

SD-JWT

IETF OAuth WG Draft

<https://datatracker.ietf.org/doc/draft-fett-oauth-selective-disclosure-jwt/>



Daniel Fett
yes.com → Authlete

Kristina Yasuda

Brian Campbell
Ping

‘Simple’ is a feature.

Design Principles

SD-JWT

Complexity	Selective disclosure, as simple as possible
Algorithms	Standard cryptography: JWS Signature + Hash function
Format	JWT & JSON
Security	Security-by-design Easy to understand & verify Hardware binding possible Cryptographic agility
Availability	Widely-available JWT libraries can be leveraged Already five independent implementations
Use Cases	Universal (beyond identity use cases)

SD-JWT in 5 Simple Steps

Step 1: Prepare User Data

```
{
  "iss": "https://example.com",
  "type": "IdentityCredential",
  "cnf": { "jwk": { "kty": "RSA", "n": "0vx....Kgw", "e": "AQAB" } },
  "credentialSubject": {
    "given_name": "Max",
    "family_name": "Mustermann",
    "email": "mustermann@example.com",
    "address": {
      "street_address": "Musterstr. 23",
      "locality": "Berlin",
      "country": "DE"
    }
  }
}
```

SD-JWT in 5 Simple Steps

Step 2: Create *Disclosures*

```
{
  "iss": "https://example.com",
  "type": "IdentityCredential",
  "cnf": { "jwk": { "kty": "RSA", "n": "0vx....Kgw", "e": "AQAB" } },
  "credentialSubject": {
    "given_name": "Max", ..... ["G00r26n0-iw50ZcAoOilFw", "given_name", "Max"]
    "family_name": "Mustermann", ..... ["cSlbR135i0NjhsouMxrjjg", "family_name", "Mustermann"]
    "email": "mustermann@example.com", ..... ["oHDt43Vwuhpo8mzaprgCcw", "email", "mustermann@example.com"]
    "address": {
      "street_address": "Musterstr. 23", ..... ["rGc0KtY6WmflywTTKEWIEQ", "street_address", "Musterstr. 23"]
      "locality": "Berlin", ..... ["pGQMqx-2tH2XwC_eQCFn4g", "locality", "Berlin"]
      "country": "DE" ..... ["TI15M8G5UIxPiWNZ-VLYBA", "country", "DE"]
    }
  }
}
```

↑ salt
 ↑ claim name
 ↑ claim value

SD-JWT in 5 Simple Steps

Step 3: Hash Disclosures & Replace Original Claims

```
{
  "iss": "https://example.com",
  "type": "IdentityCredential",
  "cnf": { "jwk": { "kty": "RSA", "n": "0vx....Kgw", "e": "AQAB" } },
  "credentialSubject": {
    "_sd": [ "EW1o0egqa5mGcbytT5S-kAubcEjYEUwRkXlu2vC5l20",
             "FEx-ITht41I8_cn0SS-hvoLneX_RGlJo_8o2xRNhfdk",
             "igg7H5fn2eBEMIEkE5Ckbn23QuwDJlTYoKRip08dYIc" ],
    "address": {
      "_sd": [ "gqB5kmAwryr88aHjaAeO-USX6JOMaojukKsheo3800c",
               "w8InvxsPXdkoowuVpyBMgl1b9_R2b6Xpa30Y0IjgQro",
               "v0nlytcjr872fP3Wa750z17c-6_MOVdIUNtwLKKxZw0" ]
    }
  }
}
```

```
← ["G00r26n0-iW50ZcAoOilFw", "given_name", "Max"]
← ["cSlbR135i0NjhsouMxrjjg", "family_name", "Mustermann"]
← ["oHDt43Vwuhpo8mzaprgCcw", "email", "mustermann@example.com"]

← ["rGc0KtY6WmflywTTKEWIEQ", "street_address", "Musterstr. 23"]
← ["pGQMqx-2tH2XwC_eQCFn4g", "locality", "Berlin"]
← ["TI15M8G5UIxPiWNZ-VLYBA", "country", "DE"]
```

SD-JWT in 5 Simple Steps

Step 4: Sign SD-JWT & Encode for Transport

```
"iss": "https://example.com",  
eyJhbGciOiAiUmlrNTYiLCIAia2lkIjogImNBRUlVcUowY21MekQxa3pHemhlaUJhZzBZ  
UkF6VmRsZnhOMjgwTmdlYUEifQ.eyJpc3MiOiAiaHR0cHM6Ly9leGFtcGx1LmNvbS9pc3N1ZXIiLCIAiY25mIjokeyyJqd2siOiB7imt0eSi6ICJSU0EiLAib1i6ICIwdng3YWdvZWJHY1FTdS4uLi4tY3NGQ3VyLWtFZ1U4YXdhcEp6S25xREtndyIsICJJIjogIkFRQUIifX0Scix0eXB1IjogIkklZw50axRSZ3JlZGVudGlibHBCisICJjcVkwZ50awFSz3ViamVjdC16IHsiX3NKIjogIiwJVfVzFvMGVncWE1bUdjYn10VDVTlWtBDwjJrJRpZRVRV3UmtYbHUydKm1bDIwiIiwgIkZFec1JEh0NDFOJF9jbJBtUY1odm9MbmvYX1JHBepvXzhvMnhSTmhMZGsilCAiUXhkVi0yvJfIOG1jbHRSNNzWQZrTM3JLVTVhTKg5d2RkeJJVZG1Sb0kxRSISiCJhdFVuMVRzd1JBbdDRHUTdQZUVOWGFNdZJmNHVJVG1KclgOODV3TTht2NjdfFiwgImZUTCxczdmtRUX3TDFYTnVZSZshIN3pCS0NIIdV91awY2MFNsRzfweVhJVVEiLAiaawdnN0g1Zm4yZUJUFTU1FaOU1Q2tibTizUXVP3REpsVF1vS1JpcDA4ZFllJyIsICJC0cFVObdCwaHBVX3hucnzAAtblHaEdvUlIxam10MXpZZ3ZNULZMEF4N0tjl10sICJhZGRZYXNX1jogeyJfc2QiOiBbImdxQjVrbUF3exJ5ODhhSGphqWVPLVVTWDZKT01hb2p1a0tzaGvvMzhPMGMilCAidk9ubF1OY2pyODcyZlAzV2E3NU96bDdjLTZftU9WZE1VtnR3TEtleFp3MCIsICJ30EludhzhzUfhkS29vd3VWhClCTWdsMWI5X1IyYjZYcGEZT1lPSWpnUXJvIl19fSwgIm1hdC16IDE1MTYyMzkwmjIjsICJleHAiOiAxNTE2MjQ3MDIyLCAic2RfZGlInZXNOX2R1cm12YXRpb25fYWNxInJogInoYS0yNYTYfQ.1UEHTLLUXOT51jh3gg-3C-ZidWzsB9Un-VxmMVdtQtBLlhWdBtJ6HJtt15p43YCXITzdpiZxtDi6fr07TP0dy_Umg3Q5_FxFj4WHnsVuVzuASU8cFlGPi6xgH9D3w1G2hqepBS8DyQ5ba_p5kN_tKJVoP1xlwhcQuJRJ8kkEKQsRia4FhrBlld18f41wgu_ipPqh1Ix4BV17GJC1ZNx94nwPT7JUFKi6Y6JkahLf3S6gBOMxtmLaEY0qkuz8VeOZNF1_CDog55kVTKArorfol6D6TEji__-w6YyU0PNIRJXJOwrYfoyhN18LKAP38QYmpdr7z_rsvHPqHzFAPTmevnHDg
```

```

← ["G00r26n0-iW50ZcAo0ilFw", "given_name", "Max"]
← ["cSlBr135i0NjhsouMxrjjg", "family_name", "Mustermann"]
← ["oHDt43Vwuhpo8mzaprgCcw", "email", "mustermann@example.com"]

← ["rGc0KtY6WmflywTTKEWIEQ", "street_address", "Musterstr. 23"]
← ["pGQMqX-2tH2XwC_eQCFn4g", "locality", "Berlin"]
← ["TI15M8G5UIxPiWNZ-VLYBA", "country", "DE"]

```


SD-JWT in 5 Simple Steps

Step 5: Base64url-encode Disclosures for Transport

```
{
  "iss": "https://example.com",
  eyJhbGciOiAiA1MyNTYiLCJkaXIjOiJjogImNBRU1VcUowY21MekQxa3pHemhlaUJhZzBZ
  UKF6VmRsZnhOMjgwTmdIYUEifQ.eyJpc3MiOiAiaHR0cHM6Ly9leGFtcGxlLmNvbS9pc
  3N1ZXIiLCJkaXIjOiJjogeyJqd2s0iB7Imt0eSI6ICJSU0EiLCJkaXIjOiJjogeyJqd2s0
  WJHY1FTdS4uLi4tY3NGQ3VyLWtFZ1U4YXdhcEp6S25xREtndyIsICJlIjogIkFRQUIif
  X0sICJ0eXB1IjogIk1kZW50aXR5Q3JlZGVudG1hbCI6ICJjcmVhZzU0aW50aW50aW50aW50
  C16IHsiX3NkIjogWyJFVzFvMGVhcnE1bUdjYn10VDVTLWtBdWJjRwpZRVV3UmtYbHUyd
  km1bDIiIiwgIkZFeC1JVEh0NDFJOF9jbJBTUy1odm9MbWVYX1JHbEpvXzhvMnhSTmhmZ
  GsiLCJkaXIjOiJjogeyJqd2s0iB7Imt0eSI6ICJSU0EiLCJkaXIjOiJjogeyJqd2s0
  CJhdFVUwMVRZd1JBbDRHUTdQZUVOWGFNzJmNHNHJVJG1Kclg0ODV3TTh2NjdfIiwgImZUT
  XczdmtrRUx3TDYfYnVZSzhIN3pCS0NidV91aWY2MFNsRzFweVhJVVEiLCJkaXIjOiJjogeyJfc
  m4yZUJFTU1Fa0U1Q2tibTIZUXV3REpsVF1vS1JpcDA4ZFlJYyIsICJ0cFV0bDcwaHBVX
  3hucnZaaTBhaEdvUlIxaM10MxpZZ3Z2NU1ZMEF4N0tjI10sICJhZGRyZXNzIjogeyJfc
  2Q0iOiBbImdxQjVrbUf3eXJ5ODhhSGphQWVPLVVTWDZKT01hb2p1a0tzaG5vMzhPMGMiL
  CAidk9ubFl0Y2pyODcyZ1AzV2E3NU96bDdjLTZftU9WZElVTnR3TetLeFp3MCIsICJ30
  EludnhzUFHkS29vd3VwchLCTWdsMWI5X1IyYjZyCGEzT1lPSWpnUxJvI19fSwgImld
  CI6IDE1MTYyMzkwMjIsICJleHAiOiA1NTE2MjQ3MDIyLCJkaXIjOiJjogeyJqd2s0iB7Imt0eSI6ICJSU0EiLCJkaXIjOiJjogeyJqd2s0
  XRpb25fYXNjIjogInNoYS0yNTYifQ.1UHEPtLUXOT51jH3gg-3C-ZidWzsB9Un-VxmM
  VdQtTbLLhwdTB6HJtt15p43yCXtZdpiZxtDI6fr07Tp0Dy_Umg3Q5_FxFj4WnhsVuVzu
  ASU8cFLGPi6xgH9D3w1G2hqepBS8DyQ5bA_p5kN_tKJVoP1xwhcQujRJ8kkEKQsRia4F
  hrBld18f41wgu_ipPqh1Ix4BVI7GJC1ZNx94nWPT7JUFkI6Y6JkahLf3S6gB0MxtmLAe
  Y0qkuz8Ve0ZNf1_CDog55kVTkArorfoL6D6TEjI_-w6YyU0PnIRJXJ0wrYfoyhN18LK
  AP38QYmpdR7z_rsvHqHzFAPTmevnhDg
}
```

```
- ["G00r26n0-iw50ZcAoOilFw", "given_name", "Max"]
```

```
~WyJHTzByMjZuTy1pVzUwWmNBb09pbEZ3IiwgImdpdmVuX25hbWUiLCJkaXIjOiJjogeyJqd2s0iB7Imt0eSI6ICJSU0EiLCJkaXIjOiJjogeyJqd2s0
~WyJjU2xiUjEzNWkwImpoc291TXhyampnIiwgImZhbWlseV9uYW1lIiwgIk11c3Rlcm1hbm4iXQ
```

```
~WyJvSER0NDNwd3VocG84bXphcHJnQ2N3IiwgImVtYWlsIiwgIm11c3Rlcm1hbm5AZXhhbXBsZS5jb20iXQ
```

```
~WyJyR2MwS3RZNldtZmx5d1RUS0VXSUVRIiwgInN0cmVldF9hZGRyZXNzIiwgIk11c3RlcnN0ci4gMjMiXQ
```

```
~WyJwR1FNUXgtMnRIMlMh3Q19lUUNGbJnRnIiwgImxvY2FsaXR5IiwgIk11c3Rlcm1hbm5AZXhhbXBsZS5jb20iXQ
```

```
~WyJUSTE1TThHNvVJeFBpV05aLVZMwUJBIiwgImNvdW50cnkiLCJkaXIjOiJjogeyJqd2s0iB7Imt0eSI6ICJSU0EiLCJkaXIjOiJjogeyJqd2s0
```

→ Done!

Issuer

Issuance



SD-JWT

plain-text claims
+ hashed Disclosures

```
{
  "iss": "https://example.com",
  "exp": 1612214400,
  "nbf": 1612214400,
  "sub": "1234567890",
  "aud": "1234567890",
  "iat": 1612214400,
  "jti": "1234567890",
  "claims": {
    "name": "John Doe",
    "age": 30,
    "email": "john.doe@example.com"
  },
  "disclosures": [
    {
      "salt": "1234567890",
      "claim": "name",
      "value": "John Doe"
    },
    {
      "salt": "1234567890",
      "claim": "age",
      "value": "30"
    },
    {
      "salt": "1234567890",
      "claim": "email",
      "value": "john.doe@example.com"
    }
  ]
}
```

✓ signed
by Issuer

Disclosures

salt + claim name + claim value

```
WyJrSEhwOTUtdEFadDhtOUU0Smw0W6JRIiwgImdpdmVUX25hbWU1LCAiSm9obiJld
WyJQak1xcEdxbDRlQjRRcm9EaHFRdzB3IiwgImZhbm1seV9uYW11IiwgIkRvZS5Jd
WyJ4bm1QNEpadeEKSUgtTGtFRHQtby1BIiwgImN0cmVldF9hZGRyZXNzIiwgIkRvZXN0cmVldCAxI10
WyJldGZzeHhUbTJtdzBZTFVjS1pVOHRBIiwgImxvY2FsaXR5IiwgIkFueXRvd241XQ
```

End-User
(Holder)

Presentation



Verifier

Issuer

Issuance

End-User
(Holder)

Presentation

Verifier

SD-JWT
plain-text claims
+ hashed Disclosures

```
{  
  "iss": "https://example.com",  
  "exp": 1717141800, "iat": 1717141800, "sub": "1",  
  "aud": "https://example.com",  
  "sd": {  
    "disclosures": {  
      "disclosure1": {  
        "salt": "1",  
        "claim_name": "name",  
        "claim_value": "value",  
        "hash": "sha256(1|name|value)"  
      },  
      "disclosure2": {  
        "salt": "2",  
        "claim_name": "age",  
        "claim_value": "42",  
        "hash": "sha256(2|age|42)"  
      }  
    }  
  }  
}
```

✓ signed
by Issuer

Disclosures
salt + claim name + claim value

```
WyJrSEhwOTUtdEFadDhtOUU0Smw0WGRlIiwgImdpdmVux25hbWU1LCAiSm9obiJld  
WyJQak1xcEdXbDRlQjRRcm9EaHFRdzB3IiwgImZhbnw1seV9uYw11IiwgIkRvZS5ld  
WyJ4bm1QNEpadeE5XSUgtTgtFRHQtby1BIiwgImN0cmVldF9hZGRyZXNzIiwgIkRvZS5ld  
WyJldGZzeHhUbTJtdzBZTFVjS1pVOHRBIiwgImxvY2FsaXR5IiwgIkFueXRvd241XQ
```

SD-JWT
plain-text claims
+ hashed Disclosures

```
{  
  "iss": "https://example.com",  
  "exp": 1717141800, "iat": 1717141800, "sub": "1",  
  "aud": "https://example.com",  
  "sd": {  
    "disclosures": {  
      "disclosure1": {  
        "salt": "1",  
        "claim_name": "name",  
        "claim_value": "value",  
        "hash": "sha256(1|name|value)"  
      },  
      "disclosure2": {  
        "salt": "2",  
        "claim_name": "age",  
        "claim_value": "42",  
        "hash": "sha256(2|age|42)"  
      }  
    }  
  }  
}
```

✓ signed
by Issuer

Selected Disclosures
salt + claim name + claim value

```
WyJrSEhwOTUtdEFadDhtOUU0Smw0WGRlIiwgImdpdmVux25hbWU1LCAiSm9obiJld  
WyJQak1xcEdXbDRlQjRRcm9EaHFRdzB3IiwgImZhbnw1seV9uYw11IiwgIkRvZS5ld
```



Verification

- Verify SD-JWT signature
- Hash over disclosed Disclosures
- Find hash digests in SD-JWT
- Replace disclosed claims in SD-JWT
- Check holder binding, if required.



Verification requires hash check!

Done!

(Selected) Changes since -02

- **Nested disclosures** allow for arbitrary granularity for selectively disclosable claims
- **W3C VC-Data-Model** credential with JSON-LD
- **Improved various parts of the spec:**
 - Concepts section,
 - Security & Privacy Considerations,
 - Discussion on Holder Binding,
 - Context in Introduction,
 - Discussion on Canonicalization,
 - etc.

Discussion Points for -04

- Media Types
- Selective Disclosure of elements in the arrays
- SD-JWT with JWS using JSON serialization

Media Types

- As an identifier to signal it is an SD-JWT, since processing rules are different from JWS/JWT
- To be used in
 - HTTP requests and responses
 - typ in the Header of an SD-JWT
 - Payload of an SD-JWT can be any JSON.
W3C VC WG expected to define a media type for a payload (cty)
 - cty in the Header of JWE
- Proposals/ideas:
 - For the **HTTP requests and responses**, define sd-jwt-issuance and sd-jwt-presentation?
 - For the **typ JOSE Header**, define sd-jwt media type?
 - For **general purpose**, define +sd-jwt media type structured suffix?

GH Issues [#236](#), [#74](#); PR [#229](#)

Selective Disclosure for Individual Array Elements

Does the WG believe this feature is needed?

Some use cases for SD in arrays:

- Multi-value claims, like “nationalities”
- Selectively releasing a subset of the “evidence” documents supporting identity claims
- Using one holder binding method without releasing data for the others (e.g., release biometrics, but not public key)

[GH Issue #194](#)

Selective Disclosure of elements in the arrays?

Two approaches how this could be done, if the WG agreed the feature is needed:

Approach 1:

```
"nationalities": [  
  "_sd",  
  "7pHe1uQ5uSClgAxXdG0E6dKnBgXcxEO1zvoQ09E5Lr4",  
  "9-VdSnvRTZND0-4Bxcp3X-V9VtLOCRUkR6oLWZQ181I"  
]
```

New leading element indicates array of SD digests.

Approach 2:

```
"nationalities": [  
  "_sd:7pHe1uQ5uSClgAxXdG0E6dKnBgXcxEO1zvoQ09E5Lr4",  
  "_sd:9-VdSnvRTZND0-4Bxcp3X-V9VtLOCRUkR6oLWZQ181I"  
]
```

String with defined prefix indicates SD hash.

SD-JWT with JWS using JSON Serialization?

Some use cases require JWS JSON serialization.

Example: ETSI JAdES signatures.

Proposal: Introduce JWS JSON serialization format for SD-JWT as optional feature (won't be a breaking change)

[GH Issue #198](#)

SD-JWT with JWS using JSON serialization?

```
{  
  "payload": "eyJpc3MiOiAiaHR0cHM6L...ZONGpUOUYySFpRIn19fQ",  
  "protected": "eyJhbGciOiAiRVMyNTYifQ",  
  "header": {  
    "kid": "e9bc097a-ce51-4036-9562-d2ade882db0d"  
  },  
  "signature": "mcndQ15m-4FbIzyfB...U2ZX7g",  
  "disclosures": [  
    "WyJkcVR2WE14UzBHYTNEb2FHbmU5eDBRIiwgInN1YiIsICJqb2huX2RvZV80MiJd",  
    "WyIzanFjYjY3ejl3a3MwOHp3aUs3RXlRIiwgImdpdmVuX25hbWUiLCAiSm9obiJd",  
    "WyJxUVdtakpsMXMxUjRscWhFTkxScnJ3IiwgImZhbWlseV9uYW1lIiwgIkRvZSJD"  
  ]  
}
```

Payload as in SD-JWT

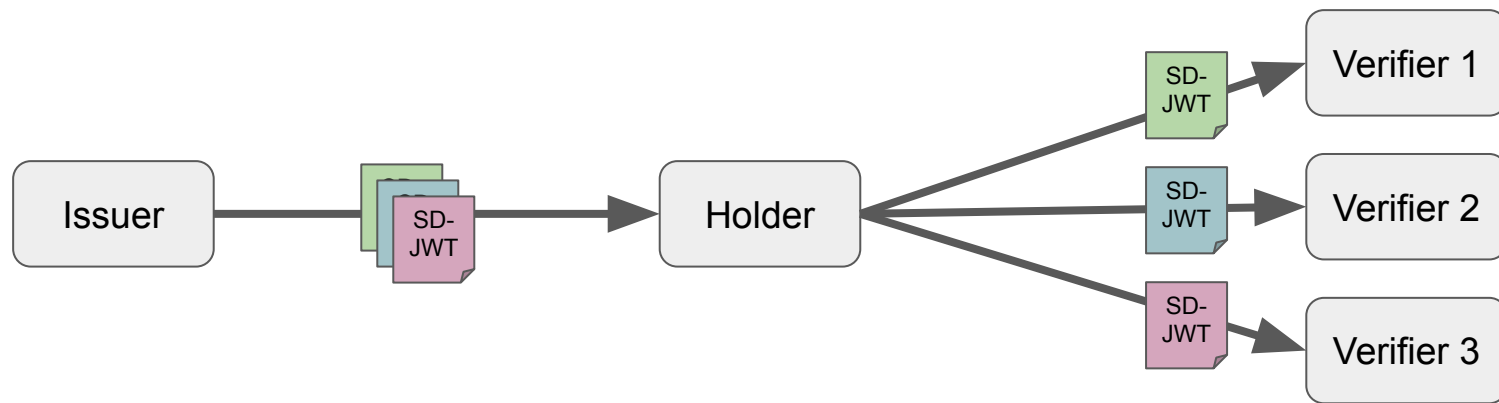
Disclosures

Unlinkability and Hash-based Selective Disclosure?

Inherent problem: Colluding verifiers can identify same user.

Solution: SD-JWTs are cheap!

- Issuer issues many SD-JWTs to Holder with same claims
- Holder uses a fresh SD-JWT per Verifier



Cryptographic Agility

SD-JWT does not prescribe any algorithms.

Signature Algorithm: All JOSE algorithms

Post-Quantum: see [draft-ietf-cose-post-quantum-signatures]

Hash Algorithm: Default SHA-256, algorithm identifier in SD-JWT

Compatibility

- Can be used with any JSON-based data format
 - W3C-VC Data Model
 - OpenID Connect for Identity Assurance (OIDC4IA)
- Flexibility regarding holder binding
 - External signature
 - Key distribution
- Makes no assumptions on the transport protocol
 - E.g., OIDC4VC

Available, Testable, Auditable

All examples in specification generated via [reference implementation](#):
[oauthstuff/draft-selective-disclosure-jwt](#) (Python)

tooling might be separated into
another GH repo in the future

```
### Produce SD-JWT
sdjwt = SDJWT(
    user_claims,
    issuer,
    ISSUER_KEY,
    HOLDER_KEY,
    iat,
    exp,
)
```

Independent open-source implementations:

- Kotlin: [IDunion/SD-JWT-Kotlin](#)
- Rust: [kushaldas/sd_jwt](#)
- TypeScript: [christianpaquin/sd-jwt](#)
- TypeScript: [chike0905/sd-jwt-ts](#)
- Typescript: [OR13/vc-sd-jwt](#) **NEW**
- Java: [authlete/sd-jwt](#) **NEW**