCI6WyJodHRwczovL3
ZoPAKQJLKK1gWHsMh5Ge1I99vhZZ61vsGBfjLO0gFhLBwpriLMW7YkJnKD4QoTv-
RxBX3JCakUCE_vkSceUOeRUfJKfEEfbyAAMjBnRZsbeH7xt5MLrs482TxYx2HhSdNkxVZU4UHK0hGSauoGfZrHV5e7XT4N2q4vXIRfN3iihYbw4-
27sSDgNwOkuY34lWwRZSQsP3PoBneJcH0KDvEPgKvOt8V9ZM78wbyH9NIae8qAEKwVNF61cs3XQx6-0bqI6h0n9I4C93ShXxrqmjgTA
```

To view the header, payload and signature of this example on JWT.io [click here](https://jwt.io/#debugger-io?
token=eyJhbGciOiJQUzI1NiIsInR5cCI6InZjK2xkK2pzb24rand0IiwiY3R5IjoidmMrbGQranNvbiIsImtpZCI6ImRpZDp3ZWI6ZXhhbXBsZS5vcmcjSldLMjAyMC1SU0EifQ.eyJAY29udG
ZoPAKQJLKK1gWHsMh5Ge1I99vhZZ61vsGBfjLO0gFhLBwpriLMW7YkJnKD4QoTv-
RxBX3JCakUCE_vkSceUOeRUfJKfEEfbyAAMjBnRZsbeH7xt5MLrs482TxYx2HhSdNkxVZU4UHK0hGSauoGfZrHV5e7XT4N2q4vXIRfN3iihYbw4-
27sSDgNwOkuY34lWwRZSQsP3PoBneJcH0KDvEPgKvOt8V9ZM78wbyH9NIae8qAEKwVNF61cs3XQx6-0bqI6h0n9I4C93ShXxrqmjgTA).

### 3.5.8.1 Header

The VC-JWT header MUST contain the following fields:

- `alg`, the signature algorithm (ie. `PS256`)
- `typ`, the media type of the JWT which must be set to `vc+ld+json+jwt`
- `cty`, the content type of the payload which must be set to `vc+ld+json`
- `kid`, the `did:web` or URL reference to the verification method in a DID document

Additional headers that aren't described in the VC-JWT, JWT or JWS specifications should be ignored.

### 3.5.8.2 Payload

The payload of the VC-JWT is a standard verifiable credential with claims as described in the Verifiable Credential Data Model v2.0 specification.

Some payload claims from the JWT specification MUST be replaced by the described verifiable credential fields such as:

- `iss` will be replaced by the verifiable credential's `issuer`
- `jti` will be replaced by the verifiable credential's `id` or `@id`
- `sub` will be replaced by the verifiable credential's `credentialSubject.id` or `credentialSubject.@id`

The `vc` and `vp` payload claims MUST NOT be present.

> The `iat` and `exp` payload claims represent the JWT's signature validity period whereas the `validFrom` and the `validUntil` verifiable credential payload claims represent the verifiable credential's data validity period. Therefore these claims can cohabit in the payload.

If the `@type` is “VerifiableCredential”, the property `credentialSubject` MUST be defined. The value of `credentialSubject` can be a Credential or an array of Credentials. A Verifiable Credential MUST have :

- an `@id`,
- an `issuer`,
- a `@type`, and
- a `credentialSubject` object or a `credentialSubject` array.

> NB: The `@id` and `@type` keywords are aliased to `id` and `type` respectively. Consequently, we MAY also use these aliases.

#### 3.5.8.3 Signature

The last element of a VC-JWT is the signature which is cryptographically secured to ensure integrity hence making the Verifiable Credential tamper-proof.

A JWS is signed using the issuer’s private key and can be verified by using the issuer’s public key which is obtainable through the issuer’s DID document (referenced in the `kid` JWS header).

VC-JWT signatures are created by following the JSON Web Signature (JWS) specification. Many libraries are available online to manage JWS creation.

#### 3.5.8.4 Credential Subject

The `credentialSubject` can be an object or array of objects, containing claims.

The claims about one Gaia-X entity MAY be spread over multiple Credentials and their subjects.

Each credential subject MUST have an `@id`.

A credential subject MAY be described *by value*, i.e., by stating one or more claims about it in place. In this case, it MUST have a `@type` as specified below.

Alternatively, a credential subject MAY be described *by reference*. In this case, the `@id` MUST be resolvable to an RDF resource that has the same `@id`, a `@type`, and one or more claims. See Identifiers section for more details.

The value of the `@type` property dictates the vocabulary available in the Trust Framework document for the definition of claims about the credential subject. E.g., `LegalPerson`, `ServiceOffering`, `DataResource`, …

**Example of credentialSubject**

```
{
  "@id": "https://example.com/legalPersonABC?vcid=c93b5075b3988eda4a529afce7e7c127f607b55dc08bb12e8c9adc9e33fe814f",
  "@type": "gx:legalPerson",
  "gx:legalName": "Legal Person ABC",
  "gx:legalRegistrationNumber": {
    "@id": "https://gaia-x.eu/legalRegistrationNumber_VC.json"
  },
  "gx:headquarterAddress": {
    "gx:countrySubdivisionCode": "FR-IDF"
  },
  "gx:legalAddress": {
    "gx:countrySubdivisionCode": "FR-IDF"
  }
}
```

### 3.5.9 Enveloped Verifiable Credential

An Enveloped Verifiable Credential is a convenient way of describing a Verifiable Credential that has been encoded with an enveloping proof such as VC-JWT.

It’s represented as a basic JSON object with three fields:

- `@context` which is usually set to `https://www.w3.org/ns/credentials/v2`
- `id` containing the data of the VC-JWT in the form of an `application/vc+ld+json+jwt` data URL
- `type` which must be set to `EnvelopedVerifiableCredential`

This type of Verifiable Credential is very useful in the context of a Verifiable Presentation to embed multiple Verifiable Credentials.

**Example Enveloped Verifiable Credential**

Below is an example representing the [Verifiable Credential example](#verifiable-credential) as an Enveloped Verifiable Credential.

```
{
  "@context": "https://www.w3.org/ns/credentials/v2",
  "id":
"data:application/vc+ld+json+jwt;eyJhbGciOiJQUzI1NiIsInR5cCI6InZjK2xkK2pzb24rand0IiwiY3R5IjoidmMrbGQranNvbiIsImtpZCI6ImRpZDp3ZWI6ZXhhbXBsZS5vcmc6bGVnYWxQZXJzb25BQkM
euaBG7fWGTQ_F6yqWSPeQ6veHqkxKkvtdLIkSSpxZRJCtQs2HiORQX3tc21dkqtziKJIDJhmIBIq-2zDToPb5D4Yb_ryP0aTgcnBavAuiNCf7x3_gS6tBtYd_ZNnh3cifFiLGLop6PUhqhaTEYBlw1ou-
28XUCHPeaarGrmxyZzxiBV_3J5hAe8XvfnFo9Y__LcbuOjNMsU2kKhI9otw9Ll4C8IZ9Qsqdq52QFCvkbvtcvX_3IJpzyxSS7TxOXAPPwYbYV_u7tgygPRvvmQG99Q651y62tQGA_B6Eqg",
  "type": "EnvelopedVerifiableCredential"
}
```

### 3.5.10 Verifiable Presentation

If the `@type` is “VerifiablePresentation”, the property `verifiableCredential` MUST be defined. The value of `verifiableCredential` property MUST be an array of one or several Enveloped Verifiable Credentials. A Verifiable Presentation MUST have :

- a `@type`,
- a `verifiableCredential` array of `EnvelopedVerifiableCredential`,

**Example Verifiable Presentation**

Below is an example of a Verifiable Presentation containing the example from the [Enveloped Verifiable Credential](#enveloped-verifiable-credential) chapter.

```
{
  "@context": [
    "https://www.w3.org/ns/credentials/v2"
  ],
  "@id": "https://gaia-x.eu/verifiablePresentation/1",
  "type": [
    "VerifiablePresentation"
  ],
  "verifiableCredential": [
    {
      "@context": "https://www.w3.org/ns/credentials/v2",
      "id":
"data:application/vc+ld+json+jwt;eyJhbGciOiJQUzI1NiIsInR5cCI6InZjK2xkK2pzb24rand0IiwiY3R5IjoidmMrbGQranNvbiIsImtpZCI6ImRpZDp3ZWI6ZXhhbXBsZS5vcmc6bGVnYWxQZXJzb25BQkM
euaBG7fWGTQ_F6yqWSPeQ6veHqkxKkvtdLIkSSpxZRJCtQs2HiORQX3tc21dkqtziKJIDJhmIBIq-2zDToPb5D4Yb_ryP0aTgcnBavAuiNCf7x3_gS6tBtYd_ZNnh3cifFiLGLop6PUhqhaTEYBlw1ou-
28XUCHPeaarGrmxyZzxiBV_3J5hAe8XvfnFo9Y__LcbuOjNMsU2kKhI9otw9Ll4C8IZ9Qsqdq52QFCvkbvtcvX_3IJpzyxSS7TxOXAPPwYbYV_u7tgygPRvvmQG99Q651y62tQGA_B6Eqg",
      "type": "EnvelopedVerifiableCredential"
    }
  ]
}
```

### 3.5.11 Enveloped Verifiable Presentation

Just like an Enveloped Verifiable Credential, an Enveloped Verifiable Presentation is a representation of a Verifiable Presentation in the form of a basic JSON object containing an `application/vp+ld+jwt` data URL.

This data URL expresses a JWS secured Verifiable Presentation. The same headers as Verifiable Credentials are used in a Verifiable Presentation VC-JWT except:

- the `typ` header is set to `vp+ld+jwt`
- the `cty` header is set to `vp+ld+json`

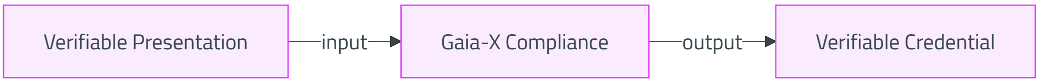
**Example Enveloped Verifiable Presentation**

Below is a representation of the [Verifiable Presentation example](#verifiable-presentation) as an Enveloped Verifiable Presentation.

```
{
  "@context": "https://www.w3.org/ns/credentials/v2",
  "id":
"data:application/vp+ld+jwt;eyJhbGciOiJQUzI1NiIsInR5cCI6InZjK2xkK2pzb24rand0IiwiY3R5IjoidmMrbGQranNvbiIsImtpZCI6ImRpZDp3ZWI6ZXhhbXBsZS5vcmc6bGVnYWxQZXJzb25BQkMja2V
765IP1mEE-tsLi2tMXQ6TeWIfw_6NkpY0vo_FUXWBBlj0IgMbxbt0gQwHRW9Ph3SVKQMCdIfp_pmdWPCEUrr_HxjkdiZpF1fa4qGSYBYl6tRSf1N0iCY0SzKvStI-
EiudwHtlSygcqjxNq1jdpZtQyjYa_golZmyBdX7BYUUkcY30vypKTjMgBLHlZOzIljdiLKcm_MfDGEBt-Ha_qxpKpwRZoMFhsq89RXeExpeAw8Vg3ZR7yWsmP3T-
7DDrZ5sadpNyCDXpryvm2UoDs__M4lEvkl3Hly9LQ",
  "type": "EnvelopedVerifiablePresentation"
}
```

## 3.6 Gaia-X compliance input/output

The Gaia-X Compliance service



### 3.6.1 Input

The input of the Gaia-X Compliance service is a VC-JWT `Verifiable Presentation` that might contain:

- one or more Enveloped Verifiable Credentials
- one or more Verifiable Credentials

Such a Verifiable Credential MAY or MAY NOT be covered by the Gaia-X Compliance rules.

The following example contains fake, placeholder attributes for `Participant` and `ServiceOffering`, which are not valid against the Gaia-X Credential Schema.

### Example of Compliance input

> :warning: Due to the lack of readability of JWTs, the following credentials are not signed, for more complete functional cases please take a look at these [examples](https://gitlab.com/gaia-x/technical-committee/identity-credentials-and-access-management-working-group/icam/-/tree/main/docs/examples/)

**compliance_input.json**

```
{
  "@context": [
    "https://www.w3.org/ns/credentials/v2"
  ],
  "type": "VerifiablePresentation",
  "issuer": "did:web:gaia-x.eu",
  "validUntil": "2024-07-11T07:51:07.168+00:00",
  "verifiableCredential": [
    {
      "id": "https://gaia-x.eu/participant.json",
      "type": "VerifiableCredential",
      "issuer": "did:web:gaia-x.eu",
      "validFrom": "2024-01-01T19:23:24Z",
      "validUntil": "2024-07-11T07:42:21.972+00:00",
      "credentialSubject": {
        "id": "https://gaia-x.eu/participant.json#cs",
        "type": "gx:LegalPerson",
        "https://schema.org/name": "GAIA-X",
        "gx:registrationNumber": {
          "id": "https://gaia-x.eu/gaia-x-lrn.json#cs"
        },
        "gx:headquartersAddress": {
          "type": "gx:Address",
          "gx:countryCode": "BE"
        },
        "gx:legalAddress": {
          "type": "gx:Address",
          "gx:countryCode": "BE"
        }
      }
    },
    {
      "id": "https://gaia-x.eu/gaia-x-lrn.json",
      "issuer": "did:web:gaia-x.eu",
      "type": "VerifiableCredential",
      "validFrom": "2024-05-15T12:10:23.900Z",
      "validUntil": "2024-07-11T07:43:05.752+00:00",
      "credentialSubject": {
        "type": "gx:VatID",
        "id": "https://gaia-x.eu/gaia-x-lrn.json#cs",
        "gx:vatID": "BE0762747721",
        "gx:countryCode": "BE"
      }

    },
    {
      "issuer": "did:web:gaia-x.eu",
      "id": "https://gaia-x.eu/gaia-x-tsandcs.json",
      "type": "VerifiableCredential",
      "validFrom": "2024-05-27T09:12:35.754Z",
      "validUntil": "2024-07-11T07:43:30.274+00:00",
      "credentialSubject": {
        "gx:termsAndConditions": "The PARTICIPANT signing the Self-Description agrees as follows:\n- to update its descriptions about any changes, be it technical, organizational, or legal - especially but not limited to contractual in regards to the indicated attributes present in the descriptions.\n\nThe keypair used to sign Verifiable Credentials will be revoked where Gaia-X Association becomes aware of any inaccurate statements in regards to the claims which result in a non-compliance with the Trust Framework and policy rules defined in the Policy Rules and Labelling Document (PRLD).",
        "type": "gx:GaiaXTermsAndConditions",
        "id": "https://gaia-x.eu/gaia-x-tsandcs.json#cs"
      }

    },
    {
      "id": "https://gaia-x.eu/service.json",
      "issuer": "did:web:gaia-x.eu",
      "type": "VerifiableCredential",
      "validFrom": "2024-07-23T13:36:54.648Z",
      "validUntil": "2024-07-25T15:17:11.243+00:00",
      "credentialSubject": {
        "type": "gx:ServiceOffering",
        "gx:providedBy": {
          "@id": "https://gaia-x.eu/participant.json#cs"
        },
        "gx:policy": "",
        "gx:termsAndConditions": {
          "gx:URL": "http://termsandconds.com",
          "gx:hash": "d8402a23de560f5ab34b22d1a142feb9e13b3143"
        },
        "gx:dataAccountExport": {
          "gx:requestType": "API",
          "gx:accessType": "digital",
          "gx:formatType": "application/json"
        },
        "@id": "https://gaia-x.eu/service.json#cs"
      }
    }
  ]
}
```

### 3.6.2 Output

The output of the Gaia-X Compliance service is a VC-JWT `VerifiableCredential` containing the `id` and `hash` of the compliant `VerifiableCredential` from the input.

**Example of Compliance output**

```
{
  "@context": [
    "https://www.w3.org/ns/credentials/v2",
    "https://w3id.org/gaia-x/development#"
  ],
  "type": [
    "VerifiableCredential",
    "gx:ComplianceCredential"
  ],
  "id": "https://storage.gaia-x.eu/credential-offers/b3e0a068-4bf8-4796-932e-2fa83043e203",
  "issuer": "did:web:compliance.lab.gaia-x.eu:development",
  "validFrom": "2024-07-24T15:18:28.376Z",
  "validUntil": "2024-10-22T15:18:28.355Z",
  "credentialSubject": {
    "id": "https://storage.gaia-x.eu/credential-offers/b3e0a068-4bf8-4796-932e-2fa83043e203#cs",
    "gx:evidence": [
      {
        "id": "https://gaia-x.eu/participant.json",
        "type": "gx:ComplianceEvidence",
        "gx:integrity": "sha256-578b2fa4ec5d83317f7356dfb11f656c14ac3b1705ee276d09ed76871bf53b29",
        "gx:integrityNormalization": "RFC8785:JCS",
        "gx:engineVersion": "2.2.0",
        "gx:rulesVersion": "PRLD-24.04_pre",
        "gx:originalType": "gx:LegalPerson,VerifiableCredential"
      },
      {
        "id": "https://gaia-x.eu/gaia-x-tsandcs.json",
        "type": "gx:ComplianceEvidence",
        "gx:integrity": "sha256-3962bfc58471f19e8e5d5ea05652725ca0bb8b62af27cfdd8d9022c69b585387",
        "gx:integrityNormalization": "RFC8785:JCS",
        "gx:engineVersion": "2.2.0",
        "gx:rulesVersion": "PRLD-24.04_pre",
        "gx:originalType": "gx:GaiaXTermsAndConditions,VerifiableCredential"
      },
      {
        "id": "https://gaia-x.eu/gaia-x-lrn.json",
        "type": "gx:ComplianceEvidence",
        "gx:integrity": "sha256-3e374271b13e1241eda27c672de70c3d4d497a4b4c8f02f287fe3aa61d789fd9",
        "gx:integrityNormalization": "RFC8785:JCS",
        "gx:engineVersion": "2.2.0",
        "gx:rulesVersion": "PRLD-24.04_pre",
        "gx:originalType": "gx:VatID,VerifiableCredential"
      },
      {
        "id": "https://gaia-x.eu/service.json",
        "type": "gx:ComplianceEvidence",
        "gx:integrity": "sha256-1b1a7bb3545891ca912ee29425b74d842eec09c1c4f6847571708d0888601457",
        "gx:integrityNormalization": "RFC8785:JCS",
        "gx:engineVersion": "2.2.0",
        "gx:rulesVersion": "PRLD-24.04_pre",
        "gx:originalType": "gx:ServiceOffering,VerifiableCredential"
      }
    ]
  }
}
```

# 4 TrustAnchor Credential

The Trust Anchor Credential is based on the Verifiable Credentials Data Model v2.0 and has the purpose to provide a machine-readable representation of the accreditation of a Trust Anchor for a specific scope. This credential enables the usage of Party Credentials, described later in this chapter, by defining their accredited issuers. Furthermore, the Trust Anchor credential supports the cooperation and interoperability between organization/ecosystems/data spaces, by easing the use of external Trust Anchors.

The **TrustAnchorCredential** mainly defines:

- The **Scope** within which the Trust Anchor is accredited (within which the issued credentials are considered valid).
- The **TrustedIssuer** entitled to issue **Credentials** in the above **Scope**
- The **Vocabularies** (expressed in SHACL) that semantically define the **Credentials** which can be issued by the **TrustedIssuer** in the defined **Scope**.
- The **Trusted List** used to check DIDs issued by the Trust Anchor.

The `Trust Anchor Credential` is defined by the following attributes:

| Attribute | Type.Value/Voc | Mandatory | Comment |
|---|---|---|---|
| `gx:scopeDescription` | String | No | A description of the scope of the Trust Anchor |
| `gx:trustIssuer` | DID | Yes | A resolvable link to the holder verificationMethod to be used to uniquely identify ONE and ONLY key pair |
| `gx:vocabularies` | URI[] | No | A list of URIs pointing to the vocabularies (SHACL) semantically describing the Trust Anchor's scope |
| `gx:trustedListKind` | KindOfTrustedList | No | Which kind of implementation of the trusted list is used to check DID issued by Trust Anchors |
| `gx:trustedListEndpoint` | URI | No | The address of the above trusted list |

The **KindOfTrustedList** type defines the list of the identified implementations of Trusted Lists:

- Gaia-X Trusted List Generic REST API Specification
- Gaia-X IPFS (with ETSI TS 119 612 format)
- TRAIN
- EBSI
- (other possible implementations)

**Important Note** The *Gaia-X Trusted List Generic REST API Specification* is meant to be a simple API contract that for a given DID can return true if it is still valid. This will be very useful to enable integration of any custom trusted list not included in the defined list.

# 5 Party Credential

The Party Credential is based upon the Verifiable Credentials Data Model v2.0 and is the basement for all IAA Parties such as **Natural Persons** , **Services**, **Legal Persons**, etc. The general purpose party credential is intended to be extended into specialized Credentials (see **Party Credential Specializations**)

The `Party Credential` is defined by the following attributes:

| Attribute | Type.Value/Voc | Mandatory | Comment |
|---|---|---|---|
| `gx:holder` | URI | Yes | A resolvable link to the holder verificationMethod to be used to uniquely identify ONE and ONLY key pair |
| `odrl:hasPolicy` | policy[] in ODRL | No | A list of `policy` expressed using ODRL |
| `gx:identityAttributes` | String[] | No | A list of literals representing Identity Attributes to be used in a ABAC context |
| `gx:identityRoles` | String[] | No | A list of literals representing Identity Roles to be used in a RBAC context |
| `gx:parentPartyCredential` | URI | Yes if is delegated by another existing Party Credential | A resolvable link to the parent Party Credential where the gx:holder MUST be equal to the signer(issuer verificationMethod) of this credential |

**VERY IMPORTANT NOTE**

The **credentialSubject.id** must identify the **DID Document** that contains the **verificationMethod** (keypair) referenced by `gx:holder` property that *belong to/is controlled by* the **Credential Holder**. With this solution, the PartyCredential VC can be either publicly published in case it contains no sensitive data or kept private in case it contains PII - Personal Identifiable Information.

## 5.1 Private Party Credential

The **Party Credentials** containing PII are not considered to be published and reachable via their **id** to everybody, instead, they are intended to be stored in secure storage such as a wallet, secure storage device, secure vault storage, etc. An example of this type of credential is the **NaturalPersonCredential**, issued by a Legal Participant to one of his users/employees, with the purpose to entitle a Natural Person to interact with Relying Parties(RP) in a certain context This credential contains Name, Surname, identityAttributes, Roles, etc. and MUST NOT be published as Public Party Credentials (see later in this section) are. “Selective disclosure” during interaction with RP can be considered.

### 5.1.1 Private Party Credential Example

The following **NaturalPersonPartyCredential** example represents the scenario where the Participant **did:web:did.actor:alice** is issuing a credential that asserts several claims (givenName, surname, idRoles etc). This Credential is not published (the **id** is not public) and must be stored in the wallet of the holder.

Note that the `gx:holder` property is a did:key referencing a keypair owned by the **Holder** that identifies the public key of the target wallet.

**party_credential.json**

```
{
  "@context": [
    "https://www.w3.org/ns/credentials/v2",
    "https://w3id.org/gaia-x/development#"
  ],
  "id": "did:web:did.actor:alice:credentials:private#1222331234",
  "type": ["VerifiableCredential"],
  "issuer": "did:web:did.actor:alice",
  "validFrom": "2023-08-28T23:00:00Z",
  "credentialSubject": {
    "id":
"did:key:z4MXj1wBzi9jUstyPMS4jQqB6KdJaiatPkAtVtGc6bQEQEEsKTic4G7Rou3iBf9vPmT5dbkm9qsZsuVNjq8HCuW1w24nhBFGkRE4cd2Uf2tfrB3N7h4mnyPp1BF3ZttHTYv3D
    "type": "gx:NaturalPersonPartyCredential",
    "odrl:hasPolicy": [],
    "gx:holder":
"did:key:z4MXj1wBzi9jUstyPMS4jQqB6KdJaiatPkAtVtGc6bQEQEEsKTic4G7Rou3iBf9vPmT5dbkm9qsZsuVNjq8HCuW1w24nhBFGkRE4cd2Uf2tfrB3N7h4mnyPp1BF3ZttHTYv3D
    "gx:identityAttributes": ["IdAttributeOne","IdAttributeTwo"],
    "gx:identityRoles": ["IdRoleOne","IdRoleTwo"],
    "gx:givenName": "John",
    "gx:surname": "Doe"
  },
  "credentialStatus": [{
    "id": "https://did.actor/alice/credentials/status/3#94567",
    "type": "BitstringStatusListEntry",
    "statusPurpose": "revocation",
    "statusListIndex": "94567",
    "statusListCredential": "https://did.actor/alice/credentials/status/3"
  },{
    "id": "https://did.actor/alice/credentials/status/4#23452",
    "type": "BitstringStatusListEntry",
    "statusPurpose": "suspension",
    "statusListIndex": "23452",
    "statusListCredential": "https://did.actor/alice/credentials/status/4"
  }],
  "proof": {
    "type": "JsonWebSignature2020",
    "created": "2023-08-28T13:25:35.827Z",
    "proofPurpose": "assertionMethod",
    "verificationMethod": "did:web:did.actor:alice#JWK2020",
    "jws": "eyJhbGciOiJQUzI1NiIsImI2NCI6ZmFsc2UsImNyaXQiOlsiYjY0Il19..Sq5VJHCFuIP-
cC86EknRnB91WQxNI5X4QtI0mMR3Xzl8VY5bYfOAtpEsejYeDUbi3Oed0VwQBRCqd11HL7NFF-
KH02D9I97nHBftBaXo8e0uWQTRk6TA8xq9oQRuNdnm15eR2zudOzKlH4ArXcBo-hUxUH6EH7YimT0Uu5NbsofN-5C2ovksogbugl-NkW3MKrGkAYzsyaEcgh-
vSiRwSl4vwE55sDkn16QgMtsccwo9PR0kzECHp8KQZTM3Nwnv4jNN-F3zP-3Vn0B-cm-
UgHPz1RYX6uKrc3A4TlJvk9rxQNfLbNot8ZaQBPoLMnd98bV6giNaGIbekVOUuBxUKg"
  }
}
```

## 5.2 Public Party Credential

The **Party Credential** contains data that can be publicly accessed and queried.

# 6 Party Credential Lifecycle

- **expired** if the expiration datetime is older than the current datetime or the certificate containing the key used to sign the claim has expired.
- **revoked**
- if the key used to sign the array is revoked.
- if the **credentialStatus** has the **statusPurpose** property set to **“revocation”** and the value of status at position **credentialIndex** is **true**
- **suspended** if the **credentialStatus** has the **statusPurpose** property set to **“suspension”** and the value of status at position **credentialIndex** is **true**
- **deprecated** if another verifiable credential with the same identifier and the same signature issuer has a newer issuance datetime.
- **active** only if none of the above.

# 7 Party Credential Status

Verifiable Credentials are a fundamental component of secure data and identity systems, enabling the issuance and presentation of trustworthy and tamper-proof credentials. However, in dynamic and evolving environments, it is crucial to establish mechanisms for the timely revocation or suspension of these credentials in case of compromised or outdated information.

The use of Credential Status Lists (CSL), specifically the W3C Verifiable Credentials Bitstring Status List, addresses this need by providing a standardized approach to manage and communicate the revocation status of verifiable Credentials.

When a Verifiable Credential is issued, the issuer has the option to embed a reference to the Credential Status List (CSL) entry associated with the credential. This reference, often in the form of a Uniform Resource Identifier (URI), enables relying parties (commonly verifiers) to promptly determine the current status of the credential's validity.

To validate the Verifiable Credential, the relying party retrieves the referenced Credential Status List entry using the provided URI. This entry contains information about the status of the credential, allowing to check if the credential is still valid, has been revoked/suspended, or has any other relevant status.

Relying parties can periodically update their local copy of the Credential Status List from trusted sources to ensure they possess the most current revocation status information. This practice prevents reliance on outdated or incorrect information, enhancing the overall security of the ecosystem.

**party_credential_revocation.json**

```
{
  "@context": [
    "https://www.w3.org/ns/credentials/v2",
    "https://w3id.org/gaia-x/development#"
  ],
  "id": "https://did.actor/alice/credentials/status/3",
  "type": ["VerifiableCredential", "BitstringStatusListCredential"],
  "issuer": "did:web:did.actor:alice",
  "issued": "2021-04-05T14:27:40Z",
  "credentialSubject": {
    "id": "https://example.com/status/3#list",
    "type": "BitstringStatusList",
    "statusPurpose": "revocation",
    "encodedList": "H4sIAAAAAAAAA-3BMQEAAADCoPVPbQwfoAAAAAAAAAAAAAAAAAAIC3AYbSVKsAQAAA"
  }
}
```

# 8 OpenID Connect for Verifiable Credentials

OpenID Connect for Verifiable Credentials (OIDC4VC) is a non-official name for collection of OpenID Connect specifications allowing Verifiable Credential and Verifiable Presentation exchanges.

The two main specifications that are relevant in the Gaia-X Ecosystem are: - OpenID Connect for Verifiable Credential Issuance (OIDC4VCI) - OpenID Connect for Verifiable Presentations (OIDC4VP)

## 8.1 OpenID Connect for Verifiable Credential Issuance

This specification is based on the OAuth 2.0 specification and allows an issuer to communicate with a holder and its wallet in order to issue Verifiable Credentials in a secure manner.

It's a great protocol for machine-to-end-user credential issuance but also from machine-to-machine issuance by using OAuth 2.0's battle tested and widely adopted standards.

Verifiable Credentials will be exchanged by using the VC-JWT format.

> The current OIDC4VCI specification is still a draft meaning that the implementation might evolve with time. The Gaia-X Lab is headed towards the first approved specification, meanwhile the latest draft will be implemented.

## 8.2 OpenID Connect for Verifiable Presentations

Just like OIDC4VCI, OIDC4VP is based on the OAuth 2.0 specification in order to allow a holder and its wallet to present one or multiple Verifiable Credentials to a verifier through a Verifiable Presentation.

This can also be a machine-to-end-user protocol in addition to a machine-to-machine protocol thanks to OAuth 2.0 standards.

Verifiable Presentations will be exchanged by using the VC-JWT format.

> The current OIDC4VP specification is still a draft meaning that the implementation might evolve with time. The Gaia-X Lab is headed towards the first approved specification, meanwhile the latest draft will be implemented.

## 8.3 Usage

Both these protocols will be used each time a Verifiable Credential needs to be produced or consumed by the Gaia-X Clearing House hence securing the exchange with an authentication and proof of possession mechanism.

## 8.4 Cloud/Enterprise Wallet

As most of the exchanges will be in a machine-to-machine environment, a cloud or enterprise wallet will be used although a specific solution hasn't been chosen to this date.

# 9 Signature Credential

## 9.1 Multiple Signatures using SignatureCredential specializations

The **SignatureCredential** in the trust model defined by the ICAM Document, represents a general purpose and machine readable *signature* that Participants are asked to provide in several contexts, one of the most important of these contexts is represented by the Data Transaction section of the Data Exchange Document(version 23.11) where this credential will be used to sign **Data Usage Agreement** and **Data Product Usage Contract**.

The concept behind this credential is the same as the credentials issued by the compliance service, to issue Participant Credentials and Service Offering Credentials (using the “credentialSubject” claim), where the **credentialSubject** consists in:

| Attribute | Type.Value/Voc | Mandatory | Comment |
|---|---|---|---|
| `type` | string | Yes | Indicating the *type* of the subject to be Signed |
| `id` | URI | Yes | A resolvable single URI as defined in the VC Specification identifying the subject to be Signed |
| `digestSRI` | String | Yes | Integrity of related Resources as defined in W3C VC Data Model v2.0 to ensure that referenced resource is not changed/modified/tempered(still identical in time) |

Another important factor is represented by the fact that, issuing and signing this credential with a **verificationMethod** bound to a certificate that has legal relevance (e.g eIDAS) gives it the same level of trust, enabling the possibility to check the `gx:legalValidity` property of the **Signature Check Type** (see `gx:signers` property).

### 9.1.1 SignedAgreementCredential

The **SignedAgreementCredential** is a specialization of **SignatureCredential** that represents a *Revocable Signed Agreement* given by a participant to a specific subject (e.g. Data Usage Agreement, GDPR Agreement, etc.) and the only difference stands in the fact that it can be revoked.

### 9.1.2 Data Usage Agreement Example

In a Data Transaction the **Data Provider**(did:web:provider.com) and a **Data Consumer**(did:web:consumer.com) must agree to a **Data Usage Agreement** (**id** “https://provider.com/data-usage-contract.654321”) and this is done treating “Signatures” as a **Verifiable Credential** that each Participant can issue and sign. In this way, when all Participants have issued their SignatureCredential referencing the contract it will be possible to create Verifiable Presentations that contain all the Signatures.

Here is an example of how CredentialSignature can be used:

```
{
  "@context": [
    "https://www.w3.org/ns/credentials/v2",
    "https://w3id.org/gaia-x/development#"
  ],
  "type": [
    "VerifiableCredential"
  ],
  "id": "https://consumer.com/data-usage-contract-signatures.123456",
  "issuer": "did:web:consumer.com",
  "validFrom": "2023-07-28T12:31:49.074Z",
  "validUntil": "2023-10-26T12:31:49.074Z",
  "credentialSubject": {
    "type": "gx:dataUsageAgreement",
    "id": "https://provider.com/data-usage-contract.654321",
    "digestSRI": "sha384-lHKDHh0msc6pRx8PhDOMkNtSI8bOfsp4giNbUrw71nXXLf13nTqNJoRp3Nx+ArVK"
  },
  "credentialStatus": {
    "id": "https://consumer.com/status/1#127",
    "type": "BitstringStatusListEntry",
    "statusPurpose": "revocation",
    "statusListIndex": "127",
    "statusListCredential": "https://consumer.com/credentials/status/1"
  }
}
```

```
{
 "@context": [
  "https://www.w3.org/ns/credentials/v2",
  "https://w3id.org/gaia-x/development#"
 ],
 "type": [
  "VerifiableCredential"
 ],
 "id": "https://provider.com/data-usage-contract-signatures.987",
 "issuer": "did:web:provider.com",
 "validFrom": "2023-07-25T10:31:49.074Z",
 "validUntil": "2023-10-25T10:31:49.074Z",
 "credentialSubject": {
  "type": "gx:dataUsageAgreement",
  "id": "https://provider.com/data-usage-contract.654321",
  "digestSRI": "sha384-lHKDHh0msc6pRx8PhDOMkNtSI8bOfsp4giNbUrw71nXXLf13nTqNJoRp3Nx+ArVK"
 },
 "credentialStatus": {
  "id": "https://provider.com/status/3#125221",
  "type": "BitstringStatusListEntry",
  "statusPurpose": "revocation",
  "statusListIndex": "125221",
  "statusListCredential": "https://provider.com/credentials/status/3"
 }
}
```

## 9.2 Multiple Signatures using Proof Set and Proof Chain

In cases where a credential must be signed by multiple parties in a non-revocable way, whether the signing order matters or not, the proof chain and proof set mechanism specified in the VC Data Integrity standard must be used. This mechanism is particularly essential for contract negotiation and compliance services credentials involving many entities.

To ensure cryptographic integrity and crypto agility, the signature must not be embedded inside the credentialSubject. The definition provided by the Gaia-X Ontology is reserved for special use cases and should not be used in such scenarios.

For compatibility reasons the multiple-signed VC should be W3C credential and wrapped by a VC-JWT credential signed by the issuer

# 10 Trustframework implementation

## 10.1 Trust Framework Implementation

This section is dedicated to illustrate how to use, extend, and customize the semantic model of the trust framework. This is achievable in 2 steps:

1. define and declare the trust anchor through the use of **TrustAnchorCredential specialisation** used to define trust issuers, the semantic model defined in the scope of the trust anchor(vocabularies), a preferred trusted list tecnology/mode.
2. define the delegation chain through the use of **PartyCredential specialisation**

## 10.2 Trust Anchor Credential specialisation examples

Several specialisations of the **TrustAnchorCredential** can be defined, for example:

- **Organization Trust Anchor** – a self-issued credential that entitles an organization to define:
- **scope** - *Organization Credential Management* (OCM).
- **trusted issuer** - the organisation itself.
- **vocabularies** - defines the semantics of *Roles/Identity Attributes* and **Domain Specific Credentials** that are valid in the defined scope.
- **trusted list** - to assign and revoke *Roles/Identity Attributes* and **Domain Specific Credentials** to its parties (users, natural persons, endpoint services, etc) by issuing **PartyCredentials**.
- **Ecosystem Trust Anchor** - a self-issued credential that entitles an Ecosystem operator to define:
- **scope** - manage an *Ecosystem* (onboarding/offboarding/role assignment etc.).
- **trusted issuer** - the Ecosystem itself.
- **vocabularies** - defines the semantic of *Roles/Identity Attributes* and **Domain Specific Credentials** valid in the defined scope.
- **trusted list** - to assign and revoke *Roles/Identity Attributes* and **Domain Specific Credentials** to its members (other Participants) issuing **MembershipPartyCredentials**.
- **Gaia-X Compliance Trust Anchor** - a Gaia-X-issued credential that entitles an organisation to issue attestations about specific claims in the context of Gaia-X Compliance.
- **scope** - issue attestations used to attest Gaia-X Compliance.
- **trusted issuer** - the organisation accredited by Gaia-X to issue attestations in the defined scope.
- **vocabularies** - defines the semantic applicable to issue attestations in the defined scope.
- **trusted list** - to assign and revoke attestations issued to Gaia-X Providers.

## 10.3 Party Credential Specialisation examples

Here, some of the possible Party Credential specialisations are defined.

### 10.3.1 Natural Person Party Credential

This credential is issued by a Participant to entitle a natural person (usually one of his users/employees) to interact with other Relying Parties belonging to other Participants.

This is how `Natural Person Party Credential` will be defined by the following attributes in addition to Party Credential attributes:

| Attribute | Type.Value/Voc | Mandatory | Comment |
|---|---|---|---|
| `gx:givenName` | String | Yes | Name of the natural person |
| `gx:surname` | String | Yes | Surname of the natural person |

### 10.3.2 Legal Person Party Credential

This credential is issued by a Legal Person Participant to entitle another legal person (usually one of his users ) to interact with other Relying Parties belonging to other Participants on its behalf.

This is how `Legal Person Party Credential` will be defined by the following attributes in addition to Party Credential attributes:

| Attribute | Type.Value/Voc | Mandatory | Comment |
|---|---|---|---|
| `gx:organizationIdentifier` | String | Yes | Organization Identifier used in the eIDAS rules |

### 10.3.3 Service Party Credential

This credential is issued by a Participant to entitle an automated service (usually an automated process) to interact with other Relying Parties belonging to other Participants.

This is how `Service Party Credential` will be defined by the following attributes in addition to Party Credential attributes

| Attribute | Type.Value/Voc | Mandatory | Comment |
|---|---|---|---|
| `gx:baseURL` | URI | Yes | The base URL endpoint where the service is accessible |

### 10.3.4 Membership Party Credential

This credential is issued by a LegalParticipant that runs an ecosystem to another Participant in order to attest his Membership status.

## 10.4 Access rights delegation example - Employee Authentication

### 10.4.1 Problem Statement

Consider the scenario where a company or a department wants to provide its employees with secure access to various cloud services (e.g., email, document storage, collaboration tools) without relying on a central identity provider. The ecosystem uses the decentralized identity and access management based on Self-Sovereign Identity (SSI) to enhance security, privacy, and flexibility. The goal is to design a system where employees can authenticate themselves to an ecosystem where the company/department is member of and access cloud services based on their roles and entitlements, without the need for a central identity provider. For security considerations, with only one single step all issued access rights must be revoked when an employee leaves the company or changes roles. The management of the access rights should be done by the respective manager and the HR department and not from a central administrator.

### 10.4.2 Types of Credentials and Issuers involved

#### 10.4.2.1 Employee Credential (specialised PartyCredential):

The company HR department is managing the employees and therefore issues an `EmployeeCredential` to each employee.

This employee credential contains the employee's unique identifier, role, department and unique company ID (unique identifier used in the onboarding phase of the company to the ecosystem). The HR department is the issuer of this credential, and it serves as the foundation for the employee's identity within the organization. The `EmployeeCredential` is used to verify the employee's identity and affiliation with the company. The Company ID should be registered in the membership trustlist of the ecosystem and in parallel used in the separate issued membership credential. The HR must revoke the employee credential in case he/she leaves the company and the employee should no longer be able to access any cloud services on behalf of the company.

- **Type**: `EmployeeCredential (Specialised Party-Credential)`
- **Issuer**: HR Department
- **Subject**: Employee
- **Company ID**: Global unique identifier for the company or organization (eg. GLEIF ID, VAT,), same across all employees. The Company ID should be used in the membership trustlist and membership credential used to verify the authenticity of the credential.
- **Emp Attributes**: Like, Employee ID, Name, Role, Department

The `EmployeeCredential` is a prerequisite for receiving `AccessEntitlementCredentials` and is used to establish the employee's identity within the organization. The Employee Credential type and the attributes must be standardized

### 10.4.3 Access Entitlement Credentials:

- **Type**: `AccessEntitlementCredential (for each access right individual credentials must be issued)`
- **Issuer**: Employee Manager
- **Attributes**: Employee ID, Access Rights (specific cloud services the employee can access)
- **Purpose**: Specifies the services the employee is entitled to access.

- **Scope**: Issued by the respective manager to grant access to specific services based on the employee's role and responsibilities.
- **Revocation**: Manager who can revoke the credential to dynamically manage access rights. <!–could they differ from the Issuer/Employee manager?>
- **Employee ID**: The employee's `EmployeeCredential` is a prerequisite for receiving `AccessEntitlementCredentials`.

The `AccessEntitlementCredential` is issued by the respective manager to grant access to specific cloud services based on the employee's role and responsibilities. The manager can revoke these credentials to dynamically manage access rights. The `EmployeeCredential` is a prerequisite for receiving `AccessEntitlementCredentials`. the Manager must be entitled to issue such credential, therefore the manager must have a valid `ManagerCredential` issued by the HR department. In addition, the manager must be a managing member of the ecosystem and the company ID must be part of the credential to ensure the manager is a valid member of the company. Therefore, the manager is registered in the membership trustlist of the specific ecosystem services and in parallel he holds a valid issued membership credential.

### 10.4.4 Authentication Process

1. **Employee Presents Credentials**: When accessing a service, the employee presents its `EmployeeCredential` and `AccessEntitlementCredential`.
2. **Verification**: The service verifies:
3. The authenticity of the credentials (ensuring they are issued by the HR department and the respective manager).
4. The validity of the credentials (not revoked).
5. The attributes match the access control policies (e.g., the employee's role and entitlements align with the service being accessed).

### 10.4.5 Revocation

- **Revocation List or Registry**: The HR department and managers can revoke credentials by adding them to a revocation list or registry that the services check during the authentication process.
- **Issuer**: HR Department for `EmployeeCredential`, Manager for `AccessEntitlementCredential`.
- **Mechanism**: Could use a decentralized identifier (DID) revocation mechanism or a centralized revocation service, depending on the architecture.

### 10.4.6 Implementation Considerations

- **Decentralized Identifiers (DIDs)**: Use DIDs for employees, HR, and managers to ensure a decentralized and self-sovereign identity system.
- **Verifiable Data Registry**: Implement a verifiable data registry to store DIDs for validation.
- **Privacy**: Ensure that the system only reveals the minimum necessary information for authentication and access control to protect employee privacy.
- **Interoperability**: Design the credential schema to be interoperable with W3C Verifiable Credentials standards and compatible with the cloud services' access control mechanisms.
- **Trust** An ecosystem may have many trustlists (Gaia-X , XFCS Ecosystem Member, Service Manager).

# 11 Changelog

## 11.1 2024 July release (24.07)

- Updated chapter “Credential Format”
- New chapter “Trust Anchor Credential and Party Credential”
- New chapter “OpenID Connect for Verifiable Credentials”
- New chapter “Signature Credential”
- New chapter “Trust Framework implementation”, containing Trust Anchor Credential specialisation examples and Party Credential specialisation examples and an access rights delegation example.