# SPICE SD-CWT

draft-ietf-spice-sd-cwt-01

M. Prorock , O. Steele , H. Birkholz , R. Mahy
IETF 121 Dublin
05 November 2024

# Agenda

- Intro to Selective Disclosure (Mike)
- CBOR / CWT Syntax Options (Rohan)
- Tagging Redactions in CBOR (Henk)
- Best place for disclosures (Orie)
- Next Steps (Orie)

# Intro to Selective Disclosure

# Intro to Selective Disclosure

- Solution statement
  - Allow a **Holder** to get a CBOR Web Token from an **Issuer** and then safely blind (“redact”) or reveal (“disclose”) some of its claims to a **Verifier**
- Holder can disclose *all, some, or none* of the disclosable claims.

- Every SD-CWT is a valid CWT, but not vice versa
- How Issuer decides which claims to selectively disclose is out-of-scope. Issuer does not have to make claims disclosable.
- Includes Holder’s public key or reference to it (`cnf`)

# What can be disclosed?

**No:** Any CWT claims that are part of the validation of the CWT (iss, aud, exp, nbf, iat, cti, cnf, nonce), **except** the subject.

**Yes:** The subject; Any other claims at *any level of hierarchy*; Any array items of any CWT claims (ex: `swversion` below).

```
{
   / sub / 1: "https://device.holder.example",
   / cnf / 8 : {
      / ckt / 5 : h'496bd8a…3bda88fadd1669da253ec'
   },
   …
   / name /  170 : "Alice Smith",
   / age_at_least_18 /  500 : true,
   / age_at_least_21 /  501 : false,
   / swversion / 271 : [
      "3.5.5",
      "4.1.7"
   ],
   / address /  187 : {
      "country"     : "us",          / United States /
      "region"      : "ca",          / California /
      "locality"    : "San Francisco",
      "postal_code" : "94188"
}
```

# Forming the disclosures — claim keys+values

For each CWT claim key, generate a random salt of at least 16-bytes.

Make a CBOR array with the salt and the selectively disclosed info

```
<<[
  /salt/ h'8d5c15fa86265d8ff77a0e92720ca837',
  /claim/  500,  / age_at_least_18 /
  /value/  true
]>>
```

```
56 83                                  # bytes(22) array(3)
   50 8D5C15FA86265D8FF77A0E92720CA837    # bytes(16)
   19 01F4                             # unsigned(500)
   F5                                  # true
```

# Forming the disclosures — Array items

For each  array item, generate a random salt of at least 16-bytes.
Make a CBOR array with the salt and the selectively disclosed info

```
<<[
  /salt/ h'86c84b9c3614ba27073c7e5a475a2a13',
  /value/  "4.1.7"
]>>
```

```
   58 18  82                           # bytes(24) array(2)
      50 86C84B9C3614BA27073C7E5A475A2A13    # bytes(16)
      65 342E312E37                    # text(5) "4.1.7"
```

# Hashing the Disclosures

The **hash** of each disclosure becomes the **redacted form** of that claim, array element (or decoy).
These hashes are placed in the *payload* of the CWT, which is signed by the issuer. (Example syntax)

```
/ cose-sign1 / 18([
  / protected   / <<{
    / key / 4 : h'496b…53ec', # public key thumbprint
    / algorithm / 1 : -7,     # ES256
  }>>,
  / unprotected / {},
  / payload     /<<{
    / iss / 1 : "https://issuer.example",
    / sub / 2 : "https://subject.example",
    / iat / 6: 1729706150,
    / redacted_map_keys / TBD : [
      / hash of disclosure for age_at_least_18 / h'db6e…76e2',
      / hash of disclosure for license_number  / h'c234…3b04'
    ]
  }>>,
  / signature   / h'2544f2ed...5840893b'
])
```



IETF

# Disclosing to a Verifier

Holder presents to a Verifier:

- the Issuer CWT
- any disclosures
- the Holder Key Binding Token (KBT) - *REQUIRED*

Can include an *audience* specific to the Verifier, and/or *nonce* scoped to the Verifier

# Key Binding

Key Binding prevents copy and paste or replay attacks on selective disclosure.

The Holder proves possession of their private key and choice of disclosures.

```
/ sd_kbt    / 18 : << 18([
  / protected / << {
        / alg / 1 : -7 / ES256 /,
        / typ / 16 : "application/kb+cwt"
  } >>,
  / unprotected / {},
  / payload / << {
        / cnonce / 39   : h'e0a156...bb3f',
        / aud   / 3 : "https://verifier.example",
        / iat   / 6 : 1783000000,
        / sd_alg  / 12 : -16,  /SHA-256/
        / sd_hash / 11 : h'c341bb...4a5f3f'  /hash of sd_claims   /
  } >>,
  / signature / h'1237af2e...678945'
]) >>
```

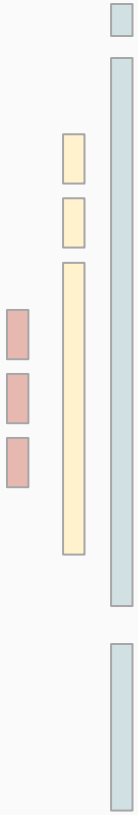


IETF

# Nested disclosures

## Disclosures can refer to nested data

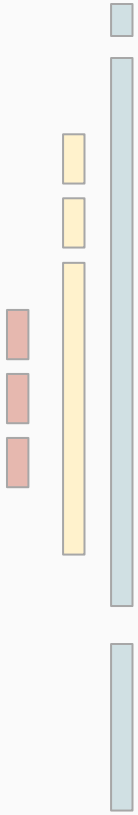
```
{
  / name / 170: "Alice Smyth",
  / address / 187: {
    / region   / 2: "jp",
    / locality / 3: "Tokyo",
    / physical / 8: {
      / block / 1: "2 Chome-5-2",
      / bldg /  2: "Mitsubishi Building"
      / floor / 3: "B1F"
    }
  },
  / nationalities / 197: [
    "jp", "ie"
  ]
}
```



# Nested disclosures

## Disclosures can refer to nested data

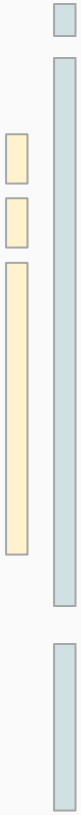
```
{
  / name / 170: "Alice Smyth",
  / address / 187: {
    / region   / 2: "jp",
    / locality / 3: "Tokyo",
    / physical / 8: { TBD : [
      h'f41a...8246',
      h'1562...228d',
      h'277c...839f' ]
    }
  },
  / nationalities / 197: [
    "jp", "ie"
  ]
}
```



# Nested disclosures

## Disclosures can refer to nested data

```
{
  / name / 170: "Alice Smyth",
  / address / 187: { TBD : [
    h'ce5a...45fe',
    h'4d5a...cf14',
    h'08fc...a01b'
    ]
  },
  / nationalities / 197: [
    69(h'7ef9...2388'), 69(h'2782...65af')
  ]
}
```



# Nested disclosures

## Disclosures can refer to nested data

```
{ TBD : [
  h'6194...d892',
  h'56b8...0389',
  h'3ebc...c52e' ]
}
```

# Forming Decoy disclosures

For each decoy, generate a random salt of at least 16-bytes.

Make a CBOR array with just the salt

Can be used to hide the information structure

```
<<[
  /salt/ h'613b281240e45f5d64b204555f91ba42'
]>>
```

```
   52 81                                  # bytes(18) array(1)
      50 613B281240E45F5D64B204555F91BA42    # bytes(16)
```

# SD-CWT Syntax Options

# CBOR / CWT SD Syntax

```
/ cose-sign1 / 18([
  / protected   / <<{
    / key / 4 : h'496b…53ec', # public key thumbprint
    / algorithm / 1 : -7,     # ES256
  }>>,
  / unprotected / {},
  / payload     /<<{
    / iss / 1 : "https://issuer.example",
    / sub / 2 : "https://subject.example",
    / iat / 6: 1729706150,
    / redacted_map_keys /  TBD : [
      / age_at_least_18 / h'c234…3b04',
      / license_number  / h'd234…3b04'
    ]
  }>>,
  / signature   / h'2544f2ed...5840893b'
])
```

# CBOR / CWT SD Syntax

```
Holder Presented Disclosure:

<<[
    /salt/   h'8d5c…a837',
    /claim/  500,  / age_at_least_18 /
    /value/  true
]>>

Issuer Signed Redaction:

h'c234…3b04'
```

# Tagging Redactions

# Tagging Redactions

- No Breaking Changes for CWT Processors
  - Map keys MUST be ***integers*** or ***strings***.
- Redacted payload size should be minimized
- Redaction should not leak information, or enable an issuer to hint at redacted values by varying their signed encoding.

# Tagging Key Redactions

Option 1

Map integers that are rare enough to not collide:

```
{
  / iss / 1 : “https://issuer.example”,
  / sub / 2 : “https://subject.example”,
  / iat / 6: 1729706150,
  / redacted_map_keys / -65536 : [
    / age_at_least_18 / h'c234…3b04',
    / license_number  / h'd234…3b04'
  ]
}
```

# Tagging Key Redactions

Option 2

Map integers as content format tags.

```
{
  / iss / 1 : "https://issuer.example",
  / sub / 2 : "https://subject.example",
  / iat / 6: 1729706150,
  / redacted_map_keys /  1668546817 : [
    / age_at_least_18 / h'c234…3b04',
    / license_number  / h'd234…3b04'
  ]
}

/ where 1668546817 is #6.LARGE_TBD(false)/
```

# Tagging Key Redactions

Option 3

Map tagged values. Not compatible with regular CWT. Requires duplicate keys

```
{
  / iss / 1 : “https://issuer.example”,
  / sub / 2 : “https://subject.example”,
  / iat / 6: 1729706150,
  / redacted age_at_least_18 /
  #TBD(false) : / h'c234…3b04',
  / redacted license_number /
  #TBD(false) : / h'd234…3b04'
}
```

# Tagging Redactions

Option 4

Map integers that are rare enough to not collide. Need a large range. May leak structure

```
{
  / iss / 1 : “https://issuer.example”,
  / sub / 2 : “https://subject.example”,
  / iat / 6: 1729706150,
  / redacted age_at_least_18 /
  76543 : / h'c234…3b04',
  / redacted license_number /
  76544 : / h'd234…3b04'
}
```

# Tagging Redactions

Option 5

Use duplicate keys using a large key. Use the existence of a tag on the value to confirm redaction. **Preprocess** these (potentially duplicate keys) to either eliminate or replace with a disclosed value. Not compatible with generic CWT libraries without preprocessing.

```
{
  / iss / 1 : "https://issuer.example",
  / sub / 2 : "https://subject.example",
  / iat / 6: 1729706150,
  / redacted age_at_least_18 /
  -65536 : / 68(h'c234…3b04'),
  / redacted license_number /
  -65536 : / 68(h'd234…3b04'),
}
```

# Best Place for Disclosures

# Where are disclosures presented today?

The unprotected header of the Issuer CWT

The Key Binding Token is also placed in the unprotected header of the Issuer CWT.

# Alternative: Disclosures in KBT

If disclosures were in the **protected header** or **payload** of the KBT they would be covered by the KBT signature.

Pros:

- Allows client to send more disclosures
  (ex: sign up/order after browsing)
  without resending the whole Issuer CWT
- Eliminates the need for the sd_hash claim in the KBT

Cons:

- Diverges more from SD-JWT (where KBT is optional)

# Next Steps

# Next Steps

- Decide how to represent redacted map keys
- Decide how to represent redacted array elements
- Decide where disclosures are presented
- Update prototypes, generate new examples