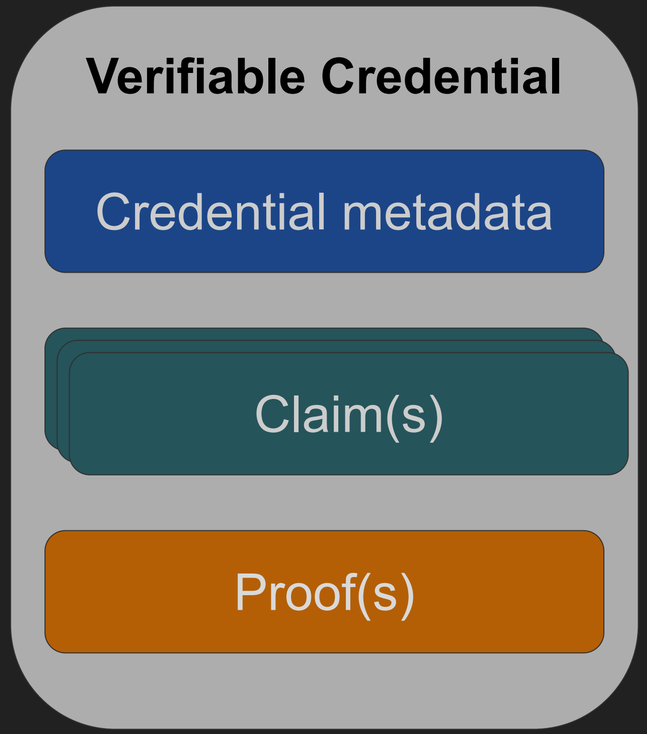
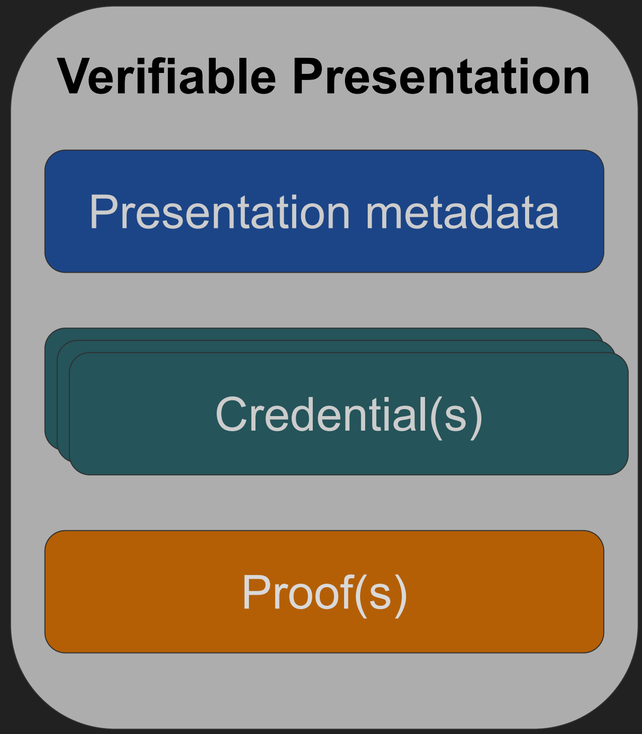
What’s new in W3C Verifiable Credentials?

Pierre-Antoine Champin (W3C / Inria)
Ivan Herman (W3C)

Verifiable Credentials Workshop in Japan - Tokyo 6 October 2025

The basic terminology





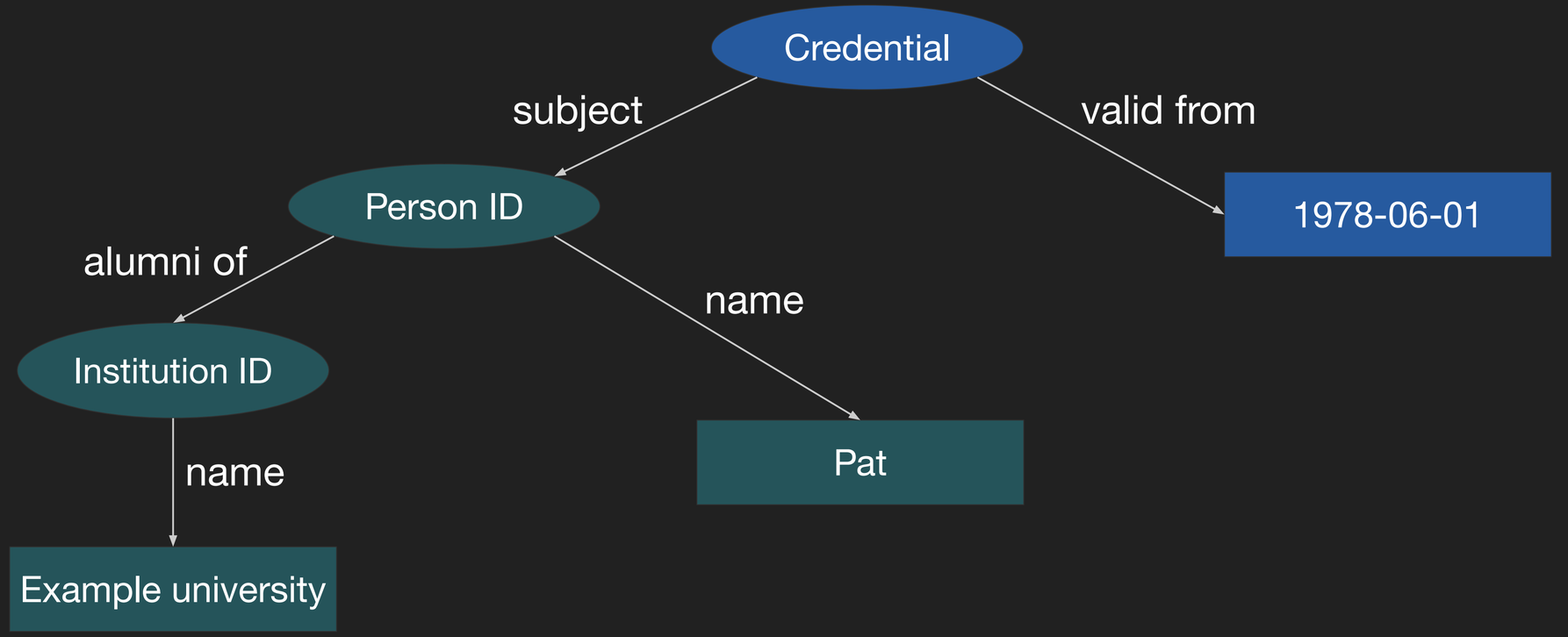
Verifiable presentations are essential!

- The holder may present:
  - a modified credential (e.g., for selective disclosure)
  - a combination of several credentials on the same subject (e.g. diploma & work permit)
  - a series of credentials for different subjects (e.g., travel documents for a family)

- Proofs on the credentials and on the presentation are different:
  - the credential proofs are provided by, or are derived from, the issuer
  - the presentation proofs are provided by the holder
  - both are required for a presentation to ensure data integrity

The VC model: collection of claims

- Is a simple model of individual claims (a.k.a. statements):
  - `subject → predicate → object`
- By combining such statements, we get a graph
- In VC, the graph is serialized in JSON



Advantages of a graph based model

- It is conceptually easy to *combine* graphs: just add new connections binding them together
  - e.g., combining standard terms with application specific ones

- Thinking in terms of atomic claims is needed for **selective disclosure** schemes, and this model facilitates this
  - e.q. buying age-restricted products without discussing exact date of birth, name, address…

In VC, the graph is serialized in JSON

- More exactly, a dialect of JSON called JSON-LD, specialized in representing of such graphs (Linked Data)
  - the standard processing steps in the recommendations are expressed in terms of JSON
  - can be implemented with standard JSON tools

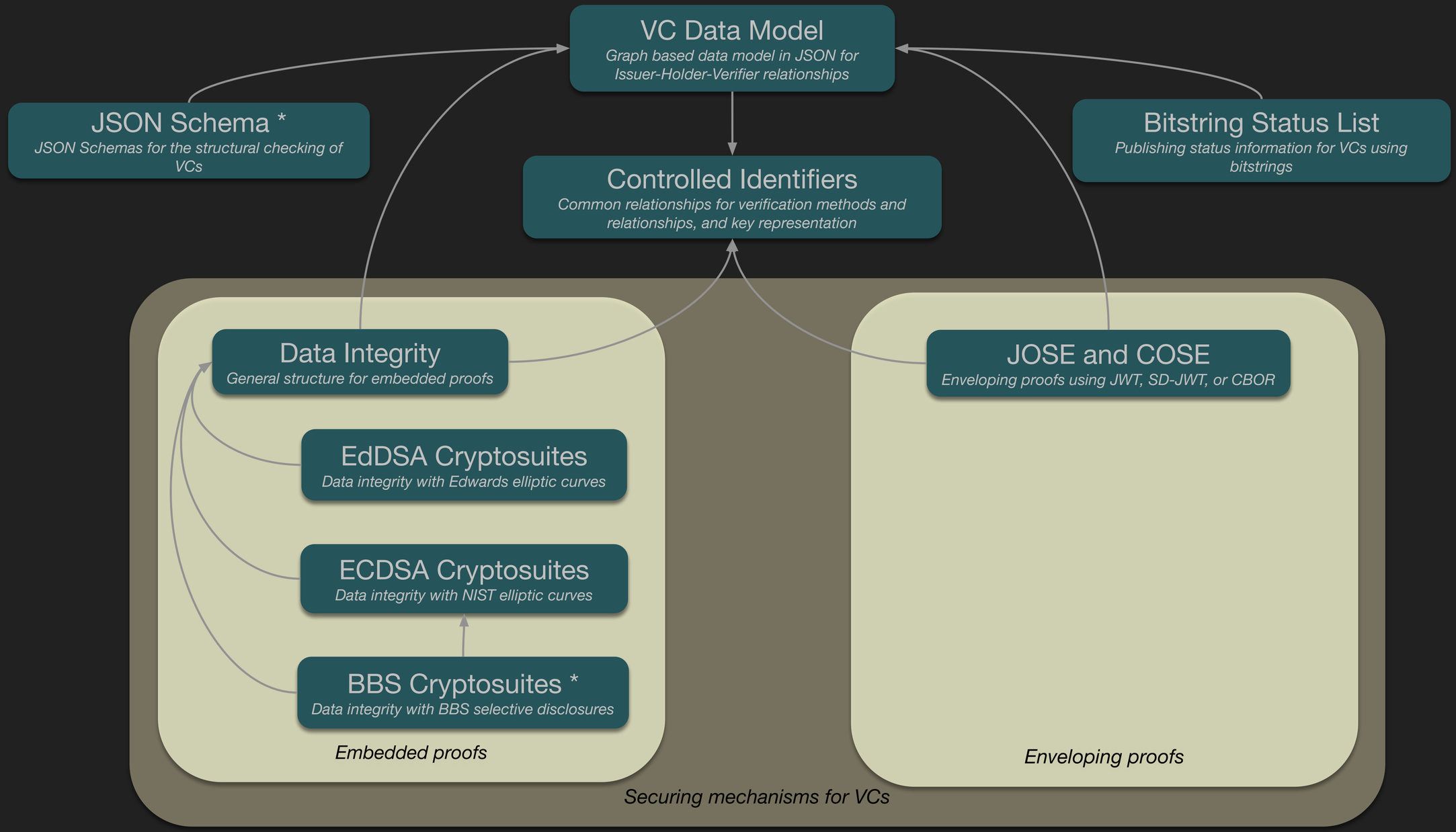
- There is work going on to define specialized CBOR conversions to ensure efficient storage and transmission

Advantages of a Linked Data graph based model

- Identifiers and attributes are expressed as URLs
  - can be HTTPS, UUID, DID, DOI…
  - makes it possible to reuse existing vocabularies / ontologies
    - developed at W3C (e.g. DPV), ETSI (e.g. SAREF), etc.
  - and to combine / integrate vocabularies developed independently
    - very important when combining graphs (e.g. university degree & work permit)
    - or when reusing VCs outside their original context (e.g., **cross-border**)

- Interoperable with other initiatives
  - Dataspaces (IDSA, Gaia-X)
  - Digital Product Passport (GS1, UN Transparency Protocol)
  - Industry 4.0 (RAMI 4.0, WoT, SAREF)

W3C VC Specifications



W3C VC Specifications

- Published as W3C Recommendations on 15 May 2025
  - Except for
    - Verifiable Credentials JSON Schema Specification
      (lack of implementation feedback so far)
    - Data Integrity BBS Cryptosuites v1.0
      (pending the official publication of BBS by IETF)
- Overview document:
  https://www.w3.org/TR/vc-overview/

The Digital Credential API

- https://www.w3.org/TR/digital-credentials/
- by the Federated Identity Working Group (still WIP)

- “This document specifies **an API to enable user agents to mediate presentation and issuance of digital credentials** such as a driver's license, government-issued identification card, and/or other types of digital credential.”

- “The API design is agnostic to both credential presentation exchange protocols, credential issuance protocols and credential formats.

Appendix: some more details with an example

An example of a basic credential (without proof)

```
{
 "@context": [
  "https://www.w3.org/ns/credentials/v2",
  "https://www.example.org/vocabs/alumni"
 ],
 "id": "https://uni.example/Credential12",
 "type": ["VerifiableCredential", "ExampleAlumniCredential"],
 "issuer": "did:example:2g55q91",
 "validFrom": "2010-01-01T00:00:00Z",
 "credentialSubject": {
  "id": "https://www.example.org/persons/pat",
  "name": "Pat",
  "alumniOf": {
   "id": "did:example:c276e12ec21ebfeb1f712ebc6f1",
   "name": "Example University"
  }
 },
 …
```

An example of a basic credential (without proof)

Identification of

- the terminologies
- the credential itself
- the type of the credential

```
{
 "@context": [
  "https://www.w3.org/ns/credentials/v2",
  "https://www.example.org/vocabs/alumni"
 ],
 "id": "https://uni.example/Credential12",
 "type": ["VerifiableCredential", "ExampleAlumniCredential"],
 "issuer": "did:example:2g55q91",
 "validFrom": "2010-01-01T00:00:00Z",
 "credentialSubject": {
  "id": "https://www.example.org/persons/pat",
  "name": "Pat",
  "alumniOf": {
   "id": "did:example:c276e12ec21ebfeb1f712ebc6f1",
   "name": "Example University"
  }
 },
 …
```

An example of a basic credential (without proof)

Credential metadata

```
{
 "@context": [
  "https://www.w3.org/ns/credentials/v2",
  "https://www.example.org/vocabs/alumni"
 ],
 "id": "https://uni.example/Credential12",
 "type": ["VerifiableCredential", "ExampleAlumniCredential"],
 "issuer": "did:example:2g55q91",
 "validFrom": "2010-01-01T00:00:00Z",
 "credentialSubject": {
  "id": "https://www.example.org/persons/pat",
  "name": "Pat",
  "alumniOf": {
   "id": "did:example:c276e12ec21ebfeb1f712ebc6f1",
   "name": "Example University"
  }
 },
 …
```

An example of a basic credential (without proof)

Credential claims

```
{
 "@context": [
  "https://www.w3.org/ns/credentials/v2",
  "https://www.example.org/vocabs/alumni"
 ],
 "id": "https://uni.example/Credential12",
 "type": ["VerifiableCredential", "ExampleAlumniCredential"],
 "issuer": "did:example:2g55q91",
 "validFrom": "2010-01-01T00:00:00Z",
 "credentialSubject": {
  "id": "https://www.example.org/persons/pat",
  "name": "Pat",
  "alumniOf": {
   "id": "did:example:c276e12ec21ebfeb1f712ebc6f1",
   "name": "Example University"
  }
 },
 …
```

Securing mechanisms

Enveloping proofs: JOSE/COSE

- The JSON-LD representation is fed into a JWT pipeline
  - reusing the JOSE toolkit, registered signature mechanisms, etc.
- Using COSE (i.e., CBOR) means a very small footprint for credentials
- Using IETF's SD-JWT provides selective disclosure

Enveloping proofs: this is how it looks like (roughly)

```
JWT Header:
{
  "kid": "ExHkBMW9fmbkvV…"
  "alg": "ES256v
}
JWT Payload (application/vc):
{
  "@context": [
    "https://www.w3.org/ns/credentials/v2",
    "https://www.example.org/vocabs/alumni"
  ],
  "id": "https://uni.example/Credential12",
  "type": ["VerifiableCredential", "ExampleAlumniCredential"],
  …
}
JWT Proof (application/vc+jwt):
eyJraWQiOiJFeEhrQk1XOWZtYmt2VjI2Nm1ScHVQMnNVWV9OX0VXSU4xbGFwVXpPOHJvIi
wiYWxnIjoiRVMyNTYifQ.eyJAY29udGV4dCI6WyJodHRwczovL3d3dy53My5vcmcvbnMvY
3JlZGVudGlhbHMvdjIiLCJodHRwczovL3d3dy53My5vcmcvbnMvY3JlZGVudGlhbHMvZXh
hbXBsZXMvdjIiXSwiaWQiOiJodHRwczovL3VuaXZlcnNpdHkuZXhhbXBsZS9DcmVkZW50a
WFsMTIzIiwidHlwZSI6WyJWZXJpZmlhYmxlQ3JlZGVudGlhbCIsIkV4YW1wbGVBbHVtbml
DcmVkZW50aWFsIl0sImlzc3VlciI6ImRpZDpl
```

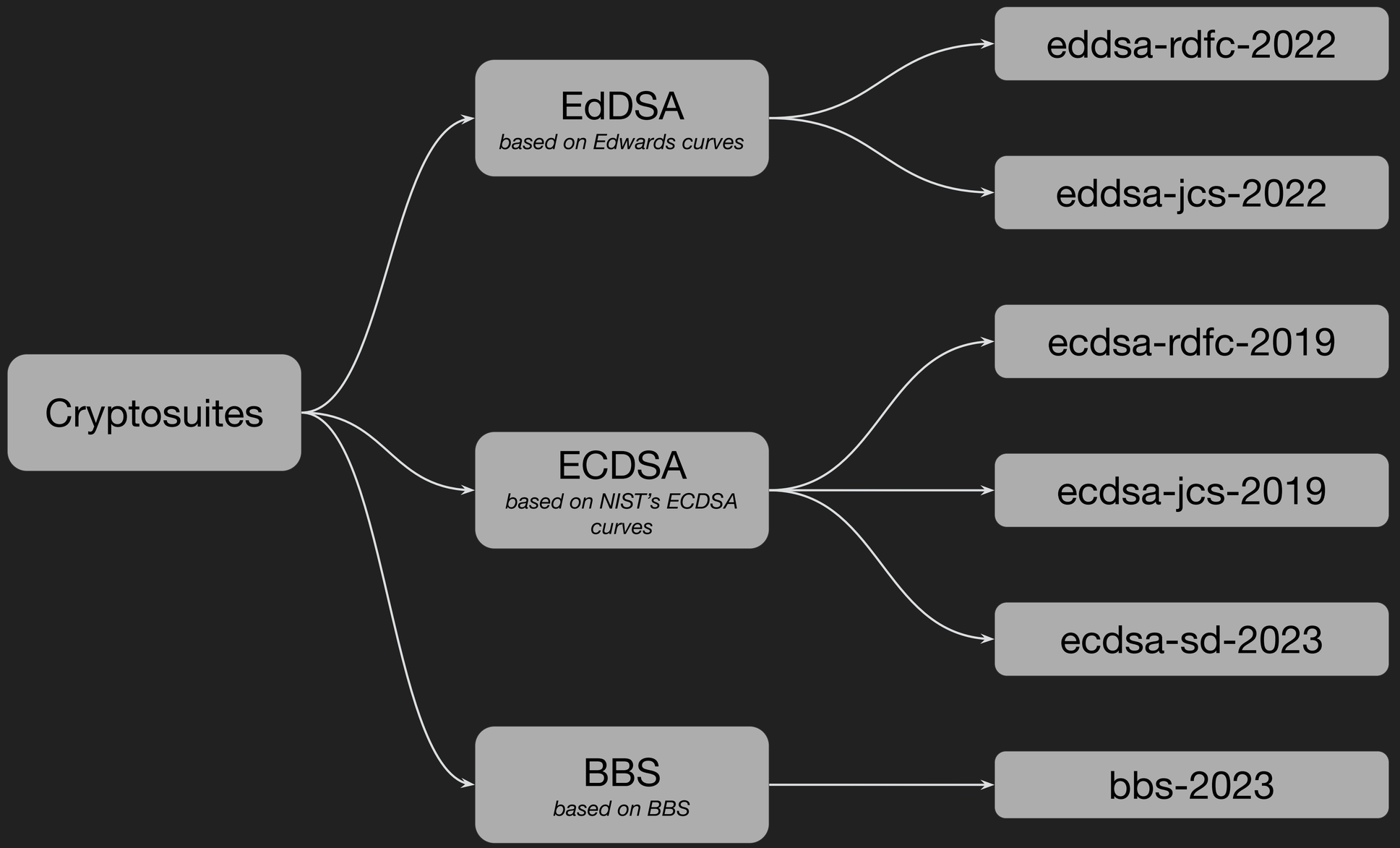
Enveloping proofs: JOSE/COSE

- Using JWT means reusing off-the shelf tools
  - means easier deployment
- Relies on a centralized registry for cryptographic schemes that can be used
- Enveloping Verifiable *Presentations* becomes awkward
  - remember that a VP contains separate issuer and holder proofs…

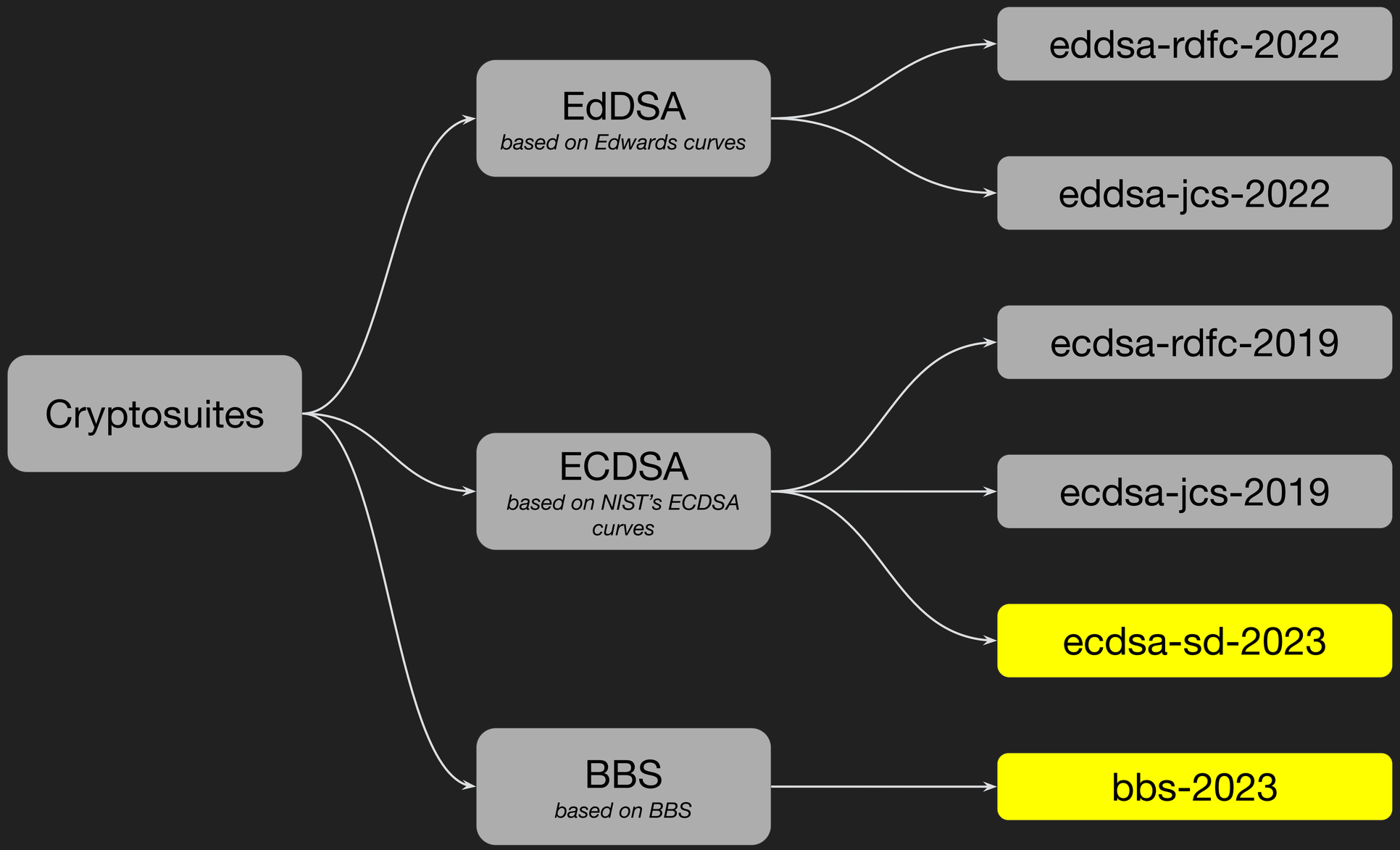
Embedded proofs: the proof is “in” the credential

- The Data Integrity specification provides a general framework to represent proofs within the JSON-LD structure
- The various “cryptosuite” specifications map cryptographic schemes to this framework
- Communities may add their cryptosuite to use other schemes (there is no central registration mechanism)
  - for example, the Chinese community may want to use SM2 instead of the NIST curves

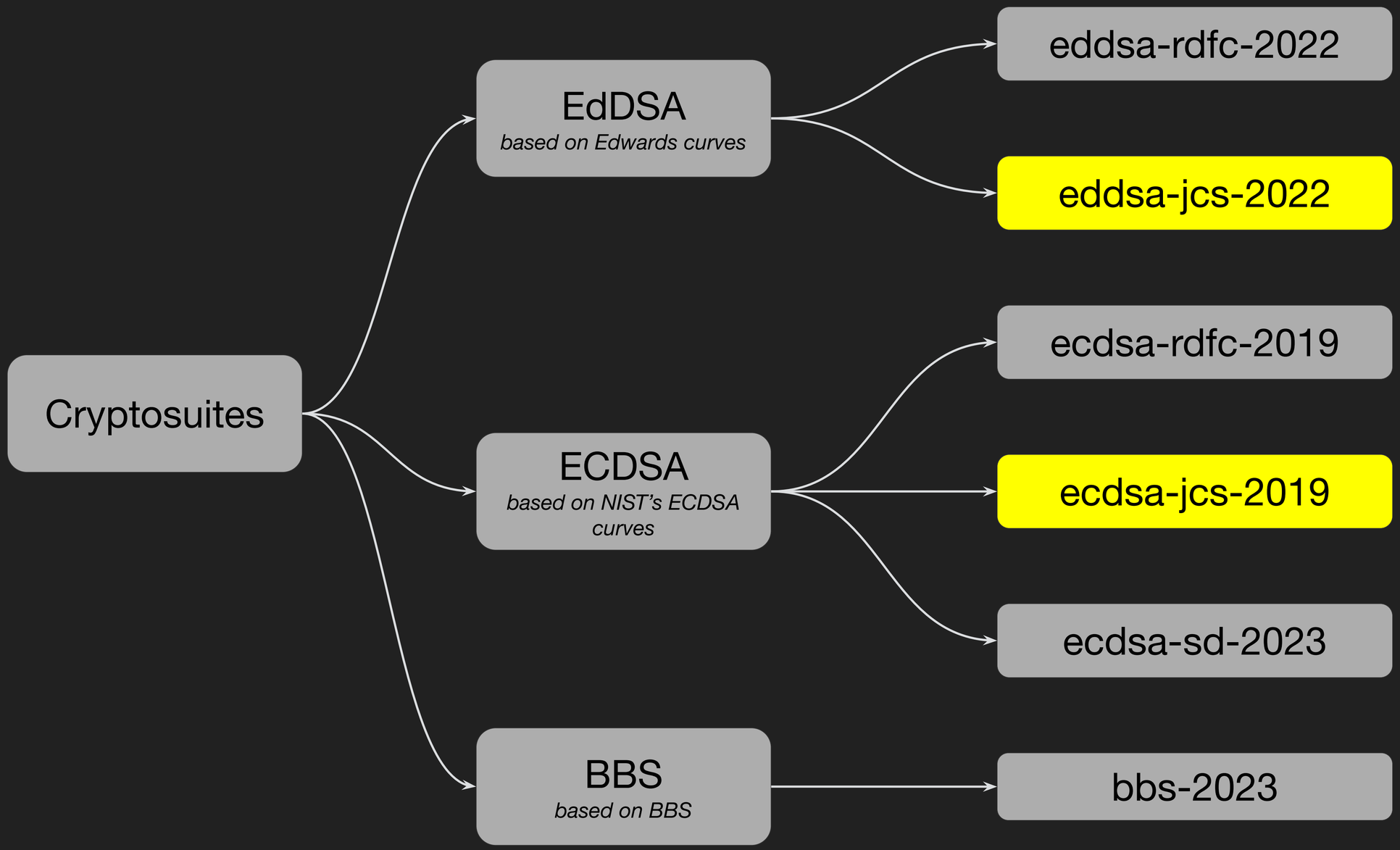
Cryptosuites defined by the VC Working Group



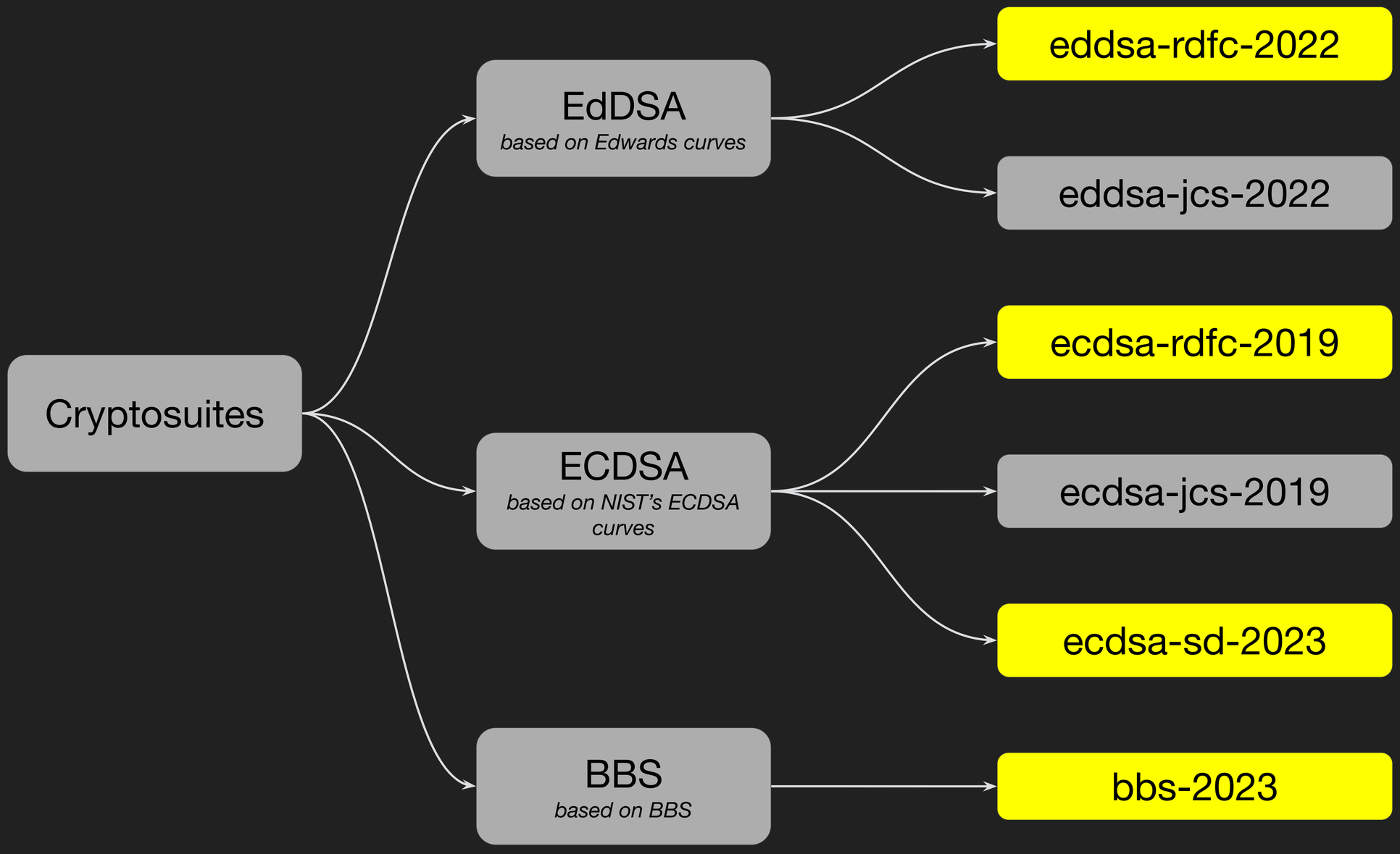
Cryptosuites with selective disclosure



Cryptosuites using JCS for canonicalization



Cryptosuites relying on the graph model



Example: Basic credential…

```
{
 "@context": [
  "https://www.w3.org/ns/credentials/v2",
  "https://www.example.org/vocabs/alumni"
 ],
 "id": "https://uni.example/Credential12",
 "type": ["VerifiableCredential", "ExampleAlumniCredential"],
 "issuer": "did:example:2g55q91",
 "validFrom": "2010-01-01T00:00:00Z",
 "credentialSubject": {
  "id": "https://www.example.org/persons/pat",
  "name": "Pat",
  "alumniOf": {
   "id": "did:example:c276e12ec21ebfeb1f712ebc6f1",
   "name": "Example University"
  }
 },
 …
```

…and signed using ECDSA

```
 …
 "proof": {
  "type": "DataIntegrityProof",
  "cryptosuite": "ecdsa-rdfc-2019",
  "created": "2010-01-01T00:00:00Z",
  "expires": "2040-01-01T00:00:00Z",
  "verificationMethod: "did:example:2…q91#ecdsa-public-key",
  "proofPurpose": "assertionMethod",
  "proofValue": "zQeVb…Wx"
 }
}
```

…and signed using ECDSA

Proof metadata

```
 …
 "proof": {
  "type": "DataIntegrityProof",
  "cryptosuite": "ecdsa-rdfc-2019",
  "created": "2010-01-01T00:00:00Z",
  "expires": "2040-01-01T00:00:00Z",
  "verificationMethod: "did:example:2…q91#ecdsa-public-key",
  "proofPurpose": "assertionMethod",
  "proofValue": "zQeVb…Wx"
 }
}
```

…and signed using ECDSA

Reference to the ECDSA public key

```
 …
 "proof": {
  "type": "DataIntegrityProof",
  "cryptosuite": "ecdsa-rdfc-2019",
  "created": "2010-01-01T00:00:00Z",
  "expires": "2040-01-01T00:00:00Z",
  "verificationMethod: "did:example:2…q91#ecdsa-public-key",
  "proofPurpose": "assertionMethod",
  "proofValue": "zQeVb…Wx"
 }
}
```

…and signed using ECDSA

The ECDSA signature itself

```
 …
 "proof": {
  "type": "DataIntegrityProof",
  "cryptosuite": "ecdsa-rdfc-2019",
  "created": "2010-01-01T00:00:00Z",
  "expires": "2040-01-01T00:00:00Z",
  "verificationMethod: "did:example:2…q91#ecdsa-public-key",
  "proofPurpose": "assertionMethod",
  "proofValue": "zQeVb…Wx"
 }
}
```