ZKLang – Implementation and Standardization

Jan Camenisch[1], Manu Drijvers[1], Maria Dubovitskaya, [1] Jason Law[2], ...

1: IBM Research – Zurich

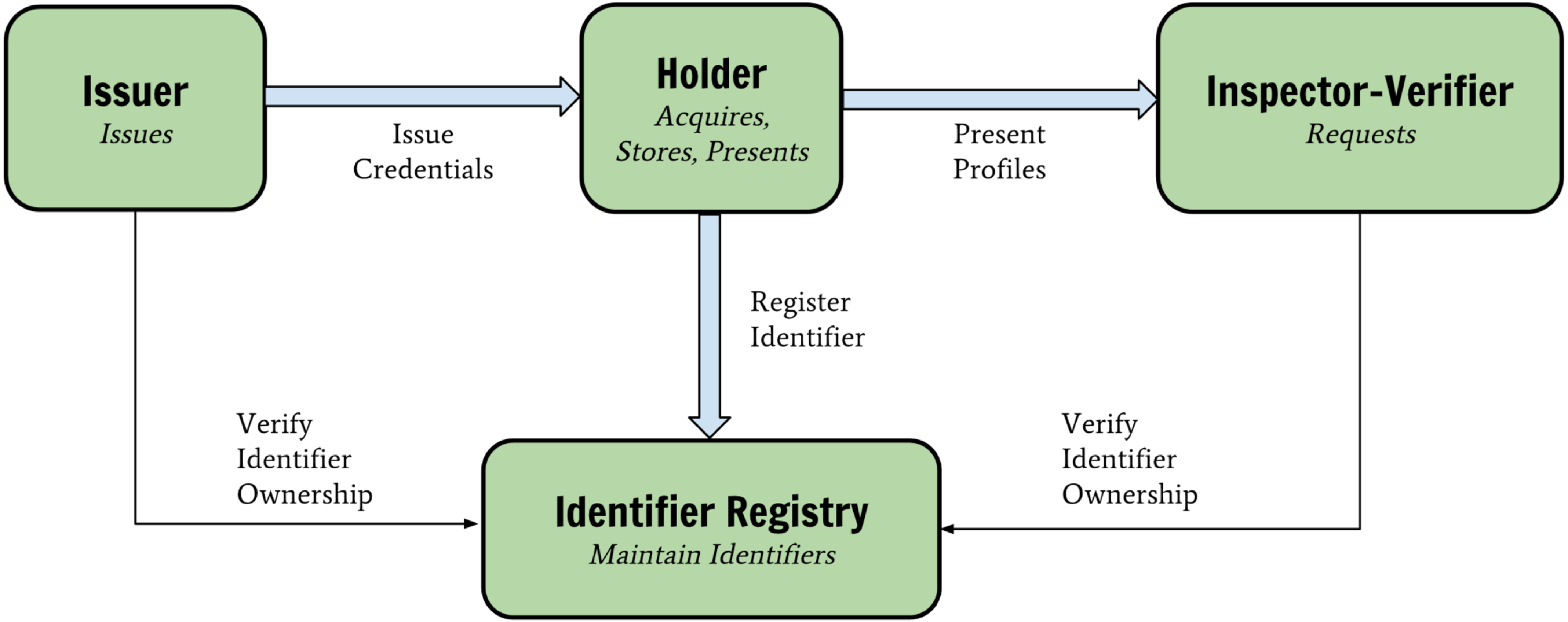
2: Evernym

W3C Verifiable Claims (VC)

- An effort for standardizing protocols and languages for authentication and identity management
- Supports different levels of privacy preservation

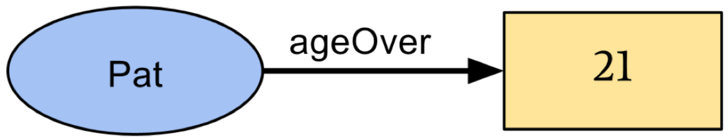
- A holder collects credentials from different issuers
- A verifiable credential reveals multiple claims about the holder to service providers
- A claim can reveal different attributes (e.g., email address) or just facts (e.g., Older18) about the holder
- Revocation and Inspection are supported

W3C Verifiable Claims: Entities

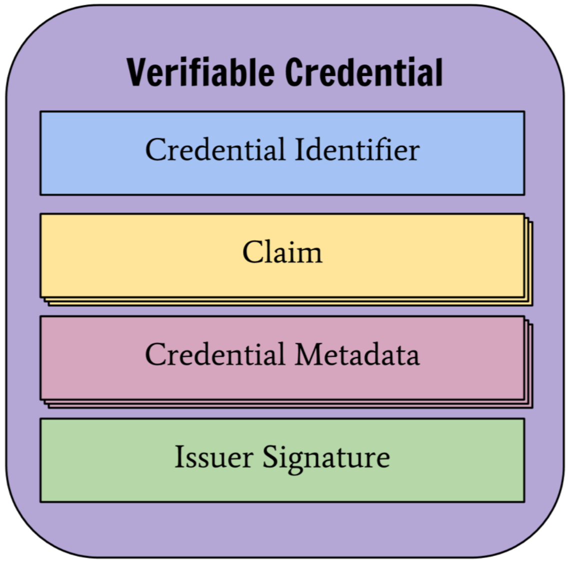


W3C Verifiable Claims: Data Model

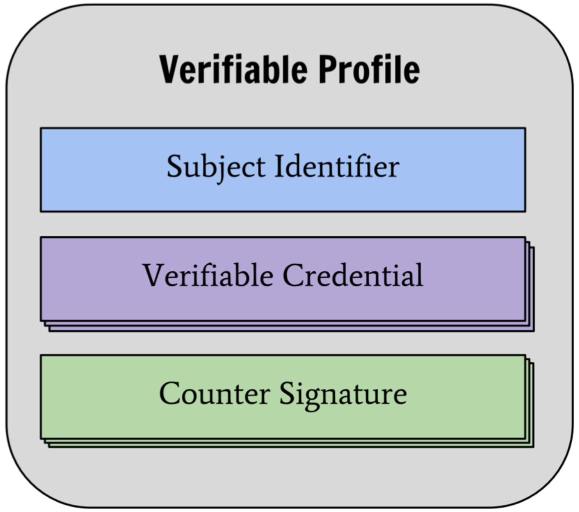
- Claim



- Verifiable Credential



- Verifiable Profile



Cryptographic Protocols to Realize VC

- We can use advanced crypto to get privacy-friendly VC

- Issuer signs subject's attributes using special type of signature (CL signature)
- Non-Interactive Zero-Knowledge Proofs (NIZK) to generate verifiable credentials/profiles
- Verifiable Encryption to conditionally reveal attributes only to certain entities (revocation/auditability)

Example: Proving Knowledge of BBS+ Signature

PoK of Signature (A, e, s) on message m w.r.t. issuer public key $y = {g'}^{x}$

- $A' \leftarrow A^r$
- $\bar{A} \leftarrow A'^{-e} \cdot (g_1 \cdot {h_0}^{s} \cdot {h_1}^{m})^r \qquad (= {A'}^{x})$
- $d \leftarrow (g_1 \cdot {h_0}^{s} \cdot {h_1}^{m})^r \cdot {h_0}^{r'}$

$$SPK\left\{(m, e, s', r, r', r''): \frac{\bar{A}}{d} = A'^{-e} \cdot {h_0}^{r'} \;\wedge\; g_1 = d^{r''} \cdot h_0^{-s'} \cdot h_1^{-m}\right\}$$

Implementing even a simple verifiable claim results in a complicated NIZK statement and requires orchestration of different cryptographic building blocks

Problem: Gap Between high-level W3C VC language and Complex Cryptographic Algorithms

EXAMPLE 2: Usage of signature property

```
{
  "id": "http://example.gov/credentials/3732",
  "type": ["Credential", "ProofOfAgeCredential"],
  "issuer": "https://dmv.example.gov",
  "issued": "2010-01-01",
  "claim": {
    "id": "did:example:ebfeb1f712ebc6f1c276e12ec21",
    "ageOver": 21
  },
  "signature": {
    "type": "LinkedDataSignature2017",
    "created": "2017-06-18T21:19:10Z",
    "creator": "https://example.com/jdoe/keys/1",
    "nonce": "c0ae1c8e-c7e7-469f-b252-86e6a0e7387e",
    "signatureValue": "BavEll0/I1zpYw8XNi1bgVg/sCneO4Jugez8RwDg/+
    MCRVpjOboDoe4SxxKjkCOvKiCHGDvc4krqi6Z1n0UfqzxGfmatCuFibcC1wps
    PRdW+gGsutPTLzvueMWmFhwYmfIFpbBu95t501+rSLHIEuujM/+PXr9Cky6Ed
    +W3JT24="
  }
}
```

?

Signature (A, e, s)

- $A' \leftarrow A^r$
- $\bar{A} \leftarrow A'^{-e} \cdot (g_1 \cdot h_0{}^s \cdot h_1{}^m)^r \qquad (= A'^x)$
- $d \leftarrow (g_1 \cdot h_0{}^s \cdot h_1{}^m)^r \cdot h_0{}^{r'}$

$$SPK\left\{(m, e, s', r, r', r''): \frac{\bar{A}}{d} = A'^{-e} \cdot h_0{}^{r'} \;\wedge\; g_1 = d^{r''} \cdot h_0^{-s'} \cdot h_1^{-m}\right\}$$

Solution: ZKLang

EXAMPLE 2: Usage of signature property

```
{
  "id": "http://example.gov/credentials/3732",
  "type": ["Credential", "ProofOfAgeCredential"],
  "issuer": "https://dmv.example.gov",
  "issued": "2010-01-01",
  "claim": {
    "id": "did:example:ebfeb1f712ebc6f1c276e12ec21",
    "ageOver": 21
  },
  "signature": {
    "type": "LinkedDataSignature2017",
    "created": "2017-06-18T21:19:10Z",
    "creator": "https://example.com/jdoe/keys/1",
    "nonce": "c0ae1c8e-c7e7-469f-b252-86e6a0e7387e",
    "signatureValue": "BavEll0/I1zpYw8XNi1bgVg/sCneO4Jugez8RwDg/+
    MCRVpjOboDoe4SxxKjkCOvKiCHGDvc4krqi6Z1n0UfqzxGfmatCuFibcC1wps
    PRdW+gGsutPTLzvueMWmFhwYmfIFpbBu95t501+rSLHIEuujM/+PXr9Cky6Ed
    +W3JT24="
  }
}
```

ZKLang

Signature (A, e, s)

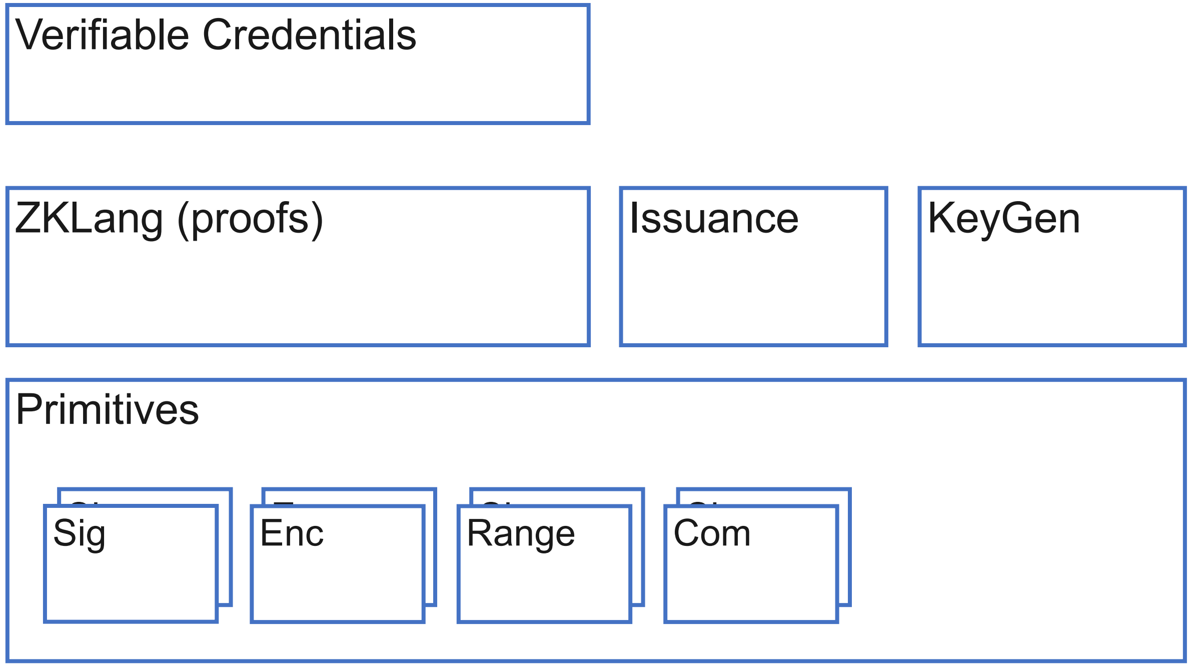
- $A' \leftarrow A^r$
- $\bar{A} \leftarrow A'^{-e} \cdot (g_1 \cdot {h_0}^s \cdot {h_1}^m)^r \qquad (= A'^x)$
- $d \leftarrow (g_1 \cdot {h_0}^s \cdot {h_1}^m)^r \cdot {h_0}^{r'}$

$$SPK\left\{(m, e, s', r, r', r''): \frac{\bar{A}}{d} = A'^{-e} \cdot {h_0}^{r'} \ \wedge \ g_1 = d^{r''} \cdot h_0^{-s'} \cdot h_1^{-m}\right\}$$

Overview and Goal

- ZKLang: language mapping W3C verifiable claims to cryptographic algorithms
  - Prove claims in a privacy-preserving way (using ZKP)
  - Abstracts cryptographic algorithms
    - (mapping to crypto algorithms needs to be specified)
  - Translates verifiable claims
    - (mapping between verifiable claims and ZKLang needs to be specified)
- Goal: define and implement ZKLang

Overview and Goal



ZKLang: Notation and Examples

Non Interactive Zero-knowledge proof of Knowledge (NIZK) statements:

- NIZK{(m_1,m_2,m_3)[m_4]: Statement(constants, m_1,m_2,m_3,m_4)}
  - (m_1, m_2, …) are hidden messages (encoded as integers);
  - [m_4] are messages (attributes) that are revealed

- NIZK{(m_1,m_2,m_3)[m_4]: Credential(PK_{issuer}, m_1,m_2,m_3,m_4)} – possession of a credential
- NIZK{(m_2): Interval(m_2, constant, constant)} – range proof
- NIZK{(m_3): Enc($PK_{auditor}$, ciphertext,m_3)} – verifiable encryption for auditing
- NIZK{(): Nym(PPK)} – pseudonymous user public key
- NIZK{(): ScopeNym(PPK,scope)} – nym, but unique per scope
- NIZK{(m_1,m_2,m_3): Polyrel(m_1= m_1 - 4m_2 + constant)} – linear relations

ZKLang: Notation and Examples

Terms can be combined

- `NIZK{(m`$_1$`,m`$_2$`,m`$_3$`)[m`$_4$`]:`

  `Credential(PK`$_{issuer}$`, m`$_1$`,m`$_2$`,m`$_3$`,m`$_4$`) AND`

  `Enc(PK`$_{auditor}$`, ciphertext, m`$_3$`) AND`

  `Interall (today-m`$_2$`,0,18*365) AND`

  `Nym(PPK)}`

- prove possession of a credential with four attributes issued by an issuer with Pk_{issuer},
- reveal attribute #4,
- verifiably encrypt attribute #3 under auditor's key $PK_{auditor}$

Mapping Verifiable Claims to ZKLang

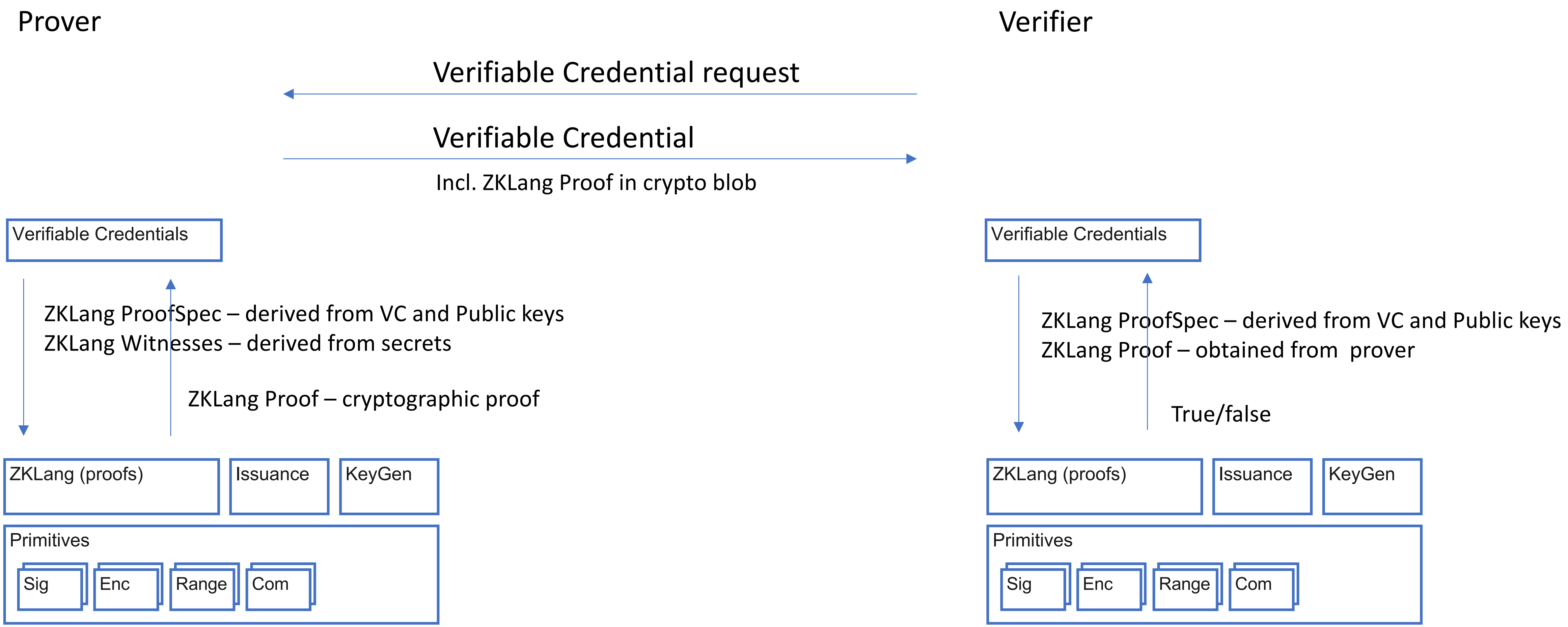
- Map Issuer name to issuer public key ($\texttt{PK}_{\texttt{issuer}}$)
- Map higher level data format (strings, dates, names, etc) to integers
- Translate predicates such as `Over18` into $\texttt{Larger(today-m}_2\texttt{,18)}$
  - $\texttt{m}_2$ is an attribute that encodes the year of birth

Mapping to Cryptographic algorithms

- Multiple options possible (RSA, ECC, DL)
  - Different cryptographic assumptions
  - Different implementations
- Different building blocks are realized in different groups
- Need to be carefully defined to allow for interoperability

- Signatures:
  - CL-signatures (RSA/ECC), U-Prove (Brands) signatures
- Range proofs:
  - `Smaller/Larger` can be realized in RSA groups

ZKLang Objects



JSON Objects for ZKLang (somewhat misformated)

ZKL-ProofSpec:{ "attributeCount": 10,

"disclosed": [{ "index": 3, "value": 500}, {"index": 9, "value": 20}],

"clauses": [{"type": "Credential", "dataclauseData": { "pk": "<ipk1>", "attrs": [0, 1, 2, 3] },

{"type": "Credential", "clauseData": { "pk": "<ipk2>","attrs": [0, 4, 5, 6, 7, 8, 9]} },

{"type": "Interval", "clauseData": { "attrs": [2], "min": 6, "max": 10, "pk": "<rpk>}] }

ZKL-Witness:{ "attributeValues": ["av0","av1","av2","av3","av4","av5","av6","av7","av8"],

"clauseSecrets": ["<cred1>", "<cred2>", "<enc randomness>", "<nym randomness>", null] }

ZKL-Proof:{ "chal": "<c>", "s": [s0, s1, s2, s4, s5, s6, s7, s8],

"clauseOut": ["<out0>", "<out1>", "<out2>", "<out3>", "<out4>", "<out5>”],

"clauseProof": ["<proof0>", "<proof1>", "<proof2>", "<proof3>", "<proof4>", "<proof5>”]}

Next Steps

- Finishing ZKLang Spec
- Specify mapping to crypto
- Specify crypto algorithms
- Implement it…

Backup slides

W3C Verifiable Claims: Examples

EXAMPLE 6: A simple entity profile


```
{
  "@context": "https://w3id.org/identity/v1",
  "id": "did:example:ebfeb1f712ebc6f1c276e12ec21",
  "type": ["Entity", "Person"],
  "name": "Alice Bobman",
  "email": "alice@example.com",
  "birthDate": "1985-12-14",
  "telephone": "12345678910"
}
```


EXAMPLE 7: A simple claim


```
{
  "@context": "https://w3id.org/identity/v1",
  "id": "http://example.gov/credentials/3732",
  "type": ["Credential", "ProofOfAgeCredential"],
  "issuer": "https://dmv.example.gov",
  "issued": "2010-01-01",
  "claim": {
    "id": "did:example:ebfeb1f712ebc6f1c276e12ec21",
    "ageOver": 21
  }
}
```

W3C Verifiable Claims: Examples

EXAMPLE 8: A simple verifiable claim


```
{
  "@context": [
    "https://w3id.org/identity/v1",
    "https://w3id.org/security/v1"
  ],
  "id": "http://example.gov/credentials/3732",
  "type": ["Credential", "ProofOfAgeCredential"],
  "issuer": "https://dmv.example.gov",
  "issued": "2010-01-01",
  "claim": {
    "id": "did:example:ebfeb1f712ebc6f1c276e12ec21",
    "ageOver": 21
  },
  "signature": {
    "type": "LinkedDataSignature2015",
    "created": "2016-06-18T21:10:38Z",
    "creator": "https://example.com/jdoe/keys/1",
    "domain": "json-ld.org",
    "nonce": "6165d7e8",
    "signatureValue": "g4j9UrpHM4/uu32NlTw0HDaSaYF2sykskfuByD
7UbuqEcJIKa+IoLJLrLjqDnMz0adwpBCHWaqqpnd47r0NKZbnJarGYrBFcRTw
PQSeqGwac8E2SqjylTBbSGwKZkprEXTywyV7gILlC8a+naA7lBRi4y29FtcUJ
BTFQq4R5XzI="
  }
}
```