Web Authorization Protocol
Internet-Draft
Intended status: Standards Track
Expires: 26 October 2026

O. Terbu
MATTR
D. Fett
Authlete Inc.
B. Campbell
Ping Identity
24 April 2026

SD-JWT-based Verifiable Digital Credentials (SD-JWT VC)
draft-ietf-oauth-sd-jwt-vc-16

Abstract

This specification describes data formats as well as validation and processing rules to express Verifiable Digital Credentials with JSON payloads with and without selective disclosure based on the SD-JWT format.

Discussion Venues

This note is to be removed before publishing as an RFC.

Discussion of this document takes place on the Web Authorization Protocol Working Group mailing list (oauth@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/oauth/.

Source for this draft and an issue tracker can be found at https://github.com/oauth-wg/oauth-sd-jwt-vc.

Status of This Memo

Table of Contents

1. Introduction

1.1. Issuer-Holder-Verifier Model

In the so-called Issuer-Holder-Verifier Model, Issuers issue Verifiable Digital Credentials to a Holder, who can then present the credentials to Verifiers. Verifiable Digital Credentials are cryptographically secured statements about a Subject, typically the Holder.

Figure 1 depicts the Issuer-Holder-Verifier model.

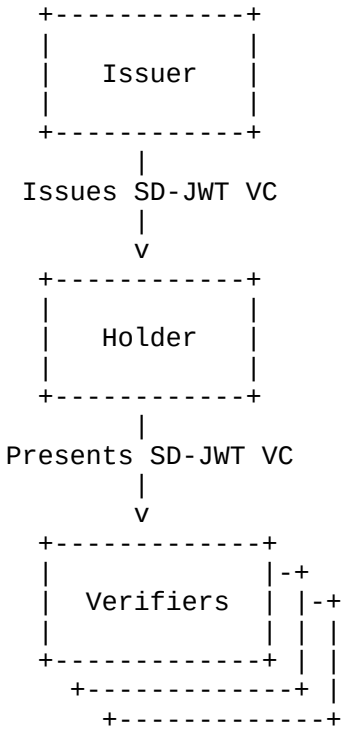

Figure 1: Abstract Depiction of the Issuer-Holder-Verifier Model

Verifiers can check the authenticity of the data in Verifiable Digital Credentials. Verifiers can also optionally enforce Key Binding, which requires the Holder to prove they are the intended Holder of the Verifiable Digital Credential. This proof is typically done by demonstrating possession of a cryptographic key referenced in the credential. This process is further described in [RFC9901].

1.2. SD-JWT as a Credential Format

JSON Web Tokens (JWTs) [RFC7519] can in principle be used to express Verifiable Digital Credentials in a way that is easy to understand and process as it builds upon established web primitives.

Selective Disclosure JWT (SD-JWT) [RFC9901] is a specification that introduces conventions to support selective disclosure for JWTs: For an SD-JWT document, a Holder can decide which claims to release (within bounds defined by the Issuer).

SD-JWT is a superset of JWT. It can also be used when there are no selectively disclosable claims. Furthermore, SD-JWT supports JWS JSON serialization, which is useful for long-term archiving and multi-signatures. However, SD-JWT itself does not define the claims that must be used within the payload or their semantics.

This specification uses SD-JWT and the well-established JWT content rules and extensibility model as basis for representing Verifiable Digital Credentials with JSON payloads. These Verifiable Digital Credentials are called SD-JWT VCs. The use of selective disclosure in SD-JWT VCs is optional.

SD-JWTs VC can contain claims that are registered in "JSON Web Token Claims" registry as defined in [RFC7519], as well as public and private claims.

1.3. Requirements Notation and Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

1.4. Terms and Definitions

This specification uses the terms "Holder", "Issuer", "Verifier", "Disclosure", "Selectively Disclosable JWT (SD-JWT)", "Key Binding", "Key Binding JWT (KB-JWT)", "Selectively Disclosable JWT with Key Binding (SD-JWT+KB)" defined by [RFC9901].

Consumer: An application using the Type Metadata specified in Section 4 is called a Consumer. This typically includes Issuers, Verifiers, and Holders.

Publisher: An entity that publishes Type Metadata or other auxiliary documents referenced by an SD-JWT VC (for example via a vct URI), but that is not necessarily the Issuer of the SD-JWT VC. A Publisher can be a standardization body, community, ecosystem authority, or any other party defining credential types or associated metadata.

Verifiable Digital Credential: An assertion with claims about a Subject that is cryptographically secured by an Issuer (usually by a digital signature).

SD-JWT-based Verifiable Digital Credential (SD-JWT VC): A Verifiable Digital Credential encoded using the format defined in [RFC9901]. It may or may not contain selectively disclosable claims. Note that the three-word term “Verifiable Digital Credential” is used to distinguish it from the W3C Verifiable Credentials Data Model and to better align with preferred terminology emerging elsewhere

(see, for example, [NIST-BLOG.VDCE]). However, the “D” is omitted in the short form “SD-JWT VC” in order to preserve the name that has been widely used to refer to this specification since its inception.

Unsecured Payload of an SD-JWT VC: A JSON object containing all selectively disclosable and non-selectively disclosable claims of the SD-JWT VC. The Unsecured Payload acts as the input JSON object to issue an SD-JWT VC complying to this specification.

2. Verifiable Digital Credentials based on SD-JWT

This section defines encoding, validation and processing rules for SD-JWT VCs.

2.1. Media Type

SD-JWT VCs compliant with this specification MUST use the media type application/dc+sd-jwt, where the base base subtype name dc stands for "digital credential".

2.2. Data Format

Issuers MUST encode an SD-JWT VC using the SD-JWT format defined in Section 4 or Section 8 of [RFC9901]. By default, the format defined in Section 4 of [RFC9901] is used, whereas support for the JWS JSON Serialization in Section 8 of [RFC9901] is OPTIONAL.

Note that in some cases, an SD-JWT VC MAY have no selectively disclosable claims, and therefore the encoded SD-JWT will not contain any Disclosures.

A presentation of an SD-JWT VC MUST be encoded as an SD-JWT or as an SD-JWT+KB defined in Section 4 or Section 8 of [RFC9901]. By default, the format defined in Section 4 of [RFC9901] is used, whereas support for the JWS JSON Serialization in Section 8 of [RFC9901] is OPTIONAL.

2.2.1. JOSE Header

This section defines JWT header parameters for the SD-JWT component of the SD-JWT VC.

The Issuer MUST include the typ header parameter in the SD-JWT. The typ value MUST use dc+sd-jwt. This indicates that the payload of the SD-JWT contains plain JSON and follows the rules defined in this specification. It further indicates that the SD-JWT is an SD-JWT component of an SD-JWT VC.

Figure 2 is an example of a decoded SD-JWT header:

```
{
  "alg": "ES256",
  "typ": "dc+sd-jwt"
}
```

Figure 2: Decoded SD-JWT VC Header

Note that this draft used vc+sd-jwt as the value of the typ header from its inception in July 2023 until November 2024 when it was changed to dc+sd-jwt to avoid conflict with the vc media type name registered by the W3C's Verifiable Credentials Data Model draft. In order to facilitate a minimally disruptive transition, both vc+sd-jwt and dc+sd-jwt should be accepted as the value of the typ header for a reasonable transitional period.

2.2.2. JWT Claims Set

This section defines the claims that can be included in the payload of SD-JWT VCs.

2.2.2.1. Verifiable Digital Credential Type - vct Claim

This specification defines the new JWT claim vct (for verifiable credential type). Its value MUST be a case-sensitive string serving as an identifier for the type of the SD-JWT VC. The vct value MUST be a Collision-Resistant Name as defined in Section 2 of [RFC7515].

A type is associated with rules defining which claims are permitted or required to appear in the Unsecured Payload of the SD-JWT VC and whether selective disclosure is permitted, necessary, or prohibited for those claims.

This specification does not define any vct values; instead it is expected that ecosystems using SD-JWT VCs define such values including the semantics of the respective claims and associated rules (e.g., policies for issuing and validating credentials beyond what is defined in this specification).

For example, a value of https://credentials.example.com/identity_credential can be associated with rules that define that at least the registered JWT claims given_name, family_name, birthdate, and address must appear in the Unsecured Payload. Additionally, the registered JWT claims email and phone_number, and the private claims is_over_18, is_over_21, and is_over_65 may be used. The type might also indicate that any of the aforementioned claims can be selectively disclosable. The content in Figure 3 shows that vct value used to express the example type.

```
{
  "vct": "https://credentials.example.com/identity_credential",
  "given_name": "Russ",
  "family_name": "Lasky",
  "birthdate": "1983-07-29",
  "address": {
    "street_address": "285 W Huntington Dr.",
    "locality": "Arcadia",
    "region": "CA",
    "postal_code": "91007"
  },
  "email": "sorryfolks@ww.example",
  "exp": 1777707000,
  "cnf": {
    "jwk": {
      "kty": "EC", "crv": "P-256",
      "x": "TCAER19Zvu3OHF4j4W4vfSVoHIP1ILilDls7vCeGemc",
      "y": "ZxjiWWbZMQGHVWKVQ4hbSIirsVfuecCE6t4jT9F2HZQ"
    }
  }
}
```

Figure 3: Example Unsecured Payload with vct

The vct value also effectively identifies the version of the credential type definition, as it ties a particular instance of a credential to a specific structure, set of semantics, and rules. When evolving a credential type without updating the version, changes to the structure or meaning of the associated claims need to be made in a way that preserves compatibility with existing implementations.

If a change alters the meaning of existing content, adds new required claims, removes previously required elements, or otherwise introduces incompatibilities, it is generally advisable to treat that as a new version of the credential type and to convey it using a new vct value. This allows different versions of a credential type to coexist and helps ensure that participants interpret credentials consistently. For example, if such a need came about for the hypothetical type urn:example:eudi:pid:aendgard:1, a new version could be defined using a 'vct' of urn:example:eudi:pid:aendgard:2.

2.2.2.2. Registered JWT Claims

SD-JWT VCs MAY use any claim registered in the "JSON Web Token Claims" registry as defined in [RFC7519].

The following registered JWT claims are used within the SD-JWT component of the SD-JWT VC and MUST NOT be included in the Disclosures, i.e., cannot be selectively disclosed:

* iss: OPTIONAL. As defined in Section 4.1.1 of [RFC7519] this claim explicitly indicates the Issuer of the Verifiable Digital Credential when it is not conveyed by other means (e.g., the subject of the end-entity certificate of an x5c header).
* nbf: OPTIONAL. The time before which the Verifiable Digital Credential MUST NOT be accepted before validating. See [RFC7519] for more information.
* exp: OPTIONAL. The expiry time of the Verifiable Digital Credential after which the Verifiable Digital Credential is no longer valid. See [RFC7519] for more information.
* cnf: OPTIONAL unless cryptographic Key Binding is to be supported, in which case it is REQUIRED. Contains the confirmation method identifying the proof of possession key as defined in [RFC7800]. It is RECOMMENDED that this contains a JWK as defined in Section 3.2 of [RFC7800]. For proof of cryptographic Key Binding, the KB-JWT in the presentation of the SD-JWT MUST be secured by the key identified in this claim.
* vct: REQUIRED. The type of the Verifiable Digital Credential, e.g., https://credentials.example.com/identity_credential, as defined in Section 2.2.2.1.
* vct#integrity: OPTIONAL. The hash of the Type Metadata document to provide integrity as defined in Section 5.
* status: OPTIONAL. The information on how to read the status of the Verifiable Credential. See [I-D.ietf-oauth-status-list] for more information. When the status claim is present and using the status_list mechanism, the associated Status List Token MUST be in JWT format.

The following registered JWT claims are used within the SD-JWT component of the SD-JWT VC and MAY be included in Disclosures, i.e., can be selectively disclosed:

* sub: OPTIONAL. The identifier of the Subject of the Verifiable Digital Credential. The Issuer MAY use it to provide the Subject identifier known by the Issuer. There is no requirement for a binding to exist between sub and cnf claims.
* iat: OPTIONAL. The time of issuance of the Verifiable Digital Credential. See [RFC7519] for more information.

2.2.2.3. Public and Private JWT claims

Additionally, any public and private claims as defined in Sections 4.2 and 4.3 of [RFC7519] MAY be used.

Binary data in claims SHOULD be encoded as data URIs as defined in [RFC2397]. Exceptions can be made when data formats are used that already define a text encoding suitable for use in JSON or where an established text encoding is commonly used. For example, images would make use of data URIs, whereas hash digests in base64 encoding do not need to be encoded as such.

An example of a claim containing binary data encoded as a data URI is shown in Appendix B.1.

2.2.2.4. SD-JWT VC without Selectively Disclosable Claims

An SD-JWT VC MAY have no selectively disclosable claims. In that case, the SD-JWT VC MUST NOT contain the _sd claim in the JWT body. It also MUST NOT have any Disclosures.

2.3. Example

2.3.1. Issuance

Figure 4 is an example of the user data of an Unsecured Payload of an SD-JWT VC.

```
{
  "vct": "https://credentials.example.com/identity_credential",
  "given_name": "John",
  "family_name": "Doe",
  "email": "johndoe@example.com",
  "phone_number": "+1-202-555-0101",
  "address": {
    "street_address": "123 Main St",
    "locality": "Anytown",
    "region": "Anystate",
    "country": "US"
  },
  "birthdate": "1940-01-01",
  "is_over_18": true,
  "is_over_21": true,
  "is_over_65": true
}
```

Figure 4: Unsecured Payload Example

Figure 5 shows how the Unsecured Payload above can be used in an SD-JWT where the resulting SD-JWT VC contains only claims about the Subject that are selectively disclosable:

```
{
  "_sd": [
    "09vKrJMOlyTWM0sjpu_pdOBVBQ2M1y3KhpH515nXkpY",
    "2rsjGbaC0ky8mT0pJrPioWTq0_daw1sX76poUlgCwbI",
    "EkO8dhW0dHEJbvUHlE_VCeuC9uRELOieLZhh7XbUTtA",
    "IlDzIKeiZdDwpqpK6ZfbyphFvz5FgnWa-sN6wqQXCiw",
    "JzYjH4svliH0R3PyEMfeZu6Jt69u5qehZo7F7EPYlSE",
    "PorFbpKuVu6xymJagvkFsFXAbRoc2JGlAUA2BA4o7cI",
    "TGf4oLbgwd5JQaHyKVQZU9UdGE0w5rtDsrZzfUaomLo",
    "jdrTE8YcbY4EifugihiAe_BPekxJQZICeiUQwY9QqxI",
    "jsu9yVulwQQlhFlM_3JlzMaSFzglhQG0DpfayQwLUK4"
  ],
  "iss": "https://example.com/issuer",
  "iat": 1683000000,
  "exp": 1883000000,
  "vct": "https://credentials.example.com/identity_credential",
  "_sd_alg": "sha-256",
  "cnf": {
    "jwk": {
      "kty": "EC",
      "crv": "P-256",
      "x": "TCAER19Zvu3OHF4j4W4vfSVoHIP1ILilDls7vCeGemc",
      "y": "ZxjiWWbZMQGHVWKVQ4hbSIirsVfuecCE6t4jT9F2HZQ"
    }
  }
}
```


Figure 5: Unsecured Payload Example

Note that a cnf claim has been added to the SD-JWT payload to express the confirmation method of the Key Binding.

The following are the Disclosures belonging to the SD-JWT payload above:

Claim given_name:

* SHA-256 Hash: jsu9yVulwQQlhFlM_3JlzMaSFzglhQG0DpfayQwLUK4
* Disclosure:
  WyIyR0xDNDJzS1F2ZUNmR2ZyeU5STjl3IiwgImdpdmVuX25hbWUiLCAiSm9o
  biJd
* Contents: ["2GLC42sKQveCfGfryNRN9w", "given_name", "John"]

Claim family_name:

* SHA-256 Hash: TGf4oLbgwd5JQaHyKVQZU9UdGE0w5rtDsrZzfUaomLo

* Disclosure:
  WyJlbHVWNU9nM2dTTklJOEVZbnN4QV9BIiwgImZhbWlseV9uYW1lIiwgIkRv
  ZSJd
* Contents: ["eluV5Og3gSNII8EYnsxA_A", "family_name", "Doe"]

Claim email:

* SHA-256 Hash: JzYjH4svliH0R3PyEMfeZu6Jt69u5qehZo7F7EPYlSE
* Disclosure:
  WyI2SWo3dE0tYTVpVlBHYm9TNXRtdlZBIiwgImVtYWlsIiwgImpvaG5kb2VA
  ZXhhbXBsZS5jb20iXQ
* Contents: ["6Ij7tM-a5iVPGboS5tmvVA", "email",
  "johndoe@example.com"]

Claim phone_number:

* SHA-256 Hash: PorFbpKuVu6xymJagvkFsFXAbRoc2JGlAUA2BA4o7cI
* Disclosure:
  WyJlSThaV205UW5LUHBOUGVOZW5IZGhRIiwgInBob25lX251bWJlciIsICIr
  MS0yMDItNTU1LTAxMDEiXQ
* Contents: ["eI8ZWm9QnKPpNPeNenHdhQ", "phone_number",
  "+1-202-555-0101"]

Claim address:

* SHA-256 Hash: IlDzIKeiZdDwpqpK6ZfbyphFvz5FgnWa-sN6wqQXCiw
* Disclosure:
  WyJRZ19PNjR6cUF4ZTQxMmExMDhpcm9BIiwgImFkZHJlc3MiLCB7InN0cmVl
  dF9hZGRyZXNzIjogIjEyMyBNYWluIFN0IiwgImxvY2FsaXR5IjogIkFueXRv
  d24iLCAicmVnaW9uIjogIkFueXN0YXRlIiwgImNvdW50cnkiOiAiVVMifV0
* Contents: ["Qg_O64zqAxe412a108iroA", "address", {"street_address":
  "123 Main St", "locality": "Anytown", "region": "Anystate",
  "country": "US"}]

Claim birthdate:

* SHA-256 Hash: jdrTE8YcbY4EifugihiAe_BPekxJQZICeiUQwY9QqxI
* Disclosure:
  WyJBSngtMDk1VlBycFR0TjRRTU9xUk9BIiwgImJpcnRoZGF0ZSIsICIxOTQw
  LTAxLTAxIl0
* Contents: ["AJx-095VPrpTtN4QMOqROA", "birthdate", "1940-01-01"]

Claim is_over_18:

* SHA-256 Hash: 09vKrJMOlyTWM0sjpu_pdOBVBQ2M1y3KhpH515nXkpY
* Disclosure:
  WyJQYzMzSk0yTGNoY1VfbEhnZ3ZfdWZRIiwgImlzX292ZXJfMTgiLCB0cnVl
  XQ

* Contents: ["Pc33JM2LchcU_lHggv_ufQ", "is_over_18", true]

Claim is_over_21:

* SHA-256 Hash: 2rsjGbaC0ky8mT0pJrPioWTq0_daw1sX76poUlgCwbI
* Disclosure:
  WyJHMDJOU3JRZmpGWFE3SW8wOXN5YWpBIiwgImlzX292ZXJfMjEiLCB0cnVl
  XQ
* Contents: ["G02NSrQfjFXQ7Io09syajA", "is_over_21", true]

Claim is_over_65:

* SHA-256 Hash: EkO8dhW0dHEJbvUHlE_VCeuC9uRELOieLZhh7XbUTtA
* Disclosure:
  WyJsa2x4RjVqTVlsR1RQVW92TU5JdkNBIiwgImlzX292ZXJfNjUiLCB0cnVl
  XQ
* Contents: ["lklxF5jMYlGTPUovMNIvCA", "is_over_65", true]

The SD-JWT and the Disclosures would then be serialized by the Issuer into the format shown in Figure 6 for issuance to the Holder.

eyJhbGciOiAiRVMyNTYiLCAidHlwIjogImRjK3NkLWp3dCIsICJraWQiOiAiZG9jLXNp
Z25lci0wNS0yNS0yMDIyIn0.eyJfc2QiOiBbIjA5dktySk1PbHlUV00wc2pwdV9wZE9C
VkJRMk0xeTNLaHBINTE1blhrcFkiLCAiMnJzakdiYUMwa3k4bVQwcEpyUGlvV1RxMF9k
YXcxc1g3NnBvVWxnQ3diSSIsICJFa084ZGhXMGRIRUpidlVIbEVfVkNldUM5dVJFTE9p
ZUxaaGg3WGJVVHRBIiwgIklsRHpJS2VpWmREd3BxcEs2WmZieXBoRnZ6NUZnbldhLXNO
NndxUVhDaXciLCAiSnpZakg0c3ZsaUgwUjNQeUVNZmVadTZKdDY5dTVxZWhabzdGN0VQ
WWxTRSIsICJQb3JGYnBLdVZ1Nnh5bUphZ3ZrRnNGWEFiUm9jMkpHbEFVQTJCQTRvN2NJ
IiwgIlRHZjRvTGJnd2Q1SlFhSHlLVlFaVTlVZEdFMHc1cnREc3JaemZVYW9tTG8iLCAi
amRyVEU4WWNiWTRFaWZ1Z2loaUFlX0JQZWt4SlFaSUNlaVVRd1k5UXF4SSIsICJqc3U5
eVZ1bHdRUWxoRmxNXzNKbHpNYVNGemdsaFFHMERwZmF5UXdMVUs0Il0sICJpc3MiOiAi
aHR0cHM6Ly9leGFtcGxlLmNvbS9pc3N1ZXIiLCAiaWF0IjogMTY4MzAwMDAwMCwgImV4
cCI6IDE4ODMwMDAwMDAsICJ2Y3QiOiAiaHR0cHM6Ly9jcmVkZW50aWFscy5leGFtcGxl
LmNvbS9pZGVudGl0eV9jcmVkZW50aWFsIiwgIl9zZF9hbGciOiAic2hhLTI1NiIsICJj
bmYiOiB7Imp3ayI6IHsia3R5IjogIkVDIiwgImNydiI6ICJQLTI1NiIsICJ4IjogIlRD
QUVSMTladnUzT0hGNGo0VzR2ZlNWb0hJUDFJTGlsRGxzN3ZDZUdlbWMiLCAieSI6ICJa
eGppV1diWk1RR0hWV0tWUTRoYlNJaXJzVmZ1ZWNDRTZ0NGpUOUYySFpRIn19fQ.HGNbT
pc51R_6ssbIbtXkT9FutnyoF2ekNfIhY4983amNK3B6MhnqTA5VKtgOlkrPek9ZF4VUy
6fHAwosePpTRQ~WyIyR0xDNDJzS1F2ZUNmR2ZyeU5STjl3IiwgImdpdmVuX25hbWUiLC
AiSm9obiJd~WyJlbHVWNU9nM2dTTklJOEVZbnN4QV9BIiwgImZhbWlseV9uYW1lIiwgI
kRvZSJd~WyI2SWo3dE0tYTVpVlBHYm9TNXRtdlZBIiwgImVtYWlsIiwgImpvaG5kb2VA
ZXhhbXBsZS5jb20iXQ~WyJlSThaV205UW5LUHBOUGVOZW5IZGhRIiwgInBob25lX251b
WJlciIsICIrMS0yMDItNTU1LTAxMDEiXQ~WyJRZ19PNjR6cUF4ZTQxMmExMDhpcm9BIi
wgImFkZHJlc3MiLCB7InN0cmVldF9hZGRyZXNzIjogIjEyMyBNYWluIFN0IiwgImxvY2
FsaXR5IjogIkFueXRvd24iLCAicmVnaW9uIjogIkFueXN0YXRlIiwgImNvdW50cnkiOi
AiVVMifV0~WyJBSngtMDk1VlBycFR0TjRRTU9xUk9BIiwgImJpcnRoZGF0ZSIsICIxOT
QwLTAxLTAxIl0~WyJQYzMzSk0yTGNoY1VfbEhnZ3ZfdWZRIiwgImlzX292ZXJfMTgiLC
B0cnVlXQ~WyJHMDJOU3JRZmpGWFE3SW8wOXN5YWpBIiwgImlzX292ZXJfMjEiLCB0cnV
lXQ~WyJsa2x4RjVqTVlsR1RQVW92TU5JdkNBIiwgImlzX292ZXJfNjUiLCB0cnVlXQ~

Figure 6: SD-JWT VC Example

2.3.2. Presentation

Figure 7 is an example of a presentation of the SD-JWT shown in Section 2.3 including a KB-JWT. In this presentation, the Holder provides only the Disclosures for the address and is_over_65 claims. Other claims are not disclosed to the Verifier.

```
eyJhbGciOiAiRVMyNTYiLCAidHlwIjogImRjK3NkLWp3dCIsICJraWQiOiAiZG9jLXNp
Z25lci0wNS0yNS0yMDIyIn0.eyJfc2QiOiBbIjA5dktySk1PbHlUV00wc2pwdV9wZE9C
VkJRMk0xeTNLaHBINTE1blhrcFkiLCAiMnJzakdiYUMwa3k4bVQwcEpyUGlvV1RxMF9k
YXcxc1g3NnBvVWxnQ3diSSIsICJFa084ZGhXMGRIRUpidlVIbEVfVkNldUM5dVJFTE9p
ZUxaaGg3WGJVVHRBIiwgIklsRHpJS2VpWmREd3BxcEs2WmZieXBoRnZ6NUZnbldhLXNO
NndxUVhDaXciLCAiSnpZakg0c3ZsaUgwUjNQeUVNZmVadTZKdDY5dTVxZWhabzdGN0VQ
WWxTRSIsICJQb3JGYnBLdVZ1Nnh5bUphZ3ZrRnNGWEFiUm9jMkpHbEFVQTJCQTRvN2NJ
IiwgIlRHZjRvTGJnd2Q1SlFhSHlLVlFaVTlVZEdFMHc1cnREc3JaemZVYW9tTG8iLCAi
amRyVEU4WWNiWTRFaWZ1Z2loaUFlX0JQZWt4SlFaSUNlaVVRd1k5UXF4SSIsICJqc3U5
eVZ1bHdRUWxoRmxNXzNKbHpNYVNGemdsaFFHMERwZmF5UXdMVUs0Il0sICJpc3MiOiAi
aHR0cHM6Ly9leGFtcGxlLmNvbS9pc3N1ZXIiLCAiaWF0IjogMTY4MzAwMDAwMCwgImV4
cCI6IDE4ODMwMDAwMDAsICJ2Y3QiOiAiaHR0cHM6Ly9jcmVkZW50aWFscy5leGFtcGxl
LmNvbS9pZGVudGl0eV9jcmVkZW50aWFsIiwgIl9zZF9hbGciOiAic2hhLTI1NiIsICJj
bmYiOiB7Imp3ayI6IHsia3R5IjogIkVDIiwgImNydiI6ICJQLTI1NiIsICJ4IjogIlRD
QUVSMTladnUzT0hGNGo0VzR2ZlNWb0hJUDFJTGlsRGxzN3ZDZUdlbWMiLCAieSI6ICJa
eGppV1diWk1RR0hWV0tWUTRoYlNJaXJzVmZ1ZWNDRTZ0NGpUOUYySFpRIn19fQ.HGNbT
pc51R_6ssbIbtXkT9FutnyoF2ekNfIhY4983amNK3B6MhnqTA5VKtgOlkrPek9ZF4VUy
6fHAwosePpTRQ~WyJsa2x4RjVqTVlsR1RQVW92TU5JdkNBIiwgImlzX292ZXJfNjUiLC
B0cnVlXQ~WyJRZ19PNjR6cUF4ZTQxMmExMDhpcm9BIiwgImFkZHJlc3MiLCB7InN0cmV
ldF9hZGRyZXNzIjogIjEyMyBNYWluIFN0IiwgImxvY2FsaXR5IjogIkFueXRvd24iLCA
icmVnaW9uIjogIkFueXN0YXRlIiwgImNvdW50cnkiOiAiVVMifV0~eyJhbGciOiAiRVM
yNTYiLCAidHlwIjogImtiK2p3dCJ9.eyJub25jZSI6ICIxMjM0NTY3ODkwIiwgImF1ZC
I6ICJodHRwczovL2V4YW1wbGUuY29tL3ZlcmlmaWVyIiwgImlhdCI6IDE3NzcwNTY4ND
gsICJzZF9oYXNoIjogImgyOGNpY2gwT0pYdmM5RjlYT2RNUlB1UEUwZVhCMVB4cHFTTn
liTXJMTFUifQ.d_6nlLgiiw1x7lRKwGw4ietim8eXnU59gA_G8KKMYn6eH3mFCJIf0pz
XXqwVocozSiQyUaEqK6sgE9U6rtbVGw
```

Figure 7: Presented SD-JWT+KB Example

After validation, the Verifier will have the processed SD-JWT payload in Figure 8 available for further handling.

```
{
  "iss": "https://example.com/issuer",
  "iat": 1683000000,
  "exp": 1883000000,
  "vct": "https://credentials.example.com/identity_credential",
  "cnf": {
    "jwk": {
      "kty": "EC",
      "crv": "P-256",
      "x": "TCAER19Zvu3OHF4j4W4vfSVoHIP1ILilDls7vCeGemc",
      "y": "ZxjiWWbZMQGHVWKVQ4hbSIirsVfuecCE6t4jT9F2HZQ"
    }
  },
  "is_over_65": true,
  "address": {
    "street_address": "123 Main St",
    "locality": "Anytown",
    "region": "Anystate",
    "country": "US"
  }
}
```

Figure 8: Verified SD-JWT Payload Example

The example in Figure 9 shows a presentation of a (similar but different) SD-JWT without a KB-JWT.

```
eyJhbGciOiAiRVMyNTYiLCAidHlwIjogImRjK3NkLWp3dCJ9.eyJfc2QiOiBbIjA5dkt
ySk1PbHlUV00wc2pwdV9wZE9CVkJRMk0xeTNLaHBINTE1blhrcFkiLCAiMnJzakdiYUM
wa3k4bVQwcEpyUGlvV1RxMF9kYXcxc1g3NnBvVWxnQ3diSSIsICJFa084ZGhXMGRIRUp
idlVIbEVfVkNldUM5dVJFTE9pZUxaaGg3WGJVVHRBIiwgIklsRHpJS2VpWmREd3BxcEs
2WmZieXBoRnZ6NUZnbldhLXNONndxUVhDaXciLCAiSnpZakg0c3ZsaUgwUjNQeUVNZmV
adTZKdDY5dTVxZWhabzdGN0VQWWxTRSIsICJQb3JGYnBLdVZ1Nnh5bUphZ3ZrRnNGWEF
iUm9jMkpHbEFVQTJCQTRvN2NJIiwgIlRHZjRvTGJnd2Q1SlFhSHlLVlFaVTlVZEdFMHc
1cnREc3JaemZVYW9tTG8iLCAiamRyVEU4WWNiWTRFaWZ1Z2loaUFlX0JQZWt4SlFaSUN
laVVRd1k5UXF4SSIsICJqc3U5eVZ1bHdRUWxoRmxNXzNKbHpNYVNGemdsaFFHMERwZmF
5UXdMVUs0Il0sICJpc3MiOiAiaHR0cHM6Ly9leGFtcGxlLmNvbS9pc3N1ZXIiLCAiaWF
0IjogMTY4MzAwMDAwMCwgImV4cCI6IDE4ODMwMDAwMDAsICJ2Y3QiOiAiaHR0cHM6Ly9
jcmVkZW50aWFscy5leGFtcGxlLmNvbS9pZGVudGl0eV9jcmVkZW50aWFsIiwgIl9zZF9
hbGciOiAic2hhLTI1NiJ9.FgrwYobSoTfaffNqRQ3LEACqNd9rk7R32rEhzMW-64lxLA
rQrM0_kbSqM6aq-vcbeDbZjirbtuWnFyVyOOC4bQ~WyJsa2x4RjVqTVlsR1RQVW92TU5
JdkNBIiwgImlzX292ZXJfNjUiLCB0cnVlXQ~WyJRZ19PNjR6cUF4ZTQxMmExMDhpcm9B
IiwgImFkZHJlc3MiLCB7InN0cmVldF9hZGRyZXNzIjogIjEyMyBNYWluIFN0IiwgImxv
Y2FsaXR5IjogIkFueXRvd24iLCAicmVnaW9uIjogIkFueXN0YXRlIiwgImNvdW50cnki
OiAiVVMifV0~
```

Figure 9: Presented SD-JWT Example

The Verifier will have the following processed SD-JWT payload shown in Figure 10 after validation.

```
{
  "iss": "https://example.com/issuer",
  "iat": 1683000000,
  "exp": 1883000000,
  "vct": "https://credentials.example.com/identity_credential",
  "is_over_65": true,
  "address": {
    "street_address": "123 Main St",
    "locality": "Anytown",
    "region": "Anystate",
    "country": "US"
  }
}
```

Figure 10: Processed SD-JWT Payload Example

2.4. Verification and Processing

The recipient (Holder or Verifier) of an SD-JWT VC MUST process and verify an SD-JWT VC as described in Section 7 of [RFC9901]. The check in point 2.c of Section 7.1 of [RFC9901], which validates the Issuer and ensures that the signing key belongs to the Issuer, MUST be satisfied by determining and validating the public verification key used to verify the Issuer-signed JWT, employing a key discovery and validation mechanism (defined in Section 2.5) that is permitted for the Issuer according to policy.

If Key Binding is required (refer to the security considerations in Section 9.5 of [RFC9901]), the Verifier MUST verify the KB-JWT according to Section 7.3 of [RFC9901]. To verify the KB-JWT, the cnf claim of the SD-JWT MUST be used.

If there are no selectively disclosable claims, there is no need to process the _sd claim nor any Disclosures.

If status is present in the verified payload of the SD-JWT, the status SHOULD be checked. Verifier policy decides whether to reject or accept a presentation of a SD-JWT VC based on the status of the Verifiable Digital Credential.

Additional validation rules MAY apply, but their use is out of the scope of this specification.

2.5. Issuer Verification Key Discovery and Validation

A key discovery and validation mechanism defines how a Verifier determines the appropriate key and associated procedure for verifying the signature of an Issuer-signed JWT. This allows for flexibility in supporting different trust anchoring systems and key resolution methods without changing the core processing model.

A recipient MUST determine and validate the public verification key for the Issuer-signed JWT using a supported key discovery and validation mechanism that is permitted for the given Issuer according to policy. This specification defines the following two mechanisms:

* JWT VC Issuer Metadata: A mechanism to retrieve the Issuer's public key using web-based resolution. When the value of the iss claim of the Issuer-signed JWT is an HTTPS URI, the recipient obtains the public key using the keys from JWT VC Issuer Metadata as defined in Section 3.

* Inline X.509 Certificates: A mechanism to retrieve the Issuer's public key using the X.509 certificate chain in the SD-JWT header. When the protected header of the Issuer-signed JWT contains the x5c parameter, the recipient uses the public key from the end-entity certificate of the certificates from that x5c parameter and validates the X.509 certificate chain accordingly. In this case, the Issuer of the Verifiable Digital Credential is the subject of the end-entity certificate.

To enable different trust anchoring systems or key resolution methods, separate specifications or ecosystem regulations may define additional key discovery and validation mechanisms that complement or override those defined above; however, the specifics of such mechanisms are out of scope for this specification. See Section 6.2 for related security considerations.

If a recipient cannot validate that the public verification key corresponds the Issuer of the Issuer-signed JWT using a permitted key discovery and validation mechanism, the SD-JWT VC MUST be rejected.

3. JWT VC Issuer Metadata

This specification defines the JWT VC Issuer Metadata to retrieve the JWT VC Issuer Metadata configuration of the Issuer of the SD-JWT VC. The Issuer is identified by the iss claim in the JWT. Use of the JWT VC Issuer Metadata is OPTIONAL.

Issuers publishing JWT VC Issuer Metadata MUST make a JWT VC Issuer Metadata configuration available at the location formed by inserting the well-known string /.well-known/jwt-vc-issuer between the host component and the path component (if any) of the iss claim value in the JWT. The iss MUST be a case-sensitive URL using the HTTPS scheme that contains scheme, host and, optionally, port number and path components, but no query or fragment components.

3.1. JWT VC Issuer Metadata Request

A JWT VC Issuer Metadata configuration MUST be queried using an HTTP GET request at the path defined in Section 3.

Figure 11 is an example of an HTTP request for the JWT VC Issuer Metadata configuration when iss is set to https://example.com:

```
GET /.well-known/jwt-vc-issuer HTTP/1.1
Host: example.com
```

Figure 11: Example HTTP Request for JWT VC Issuer Metadata

If the iss value contains a path component, any terminating / MUST be removed before inserting /.well-known/ and the well-known URI suffix between the host component and the path component.

Figure 12 is an example of an HTTP request for the JWT VC Issuer Metadata configuration when iss is set to https://example.com/tenant/1234:

```
GET /.well-known/jwt-vc-issuer/tenant/1234 HTTP/1.1
Host: example.com
```

Figure 12: Example HTTP Request for JWT VC Issuer Metadata

3.2. JWT VC Issuer Metadata Response

A successful response MUST use an HTTP 200 status code and return the JWT VC Issuer Metadata configuration using the application/json content type.

An error response MUST use the applicable HTTP status code value.

This specification defines the following JWT VC Issuer Metadata configuration parameters:

* issuer: REQUIRED. The Issuer identifier, which MUST be identical to the iss value in the JWT.

* jwks_uri: OPTIONAL. URL string referencing the Issuer's JSON Web Key (JWK) Set [RFC7517] document which contains the Issuer's public keys. The value of this field MUST point to a valid JWK Set document.
* jwks: OPTIONAL. Issuer's JSON Web Key Set [RFC7517] document value, which contains the Issuer's public keys. The value of this field MUST be a JSON object containing a valid JWK Set.

JWT VC Issuer Metadata MUST include either jwks_uri or jwks in their JWT VC Issuer Metadata, but not both.

It is RECOMMENDED that the Issuer-signed JWT contains a kid JWT header parameter that can be used to look up the public key in the JWK Set included by value or referenced in the JWT VC Issuer Metadata.

Figure 13 is an example of a JWT VC Issuer Metadata configuration including jwks:

```
{
   "issuer":"https://example.com",
   "jwks":{
      "keys":[
         {
            "kid":"doc-signer-05-25-2022",
            "kty": "EC", "crv": "P-256",
            "x": "b28d4MwZMjw8-00CG4xfnn9SLMVMM19SlqZpVb_uNtQ",
            "y": "Xv5zWwuoaTgdS6hV43yI6gBwTnjukmFQQnJ_kCxzqk8"
         },
         {
            "kid":"doc-signer-06-05-2025",
            "kty": "EC", "crv": "P-256",
            "x":"DfjUx4WHBds61vGbqUQhsy3FGX13fAS13QWh2EHIkX8",
            "y":"NfqJt9Kp0EA93xq9ysO80DRZ_hCGlISz-pYLgv4RFvg"
         }
      ]
   }
}
```

Figure 13: Example Metadata with a JSON Web Key Set

Figure 14 is an example of a JWT VC Issuer Metadata configuration including jwks_uri:

```
{
   "issuer":"https://example.com",
   "jwks_uri":"https://jwt-vc-issuer.example.com/my_public_keys.jwks"
}
```

Figure 14: Example Metadata with a JSON Web Key Set URI

Additional JWT VC Issuer Metadata configuration parameters MAY also be used.

3.3. JWT VC Issuer Metadata Validation

The issuer value returned MUST be identical to the iss value of the Issuer-signed JWT. If these values are not identical, the data contained in the response MUST NOT be used.

4. SD-JWT VC Type Metadata

An SD-JWT VC type, i.e., the vct value, is associated with Type Metadata defining, for example, information about the type and how credentials are displayed.

This section defines Type Metadata that can be associated with a type of an SD-JWT VC, as well as a method for retrieving the Type Metadata and processing rules. This Type Metadata is intended to be used, among other things, for the following purposes:

* Developers of Issuers and Verifiers can use the Type Metadata to understand the semantics of the type and the associated rules. While in some cases, Issuers are the parties that define types, this is not always the case. For example, a type can be defined by a standardization body or a community.
* Verifiers can use the Type Metadata to determine whether a credential is valid according to the rules of the type. For example, a Verifier can check whether a credential contains all required claims and whether the claims are selectively disclosable.
* Holders can use the metadata to display the credential in a way that is consistent with the intent of the provider of the Type Metadata.

Type Metadata can be retrieved as described in Section 4.3.

4.1. Type Metadata Example

This section only shows excerpts, the full examples can be found in Appendix B.2.

The following in Figure 15 is an example of an SD-JWT VC payload, containing a vct claim with the value https://betelgeuse.example.com/education_credential/v42.

```
{
  "vct": "https://betelgeuse.example.com/education_credential/v42",
  "vct#integrity": "sha256-1odmyxoVQCuQx8SAym8rWHXba41fM/Iv/V1H8VHGN00=",
  ...
}
```

Figure 15: Example SD-JWT VC Payload with vct Claim

Type Metadata for the type https://betelgeuse.example.com/education_credential/v42 can be retrieved using various mechanisms as described in Section 4.3. For this example, the vct value is a URL as defined in Section 4.3.1 and the Type Metadata document in Figure 16 is retrieved from it:

```
{
  "vct":"https://betelgeuse.example.com/education_credential/v42",
  "name":"Betelgeuse Education Credential - Version 42",
  "description":"This is our education credential. Don’t panic.",
  "extends":"https://galaxy.example.com/galactic-education-credential/v2",
  "extends#integrity":"sha256-ilOUJsTultOwLfz7QUcFALaRa3BP/jelX1ds04kB9yU="
}
```

Figure 16: Example Type Metadata Document

Note: The hash of the Type Metadata document shown in the second example must be equal to the one in the vct#integrity claim in the SD-JWT VC payload, 1odmyxoVQCuQx8SAym8rWHXba41fM/Iv/V1H8VHGN00= (see Section 5 for details).

4.2. Type Metadata Format

The Type Metadata document MUST be a JSON object. The following properties are defined:

* vct: REQUIRED. The verifiable digital credential type described by this type metadata document.
* name: OPTIONAL. A human-readable name for the type, intended for developers reading the JSON document.
* description: OPTIONAL. A human-readable description for the type, intended for developers reading the JSON document.
* extends: OPTIONAL. A URI of another type that this type extends, as described in Section 4.4.
* display: An array of objects containing display information for the type, as described in Section 4.5. This property is OPTIONAL.
* claims: An array of objects containing claim information for the type, as described in Section 4.6. This property is OPTIONAL.

Additionally, extends#integrity MAY be present as defined in Section 5.

A Type Metadata document MAY contain additional top level or subordinate properties. Consumers MUST ignore properties that are not understood.

An example of a Type Metadata document is shown in Appendix B.2.

4.3. Retrieving Type Metadata

A Consumer retrieving Type Metadata MUST ensure that the vct value in the SD-JWT VC payload is identical to the vct value in the reference to the Type Metadata (either in the SD-JWT VC itself or in an extends property in a Type Metadata document).

The following sections define methods to retrieve Type Metadata.

4.3.1. From a URL in the vct Claim

A URI in the vct claim can be used to express a type. If the type is a URL using the HTTPS scheme, Type Metadata MAY be retrieved from it.

The Type Metadata is retrieved using the HTTP GET method. A successful response MUST use an HTTP 200 status code and return a JSON object as defined in Section 4.2 using the application/json content type. An error response MUST use the applicable HTTP status code value.

If the claim vct#integrity is present in the SD-JWT VC, its value vct#integrity MUST be an "integrity metadata" string as defined in Section 5.

4.3.2. From a Registry

A Consumer MAY use a registry to retrieve Type Metadata for a SD-JWT VC type, e.g., if the type is not an HTTPS URL or if the Consumer does not have access to the URL. A Consumer needs to trust the registry to provide correct Type Metadata for the type.

The registry MUST provide the Type Metadata in the same format as described in Section 4.2.

4.3.3. Using a Defined Retrieval Method

Ecosystems MAY define additional methods for retrieving Type Metadata. For example, a standardization body or a community MAY define a service which has to be used to retrieve Type Metadata based on a URN in the vct claim.

4.3.4. From a Local Cache

A Consumer MAY cache Type Metadata for a SD-JWT VC type. If a hash for integrity protection is present in the Type Metadata as defined in Section 5, the Consumer MAY assume that the Type Metadata is static and can be cached indefinitely. Otherwise, the Consumer MUST use the Cache-Control header of the HTTP response to determine how long the metadata can be cached.

4.4. Extending Type Metadata

An SD-JWT VC type can extend another type. The extended type is identified by the URI in the extends property. Consumers MUST retrieve and process Type Metadata for the extended type before processing the Type Metadata for the extending type.

The extended type MAY itself extend another type. This can be used to create a chain or hierarchy of types. The security considerations described in Section 6.3 apply in order to avoid problems with circular dependencies.

Processing details when extending type metadata are described in Section 4.5.2 and Section 4.6.5.

4.5. Display Metadata

The display property is an array containing display information for the type. The array MUST contain an object for each locale that is supported by the type. The consuming application MUST use the language tag it considers most appropriate for the user.

The objects in the array have the following properties:

* locale: A language tag as defined in Section 2 of [RFC5646]. This property is REQUIRED.
* name: A human-readable name for the type, intended for end users. This property is REQUIRED.
* description: A human-readable description for the type, intended for end users. This property is OPTIONAL.
* rendering: An object containing rendering information for the type, as described in Section 4.5.1. This property is OPTIONAL.

4.5.1. Rendering Metadata

The rendering property is an object containing rendering information for the type. The object MUST contain a property for each rendering method that is supported by the type. The property name MUST be a rendering method identifier and the property value MUST be an object containing the properties defined for the rendering method.

4.5.1.1. Rendering Method "simple"

The simple rendering method is intended for use in applications that do not support SVG rendering. The object contains the following properties:

* logo: An object containing information about the logo to be displayed for the type, as described in Section 4.5.1.1.1. This property is OPTIONAL.
* background_image: An object containing information about the background image to be displayed for the type, as described in Section 4.5.1.1.2. This property is OPTIONAL.
* background_color: An RGB color value as defined in [W3C.CSS-COLOR] for the background of the credential. This property is OPTIONAL.
* text_color: An RGB color value as defined in [W3C.CSS-COLOR] value for the text of the credential. This property is OPTIONAL.

4.5.1.1.1. Logo Metadata

The logo property is an object containing information about the logo to be displayed for the type. The object contains the following properties:

* uri: A URI pointing to the logo image. This property is REQUIRED.
* uri#integrity: An "integrity metadata" string as described in Section 5. This property is OPTIONAL.
* alt_text: A string containing alternative text for the logo image. This property is OPTIONAL.

4.5.1.1.2. Background Image Metadata

The background_image property is an object containing information about the background image to be displayed for the type. The object contains the following properties:

* uri: A URI pointing to the background image. This property is REQUIRED.
* uri#integrity: An "integrity metadata" string as described in Section 5. This property is OPTIONAL.

4.5.1.2. Rendering Method "svg_templates"

The svg_templates rendering method is intended for use in applications that support SVG rendering. svg_templates MUST contain an array of objects containing information about the SVG templates available for the type. Each object contains the following properties:

* uri: A URI pointing to the SVG template. This property is REQUIRED.
* uri#integrity: An "integrity metadata" string as described in Section 5. This property is OPTIONAL.
* properties: An object containing properties for the SVG template, as described in Section 4.5.1.2.1. This property is REQUIRED if more than one SVG template is present, otherwise it is OPTIONAL.

4.5.1.2.1. SVG Template Properties

The properties property is an object containing properties for the SVG template. Consuming applications MUST use these properties to find the best SVG template available for display to the user based on the display properties (landscape/portrait) and user preferences (color scheme, contrast). The object MUST contain at least one of the following properties:

* orientation: The orientation for which the SVG template is optimized, with valid values being portrait and landscape. This property is OPTIONAL.
* color_scheme: The color scheme for which the SVG template is optimized, with valid values being light and dark. This property is OPTIONAL.
* contrast: The contrast for which the SVG template is optimized, with valid values being normal and high. This property is OPTIONAL.

4.5.1.2.2. SVG Rendering

A consuming application MUST preprocess the SVG template by replacing placeholders in the SVG template with properly escaped values of the claims in the credential. The placeholders MUST be defined in the SVG template using the syntax {{svg_id}}, where svg_id is an identifier defined in the claim metadata as described in Section 4.6.

Placeholders MUST only be used in the text content of the SVG template and MUST NOT be used in any other part of the SVG template, e.g., in attributes or comments.

A consuming application MUST ensure that all special characters in the claim values are properly escaped before inserting them into the SVG template. At least the following characters MUST be escaped:

* & as &
* < as <
* > as >
* " as "
* ' as '

If the svg_id is not present in the claim metadata, the consuming application SHOULD reject and not render the SVG template. If the svg_id is present in the claim metadata, but the claim is not present in the credential, the placeholder MUST be replaced with an empty string or a string appropriate to indicate that the value is absent.

The example in Figure 17 shows a minimal SVG with one placeholder using the svg_id value address_street_address which is defined in the example in Appendix B.2.

```
<svg xmlns="http://www.w3.org/2000/svg" width="100" height="100">
  <text x="10" y="20">Street address: {{address_street_address}}</text>
</svg>
```

Figure 17: Example SVG Template with Placeholder

When rendering the SVG template, the consuming application MUST ensure that malicious metadata providers or issuers cannot inject executable code into the SVG template and thereby compromise the security of the consuming application. The consuming application MUST NOT execute any code in the SVG template. If code execution cannot be prevented reliably, the SVG display MUST be sandboxed.

Furthermore, consuming applications MUST ensure that references to external resources (images, etc.) from within the SVG cannot be used to track users or the usage of credentials.

4.5.2. Extending Display Metadata

When an SD-JWT VC type extends another type as described in Section 4.4, the display metadata remains valid for the inheriting type unless that type defines its own display property, in which case the original display metadata is ignored.

Note that this does not affect the display properties in the claim metadata. Rules for these are defined in Section 4.6.5.

4.6. Claim Metadata

The claims property is an array of objects that provides per-claim metadata. Each object identifies which claim or set of claims in the credential is being described (path), and can specify how that claim is presented to end users (display), whether it is required to be included by the Issuer (mandatory), whether it is selectively disclosable (sd), and, for SVG rendering, which placeholder refers to it (svg_id).

The array MAY contain an object for each claim that is supported by the type. Each object contains the following properties:

* path: An array indicating the claim or claims that are being addressed, as described below. This property is REQUIRED.
* display: An array containing display information for the claim or claims that are being addressed, as described in Section 4.6.2. This property is OPTIONAL.
* mandatory: A boolean indicating that the claim must be present in the issued credential. This property is OPTIONAL. If omitted, the default value is false. See Section 4.6.3 for details.
* sd: A string indicating whether the claim is selectively disclosable, as described in Section 4.6.4. This property is OPTIONAL.
* svg_id: A string defining the ID of the claim for reference in the SVG template, as described in Section 4.5.1.2.2. The ID MUST be unique within the type metadata. It MUST consist of only alphanumeric characters and underscores and MUST NOT start with a digit. This property is OPTIONAL.

4.6.1. Claim Path

The path property MUST be a non-empty array of strings, null values, or non-negative integers. It is used to select a particular claim in the credential or a set of claims. A string indicates that the respective key is to be selected, a null value indicates that all elements of the currently selected array(s) are to be selected, and a non-negative integer indicates that the respective index in an array is to be selected.

4.6.1.1. Example

Figure 18 shows a reduced example of a credential.

```
{
  "vct": "https://betelgeuse.example.com/education_credential/v42",
  "name": "Arthur Dent",
  "address": {
    "street_address": "42 Market Street",
    "city": "Milliways",
    "postal_code": "12345"
  },
  "degrees": [
    {
      "type": "Bachelor of Science",
      "university": "University of Betelgeuse"
    },
    {
      "type": "Master of Science",
      "university": "University of Betelgeuse"
    }
  ],
  "nationalities": ["British", "Betelgeusian"]
}
```

Figure 18: Example Credential for Claim Path

The following shows examples of path values and the respective selected claims in the credential above:

* ["name"]: The claim name with the value Arthur Dent is selected.
* ["address"]: The claim address with its sub-claims as the value is selected.
* ["address", "street_address"]: The claim street_address with the value 42 Market Street is selected.
* ["degrees", null, "type"]: All type claims in the degrees array are selected.

The example in Appendix B.2 shows how the path can be used to address arrays and their elements.

4.6.1.2. Processing of path

In detail, the array components of path are processed from left to right as follows:

1. Select the root element of the credential, i.e., the top-level JSON object.
2. Process the path components from left to right:
    1. If the path component is a string, select the element in the respective key in the currently selected element(s). If any of the currently selected element(s) is not an object, abort

processing and return an error. If the key does not exist in any element currently selected, remove that element from the selection.
2. If the path component is null, select all elements of the currently selected array(s). If any of the currently selected element(s) is not an array, abort processing and return an error.
3. If the path component is a non-negative integer, select the element at the respective index in the currently selected array(s). If any of the currently selected element(s) is not an array, abort processing and return an error. If the index does not exist in a selected array, remove that array from the selection.

3. If the set of elements currently selected is empty, abort processing and return an error.

The result of the processing is the set of elements to which the respective claim metadata applies.

The path property MUST point to the respective claim as if all selectively disclosable claims were disclosed to a Verifier. That means that a consuming application which does not have access to all disclosures may not be able to identify the claim which is being addressed.

Note: This specification intentionally does not use JSON Pointer [RFC6901] for selecting claims, as JSON Pointer requires string parsing and does not support wildcard selection of array elements. It does not use JSON Path [RFC9535] as that introduces a considerable complexity and brings in many features which are not needed for the use case of selecting claims in a credential. There are also security concerns with some implementations.

4.6.2. Claim Display Metadata

The display property is an array containing display information for the claim. The array MUST contain an object for each locale that is supported by the type. The consuming application MUST use the language tag it considers most appropriate for the user.

The objects in the array have the following properties:

* locale: A language tag as defined in Section 2 of [RFC5646]. This property is REQUIRED.
* label: A human-readable label for the claim, intended for end users. This property is REQUIRED.
* description: A human-readable description for the claim, intended for end users. This property is OPTIONAL.

See Figure 29 in Appendix B.2 for an example that includes claim display metadata.

4.6.3. Claim Mandatory Metadata

The mandatory property is a boolean indicating that, if set to true, the claim MUST be included in the credential by the Issuer. If the value is false or omitted, the claim is considered optional for the Issuer to include. A claim that is mandatory can nonetheless be selectively disclosable, as described in Section 4.6.4.

4.6.4. Claim Selective Disclosure Metadata

The sd property is a string indicating whether the claim is selectively disclosable. The following values are defined:

* always: The Issuer MUST make the claim selectively disclosable.
* allowed: The Issuer MAY make the claim selectively disclosable.
* never: The Issuer MUST NOT make the claim selectively disclosable.

If omitted, the default value is allowed. It is RECOMMENDED to use either always or never to avoid ambiguity.

See Figure 29 in Appendix B.2 for an example that includes claim selective disclosure metadata.

4.6.5. Extending Claim Metadata

When an SD-JWT VC type extends another type as described in Section 4.4, all claim metadata from the extended type MUST be respected and are inherited by the child type unless it is overridden by the child type as defined in the following.

If the child type defines claim metadata with the same path as in the extended type, the child type’s properties are combined with those of the extended type, with the child type’s properties taking precedence. Limitations to this rule are defined for sd and mandatory in Section 4.6.5.1.

The contents of properties are never merged (e.g., if the base display property contains two locale objects and the extending type contains one, the resulting display property will contain only the one object defined for the child type).

Note: The rule for overriding claim metadata is different from the one for display metadata described in Section 4.5.2. This ensures that Consumers can rely on the claim metadata defined in the extended type while still allowing display customization for the visual representation of the credential.

4.6.5.1. Limitations for sd and mandatory

An extending type can specify an sd property for a claim that is marked as allowed in the extended type (or where sd was omitted), changing it to either always or never. However, it MUST NOT change a claim that is marked as always or never in the extended type to a different value.

Similarly, an extending type can set the mandatory property of a claim that is optional in the extended type to true, but it MUST NOT change a claim that is mandatory in the extended type to false.

4.6.5.2. Example for Extending Type Metadata

The base type metadata document for this example is shown in Figure 19.

```
{
  "vct": "https://example.com/base-type-metadata",
  "claims": [
    {
      "path": ["name"],
      "display": [{"label": "Full Name", "locale": "en"}]
    },
    {
      "path": ["address", "city"],
      "display": [{"label": "City", "locale": "en"}]
    }
  ]
}
```

Figure 19: Example Base Type Metadata Document

The extending type metadata document is shown in Figure 20.

```
{
  "vct": "https://example.com/custom-type-metadata",
  "extends": "https://example.com/base-type-metadata",
  "claims": [
    {
      "path": ["address", "city"],
      "display": [{"label": "Town", "locale": "en"}]
    },
    {
      "path": ["nationalities"],
      "display": [{"label": "Nationalities", "locale": "en"}]
    }
  ]
}
```

Figure 20: Example Child Type Metadata Document

In this example, the child type inherits the name claim metadata from the base type, but overrides the address.city claim metadata with its own definition. It also adds a new claim metadata for nationalities. The final effective claim metadata for the child type is shown here in Figure 21:

```
{
  "claims": [
    {
      "path": ["name"],
      "display": [{"label": "Full Name", "locale": "en"}]
    },
    {
      "path": ["address", "city"],
      "display": [{"label": "Town", "locale": "en"}]
    },
    {
      "path": ["nationalities"],
      "display": [{"label": "Nationalities", "locale": "en"}]
    }
  ]
}
```

Figure 21: Effective Claim Metadata for Child Type

5. Integrity of Referenced Documents

This section defines how integrity can be asserted for documents referenced by an SD-JWT VC and its Type Metadata. For references such as vct, extends, and uri, a corresponding #integrity value can be used to identify the expected content of the retrieved document. If such an integrity value is present, the Consumer verifies that the retrieved document matches it before processing that document.

The vct claim in the SD-JWT VC as defined in Section 2.2.2.2 and various URIs in the Type Metadata MAY be accompanied by a respective item suffixed with #integrity, in particular:

* extends as defined in Section 4.4, and
* uri as used in three places in Section 4.5.1.

The value MUST be an "integrity metadata" string as defined in Section 3 of [W3C.SRI]. If an integrity property is present for a particular claim, the Consumer of the respective document MUST verify the integrity of the retrieved document as defined in Section 3.3.5 of [W3C.SRI].

6. Security Considerations

The security considerations in the SD-JWT specification [RFC9901] apply to this specification. Additionally, the following security considerations need to be taken into account when using SD-JWT VCs:

6.1. Server-Side Request Forgery

The JWT VC Issuer Metadata configuration is retrieved from the JWT VC Issuer by the Holder or Verifier. Similar to other metadata endpoints, the URL for the retrieval MUST be considered an untrusted value and could be a vector for Server-Side Request Forgery (SSRF) attacks.

Before making a request to the JWT VC Issuer Metadata endpoint, the Holder or Verifier MUST validate the URL to ensure that it is a valid HTTPS URL and that it does not point to internal resources. This requires, in particular, ensuring that the host part of the URL does not address an internal service (by IP address or an internal host name) and that, if an external DNS name is used, the resolved DNS name does not point to an internal IPv4 or IPv6 address.

When retrieving the metadata, the Holder or Verifier MUST ensure that the request is made in a time-bound and size-bound manner to prevent denial of service attacks. The Holder or Verifier MUST also ensure that the response is a valid JWT VC Issuer Metadata configuration document before processing it.

Additional considerations can be found in [OWASP_SSRF].

6.2. Ecosystem-specific Public Key Verification Methods

When defining ecosystem-specific rules for resolution and verification of the public key, as outlined in Section 2.5, it is critical that those rules maintain the integrity of the relationship between the iss value of the SD-JWT, if present, and the public keys of the Issuer.

A Verifier MUST ensure that for any given iss value, an attacker cannot influence the type of verification process used. Otherwise, an attacker could attempt to make the Verifier use a verification process not intended by the Issuer, allowing the attacker to potentially manipulate the verification result to their advantage.

6.3. Circular "extends" Dependencies of Types

A type MUST NOT extend another type that extends (either directly or with steps in-between) the first type. This would result in a circular dependency that could lead to infinite recursion when retrieving and processing the metadata.

Consumers MUST detect such circular dependencies and reject the credential.

6.4. Robust Retrieval of Type Metadata

In Section 4.3, various methods for distributing and retrieving metadata are described. Methods relying on a network connection may fail due to network issues or unavailability of a network connection due to offline usage of credentials, temporary server outages, or denial-of-service attacks on the metadata server.

Consumers SHOULD therefore implement a local cache as described in Section 4.3.4 if possible. Such a cache MAY be populated with metadata before the credential is used.

These measures allow the Consumers to continue to function even if the metadata server is temporarily unavailable and avoid privacy issues as described in Section 7.4.

6.5. Risks Associated with Textual Information

Some claims in the SD-JWT VC and properties in the Type Metadata, e.g., display, allows issuers and providers of metadata to specify human-readable information. These can contain arbitrary textual information that may be displayed to end users and developers. As such, any consuming application MUST ensure that maliciously crafted information cannot be used to compromise the security of the application or the privacy of the user. To this end, the following considerations apply:

* The consuming application MUST ensure that the text is properly escaped before displaying it to the user or transferring it into other contexts. For example, if the data is displayed in an HTML document, the text MUST be properly escaped to prevent Cross-Site Scripting (XSS) attacks.
* The consuming application MUST ensure that the display of the user interface elements cannot be distorted by overly long text or special characters.

6.6. Credential Type Extension and Issuer Authorization

When processing credentials, it is important to recognize that the ability to extend or reference an existing credential type (e.g., a well-known identifier such as urn:ec.eu.x.y.z) does not confer any implicit authorization to issue credentials of that type or its extensions. In particular:

* *Issuer Authorization*: Verifiers and Holders MUST NOT assume that any issuer who issues a credential extending a known type is authorized to do so. The mere presence of an extension or reference to a recognized type (e.g., a national type urn:de.bla extending a European PID type) does not validate the issuer’s authority.
* *Rogue Issuers*: Attackers may issue credentials with types that extend or mimic legitimate types (e.g., urn:attacker extending urn:ec.eu.x.y.z). Such credentials MUST NOT be accepted solely based on their type hierarchy or extension relationship.
* *Processing Rules*: Implementations MUST verify the issuer’s authorization independently of the credential type or its extensions. This typically involves checking the issuer’s identity, trust status, and any relevant accreditation or registry before accepting a credential.

Verifiers and Holders MUST implement explicit checks for issuer authorization and MUST NOT rely on type extension as a proxy for trust or legitimacy. Credential acceptance decisions MUST be based on both the credential type and the verified authority of the issuer.

6.7. Trust in Type Metadata

Type Metadata associated with an SD-JWT VC, e.g., rendering metadata, is asserted by the Publisher of the Type Metadata and trust in this metadata depends on the trust relationship between its Publisher and the Consumer. A Consumer MUST NOT assume that Type Metadata is accurate or meaningful unless the Publisher is recognized as authoritative for the type in question.

Ecosystems SHOULD define governance or accreditation mechanisms that specify which Publishers are authorized to provide Type Metadata for specific verifiable digital credential types and under what conditions such metadata can be relied upon.

Consumers SHOULD treat with reduced trust any Type Metadata if the Publisher is not accredited or otherwise trusted within the applicable ecosystem.

6.8. Use of Data URIs for Claim Types

The use of data URIs allows embedding of data directly within credential payloads. Implementations SHOULD treat data URIs as untrusted input at the processing and rendering layer and apply appropriate validation and handling. Failure to properly escape, sanitize or constrain their use can lead to security issues such as unintended code execution, resource exhaustion, or misuse of embedded content. Implementations SHOULD restrict the set of accepted media types, enforce reasonable size and content limits, and avoid dereferencing or interpreting data URIs in ways that could execute or render active content, consistent with their overall security model.

7. Privacy Considerations

The privacy considerations in the SD-JWT specification [RFC9901] apply to this specification. Additionally, the following privacy considerations need to be taken into account when using SD-JWT VCs.

7.1. Unlinkability

The privacy considerations in Section 10.1 of [RFC9901] apply, especially to the cnf claim.

7.2. Verifiable Digital Credential Type Identifier

Issuers and Holders have to be aware that while this specification supports selective disclosure of claims of a given SD-JWT VC, the vct claim is not selectively disclosable. In certain situations this could lead to unwanted leakage of additional context information.

In general, Issuers are advised to choose vct values following data minimization principles. For example, government Issuers issuing an SD-JWT VC to their citizens to enable them to prove their age, might consider using a vct value that does not allow third-parties to infer additional personal information about the Holder, e.g., country of residency or citizenship.

Additionally, Holders have to be informed that, besides the actual requested claims, the vct information is shared with the Verifier.

7.3. Issuer Phone-Home

A malicious Issuer can choose the Issuer identifier of the SD-JWT VC to enable tracking the usage behavior of the Holder if the Issuer identifier is Holder-specific and if the resolution of the key material to verify the Issuer-signed JWT requires the Verifier to phone home to the Issuer.

For example, a malicious Issuer could generate a unique value for the Issuer identifier per Holder, e.g., https://example.com/issuer/holder-1234 and host the JWT VC Issuer Metadata. The Verifier would create an HTTP GET request to the Holder-specific well-known URI when the SD-JWT VC is verified. This would allow the malicious Issuer to keep track where and how often the SD-JWT VC was used.

Verifiers are advised to establish trust in an SD-JWT VC by pinning specific Issuer identifiers and should monitor suspicious behavior such as frequent rotation of those identifiers. If such behavior is detected, Verifiers are advised to reject SD-JWT VCs issued by those Issuers and not to retrieve associated metadata. Holders are similarly advised to exercise caution with SD-JWT VCs if they contain unexpected or easily correlatable information in the Issuer identifier.

Another related concern arises from the use of confirmation methods in the cnf claim that involve retrieving key material from a remote source, especially if that source is controlled by the issuer. This includes, but is not limited to, the use of the x5u parameter in JWKs (Section 4.6 of [RFC7517]), the jku parameter (Section 3.5 of [RFC7800]), and cases where a URL is used in the kid parameter (Section 3.4 of [RFC7800]). Future confirmation methods may also introduce remote retrieval mechanisms. Issuers are advised not to issue SD-JWT VCs with such cnf methods, and Verifiers and Holders are advised not to follow or resolve remote references for key material in the cnf claim. Only confirmation methods that do not require remote retrieval of key material SHOULD be supported.

7.4. Privacy-Preserving Retrieval of Type Metadata

In Section 4.3, various methods for distributing and retrieving Type Metadata are described. For methods which rely on a network connection to a URL (e.g., provided by an Issuer), the Issuer and other third parties may be able to track the usage of a credential by observing requests to the Type Metadata URL.

Consumers SHOULD prefer methods for retrieving Type Metadata that do not leak information about the usage of a credential to third parties. The recommendations in Section 6.4 apply.

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC5646] Phillips, A., Ed. and M. Davis, Ed., "Tags for Identifying Languages", BCP 47, RFC 5646, DOI 10.17487/RFC5646, September 2009, <https://www.rfc-editor.org/info/rfc5646>.

[RFC5785] Nottingham, M. and E. Hammer-Lahav, "Defining Well-Known Uniform Resource Identifiers (URIs)", RFC 5785, DOI 10.17487/RFC5785, April 2010, <https://www.rfc-editor.org/info/rfc5785>.

[RFC7515] Jones, M., Bradley, J., and N. Sakimura, "JSON Web Signature (JWS)", RFC 7515, DOI 10.17487/RFC7515, May 2015, <https://www.rfc-editor.org/info/rfc7515>.

[RFC7517] Jones, M., "JSON Web Key (JWK)", RFC 7517, DOI 10.17487/RFC7517, May 2015, <https://www.rfc-editor.org/info/rfc7517>.

[RFC7519] Jones, M., Bradley, J., and N. Sakimura, "JSON Web Token (JWT)", RFC 7519, DOI 10.17487/RFC7519, May 2015, <https://www.rfc-editor.org/info/rfc7519>.

[RFC7800] Jones, M., Bradley, J., and H. Tschofenig, "Proof-of-Possession Key Semantics for JSON Web Tokens (JWTs)", RFC 7800, DOI 10.17487/RFC7800, April 2016, <https://www.rfc-editor.org/info/rfc7800>.

[RFC9901] Fett, D., Yasuda, K., and B. Campbell, "Selective Disclosure for JSON Web Tokens", RFC 9901, DOI 10.17487/RFC9901, November 2025, <https://www.rfc-editor.org/info/rfc9901>.

[W3C.CSS-COLOR] elik, T., Lilley, C., and L. D. Baron, "CSS Color Module Level 3", 18 January 2022, <https://www.w3.org/TR/css-color-3>.

[W3C.SRI] Akhawe, D., Braun, F., Marier, F., and J. Weinberger, "Subresource Integrity", 23 June 2016, <https://www.w3.org/TR/2016/REC-SRI-20160623/>.

8.2. Informative References

[EUDIW.ARF] Commission, E., "The European Digital Identity Wallet Architecture and Reference Framework", <https://github.com/eu-digital-identity-wallet/eudi-doc-architecture-and-reference-framework/releases>.

[I-D.ietf-oauth-status-list] Looker, T., Bastian, P., and C. Bormann, "Token Status List (TSL)", Work in Progress, Internet-Draft, draft-ietf-oauth-status-list-20, 20 April 2026, <https://datatracker.ietf.org/doc/html/draft-ietf-oauth-status-list-20>.

[IANA.well-known] IANA, "Well-Known URIs", <https://www.iana.org/assignments/well-known-uris>.

[NIST-BLOG.VDCE] Fisher, B. and R. Galluzzo, "Digital Identities: Getting to Know the Verifiable Digital Credential Ecosystem", 13 November 2024, <https://www.nist.gov/blogs/cybersecurity-insights/digital-identities-getting-know-verifiable-digital-credential-ecosystem>.

[OWASP_SSRF] OWASP, "Server Side Request Forgery Prevention Cheat Sheet", <https://cheatsheetseries.owasp.org/cheatsheets/Server_Side_Request_Forgery_Prevention_Cheat_Sheet.html/>.

[RFC2397] Masinter, L., "The "data" URL scheme", RFC 2397, DOI 10.17487/RFC2397, August 1998, <https://www.rfc-editor.org/info/rfc2397>.

[RFC6901] Bryan, P., Ed., Zyp, K., and M. Nottingham, Ed., "JavaScript Object Notation (JSON) Pointer", RFC 6901, DOI 10.17487/RFC6901, April 2013, <https://www.rfc-editor.org/info/rfc6901>.

[RFC9535] Gssner, S., Ed., Normington, G., Ed., and C. Bormann, Ed., "JSONPath: Query Expressions for JSON", RFC 9535, DOI 10.17487/RFC9535, February 2024, <https://www.rfc-editor.org/info/rfc9535>.

Appendix A. IANA Considerations

A.1. JSON Web Token Claims Registration

* Claim Name: "vct"
* Claim Description: Verifiable digital Credential Type identifier
* Change Controller: IETF
* Specification Document(s): [[Section 2.2.2.1 of this specification]]
* Claim Name: "vct#integrity"
* Claim Description: SD-JWT VC vct claim "integrity metadata" value
* Change Controller: IETF
* Specification Document(s): [[Section 5 of this specification]]

A.2. Media Types Registry

A.2.1. application/dc+sd-jwt

The Internet media type for an SD-JWT VC is application/dc+sd-jwt.

* Type name: application
* Subtype name: dc+sd-jwt
* Required parameters: n/a
* Optional parameters: n/a
* Encoding considerations: 8-bit code points; SD-JWT VC values are encoded as a series of base64url-encoded values (some of which may be the empty string) separated by period ('.') and tilde ('~') characters.
* Security considerations: See Security Considerations in Section 6.
* Interoperability considerations: n/a
* Published specification: [[this specification]]
* Applications that use this media type: Applications that issue, present, and verify SD-JWT-based Verifiable Digital Credentials.
* Additional information:
  - Magic number(s): n/a
  - File extension(s): n/a
  - Macintosh file type code(s): n/a

* Person & email address to contact for further information: Oliver Terbu oliver.terbu@mattr.global (mailto:oliver.terbu@mattr.global)
* Intended usage: COMMON
* Restrictions on usage: none
* Author: Oliver Terbu oliver.terbu@mattr.global (mailto:oliver.terbu@mattr.global)
* Change controller: IETF

A.3. Well-Known URI Registry

This specification requests the well-known URI defined in Section 3 in the IANA "Well-Known URIs" registry [IANA.well-known] established by [RFC5785].

A.3.1. Registry Contents

* URI suffix: jwt-vc-issuer
* Change controller: IETF
* Specification document: [[Section 3 of this specification]]
* Related information: (none)
* Status: permanent

Appendix B. Examples

Important: The following examples are not normative and provided for illustrative purposes only. In particular, neither the structure of the claims nor the selection of selectively disclosable claims are normative.

Line breaks have been added for readability.

B.1. Example Person Identification Data (PID) Credential

This example shows how the artifacts defined in this specification could be used to represent the identity information of a person from a fictional European country. It leans on the concept of Person Identification Data (PID) as defined in the "PID Rulebook" in [EUDIW.ARF], but does not intend to fully comply with that specification.

Key Binding is applied using the Holder’s public key passed in a cnf claim in the SD-JWT.

The information in Figure 22 about the citizen comprises the input data used by the Issuer:

```
{
  "iss": "https://pid-issuer.aendgard.example",
  "vct": "urn:example:eudi:pid:aendgard:1",
  "given_name": "Astrid",
  "family_name": "Holmgren",
  "birthdate": "1978-04-10",
  "address": {
    "street_address": "Sjgata 12",
    "locality": "Viken",
    "postal_code": "12001",
    "country": "Kingdom of ndgard"
  },
  "nationalities": [
    "ndgard"
  ],
  "sex": 2,
  "birth_family_name": "Jensen",
  "place_of_birth": {
    "locality": "ndholm",
    "country": "Kingdom of ndgard"
  },
  "age_equal_or_over": {
    "12": true,
    "14": true,
    "16": true,
    "18": true,
    "21": true,
    "65": false
  },
  "age_in_years": 47,
  "age_birth_year": 1978,
  "portrait": "
    WAQMAAACSfKx9AAAAAXNSR0IB2cksfwAAAARnQU1BAACxjwv8YQUAAAAgY0hSTQA
    AeiYAAICEAAD6AAAAgOgAAHUwAADqYAAAOpgAABdwnLpRPAAAAAZQTFRFAAAA///
    /pdmf3QAAAFFJREFUCNdNylENgDAQA9ASBJyEWcDBLE0CTpDCpCDhPvexrBwlC/y
    8NG3RMunw9Ai5p6/pr/ZzlcAi4WiRN1nlXAesOgheBTSO8JDnND6Z3W+d90fCnTj
    jKQAAAABJRU5ErkJggg==",
  "issuance_date": "2024-01-15",
  "expiry_date": "2034-01-15",
  "issuing_authority": "Kingdom of ndgard, Ministry of Interior",
  "issuing_country": "G"
}
```


Figure 22: Citizen’s Information

The following in Figure 23 is the issued SD-JWT:

```
eyJhbGciOiAiRVMyNTYiLCAidHlwIjogImRjK3NkLWp3dCJ9.eyJfc2QiOiBbIjJyMDA
5ZHp2SHVWcldyUlhUNWtKTW1IbnFFSEhuV2UwTUxWWnc4UEFUQjgiLCAiM05kREpZUnd
CcWFFSXlYQ0hjbjRhNVlLSmFnUGdJUDBXa2c5QW9yZ0FLZyIsICI1Zm42SWlYalVQbUd
RNG1ZSHhEZHQ2RWp2QlE3Z1J2R2NkRW5TQ0VOOEVrIiwgIjZCek1CazdGR2l2b2dibGR
PNzVmM05Md2pGbGlDcWI3bm4xR2hnQ29rclEiLCAiOTBDVDhBYUJQYm41WDhuUlhrZXN
qdTFpMEJxaFdxWjN3cUQ0akYtcURHayIsICI5bkV3VXJkZVo2a2NSYkVqNTJRSDNuT1I
tb2llMF9wOXRNTklBRXVGdk1VIiwgIkhUaDZacjJKOGFpcXBhOTYzY0xrdUtlUURmOU8
wMEZ6T0hoeWpHcFZmVGciLCAiSkNwMksyTmhILUxFNUwwajdNRTBMTngtVC1weDlRRnR
hbVNzeG1DVUN4VSIsICJMb01HcmhsRG1fYlVXVVgxZ2RoUnhUc3k0ZDR6eXdHemxTVm9
jUlNZMDFNIiwgIlVnVWpxNXdoelFld081OWZWTmowZjBtcXlWOFlQWFU1NHNwNGtMZWN
3UWsiLCAiWTBILUUzaHpLRVVuQXlyaV94OXRUOWstajRpNjVhbzloa2VFSE5leG9zSSI
sICJlekNBbWFUd0kyblBqXzFJaUIzS1pmaUVPUkxlUndoWnNNTURNWklKOGtBIiwgImk
wSF8tV0FId2ZFanQ4dHFRSDc0dU9DV3ZxdVkzRnd1WC1reDRlMlJKSDgiLCAicjFHckg
0NnlEVjYyc2V6SW1FanhhNmNKU1pNVDIxMmpyT1NIaWVjOXl0TSIsICJyVmozUkxzN0s
4emROeTd6WHoxQ3IzOVk2cm9nOHVsTm96azRGN05DdUMwIiwgInQ1a3AwSGZ4TVMxZUh
OQ0dSenJLOUdpNW9UWnJlZzVVZTlsTFFJdUx2QjQiXSwgImlzcyI6ICJodHRwczovL3B
pZC1pc3N1ZXIuYWVuZGdhcmQuZXhhbXBsZSIsICJpYXQiOiAxNjgzMDAwMDAwLCAiZXh
wIjogMTg4MzAwMDAwMCwgInZjdCI6ICJ1cm46ZXhhbXBsZTpldWRpOnBpZDphZW5kZ2F
yZDoxIiwgIl9zZF9hbGciOiAic2hhLTI1NiIsICJjbmYiOiB7Imp3ayI6IHsia3R5Ijo
gIkVDIiwgImNydiI6ICJQLTI1NiIsICJ4IjogIlRDQUVSMTladnUzT0hGNGo0VzR2ZlN
Wb0hJUDFJTGlsRGxzN3ZDZUdlbWMiLCAieSI6ICJaeGppV1diWk1RR0hWV0tWUTRoYlN
JaXJzVmZ1ZWNDRTZ0NGpUOUYySFpRIn19fQ.NsDsoawVCSfu9cjYWtq47Cs8xX83PJi0
75Ap_79UTqik8WUim8gp6GOh3PzCIvzZt8yfMdtXYradxrHdsjUIkg~WyIyR0xDNDJzS
1F2ZUNmR2ZyeU5STjl3IiwgImdpdmVuX25hbWUiLCAiQXN0cmlkIl0~WyJlbHVWNU9nM
2dTTklOEVZbnN4QV9BIiwgImZhbWlseV9uYW1lIiwgIkhvbG1ncmVuIl0~WyI2SWo3d
E0tYTVpVlBHYm9TNXRtdlZBIiwgImJpcnRoZGF0ZSIsICIxOTc4LTA0LTEwIl0~WyJlS
ThaV205UW5LUHBOUGVOZW5IZGhRIiwgInN0cmVldF9hZGRyZXNzIiwgIlNqXHUwMGY4Z
2F0YSAxMiJd~WyJRZ19PNjR6cUF4ZTQxMmExMDhpcm9BIiwgImxvY2FsaXR5IiwgIlZp
a2VuIl0~WyJBSngtMDk1VlBycFR0TjRRTU9xUk9BIiwgInBvc3RhbF9jb2RlIiwgIjEy
MDAxIl0~WyJQYzMzSk0yTGNoY1VfbEhnZ3ZfdWZRIiwgImNvdW50cnkiLCAiS2luZ2Rv
bSBvZiBcdTAwYzZuZGdhcmQiXQ~WyJHMDJOU3JRZmpGWFE3SW8wOXN5YWpBIiwgImFkZ
HJlc3MiLCB7Il9zZCI6IFsiNjBCM0N3cTA0aGxQQndzdnNmdks5SlV4dnVMRlViSlFmZ
1BCYk9fSDlyTSIsICI4eWpQUjNyOGRPNUhXTG55MWdCZU1KVFBSZ2tCY2h1cTQzcUg4V
2xfZjFjIiwgIk5hcmQ3NHcyTl85YW5WRW1TbGVjRUpscXgzanhiY3F4dlZ6QnNCRFgwe
HMiLCAiVS16RDFESS03WjB6WW94WHF5NUhTVE5VZWdhaFJDSHlkQWVzS09fc0VQYyJdf
V0~WyJsa2x4RjVqTVlsR1RQVW92TU5JdkNBIiwgIm5hdGlvbmFsaXRpZXMiLCBbIlx1M
DBjNm5kZ2FyZCJdXQ~WyJuUHVvUW5rUkZxM0JJZUFtN0FuWEZBIiwgInNleCIsIDJd~W
yI1YlBzMUlxdVpOYTBoa2FGenp6Wk53IiwgImJpcnRoX2ZhbWlseV9uYW1lIiwgIkplb
nNlbiJd~WyI1YTJXMF9OcmxFWnpmcW1rXzdQcS13IiwgImxvY2FsaXR5IiwgIlx1MDBj
Nm5kaG9sbSJd~WyJ5MXNWVTV3ZGZKYWhWZGd3UGdTN1JRIiwgImNvdW50cnkiLCAiS2l
uZ2RvbSBvZiBcdTAwYzZuZGdhcmQiXQ~WyJIYlE0WDhzclZXM1FEeG5JSmRxeU9BIiwg
InBsYWNlX29mX2JpcnRoIiwgeyJfc2QiOiBbIkpET1lLUzB3MENvZm8tSGpaR3ZyZDdI
aXJwU2ZpWURBTFU2cjhUX0xFMWciLCAib1RacjlFbXk5eG1ycGNmUTVaU1U4c3Z4ZE1Z
OHk0WElGXzVqOVAxdTF5QSJdfV0~WyJDOUdTb3VqdmlKcXVFZ1lmb2pDYjFBIiwgIjEy
IiwgdHJ1ZV0~WyJreDVrRjE3Vi14MEptd1V4OXZndnR3IiwgIjE0IiwgdHJ1ZV0~WyJI
M28xdXN3UDc2MEZpMnllR2RWQ0VRIiwgIjE2IiwgdHJ1ZV0~WyJPQktsVFZsdkxnLUFk
d3FZR2JQOFpBIiwgIjE4IiwgdHJ1ZV0~WyJNMEpiNTd0NDF1YnJrU3V5ckRUM3hBIiwg
IjIxIiwgdHJ1ZV0~WyJEc210S05ncFY0ZEFIcGpyY2Fvc0F3IiwgIjY1IiwgZmFsc2Vd
```

~WyJlSzVvNXBIZmd1cFBwbHRqMXFoQUp3IiwgImFnZV9lcXVhbF9vcl9vdmVyIiwgeyJ
fc2QiOiBbIjF0RWl5elBSWU9Lc2Y3U3NZR01nUFpLc09UMWxRWlJ4SFhBMHI1X0J3a2s
iLCAiQ1ZLbmx5NVA5MHlKczNFd3R4UWlPdFVjemFYQ1lOQTRJY3pSYW9ock1EZyIsICJ
hNDQtZzJHcjhfM0FtSncyWFo4a0kxeTBRel96ZTlpT2NXMlczUkxwWEdnIiwgImdrdnk
wRnV2QkJ2ajBoczJaTnd4Y3FPbGY4bXUyLWtDRTctTmIyUXh1QlUiLCAiaHJZNEhubUY
1YjVKd0M5ZVR6YUZDVWNlSVFBYUlkaHJxVVhRTkNXYmZaSSIsICJ5NlNGclZGUnlxNTB
JYlJKdmlUWnFxalFXejB0TGl1Q21NZU8wS3FhekdJIl19XQ~WyJqN0FEZGIwVVZiMExp
MGNpUGNQMGV3IiwgImFnZV9pbl95ZWFycyIsIDQ3XQ~WyJXcHhKckZ1WDh1U2kycDRod
DA5anZ3IiwgImFnZV9iaXJ0aF95ZWFyIiwgMTk3OF0~WyJhdFNtRkFDWU1iSlZLRDA1b
zNKZ3RRIiwgInBvcnRyYWl0IiwgImRhdGE6aW1hZ2UvcG5nO2Jhc2U2NCxpVkJPUncwS
0dnb0FBQUFOU1VoRVVnQUFBQ1FBQUFBV0FRTUFBQUNTZkt4OUFBQUFBWE5TUjBJQjJja
3Nmd0FBQUFSblFVMUJBQUN4and2OFlRVUFBQUFnWTBoU1RRQUFlaVlBQUlDRUFBRDZBQ
UFBZ09nQUFIVXdBQURxWUFBQU9wZ0FBQmR3bkxwUlBBQUFBQVpRVEZSRkFBQUEvLy8vc
GRtZjNRQUFBRkZKUkVGVUNOZE55bEVOZ0RBUUE5QVNCSnlFV2NEQkxFMENUcERDcENEa
FB2ZXhyQndsQy95OE5HM1JNdW53OUFpNXA2L3ByL1p6bGNBaTRXaVJOMW5sWEFlc09na
GVCVFNPOEpEbk5ENlozVytkOTBmQ25UampLUUFBQUFCSlJVNUVya0pnZ2c9PSJd~WyI0
S3lSMzJvSVp0LXprV3ZGcWJVTEtnIiwgImlzc3VhbmNlX2RhdGUiLCAiMjAyNC0wMS0x
NSJd~WyJjaEJDc3loeWgtSjg2SS1hd1FEaUNRIiwgImV4cGlyeV9kYXRlIiwgIjIwMzQ
tMDEtMTUiXQ~WyJmbE5QMW5jTXo5TGctYzlxTUl6XzlnIiwgImlzc3VpbmdfYXV0aG9y
aXR5IiwgIktpbmdkb20gb2YgXHUwMGM2bmRnYXJkLCBNaW5pc3RyeSBvZiBJbnRlcmlv
ciJd~WyJvdm9WMXJraF9BTm04NjFxVUFBMkF3IiwgImlzc3VpbmdfY291bnRyeSIsICJ
cdTAwYzZHIl0~

Figure 23: Issued SD-JWT

Figure 24 is the payload of that SD-JWT:

```
{
  "_sd": [
    "2r009dzvHuVrWrRXT5kJMmHnqEHHnWe0MLVZw8PATB8",
    "3NdDJYRwBqaEIyXCHcn4a5YKJagPgIP0Wkg9AorgAKg",
    "5fn6IiXjUPmGQ4mYHxDdt6EjvBQ7gRvGcdEnSCEN8Ek",
    "6BzMBk7FGivogbldO75f3NLwjFliCqb7nn1GhgCokrQ",
    "90CT8AaBPbn5X8nRXkesju1i0BqhWqZ3wqD4jF-qDGk",
    "9nEwUrdeZ6kcRbEj52QH3nOR-oie0_p9tMNIAEuFvMU",
    "HTh6Zr2J8aiqpa963cLkuKeQDf9O00FzOHhyjGpVfTg",
    "JCp2K2NhH-LE5L0j7ME0LNx-T-px9QFtamSsxmCUCxU",
    "LoMGrhlDm_bUWUX1gdhRxTsy4d4zywGzlSVocRSY01M",
    "UgUjq5whzQewO59fVNj0f0mqyV8YPXU54sp4kLecwQk",
    "Y0H-E3hzKEUnAyri_x9tT9k-j4i65ao9hkeEHNexosI",
    "ezCAmaTwI2nPj_1IiB3KZfiEORLeRwhZsMMDMZIJ8kA",
    "i0H_-WAHwfEjt8tqQH74uOCWvquY3FwuX-kx4e2RJH8",
    "r1GrH46yDV62sezImEjxa6cJSZMT212jrOSHiec9ytM",
    "rVj3RLs7K8zdNy7zXz1Cr39Y6rog8ulNozk4F7NCuC0",
    "t5kp0HfxMS1eHNCGRzrK9Gi5oTZreg5Ue9lLQIuLvB4"
  ],
  "iss": "https://pid-issuer.aendgard.example",
  "iat": 1683000000,
  "exp": 1883000000,
  "vct": "urn:example:eudi:pid:aendgard:1",
  "_sd_alg": "sha-256",
  "cnf": {
    "jwk": {
      "kty": "EC",
      "crv": "P-256",
      "x": "TCAER19Zvu3OHF4j4W4vfSVoHIP1ILilDls7vCeGemc",
      "y": "ZxjiWWbZMQGHVWKVQ4hbSIirsVfuecCE6t4jT9F2HZQ"
    }
  }
}
```


Figure 24: SD-JWT Payload

The digests in the SD-JWT payload reference the following Disclosures:

Claim given_name:

* SHA-256 Hash: 3NdDJYRwBqaEIyXCHcn4a5YKJagPgIP0Wkg9AorgAKg
* Disclosure: WyIyR0xDNDJzS1F2ZUNmR2ZyeU5STjl3IiwgImdpdmVuX25hbWUiLCAiQXN0cmlkIl0
* Contents: ["2GLC42sKQveCfGfryNRN9w", "given_name", "Astrid"]

Claim family_name:

* SHA-256 Hash: JCp2K2NhH-LE5L0j7ME0LNx-T-px9QFtamSsxmCUCxU
* Disclosure:
  WyJlbHVWNU9nM2dTTklJOEVZbnN4QV9BIiwgImZhbWlseV9uYW1lIiwgIkhv
  bG1ncmVuIl0
* Contents: ["eluV5Og3gSNII8EYnsxA_A", "family_name", "Holmgren"]

Claim birthdate:

* SHA-256 Hash: 5fn6IiXjUPmGQ4mYHxDdt6EjvBQ7gRvGcdEnSCEN8Ek
* Disclosure:
  WyI2SWo3dE0tYTVpVlBHYm9TNXRtdlZBIiwgImJpcnRoZGF0ZSIsICIxOTc4
  LTA0LTEwIl0
* Contents: ["6Ij7tM-a5iVPGboS5tmvVA", "birthdate", "1978-04-10"]

Claim street_address:

* SHA-256 Hash: 8yjPR3r8dO5HWLny1gBeMJTPRgkBchuq43qH8Wl_f1c
* Disclosure:
  WyJlSThaV205UW5LUHBOUGVOZW5IZGhRIiwgInN0cmVldF9hZGRyZXNzIiwg
  IlNqXHUwMGY4Z2F0YSAxMiJd
* Contents: ["eI8ZWm9QnKPpNPeNenHdhQ", "street_address",
  "Sj\u00f8gata
  12"]

Claim locality:

* SHA-256 Hash: Nard74w2N_9anVEmSlecEJlqx3jxbcqxvVzBsBDX0xs
* Disclosure:
  WyJRZ19PNjR6cUF4ZTQxMmExMDhpcm9BIiwgImxvY2FsaXR5IiwgIlZpa2Vu
  Il0
* Contents: ["Qg_O64zqAxe412a108iroA", "locality", "Viken"]

Claim postal_code:

* SHA-256 Hash: 60B3Cwq04hlPBwsvsfvK9JUxvuLFUbJQfgPBbO_H9rM
* Disclosure:
  WyJBSngtMDk1VlBycFR0TjRRTU9xUk9BIiwgInBvc3RhbF9jb2RlIiwgIjEy
  MDAxIl0
* Contents: ["AJx-095VPrpTtN4QMOqROA", "postal_code", "12001"]

Claim country:

* SHA-256 Hash: U-zD1DI-7Z0zYoxXqy5HSTNUegahRCHydAesKO_sEPc
* Disclosure:
  WyJQYzMzSk0yTGNoY1VfbEhnZ3ZfdWZRIiwgImNvdW50cnkiLCAiS2luZ2Rv
  bSBvZiBcdTAwYzZuZGdhcmQiXQ
* Contents: ["Pc33JM2LchcU_lHggv_ufQ", "country", "Kingdom of
  \u00c6ndgard"]

Claim address:

* SHA-256 Hash: i0H_-WAHwfEjt8tqQH74uOCWvquY3FwuX-kx4e2RJH8
* Disclosure:
  WyJHMDJOU3JRZmpGWFE3SW8wOXN5YWpBIiwgImFkZHJlc3MiLCB7Il9zZCI6
  IFsiNjBCM0N3cTA0aGxQQndzdnNmdks5SlV4dnVMRlViSlFmZ1BCYk9fSDly
  TSIsICI4eWpQUjNyOGRPNUhXTG55MWdCZU1KVFBSZ2tCY2h1cTQzcUg4V2xf
  ZjFjIiwgIk5hcmQ3NHcyTl85YW5WRW1TbGVjRUpscXgzanhiY3F4dlZ6QnNC
  RFgweHMiLCAiVS16RDFESS03WjB6WW94WHF5NUhTVE5VZWdhaFJDSHlkQWVz
  S09fc0VQYyJdfV0
* Contents: ["G02NSrQfjFXQ7Io09syajA", "address", {"_sd":
  ["60B3Cwq04hlPBwsvsfvK9JUxvuLFUbJQfgPBbO_H9rM",
  "8yjPR3r8dO5HWLny1gBeMJTPRgkBchuq43qH8Wl_f1c",
  "Nard74w2N_9anVEmSlecEJlqx3jxbcqxvVzBsBDX0xs",
  "U-zD1DI-7Z0zYoxXqy5HSTNUegahRCHydAesKO_sEPc"]}]

Claim nationalities:

* SHA-256 Hash: HTh6Zr2J8aiqpa963cLkuKeQDf9O00FzOHhyjGpVfTg
* Disclosure:
  WyJsa2x4RjVqTVlsR1RQVW92TU5JdkNBIiwgIm5hdGlvbmFsaXRpZXMiLCBb
  Ilx1MDBjNm5kZ2FyZCJdXQ
* Contents: ["lklxF5jMYlGTPUovMNIvCA", "nationalities",
  ["\u00c6ndgard"]]

Claim sex:

* SHA-256 Hash: 90CT8AaBPbn5X8nRXkesju1i0BqhWqZ3wqD4jF-qDGk
* Disclosure:
  WyJuUHVvUW5rUkZxM0JJZUFtN0FuWEZBIiwgInNleCIsIDJd
* Contents: ["nPuoQnkRFq3BIeAm7AnXFA", "sex", 2]

Claim birth_family_name:

* SHA-256 Hash: t5kp0HfxMS1eHNCGRzrK9Gi5oTZreg5Ue9lLQIuLvB4
* Disclosure:
  WyI1YlBzMUlxdVpOYTBoa2FGenp6Wk53IiwgImJpcnRoX2ZhbWlseV9uYW1l
  IiwgIkplbnNlbiJd
* Contents: ["5bPs1IquZNa0hkaFzzzZNw", "birth_family_name",
  "Jensen"]

Claim locality:

* SHA-256 Hash: JDOYKS0w0Cofo-HjZGvrd7HirpSfiYDALU6r8T_LE1g
* Disclosure:
  WyI1YTJXMF9OcmxFWnpmcW1rXzdQcS13IiwgImxvY2FsaXR5IiwgIlx1MDBj
  Nm5kaG9sbSJd
* Contents: ["5a2W0_NrlEZzfqmk_7Pq-w", "locality", "\u00c6ndholm"]

Claim country:

* SHA-256 Hash: oTZr9Emy9xmrpcfQ5ZSU8svxdMY8y4XIF_5j9P1u1yA
* Disclosure:
  WyJ5MXNWVTV3ZGZKYWhWZGd3UGdTN1JRIiwgImNvdW50cnkiLCAiS2luZ2Rv
  bSBvZiBcdTAwYzZuZGdhcmQiXQ
* Contents: ["y1sVU5wdfJahVdgwPgS7RQ", "country", "Kingdom of
  \u00c6ndgard"]

Claim place_of_birth:

* SHA-256 Hash: r1GrH46yDV62sezImEjxa6cJSZMT212jrOSHiec9ytM
* Disclosure:
  WyJIYlE0WDhzclZXM1FEeG5JSmRxeU9BIiwgInBsYWNlX29mX2JpcnRoIiwg
  eyJfc2QiOiBbIkpET1lLUzB3MENvZm8tSGpaR3ZyZDdIaXJwU2ZpWURBTFU2
  cjhUX0xFMWciLCAib1RacjlFbXk5eG1ycGNmUTVaU1U4c3Z4ZE1ZOHk0WElG
  XzVqOVAxdTF5QSJdfV0
* Contents: ["HbQ4X8srVW3QDxnIJdqyOA", "place_of_birth", {"_sd":
  ["JDOYKS0w0Cofo-HjZGvrd7HirpSfiYDALU6r8T_LE1g",
  "oTZr9Emy9xmrpcfQ5ZSU8svxdMY8y4XIF_5j9P1u1yA"]}]

Claim 12:

* SHA-256 Hash: gkvy0FuvBBvj0hs2ZNwxcqOlf8mu2-kCE7-Nb2QxuBU
* Disclosure:
  WyJDOUdTb3VqdmlKcXVFZ1lmb2pDYjFBIiwgIjEyIiwgdHJ1ZV0
* Contents: ["C9GSoujviJquEgYfojCb1A", "12", true]

Claim 14:

* SHA-256 Hash: y6SFrVFRyq50IbRJviTZqqjQWz0tLiuCmMeO0KqazGI
* Disclosure:
  WyJreDVrRjE3Vi14MEptd1V4OXZndnR3IiwgIjE0IiwgdHJ1ZV0
* Contents: ["kx5kF17V-x0JmwUx9vgvtw", "14", true]

Claim 16:

* SHA-256 Hash: hrY4HnmF5b5JwC9eTzaFCUceIQAaIdhrqUXQNCWbfZI
* Disclosure:
  WyJIM28xdXN3UDc2MEZpMnllR2RWQ0VRIiwgIjE2IiwgdHJ1ZV0
* Contents: ["H3o1uswP760Fi2yeGdVCEQ", "16", true]

Claim 18:

* SHA-256 Hash: CVKnly5P90yJs3EwtxQiOtUczaXCYNA4IczRaohrMDg
* Disclosure:
  WyJPQktsVFZsdkxnLUFkd3FZR2JQOFpBIiwgIjE4IiwgdHJ1ZV0
* Contents: ["OBKlTVlvLg-AdwqYGbP8ZA", "18", true]

Claim 21:

* SHA-256 Hash: 1tEiyzPRYOKsf7SsYGMgPZKsOT1lQZRxHXA0r5_Bwkk
* Disclosure:
  WyJNMEpiNTd0NDF1YnJrU3V5ckRUM3hBIiwgIjIxIiwgdHJ1ZV0
* Contents: ["M0Jb57t41ubrkSuyrDT3xA", "21", true]

Claim 65:

* SHA-256 Hash: a44-g2Gr8_3AmJw2XZ8kI1y0Qz_ze9iOcW2W3RLpXGg
* Disclosure:
  WyJEc210S05ncFY0ZEFIcGpyY2Fvc0F3IiwgIjY1IiwgZmFsc2Vd
* Contents: ["DsmtKNgpV4dAHpjrcaosAw", "65", false]

Claim age_equal_or_over:

* SHA-256 Hash: 2r009dzvHuVrWrRXT5kJMmHnqEHHnWe0MLVZw8PATB8
* Disclosure:
  WyJlSzVvNXBIZmd1cFBwbHRqMXFoQUp3IiwgImFnZV9lcXVhbF9vcl9vdmVy
  IiwgeyJfc2QiOiBbIjF0RWl5elBSWU9Lc2Y3U3NZR01nUFpLc09UMWxRWlJ4
  SFhBMHI1X0J3a2siLCAiQ1ZLbmx5NVA5MHlKczNFd3R4UWlPdFVjemFYQ1lO
  QTRJY3pSYW9ock1EZyIsICJhNDQtZzJHcjhfM0FtSncyWFo4a0kxeTBRel96
  ZTlpT2NXMlczUkxwWEdnIiwgImdrdnkwRnV2QkJ2ajBoczJaTnd4Y3FPbGY4
  bXUyLWtDRTctTmIyUXh1QlUiLCAiaHJZNEhubUY1YjVKd0M5ZVR6YUZDVWNl
  SVFBYUlkaHJxVVhRTkNXYmZaSSIsICJ5NlNGclZGUnlxNTBJYlJKdmlUWnFx
  alFXejB0TGl1Q21NZU8wS3FhekdJIl19XQ
* Contents: ["eK5o5pHfgupPpltj1qhAJw", "age_equal_or_over", {"_sd":
  ["1tEiyzPRYOKsf7SsYGMgPZKsOT1lQZRxHXA0r5_Bwkk",
  "CVKnly5P90yJs3EwtxQiOtUczaXCYNA4IczRaohrMDg",
  "a44-g2Gr8_3AmJw2XZ8kI1y0Qz_ze9iOcW2W3RLpXGg",
  "gkvy0FuvBBvj0hs2ZNwxcqOlf8mu2-kCE7-Nb2QxuBU",
  "hrY4HnmF5b5JwC9eTzaFCUceIQAaIdhrqUXQNCWbfZI",
  "y6SFrVFRyq50IbRJviTZqqjQWz0tLiuCmMeO0KqazGI"]}]

Claim age_in_years:

* SHA-256 Hash: LoMGrhlDm_bUWUX1gdhRxTsy4d4zywGzlSVocRSY01M
* Disclosure:
  WyJqN0FEZGIwVVZiMExpMGNpUGNQMGV3IiwgImFnZV9pbl95ZWFycyIsIDQ3
  XQ
* Contents: ["j7ADdb0UVb0Li0ciPcP0ew", "age_in_years", 47]

Claim age_birth_year:

* SHA-256 Hash: rVj3RLs7K8zdNy7zXz1Cr39Y6rog8ulNozk4F7NCuC0
* Disclosure:
  WyJXcHhKckZ1WDh1U2kycDRodDA5anZ3IiwgImFnZV9iaXJ0aF95ZWFyIiwg
  MTk3OF0

* Contents: ["WpxJrFuX8uSi2p4ht09jvw", "age_birth_year", 1978]

Claim portrait:

* SHA-256 Hash: 9nEwUrdeZ6kcRbEj52QH3nOR-oie0_p9tMNIAEuFvMU
* Disclosure:
  WyJhdFNtRkFDWU1iSlZLRDA1bzNKZ3RRIiwgInBvcnRyYWl0IiwgImRhdGE6
  aW1hZ2UvcG5nO2Jhc2U2NCxpVkJPUncwS0dnb0FBQUFOU1VoRVVnQUFBQ1FB
  QUFBV0FRTUFBQUNTZkt4OUFBQUFBWE5TUjBJQjJja3Nmd0FBQUFSblFVMUJB
  QUN4and2OFlRVUFBQUFnWTBoU1RRQUFlaVlBQUlDRUFBRDZBQUFBZ09nQUFI
  VXdBQURxWUFBQU9wZ0FBQmR3bkxwUlBBQUFBQVpRVEZSRkFBQUEvLy8vcGRt
  ZjNRQUFBRkZKUkVGVUNOZE55bEVOZ0RBUUE5QVNCSnlFV2NEQkxFMENUcERD
  cENEaFB2ZXhyQndsQy95OE5HM1JNdW53OUFpNXA2L3ByL1p6bGNBaTRXaVJO
  MW5sWEFlc09naGVCVFNPOEpEbk5ENlozVytkOTBmQ25UampLUUFBQUFCSlJV
  NUVya0pnZ2c9PSJd
* Contents: ["atSmFACYMbJVKD05o3JgtQ", "portrait", "data:image/
  png;base6
  4,iVBORw0KGgoAAAANSUhEUgAAACQAAAAWAQMAAACSfKx9AAAAAXNSR0IB2c
  ksfwAAAARnQU1BAACxjwv8YQUAAAAgY0hSTQAAeiYAAICEAAD6AAAAgOgAAH
  UwAADqYAAAOpgAABdwnLpRPAAAAAZQTFRFAAAA////pdmf3QAAAFFJREFUCN
  dNylENgDAQA9ASBJyEWcDBLE0CTpDCpCDhPvexrBwlC/y8NG3RMunw9Ai5p6
  /pr/ZzlcAi4WiRN1nlXAesOgheBTSO8JDnND6Z3W+d90fCnTjjKQAAAABJRU
  5ErkJggg==">

Claim issuance_date:

* SHA-256 Hash: ezCAmaTwI2nPj_1IiB3KZfiEORLeRwhZsMMDMZIJ8kA
* Disclosure:
  WyI0S3lSMzJvSVp0LXprV3ZGcWJVTEtnIiwgImlzc3VhbmNlX2RhdGUiLCAi
  MjAyNC0wMS0xNSJd
* Contents: ["4KyR32oIZt-zkWvFqbULKg", "issuance_date",
  "2024-01-15"]

Claim expiry_date:

* SHA-256 Hash: Y0H-E3hzKEUnAyri_x9tT9k-j4i65ao9hkeEHNexosI
* Disclosure:
  WyJjaEJDc3loeWgtSjg2SS1hd1FEaUNRIiwgImV4cGlyeV9kYXRlIiwgIjIw
  MzQtMDEtMTUiXQ
* Contents: ["chBCsyhyh-J86I-awQDiCQ", "expiry_date", "2034-01-15"]

Claim issuing_authority:

* SHA-256 Hash: 6BzMBk7FGivogbldO75f3NLwjFliCqb7nn1GhgCokrQ
* Disclosure:
  WyJmbE5QMW5jTXo5TGctYzlxTUl6XzlnIiwgImlzc3VpbmdfYXV0aG9yaXR5
  IiwgIktpbmdkb20gb2YgXHUwMGM2bmRnYXJkLCBNaW5pc3RyeSBvZiBJbnRl
  cmlvciJd

* Contents: ["flNP1ncMz9Lg-c9qMIz_9g", "issuing_authority", "Kingdom of
\u00c6ndgard, Ministry of Interior"]

Claim issuing_country:

* SHA-256 Hash: UgUjq5whzQewO59fVNj0f0mqyV8YPXU54sp4kLecwQk
* Disclosure:
WyJvdm9WMXJraF9BTm04NjFxVUFBMkF3IiwgImlzc3VpbmdfY291bnRyeSIs
ICJcdTAwYzZHIl0
* Contents: ["ovoV1rkh_ANm861qUAA2Aw", "issuing_country", "\u00c6G"]

Figure 25 shows a presentation of the SD-JWT with a Key Binding JWT that discloses only nationality and the fact that the person is over 18 years old:

```
eyJhbGciOiAiRVMyNTYiLCAidHlwIjogImRjK3NkLWp3dCJ9.eyJfc2QiOiBbIjJyMDA
5ZHp2SHVWcldyUlhUNWtKTW1IbnFFSEhuV2UwTUxWWnc4UEFUQjgiLCAiM05kREpZUnd
CcWFFSXlYQ0hjbjRhNVlLSmFnUGdJUDBXa2c5QW9yZ0FLZyIsICI1Zm42SWlYalVQbUd
RNG1ZSHhEZHQ2RWp2QlE3Z1J2R2NkRW5TQ0VOOEVrIiwgIjZCek1CazdGR2l2b2dibGR
PNzVmM05Md2pGbGlDcWI3bm4xR2hnQ29rclEiLCAiOTBDVDhBYUJQYm41WDhuUlhrZXN
qdTFpMEJxaFdxWjN3cUQ0akYtcURHayIsICI5bkV3VXJkZVo2a2NSYkVqNTJRSDNuT1I
tb2llMF9wOXRNTklBRXVGdk1VIiwgIkhUaDZacjJKOGFpcXBhOTYzY0xrdUtlUURmOU8
wMEZ6T0hoeWpHcFZmVGciLCAiSkNwMksyTmhILUxFNUwwajdNRTBMTngtVC1weDlRRnR
hbVNzeG1DVUN4VSIsICJMb01HcmhsRG1fYlVXVVgxZ2RoUnhUc3k0ZDR6eXdHemxTVm9
jUlNZMDFNIiwgIlVnVWpxNXdoelFld081OWZWTmowZjBtcXlWOFlQWFU1NHNwNGtMZWN
3UWsiLCAiWTBILUUzaHpLRVVuQXlyaV94OXRUOWstajRpNjVhbzloa2VFSE5leG9zSSI
sICJlekNBbWFUd0kyblBqXzFJaUIzS1pmaUVPUkxlUndoWnNNTURNWklOGtBIiwgImk
wSF8tV0FId2ZFanQ4dHFRSDc0dU9DV3ZxdVkzRnd1WC1reDRlMlJKSDgiLCAicjFHckg
0NnlEVjYyc2V6SW1FanhhNmNKU1pNVDIxMmpyT1NIaWVjOXl0TSIsICJyVmozUkxzN0s
4emROeTd6WHoxQ3IzOVk2cm9nOHVsTm96azRGN05DdUMwIiwgInQ1a3AwSGZ4TVMxZUh
OQ0dSenJLOUdpNW9UWnJlZzVVZTlsTFFJdUx2QjQiXSwgImlzcyI6ICJodHRwczovL3B
pZC1pc3N1ZXIuYWVuZGdhcmQuZXhhbXBsZSIsICJpYXQiOiAxNjgzMDAwMDAwLCAiZXh
wIjogMTg4MzAwMDAwMCwgInZjdCI6ICJ1cm46ZXhhbXBsZTpldWRpOnBpZDphZW5kZ2F
yZDoxIiwgIl9zZF9hbGciOiAic2hhLTI1NiIsICJjbmYiOiB7Imp3ayI6IHsia3R5Ijo
gIkVDIiwgImNydiI6ICJQLTI1NiIsICJ4IjogIlRDQUVSMTladnUzT0hGNGo0VzR2ZlN
Wb0hJUDFJTGlsRGxzN3ZDZUdlbWMiLCAieSI6ICJaeGppV1diWk1RR0hWV0tWUTRoYlN
JaXJzVmZ1ZWNDRTZ0NGpUOUYySFpRIn19fQ.NsDsoawVCSfu9cjYWtq47Cs8xX83PJi0
75Ap_79UTqik8WUim8gp6GOh3PzCIvzZt8yfMdtXYradxrHdsjUIkg~WyJlSzVvNXBIZ
md1cFBwbHRqMXFoQUp3IiwgImFnZV9lcXVhbF9vcl9vdmVyIiwgeyJfc2QiOiBbIjF0R
Wl5elBSWU9Lc2Y3U3NZR01nUFpLc09UMWxRWlJ4SFhBMHI1X0J3a2siLCAiQ1ZLbmx5N
VA5MHlKczNFd3R4UWlPdFVjemFYQ1lOQTRJY3pSYW9ock1EZyIsICJhNDQtZzJHcjhfM
0FtSncyWFo4a0kxeTBRel96ZTlpT2NXMlczUkxwWEdnIiwgImdrdnkwRnV2QkJ2ajBoc
zJaTnd4Y3FPbGY4bXUyLWtDRTctTmIyUXh1QlUiLCAiaHJZNEhubUY1YjVKd0M5ZVR6Y
UZDVWNlSVFBYUlkaHJxVVhRTkNXYmZaSSIsICJ5NlNGclZGUnlxNTBJYlJKdmlUWnFxa
lFXejB0TGl1Q21NZU8wS3FhekdJIl19XQ~WyJPQktsVFZsdkxnLUFkd3FZR2JQOFpBIi
wgIjE4IiwgdHJ1ZV0~WyJsa2x4RjVqTVlsR1RQVW92TU5JdkNBIiwgIm5hdGlvbmFsaX
RpZXMiLCBbIlx1MDBjNm5kZ2FyZCJdXQ~eyJhbGciOiAiRVMyNTYiLCAidHlwIjogImt
iK2p3dCJ9.eyJub25jZSI6ICIxMjM0NTY3ODkwIiwgImF1ZCI6ICJodHRwczovL2V4YW
1wbGUuY29tL3ZlcmlmaWVyIiwgImlhdCI6IDE3NzcwNTY4NDgsICJzZF9oYXNoIjogIm
h3UUg0bklDU2ZfLWJlNklBNlJEMEdDZVQ0dHhudFZOYzE1M1QwTVRWZ2sifQ.T5DFB9R
CoKFDWHV8s4u1_HLgdQxo7Cecpp3mue1B6_ssDlKnuixyixs7AuNFv7m7S2JOqM1oQP3
Gv3h-0r-u9g
```

Figure 25: Presented SD-JWT+KB

Figure 26 is the payload of a corresponding Key Binding JWT:

```
{
  "nonce": "1234567890",
  "aud": "https://example.com/verifier",
  "iat": 1777056848,
  "sd_hash": "hwQH4nICSf_-be6IA6RD0GCeT4txntVNc153T0MTVgk"
}
```

Figure 26: Key Binding JWT Payload

After validation, the Verifier will have the processed SD-JWT payload in Figure 27 available for further handling:

```
{
  "iss": "https://pid-issuer.aendgard.example",
  "iat": 1683000000,
  "exp": 1883000000,
  "vct": "urn:example:eudi:pid:aendgard:1",
  "cnf": {
    "jwk": {
      "kty": "EC",
      "crv": "P-256",
      "x": "TCAER19Zvu3OHF4j4W4vfSVoHIP1ILilDls7vCeGemc",
      "y": "ZxjiWWbZMQGHVWKVQ4hbSIirsVfuecCE6t4jT9F2HZQ"
    }
  },
  "age_equal_or_over": {
    "18": true
  },
  "nationalities": [
    "ndgard"
  ]
}
```

Figure 27: Processed SD-JWT Payload

B.2. Example Type Metadata

The following example for Type Metadata assumes an SD-JWT VC payload structured as follows in Figure 28:

```
{
  "vct": "https://betelgeuse.example.com/education_credential/v42",
  "vct#integrity": "sha256-1odmyxoVQCuQx8SAym8rWHXba41fM/Iv/V1H8VHGN00=",
  "name": "Zaphod Beeblebrox",
  "address": {
    "street_address": "42 Galaxy Way",
    "city": "Betelgeuse City",
    "postal_code": "12345",
    "country": "Betelgeuse"
  },
  "degrees": [
    {
      "field_of_study": "Intergalactic Politics",
      "date_awarded": "2020-05-15"
    },
    {
      "field_of_study": "Space Navigation",
      "date_awarded": "2018-06-20"
    },
    {
      "field_of_study": "Quantum Mechanics",
      "date_awarded": "2016-07-25"
    }
  ]
}
```

Figure 28: SD-JWT VC Payload

The Type Metadata for this SD-JWT VC could be defined as follows in Figure 29:

```
{
  "vct": "https://betelgeuse.example.com/education_credential/v42",
  "name": "Betelgeuse Education Credential - First Version",
  "description": "This is our first version of the education credential. Don’t panic."
,
  "extends": "https://galaxy.example.com/galactic-education-credential/v2",
  "extends#integrity": "sha256-ilOUJsTultOwLfz7QUcFALaRa3BP/jelX1ds04kB9yU=",
  "display": [
    {
      "locale": "en-US",
      "name": "Betelgeuse Education Credential",
      "description": "An education credential for all carbon-based life forms on Betel
geuse.",
      "rendering": {
        "simple": {
          "logo": {
            "uri": "https://betelgeuse.example.com/public/education-logo.png",
            "uri#integrity": "sha256-LmXfh+9cLlJNXN+TsMk+PmKjZ5t0WRL5ca/xGgX3c1U=",
            "alt_text": "Betelgeuse Ministry of Education logo"
```

```
            },
            "background_image": {
              "uri": "https://betelgeuse.example.com/public/credential-background.png",
              "uri#integrity": "sha256-5sBT7mMLylHLWrrS/qQ8aHpRAxoraWVmWX6eUVMlrrA="
            },
            "background_color": "#12107c",
            "text_color": "#FFFFFF"
          },
          "svg_templates": [
            {
              "uri": "https://betelgeuse.example.com/public/credential-english.svg",
              "uri#integrity": "sha256-I4JcBGO7UfrkOBrsV7ytNJAfGuKLQh+e+Z31mc7iAb4=",
              "properties": {
                "orientation": "landscape",
                "color_scheme": "light",
                "contrast": "high"
              }
            }
          ]
        }
      },
      {
        "locale": "de-DE",
        "name": "Betelgeuse-Bildungsnachweis",
        "description": "Ein Bildungsnachweis fr alle kohlenstoffbasierten Lebensformen a
uf Betelgeuse.",
        "rendering": {
          "simple": {
            "logo": {
              "uri": "https://betelgeuse.example.com/public/education-logo-de.png",
              "uri#integrity": "sha256-LmXfh+9cLlJNXN+TsMk+PmKjZ5t0WRL5ca/xGgX3c1U=",
              "alt_text": "Logo des Betelgeusischen Bildungsministeriums"
            },
            "background_image": {
              "uri": "https://betelgeuse.example.com/public/credential-background-de.png
",
              "uri#integrity": "sha256-9cLlJNXN+TsMk+PmKjZ5t0WRL5ca/xGgX3c1ULmXfh="
            },
            "background_color": "#12107c",
            "text_color": "#FFFFFF"
          },
          "svg_templates": [
            {
              "uri": "https://betelgeuse.example.com/public/credential-german.svg",
              "uri#integrity": "sha256-I4JcBGO7UfrkOBrsV7ytNJAfGuKLQh+e+Z31mc7iAb4=",
              "properties": {
                "orientation": "landscape",
                "color_scheme": "light",
                "contrast": "high"
              }
```

```
            }
          ]
        }
      }
    ],
    "claims": [
      {
        "path": ["name"],
        "display": [
          {
            "locale": "de-DE",
            "label": "Vor- und Nachname",
            "description": "Der Name des/der Studierenden"
          },
          {
            "locale": "en-US",
            "label": "Name",
            "description": "The name of the student"
          }
        ],
        "sd": "always",
        "mandatory": true
      },
      {
        "path": ["address"],
        "display": [
          {
            "locale": "de-DE",
            "label": "Adresse",
            "description": "Adresse zum Zeitpunkt des Abschlusses"
          },
          {
            "locale": "en-US",
            "label": "Address",
            "description": "Address at the time of graduation"
          }
        ],
        "sd": "always"
      },
      {
        "path": ["address", "street_address"],
        "display": [
          {
            "locale": "de-DE",
            "label": "Strae"
          },
          {
            "locale": "en-US",
```

```
            "label": "Street Address"
          }
        ],
        "sd": "always",
        "svg_id": "address_street_address"
      },
      {
        "path": ["degrees"],
        "display": [
          {
            "locale": "de-DE",
            "label": "Abschlsse",
            "description": "Abschlsse des/der Studierenden"
          },
          {
            "locale": "en-US",
            "label": "Degrees",
            "description": "Degrees earned by the student"
          }
        ],
        "sd": "never"
      },
      {
        "path": ["degrees", null],
        "sd": "always"
      },
      {
        "path": ["degrees", null, "field_of_study"],
        "display": [
          {
            "locale": "de-DE",
            "label": "Studienfach"
          },
          {
            "locale": "en-US",
            "label": "Field of Study"
          }
        ],
        "sd": "never"
      },
      {
        "path": ["degrees", null, "date_awarded"],
        "display": [
          {
            "locale": "de-DE",
            "label": "Verleihungsdatum"
          },
          {
```

```
            "locale": "en-US",
            "label": "Date Awarded"
          }
        ],
        "sd": "always"
      }
    ]
  }
```

Figure 29: Example Type Metadata Document

Note that in this example, there are four definitions affecting the degrees claim:

1. The degrees array itself is marked as sd: never, meaning the element degrees cannot be selectively disclosed and will always exist in the disclosed SD-JWT. Changing this to sd: always would mean that the degrees array itself is selectively disclosable.
2. Each item in the degrees array (denoted by null in the path) is marked as sd: always, meaning each degree object will be selectively disclosed as a whole.
3. The field_of_study property of each degree object is marked as sd: never, meaning that if the respective degree object is disclosed, the field_of_study property will always be included and cannot be hidden.
4. The date_awarded property of each degree object is marked as sd: always, meaning it can be selectively disclosed.

Appendix C. Acknowledgements

We would like to thank Aaron Parecki, Alen Horvat, Andres Uribe, Andrii Deinega, Annabelle Kennedy, Babis Routis, Christian Bormann, Dan Moore, Denis Pinkas, George J Padayatti, Giuseppe De Marco, Hannes Tschofenig, Lukas J Han, Lukasz Jaromin, Leif Johansson, Michael B. Jones, Mike Prorock, Mirko Mollik, Nat Sakimura, Orie Steele, Paul Bastian, Pavel Zarecky, Stefan Charsley, Tim Cappalli, Timo Glastra, Torsten Lodderstedt, Tobias Looker, and Kristina Yasuda for their contributions (some of which substantial) to this draft and to the initial set of implementations.

Appendix D. Document History

-16

* shepherd review and resulting changes
* use key discovery and validation mechanism instead of Issuer Signature Mechanism

* moved Display Metadata and Claim Metadata to be subsections of SD-JWT VC Type Metadata

-15

* remove unnecessary Relationships to Other Documents section

-14

* State more explicitly that additional Issuer Signature Mechanisms can complement or override the defined mechanisms
* Add example of a vct value that is versioned (in the text)
* Use the three-word term Verifiable Digital Credential as consistently as possible throughout the document
* Update draft-ietf-oauth-selective-disclosure-jwt references to point to RFC 9901
* More consistent use of application/json content type and 200 status code when retrieving metadata
* Editorial improvements based on review feedback of -13
* Expand the example showing the vct claim
* Number and label the examples/figures
* Fix reference to Subresource Integrity document to point to W3C Recommendation version
* Move Privacy-Preserving Retrieval of Type Metadata section to Privacy Considerations where it belongs
* Move RFC 2397 to informative (only mentioned once and behind a SHOULD)
* Add line break between the two things in the JSON Web Token Claims Registration section
* Error responses both for JWT VC Issuer Metadata and Type Metadata retrieval now MUST use appropriate HTTP status codes
* Clarified that if an integrity property is present for a particular claim/property, the Consumer MUST verify the integrity of the retrieved document.
* Fixed the description of the claim metadata extension rules to correctly reflect the intended behavior.
* Added a note explaining the difference in overriding rules between claim metadata and display metadata when extending types.
* Merged section section 4 into section 3
* Replaced the term wallet with holder for consistency reasons

-13

* Updated svg_template to match pluralised appendix example

-12

* Change lang to locale. While lang is more accurate, locale is what has traditionally been used in OpenID Connect and later related specs.
* Remove JSON schema from Type Metadata
* Introduce optional mandatory property for claims
* Explicitly mention that Type Metadata can have additional stuff that has to be ignored if not understood
* Clarify that an SD-JWT VC doesn’t contain a KB-JWT but rather might have an associated one (which makes it a SD-JWT+KB and Brian is still not sure about the term or these words, but it’s where we’ve ended up)
* Remove the requirement to ignore unknown claims, as some applications may not want to follow this rule
* Fix cnf claim and JWK references and move them to normative
* List vct as one of the required values in type metadata and ensure that the use of the document integrity claims is clear
* Remove discussion of status and Status Provider from the Introduction
* Add a background_image property to the simple rendering aligned with the definition in OpenID4VCI
* Recommend to use sd=always or sd=never to avoid ambiguity and introduce rules for sd and mandatory when extending types
* Provide some guidance on versioning via the vct value
* Add security considerations for trust in type metadata
* Require data URIs for non-JSON types
* Require x5c to be in the protected header
* Clarify presentations of SD-JWT VC do not require KB
* Updated/expanded example for Type Metadata
* Be more consistent with style for lists of claims/parameters/properties
* Update PID example to make clear that it is not normative
* Clarification on processing of display metadata

-11

* Clarify extend support for claim metadata
* Add privacy concerns regarding the use of x5u parameter in JWKs and similar remote retrieval mechanisms
* Added a section on Credential Type Extension and Issuer Authorization.
* Fixed an inconsistency to the description of display attribute of claim metadata.
* add vct#integrity to the list of claims that cannot be selectively disclosed
* Drop explicit treatment of the glue type metadata document concept
* Editorial updates and fixes.

* State that when the status claim is present and using the status_list mechanism, the associated Status List Token has to be a JWT.
* vct datatype is now just a string

-10

* Rename 'Issuer-signed JWT Verification Key Validation' to 'Issuer Signature Mechanisms' and rework some text accordingly. Provide a web-based metadata resolution mechanism and an inline x509 mechanism. A DID-based mechanism is not explicitly provided herein but still possible via profile/extension. Be explicit that the employed Issuer Signature Mechanism has to be one that is permitted for the Issuer according to policy. Be more clear that one permitted Issuer Signature Mechanism is sufficient.
* Fix [...]#integrity claim values in examples (Subresource Integrity uses regular base64 encoding and some were wrong length)

-09

* Use SD-JWT KB in place of SD-JWT with Key Binding JWT
* Editorial changes
* Document reasons for not using JSON Pointer or JSON Path (Issue #267)
* Clarify that private claim names MAY be used
* Update PID Example
* Fix section numbering in a few SD-JWT references

-08

* Fix formatting issue introduced by the reintroduction of the DID paragraph in -07

-07

* Revert change from previous release that removed explicit mention of DIDs in the Issuer-signed JWT Verification Key Validation section
* Remove the requirement to insert a .well-known part for vct URLs
* fix section numbering in SD-JWT references to align with the latest -14 version

-06

* Update the anticipated media type registration request from application/vc+sd-jwt to application/dc+sd-jwt
* Tightened the exposition of the Issuer-signed JWT Verification Key Validation section

* Add the “Status” field for the well-known URI registration per IANA early review

-05

* Include display and claim type metadata
* Added example for type metadata
* Clarify, add context, or otherwise improved the examples

-04

* update reference to IETF Status List
* Include Type Metadata
* Include schema Type Metadata
* Editorial changes
* Updated terminology to clarify digital signatures are one way to secure VCs and presentations
* Rework key resolution/validation for x5c

-03

* Include disclosure of age_equal_or_over/18 in the PID example

-02

* Made specific rules for public verification key validation conditional
* Finetuned rules for obtaining public verification key
* Editorial changes
* added Brian Campbell as co-author
* Renamed JWT Issuer Metadata to JWT VC Issuer Metadata
* ’iat’ is now optional and allowed to be selectively disclosable
* Fix inconstancy in the .well-known path construction
* Added registration request to IANA for the well-known URI
* Fix some formatting and text in the media type and JWT claim registration requests
* Clarify the optionality of the cnf claim
* Added relationships to other documents
* Added PID example

-01

* Introduce rules for type identifiers (Collision-Resistant Name)
* Rename type to vct
* Removed duplicated and inconsistent requirements on KB-JWT
* Editorial changes
* Added issuer public verification key discovery section.

-00

* Upload as draft-ietf-oauth-sd-jwt-vc-00
* Aligned terminology and descriptions with latest version of SD-JWT

[[pre Working Group Adoption:]]

-00

* Initial Version
* Removed W3C VCDM transformation algorithm
* Various editorial changes based on feedback
* Adjusted terminology based on feedback
* Added non-selectively disclosable JWT VC
* Added a note that this is not W3C VCDM

Authors' Addresses

Oliver Terbu
MATTR
Email: oliver.terbu@mattr.global

Daniel Fett
Authlete Inc.
Email: mail@danielfett.de

Brian Campbell
Ping Identity
Email: bcampbell@pingidentity.com