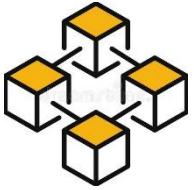


UNIT 7.5

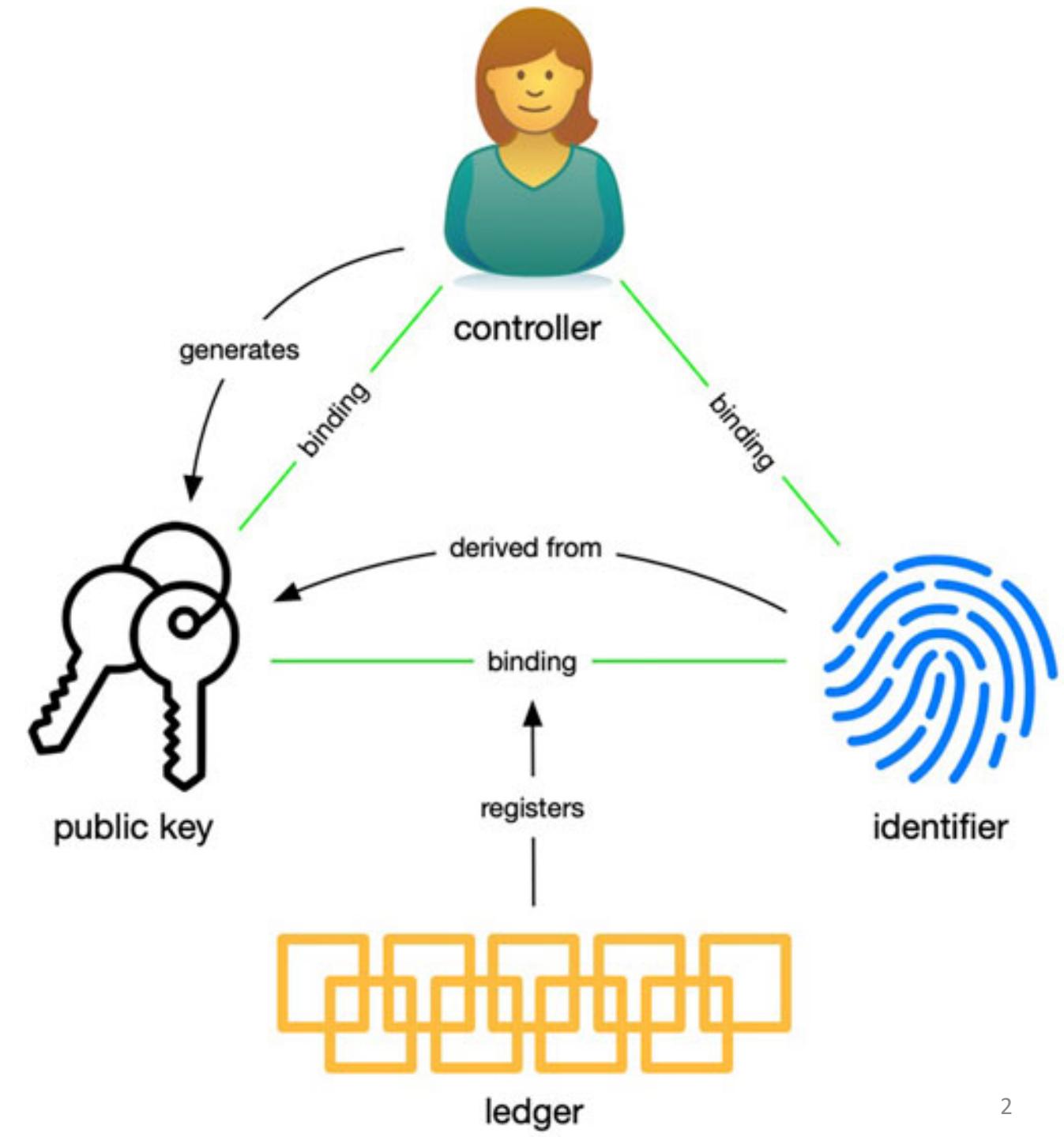
BLOCKCHAIN APPLICATIONS SELF-SOVEREIGN IDENTITY

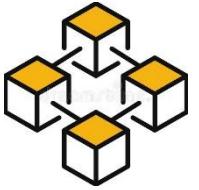
Lecturer: Ph.D Lê Quang Huy



CONTENTS

1. INTRODUCTION
2. IDENTITY MANAGEMENT
3. SELF-SOVEREIGN IDENTITY
4. DECENTRALIZED IDENTIFIERS
5. VERIFIABLE CREDENTIALS
6. SSI BLOCKCHAIN PLATFORMS
7. SUMMARY
8. DISCUSSION

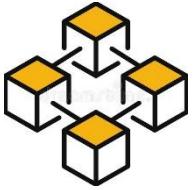




1. INTRODUCTION

OLD WORLD





2. IDENTITY MANAGEMENT

2.1. INTRODUCTION

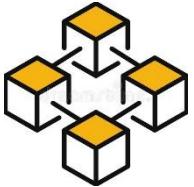
2.2. DIGITAL IDENTITY

2.3. IDENTITY MANAGEMENT

2.4. BENEFIT OF IDENTITY MANAGEMENT

2.5. IDENTITY MANAGEMENT MODELS





2.2. DIGITAL IDENTITY

Entity:

- thing with distinct independent existence.
- can have multiple identities

Identity: encompass

- an Identifier
- Attributes
- Activities

Physical Identity

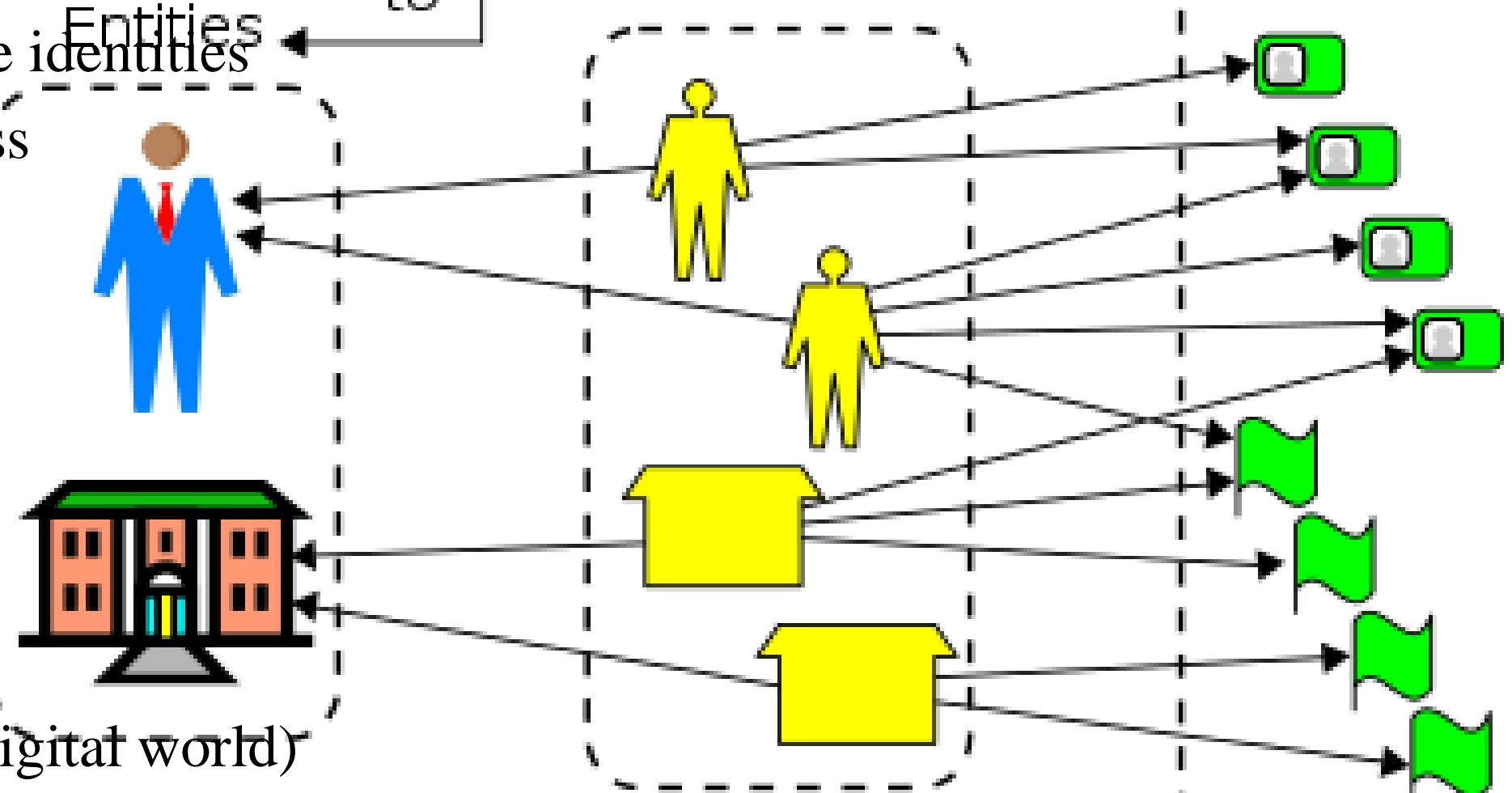
- Intrinsic
- Extrinsic

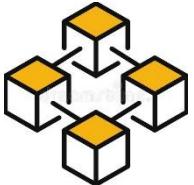
Digital Identity: (digital world)

*correspond
to*

consist
of

Attributes /
Identifiers





2.2. DIGITAL IDENTITY

Identity activities: create/destroy, use

Identification: create identity

- Submit/gather identity claim (attribute, activities).
- Verification: check identity claim
- Deduplication
- Issue identity Artifact/Credential

Authentication: Check identity credential, factors:

- Knowledge, Possession
- Inherence, Behavioral

Authorisation:

- Relying party
- Actions may be performed

Identity Proofing

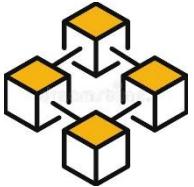
Establish identity

Authentication

using identity
Assert identity

Authorisation

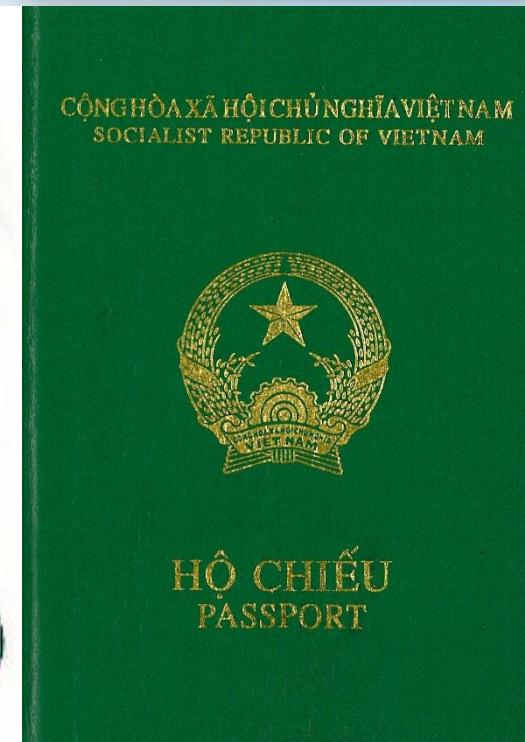
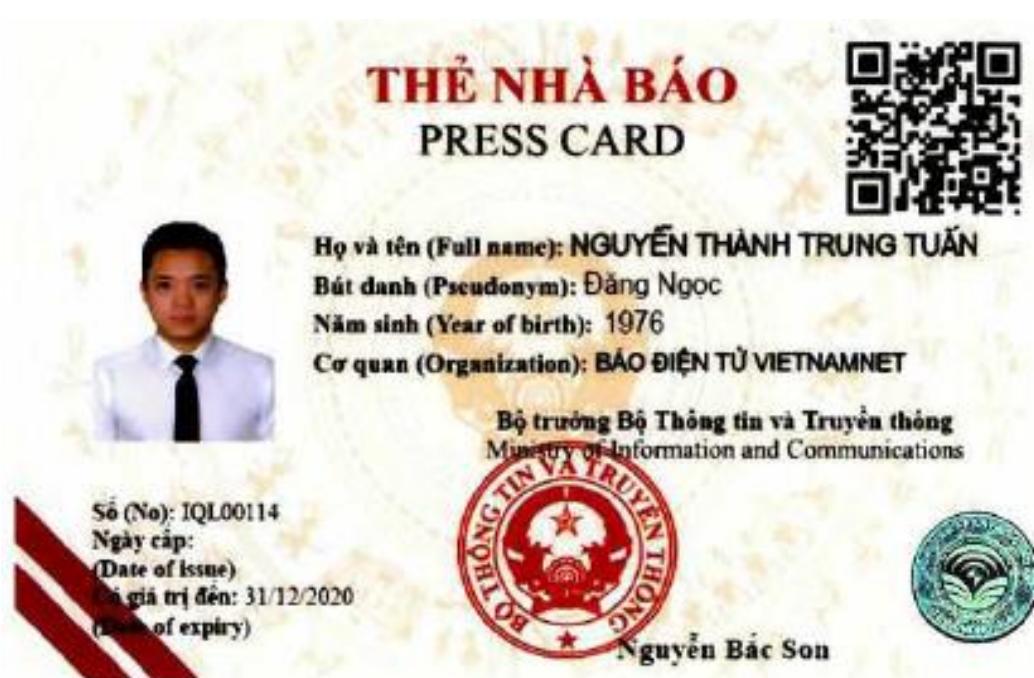
Use identity

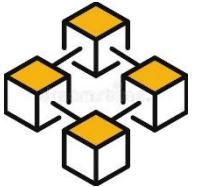


2.2. DIGITAL IDENTITY

Credentials:

- part of our daily lives;
- driver's licenses, university certificate, passports

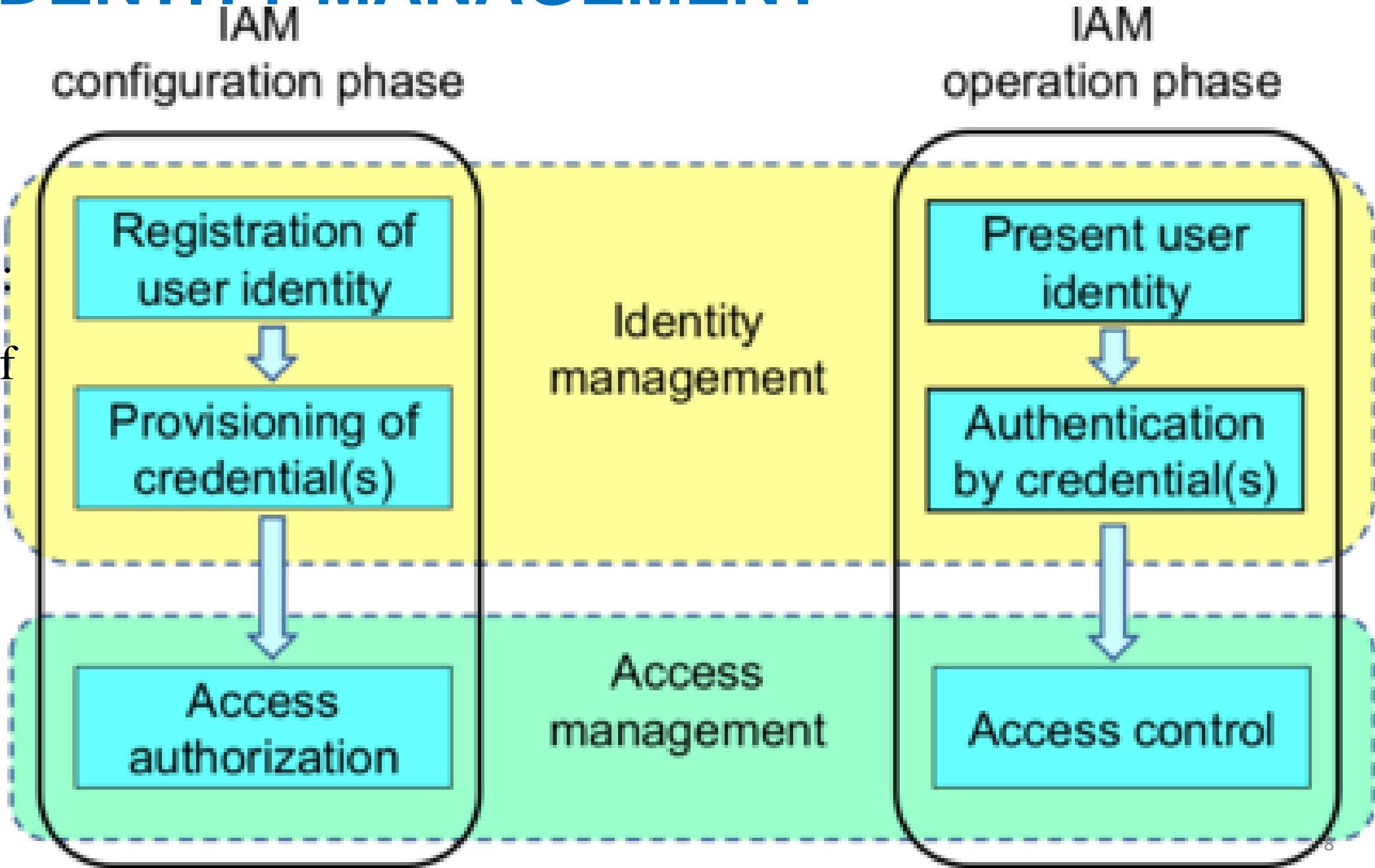


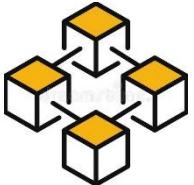


2.3. IDENTITY MANAGEMENT

Identity and Access Management (IAM) systems:

- Framework of policies and technologies
- Right users access to resources.

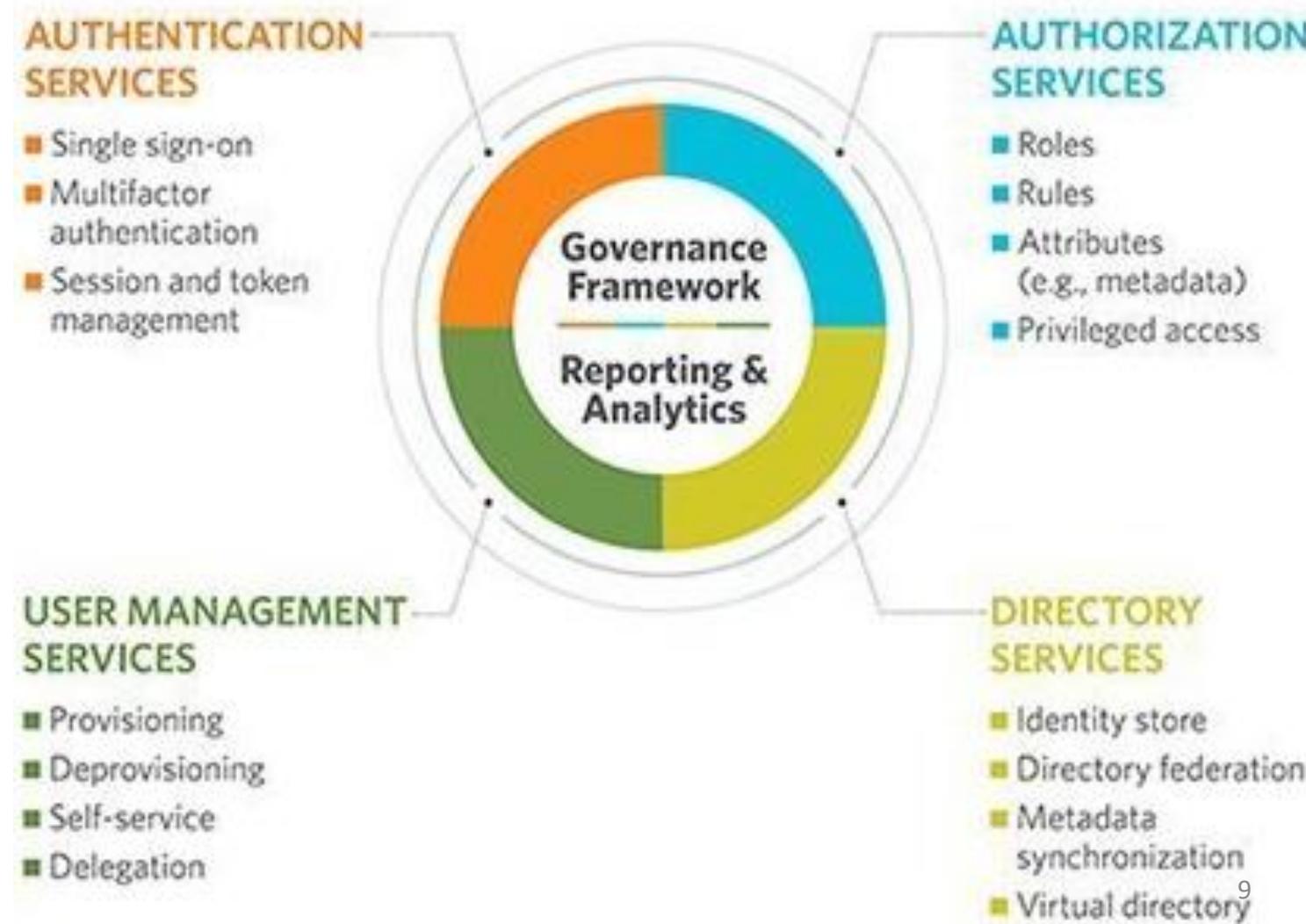


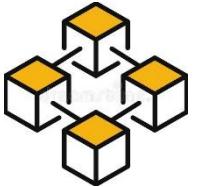


2.3. IDENTITY MANAGEMENT

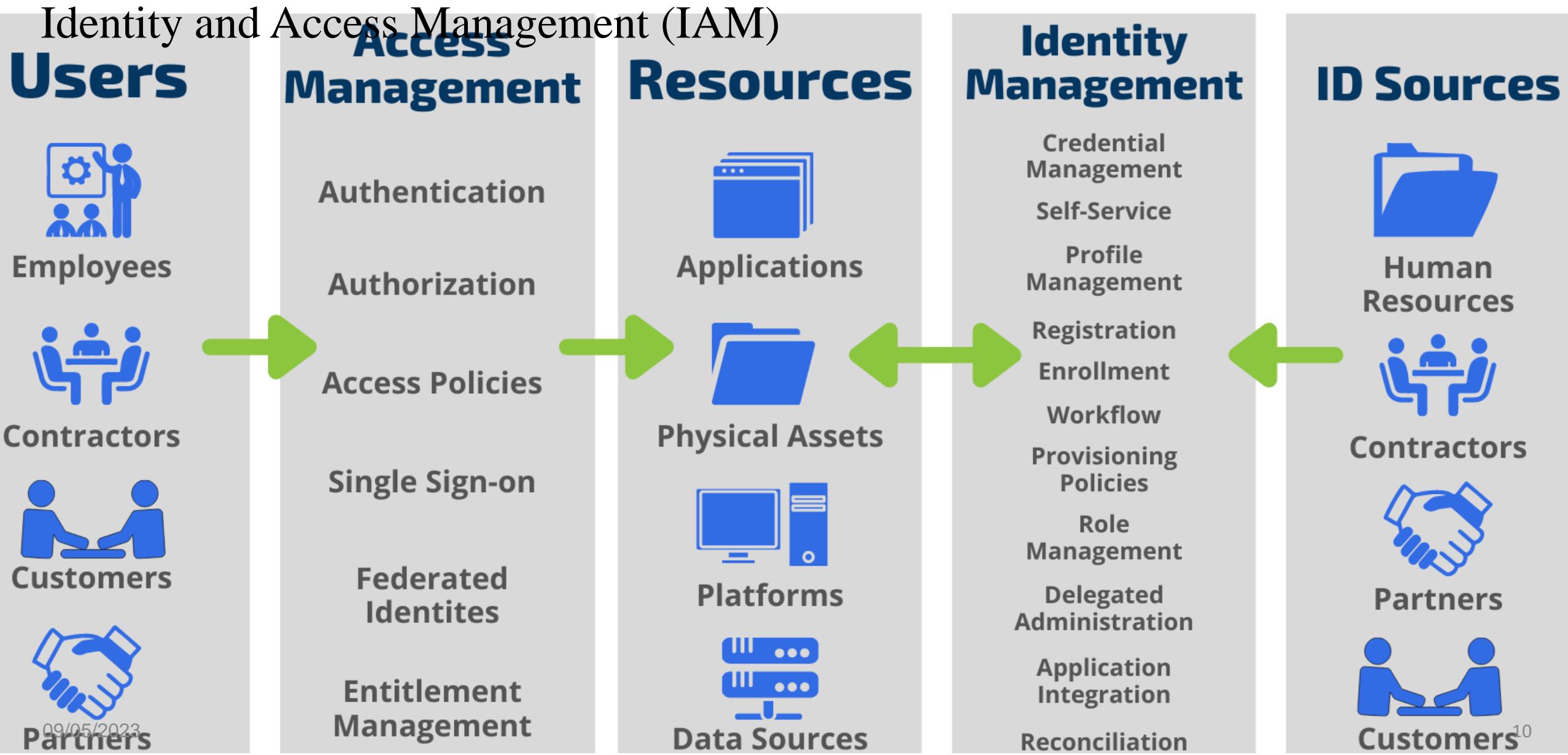
IAM service components

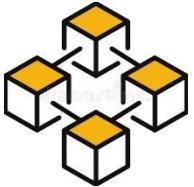
Identity and Access Management (IAM)



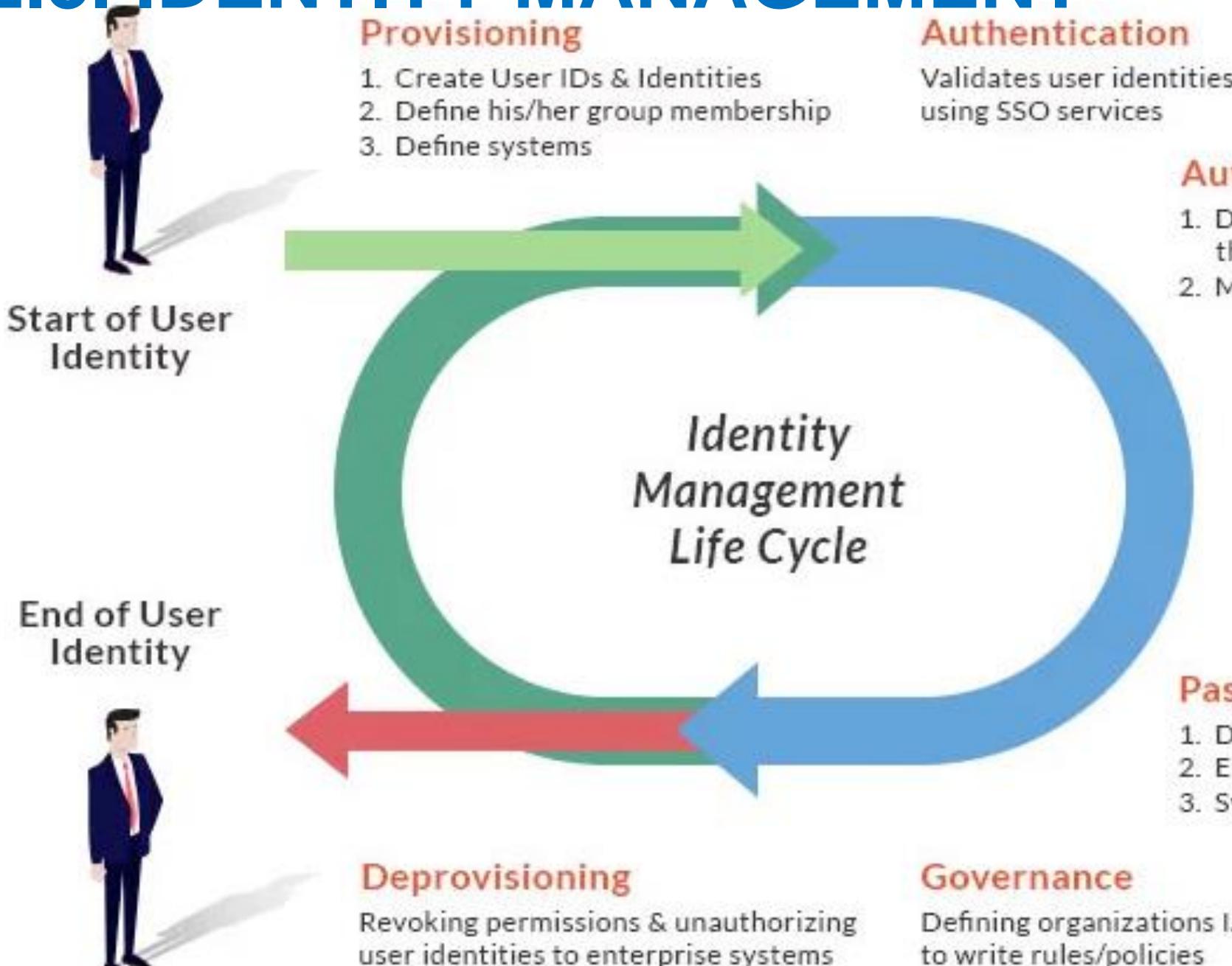


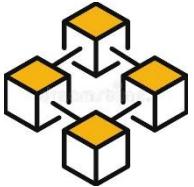
2.3. IDENTITY MANAGEMENT





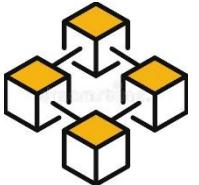
2.3. IDENTITY MANAGEMENT



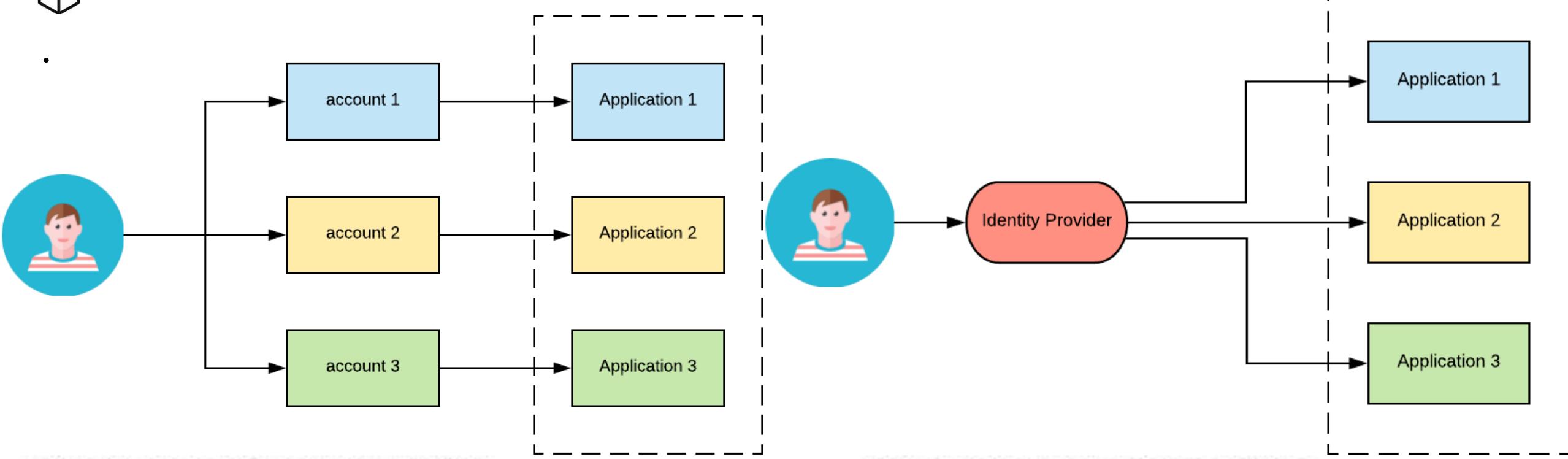


2.4. BENEFIT OF IDENTITY MANAGEMENT





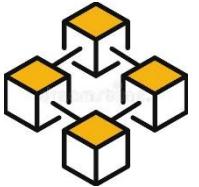
2.5. IDENTITY MANAGEMENT MODELS



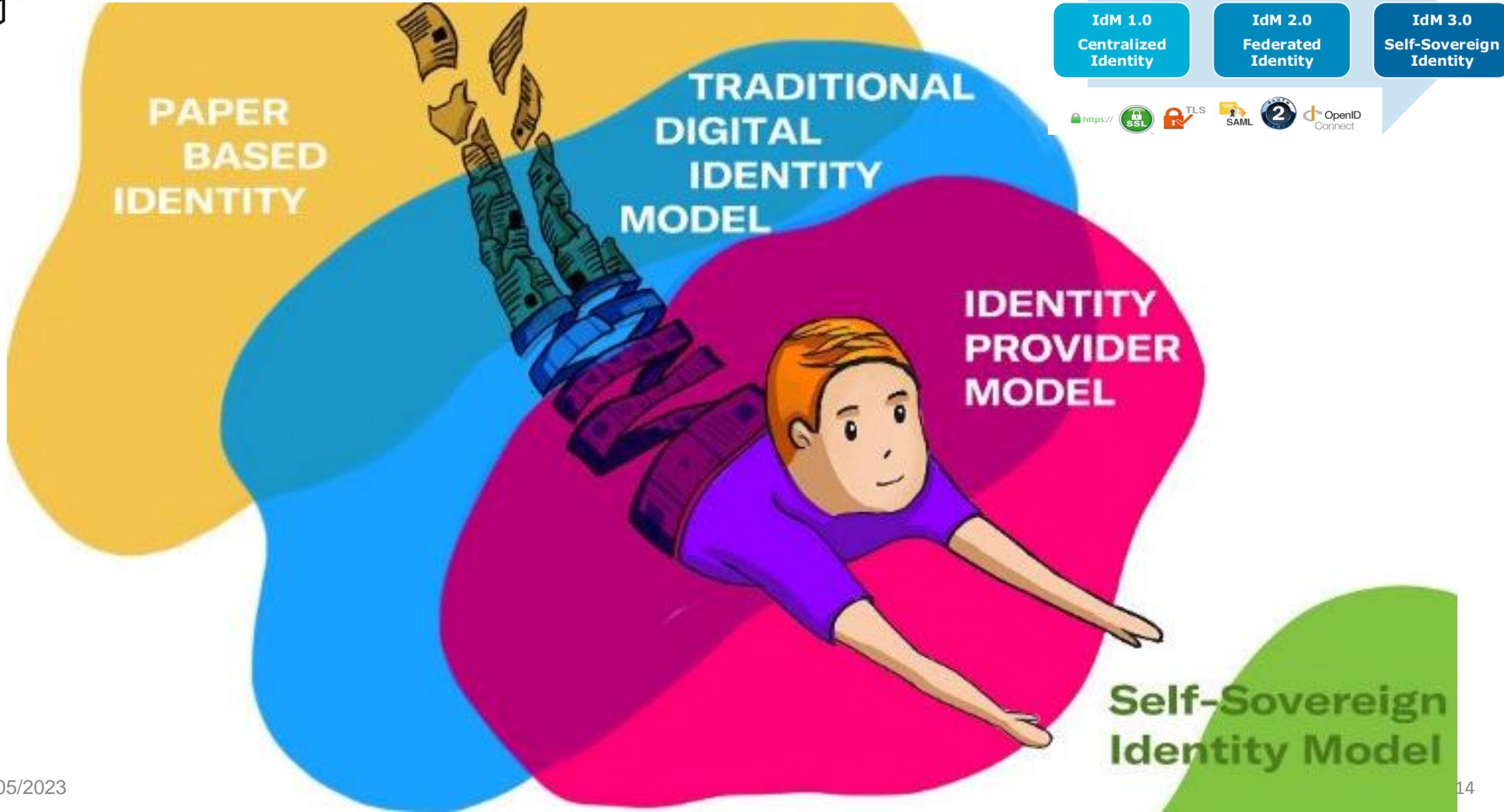
#1: Centralized Identity

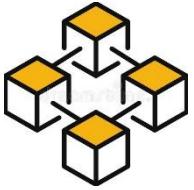
#2: Federated Identity





2.5. IDENTITY MANAGEMENT MODELS





3. SELF-SOVEREIGN IDENTITY

3.1. INTRODUCTION TO SSI

3.2. GUIDING PRINCIPALS

3.3. MODEL OF SSI

3.4. COMPONENTS OF SSI

3.5. USE CASES



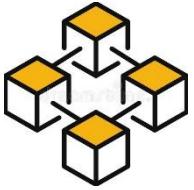
Self



Sovereign



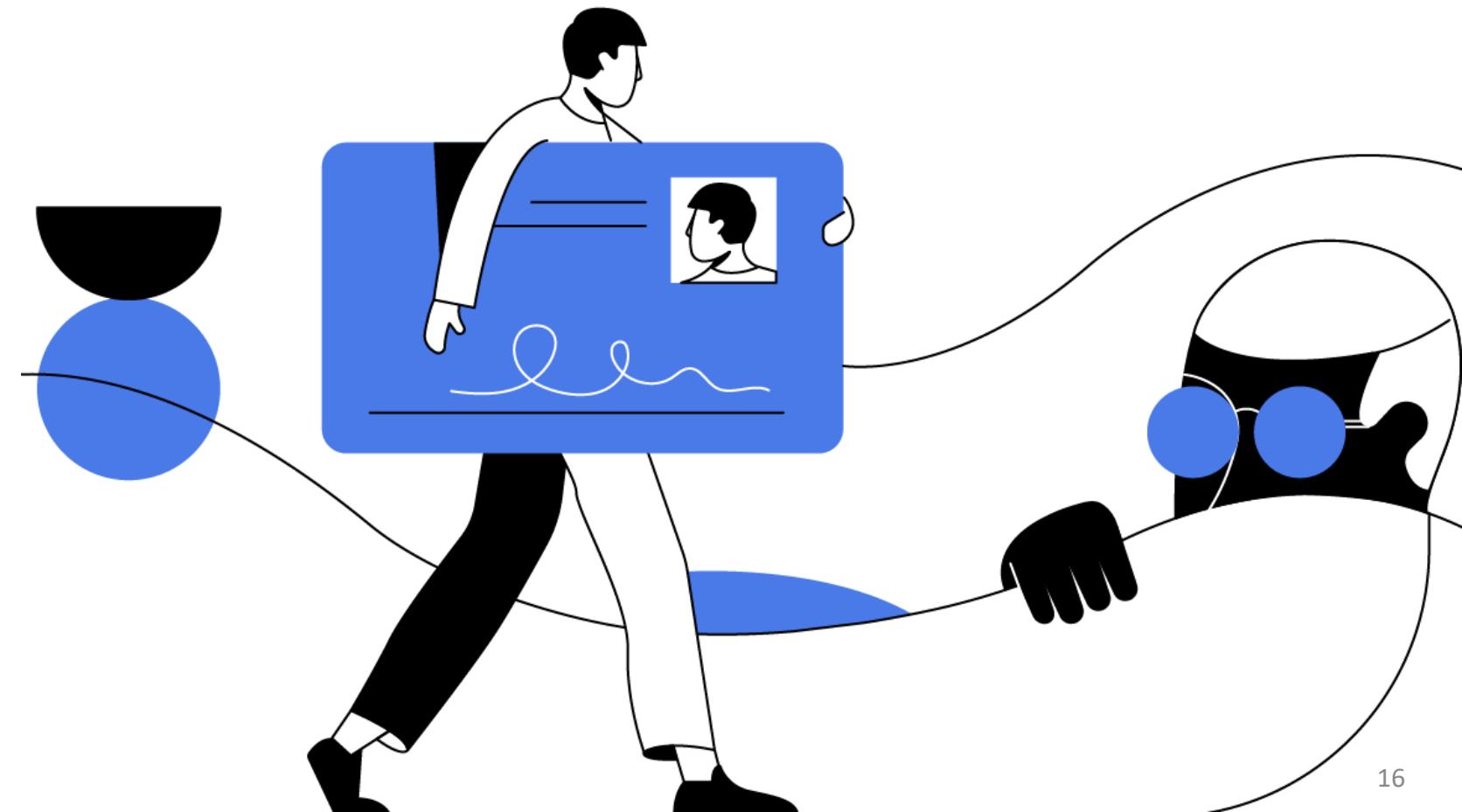
Identity

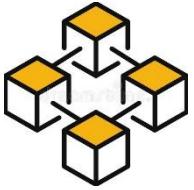


3.1. INTRODUCTION TO SSI

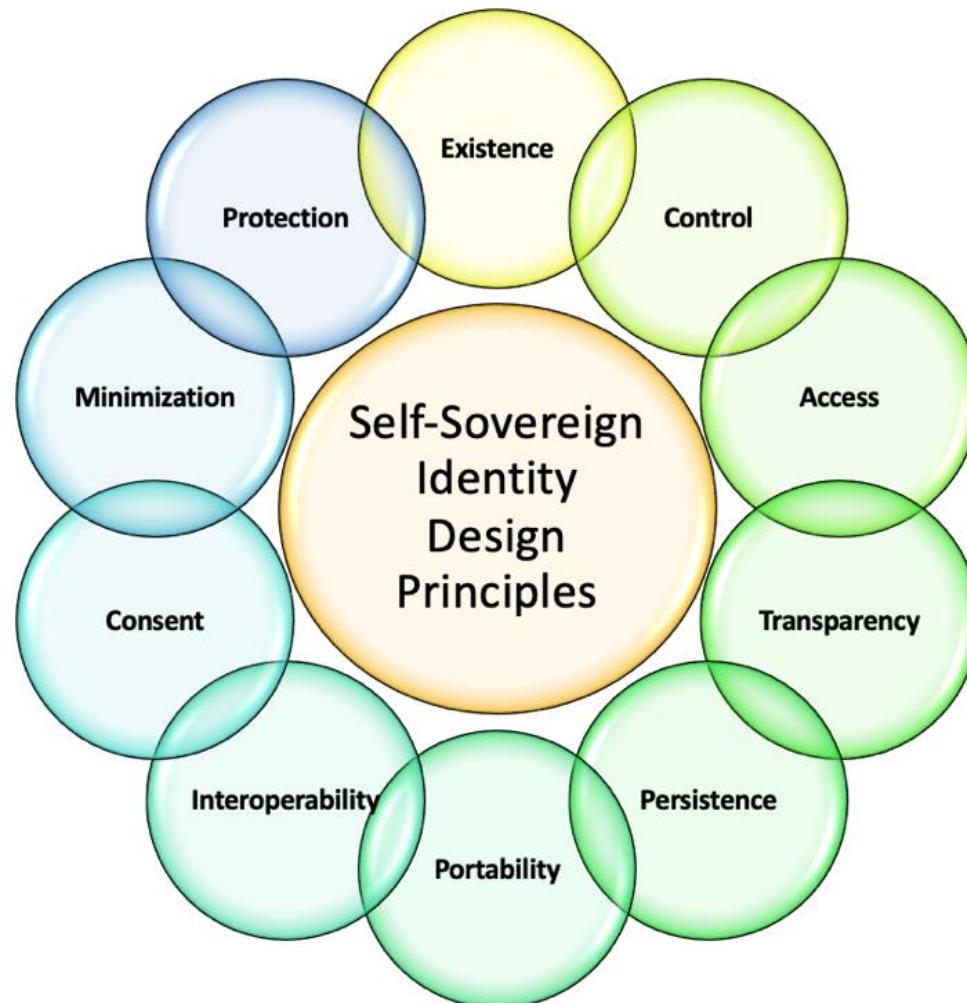
Self-sovereign identity (SSI):

- Is a digital identity.
- Individuals control identity data.

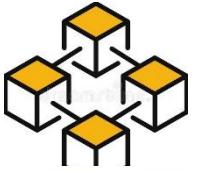




3.2. GUIDING PRINCIPALS



- **Existence:** Users must have an independent existence.
- **Control:** Users must control their identities.
- **Access:** Users must have access to their own data.
- **Transparency:** Systems and algorithms must be transparent.
- **Persistence:** Identities must be long-lived.
- **Portability:** Information and services about identity must be transportable.
- **Interoperability:** Identities should be as widely usable as possible.
- **Consent:** Users must agree to the use of their identity.
- **Minimization:** Disclosure of claims must be minimized.
- **Protection:** The rights of users must be protected.



3.2. GUIDING PRINCIPALS

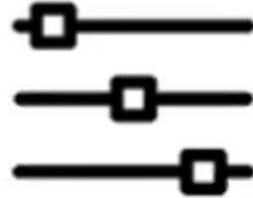
10 Principles of Self-Sovereign Identity*

Rebecca Lynn Jo



Existence

Users must have an independent existence.



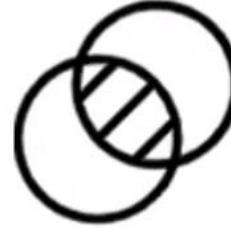
Control

Users must control their identities.



Access

Users must have access to their own data.



Transparency

Systems and algorithms must be transparent.



Persistence

Identities must be long-lived.



Portability

Information and services about identity must be transportable.



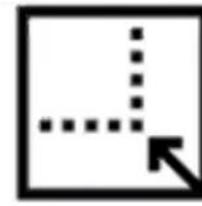
Interoperability

Identities should be as widely usable as possible.



Consent

Users must agree to the use of their identity.



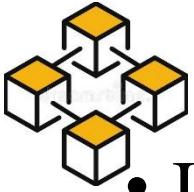
Minimization

Disclosure of claims must be minimized.



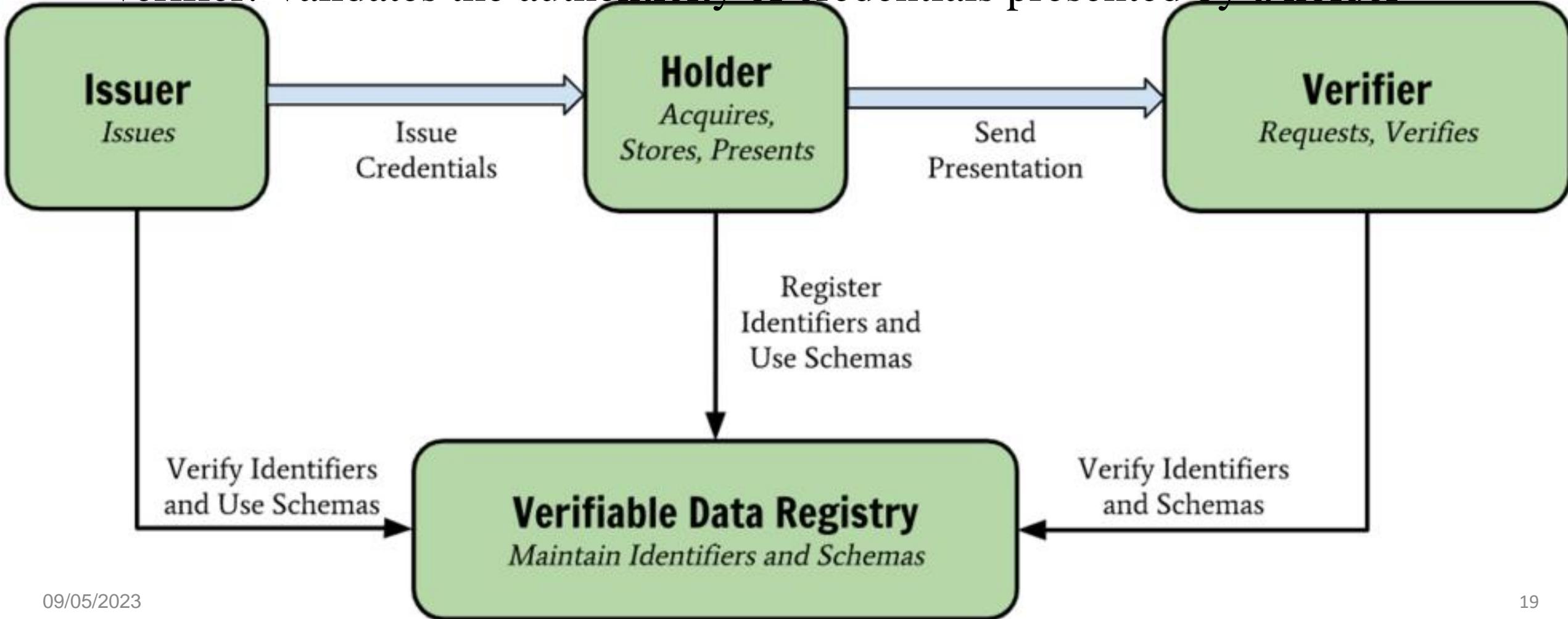
Protection

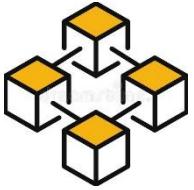
The rights of users must be protected.



3.3. MODEL OF SSI

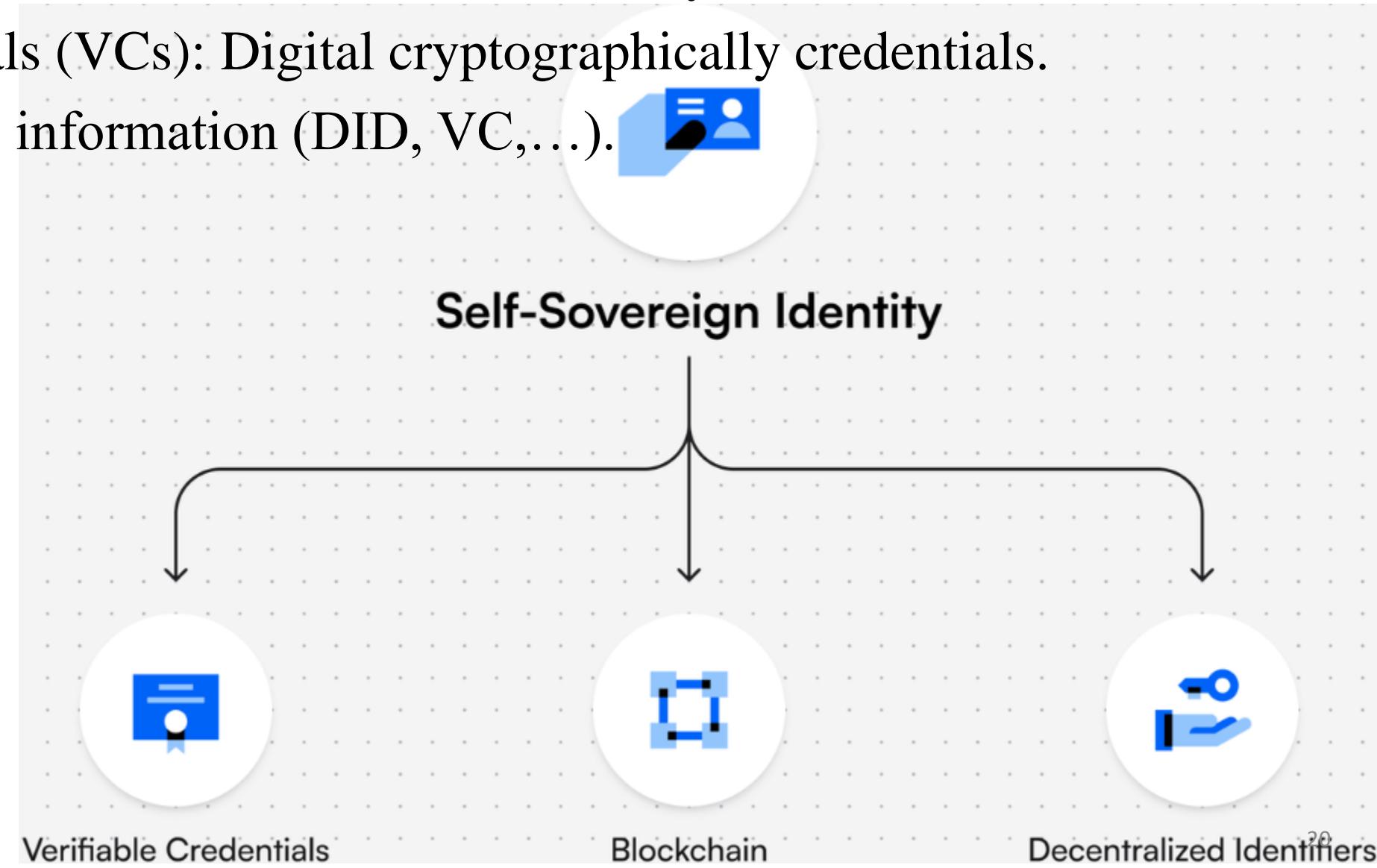
- Issuer: Generates credentials.
- Holder: individual/organization holds, present credential to proofs of identity
- Verifier: Validates the authenticity of credentials presented by a holder

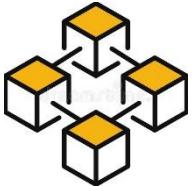




3.4. COMPONENTS OF SSI

- Decentralized Identifiers (DIDs): created, owned by user.
- Verifiable Credentials (VCs): Digital cryptographically credentials.
- Blockchain: records information (DID, VC,...).





4. DECENTRALIZED IDENTIFIER - DID

4.1. INTRODUCTION TO DID

4.2. DID ARCHITECTURE

4.3. DID & DID URL SYNTAX

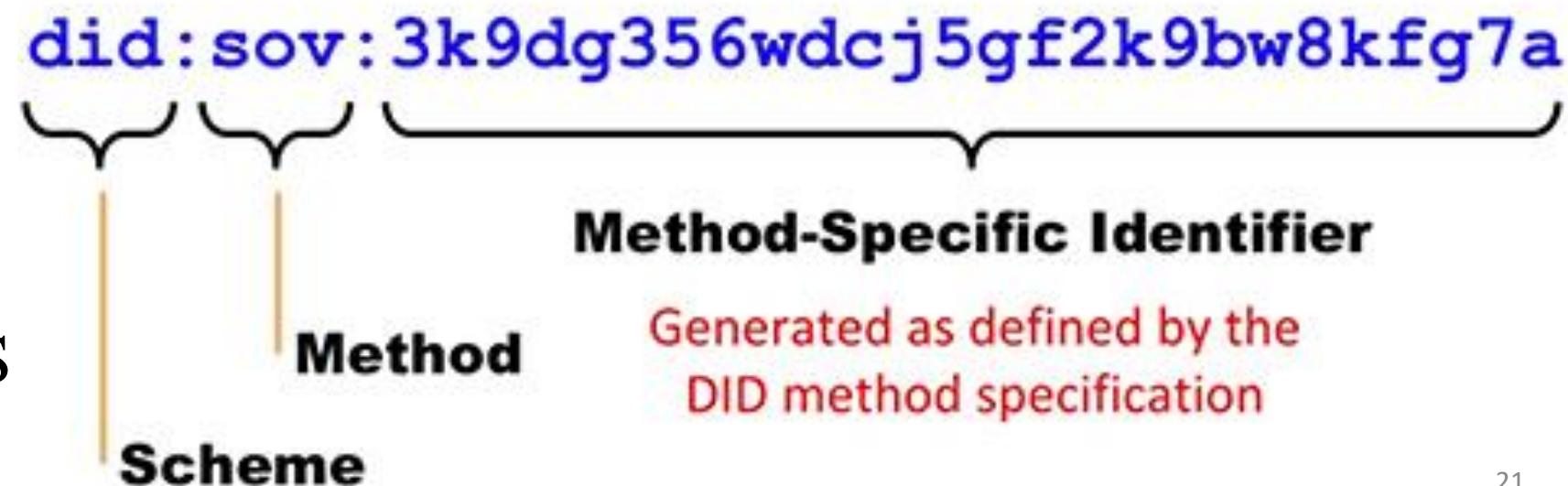
4.4. DID DOCUMENTS

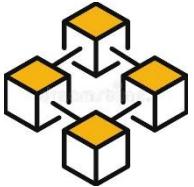
4.5. RESOLUTION

4.6. DID ACTIONS

4.7. DID USE CASES

DID Syntax (W3C)





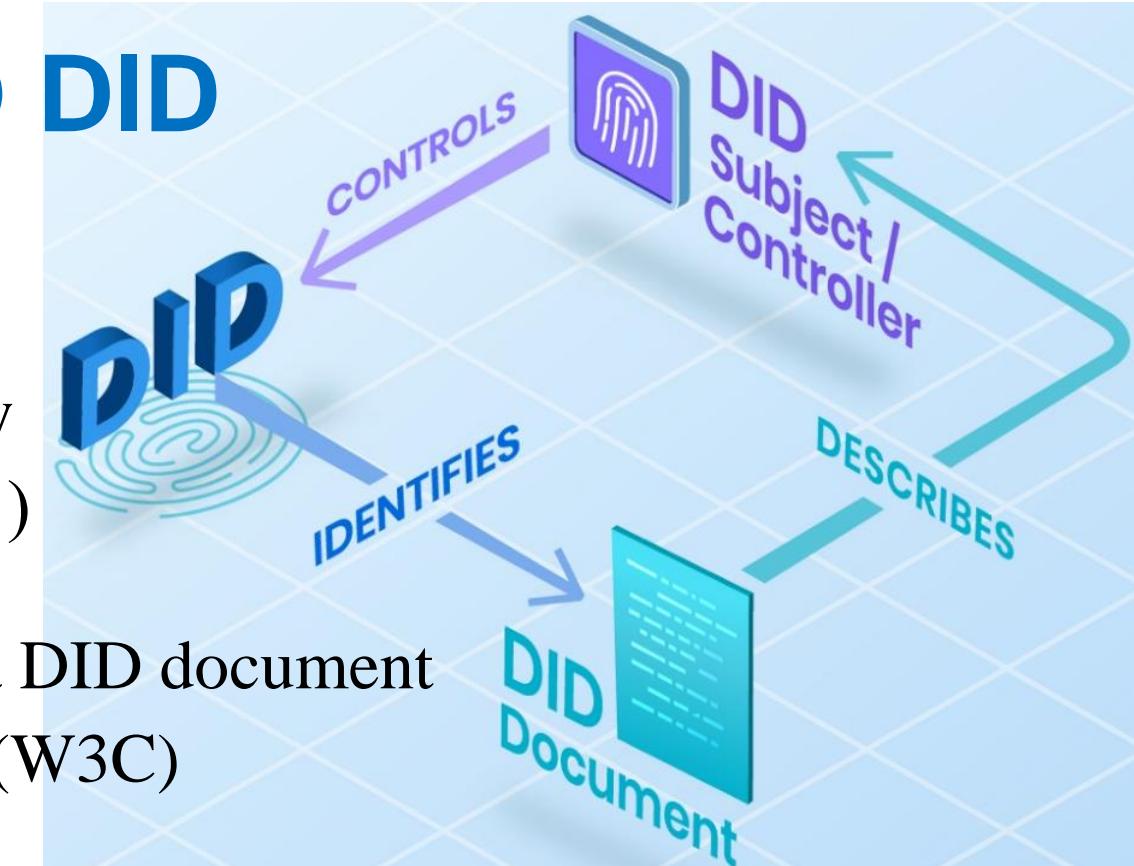
4.1. INTRODUCTION TO DID

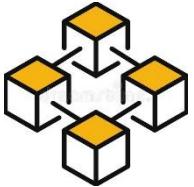
Decentralized Identifiers (DIDs):

- globally unique identifier
- enables verifiable, decentralized digital identity
- refers to subject (person, organization, thing,...) as controller of DID
- are URI (URN), associate a DID subject with a DID document
- standardized by World Wide Web Consortium (W3C)

Advantages of DID:

- Better privacy and control of data
- Higher security
- Protection against fraud
- No metadata collection





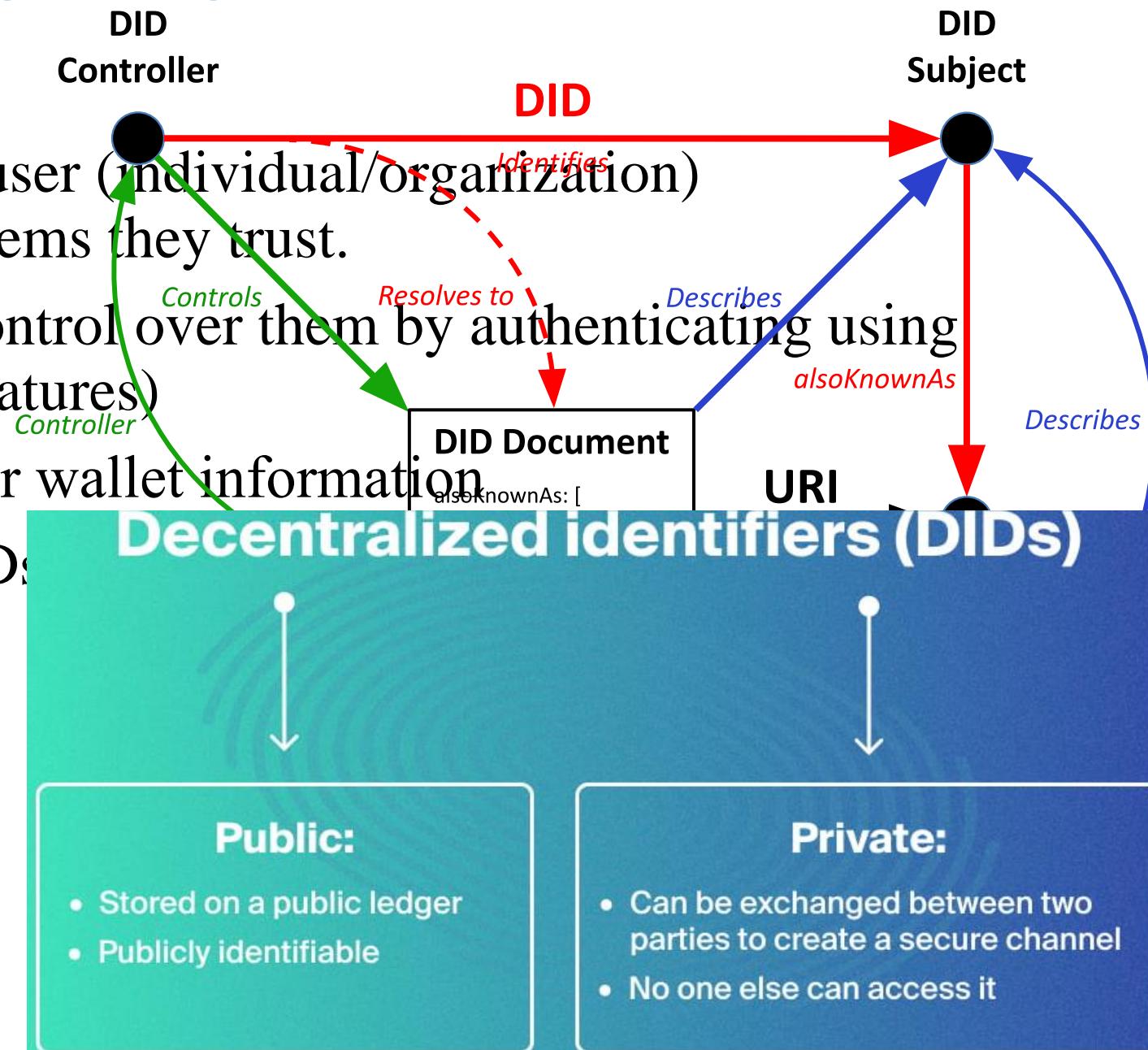
4.1. INTRODUCTION TO DID

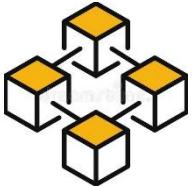
Key features of DIDs:

- Are created and managed by the user (individual/organization) without any third party using systems they trust.
- Allow owner to securely prove control over them by authenticating using cryptographic proofs (digital signatures)
- Don't contain any personal data or wallet information
- Entity/subject can have many DIDs

Types of DID:

- public
- private



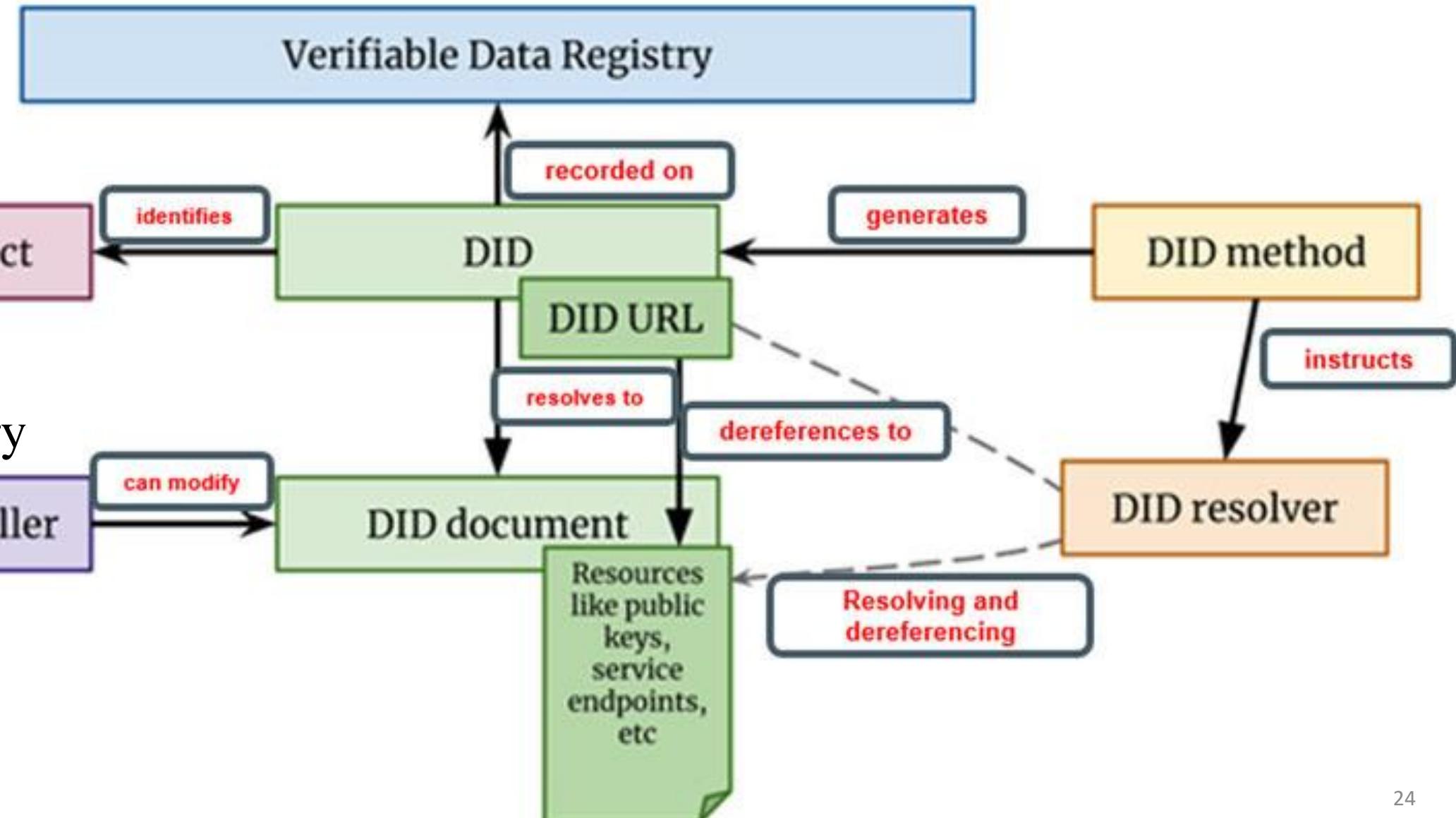


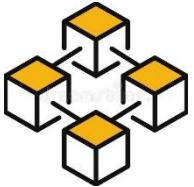
4.2. DID ARCHITECTURE

DID ecosystem:

- Subject
- Controller

- Document
- Data Registry
- Resolver
- Method

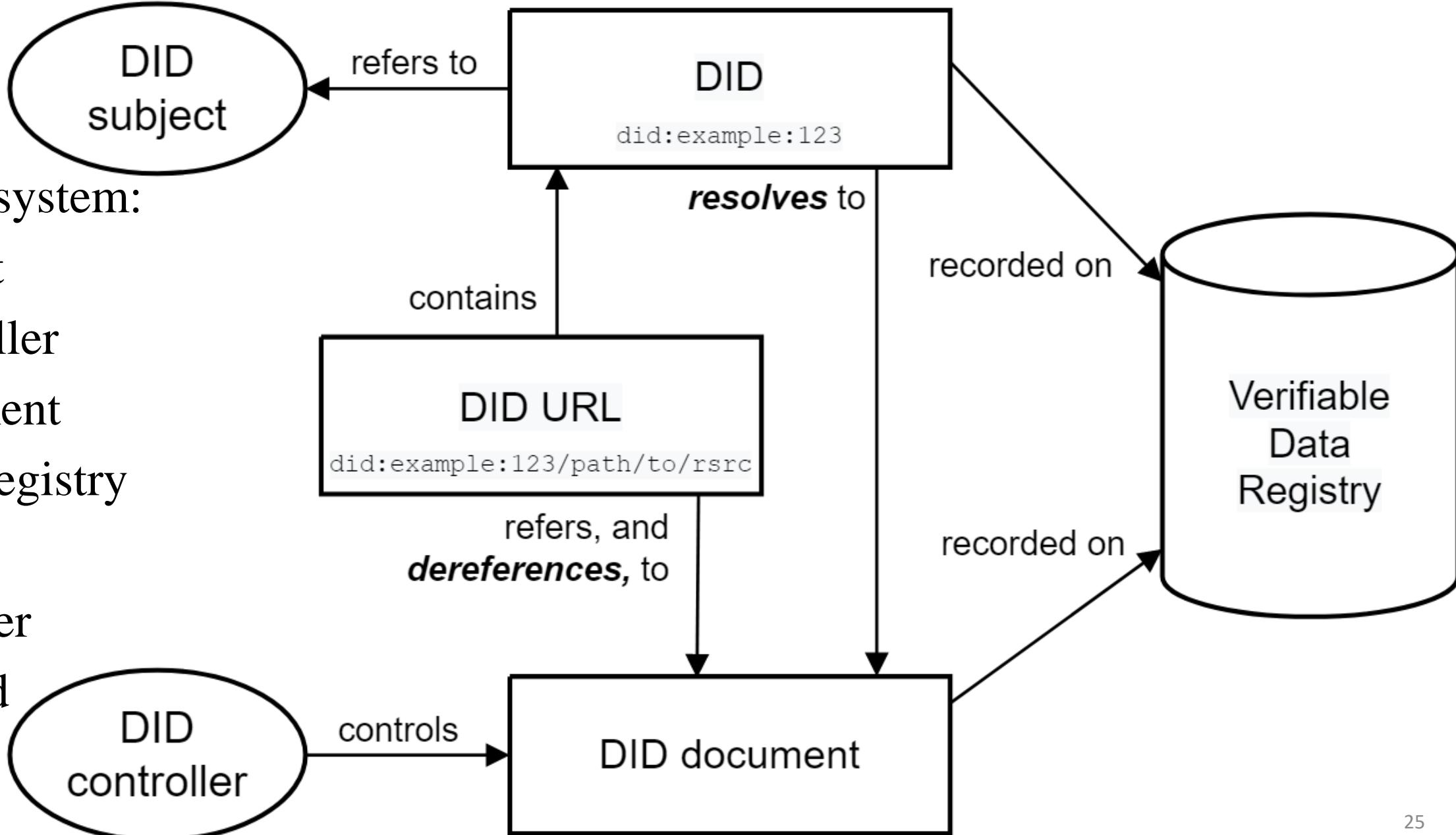


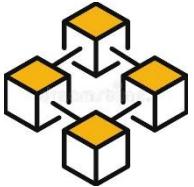


4.2. DID ARCHITECTURE

DID ecosystem:

- Subject
- Controller
- Document
- Data Registry
- Resolver
- Method

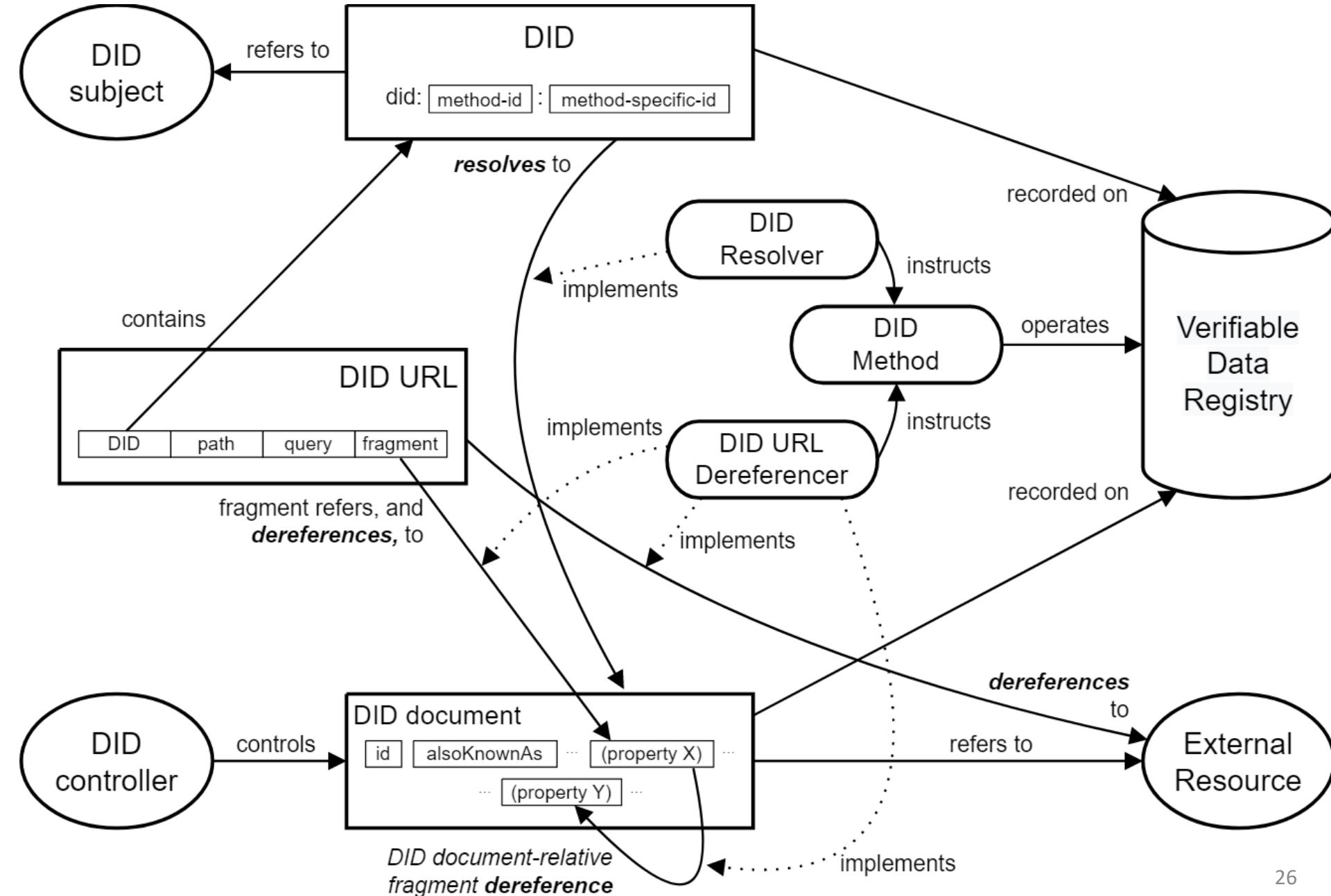


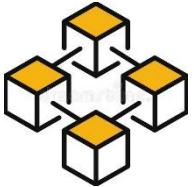


4.2. DID ARCHITECTURE

DID ecosystem:

- Subject
- Controller
- Document
- Data Registry
- Resolver
- Method



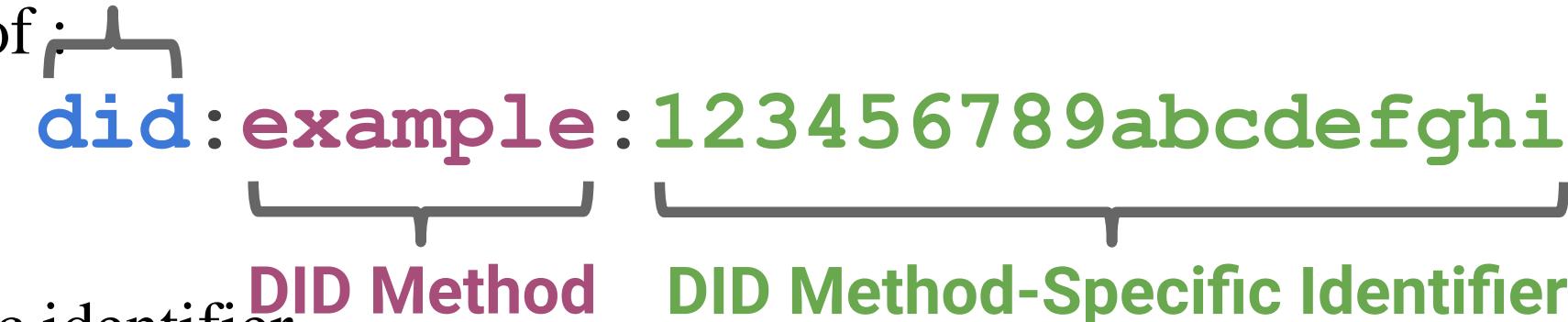


4.3. DID & DID URL SYNTAX

Scheme

DID is a URI composed of:

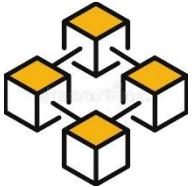
1. scheme identifier
2. method identifier
3. unique, method-specific identifier



Method	DID Prefix
Sovrin	did:sov:
Bitcoin	did:btcr:
uPort	did:uport:
VeresOne	did:v1:
IPFS	did:ipid:
IPDB	did:ipdb:
Blockstack	did:stack

DID URL extends DID to incorporate other components:
path, query, fragment
to locate a resource inside/outside DID document.

did-url = did path-abempty ["?" query] ["#" fragment]



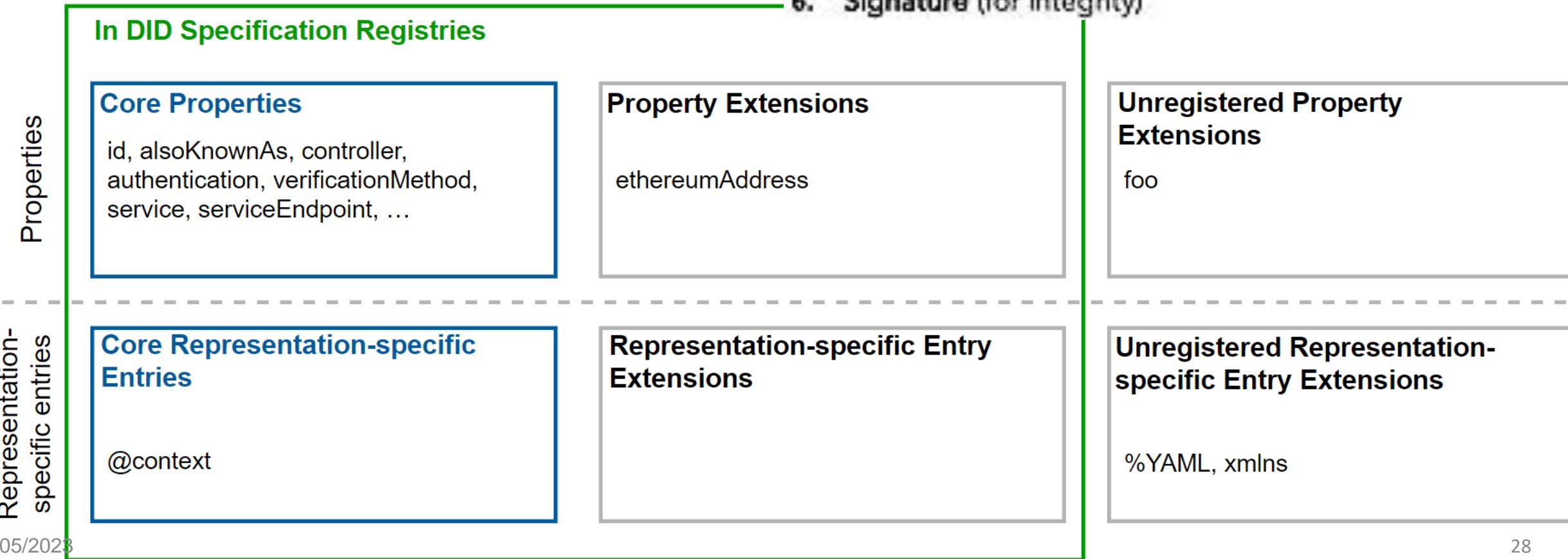
4.4. DID DOCUMENTS

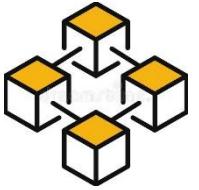
DID document consists of map of entries, contains at least two classes of entries

- Core Properties
- Representation-specific entries (Representations)

Entries in the DID Document map

- The standard elements of a DID doc
1. DID (for self-description)
 2. Set of public keys (for verification)
 3. Set of auth methods (for authentication)
 4. Set of service endpoints (for interaction)
 5. Timestamp (for audit history)
 6. Signature (for integrity)

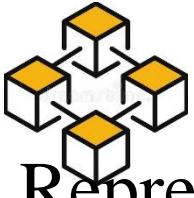




4.4. DID DOCUMENTS

Core Properties:

Property	Required?	Value constraints
id	yes	A string that conforms to the rules in 3.1 DID Syntax.
alsoKnownAs	no	A set of strings that conform to the rules of [RFC3986] for URIs.
controller	no	A string or a set of strings that conform to the rules in 3.1 DID Syntax.
verificationMethod	no	A set of Verification Method maps that conform to the rules in Verification Method properties.
authentication	no	
assertionMethod	no	
keyAgreement	no	
capabilityInvocation	no	A set of either Verification Method maps that conform to the rules in Verification Method properties) or strings that conform to the rules in 3.2 DID URL Syntax.
capabilityDelegation	no	
service 09/05/2023	no	A set of Service Endpoint maps that conform to the rules in Service properties.



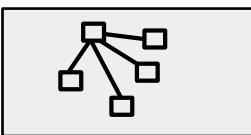
4.4. DID DOCUMENTS

Representations: serialization of DID document

resolve()

- Production (serializing): data model to representation

VERIFIABLE
DATA REGISTRY



- Consumption: representation to data model

resolveRepresentation()

Core Properties

```
«[  
  "id" → "example:123",  
  "verificationMethod" → « «[  
    "id": "did:example:123#keys-1",  
    "controller": "did:example:123",  
    "type": "Ed25519VerificationKey2018",  
    "publicKeyBase58": "H3C2AVvLMv6gmMNam3uVA"  
  ]» »,  
  "authentication" → «  
    "did:example:123#keys-1"  
  »  
]»
```

Core Representation-specific Entries (JSON-LD)

```
«[ "@context" → "https://www.w3.org/ns/did/v1" ]»
```

consume

produce

consume

produce

consume

```
{  
  "@context": ["https://www.w3.org/ns/did/v1"],  
  "id": "did:example:123",  
  "verificationMethod": [  
    {"id": "did:example:123#keys-1",  
     "controller": "did:example:123",  
     "type": "Ed25519VerificationKey2018",  
     "publicKeyBase58": "..."}],  
  "authentication": [  
    {"did": "did:example:123#keys-1"}]
```

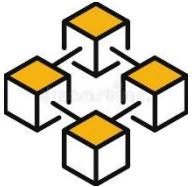
application/did+ld+json

```
{  
  "id": "did:example:123",  
  "verificationMethod": [  
    {"id": "did:example:123#keys-1",  
     "controller": "did:example:123",  
     "type": "Ed25519VerificationKey2018",  
     "publicKeyBase58": "..."}],  
  "authentication": [  
    {"did": "did:example:123#keys-1"}]
```

application/did+json

```
62a3 6469 646f 6469 653a 6178 706d  
656c 313a 3332 7672 7265 6669 6369  
7461 6f69 4d6e 7465 6f68 8164 62a4  
6469 6476 6469 653a 6178 706d 656c  
313a 3332 6b23 7965 2d73 6a31 6f63  
746e 6f72 6c6c 7265 646f 6469 653a  
...
```

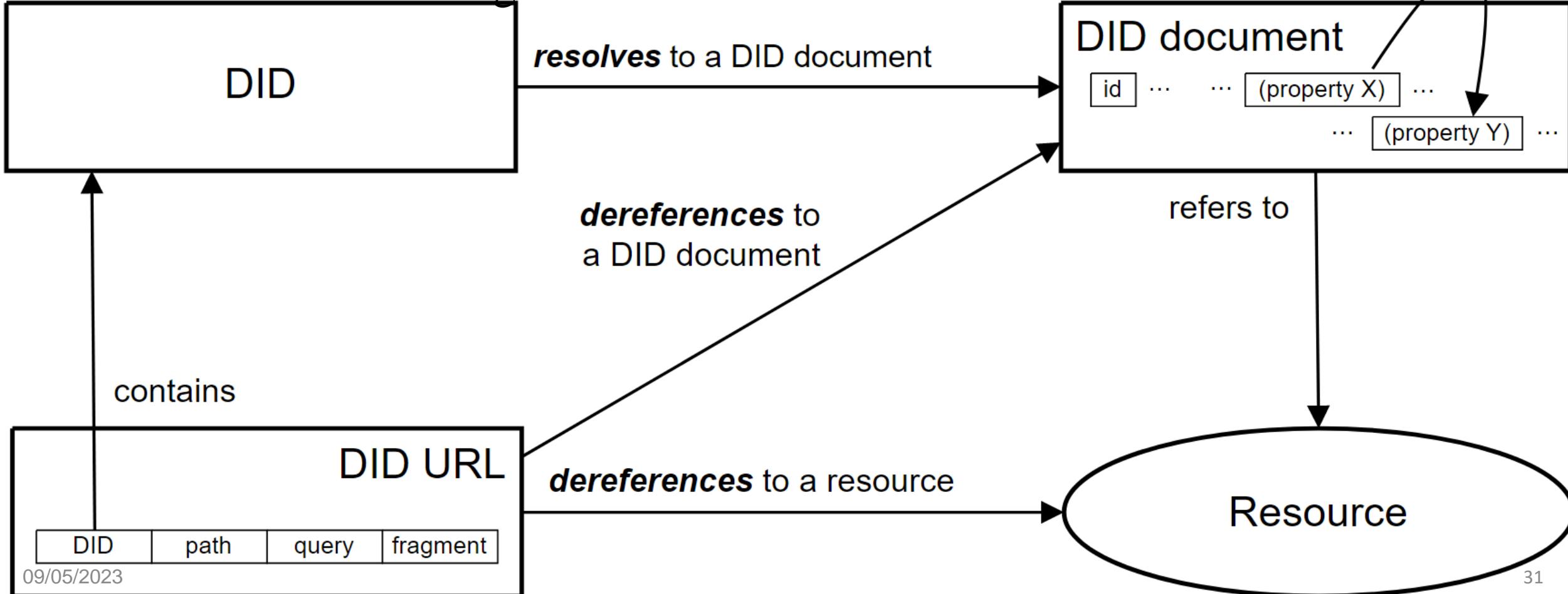
application/did+cbor

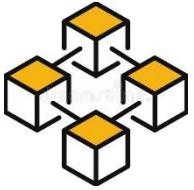


4.5. RESOLUTION

Resolution:

- DID Resolution: DID into DID document by Read
- DID URL Dereferencing: DID URL into resource





4.6. DID ACTIONS

- Create

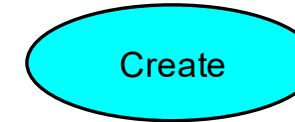


Subject



Controller

Create



Create

- Delete



DID



Register DID

Registry



Deactivate



Controller

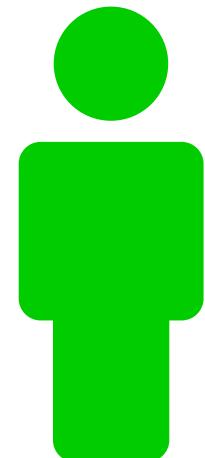
Deactivate DID



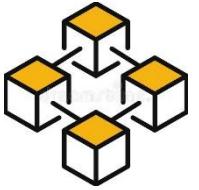
Registry

Resolve DID

'Inactive'

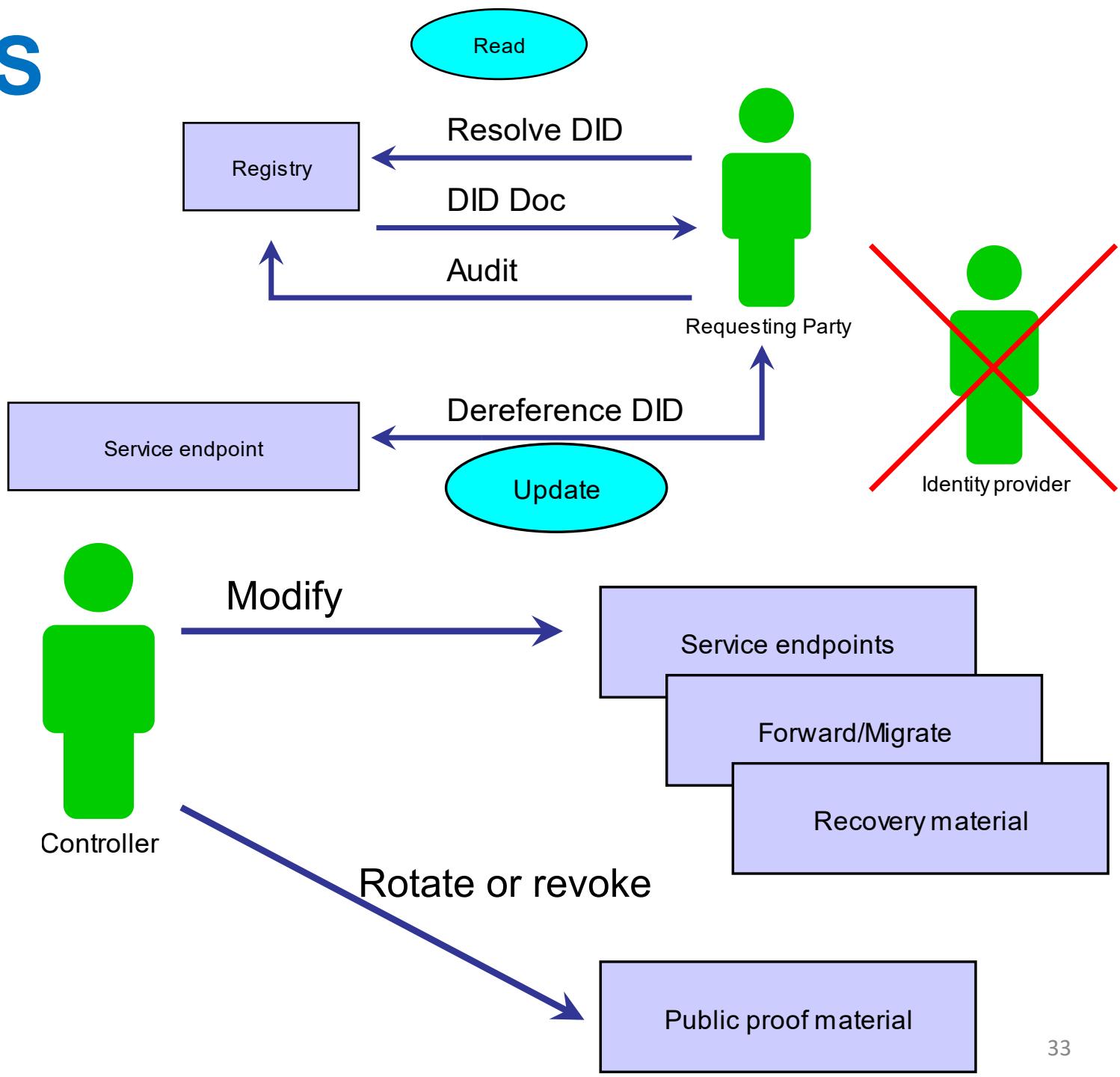


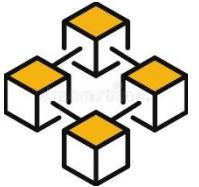
Requesting party



4.6. DID ACTIONS

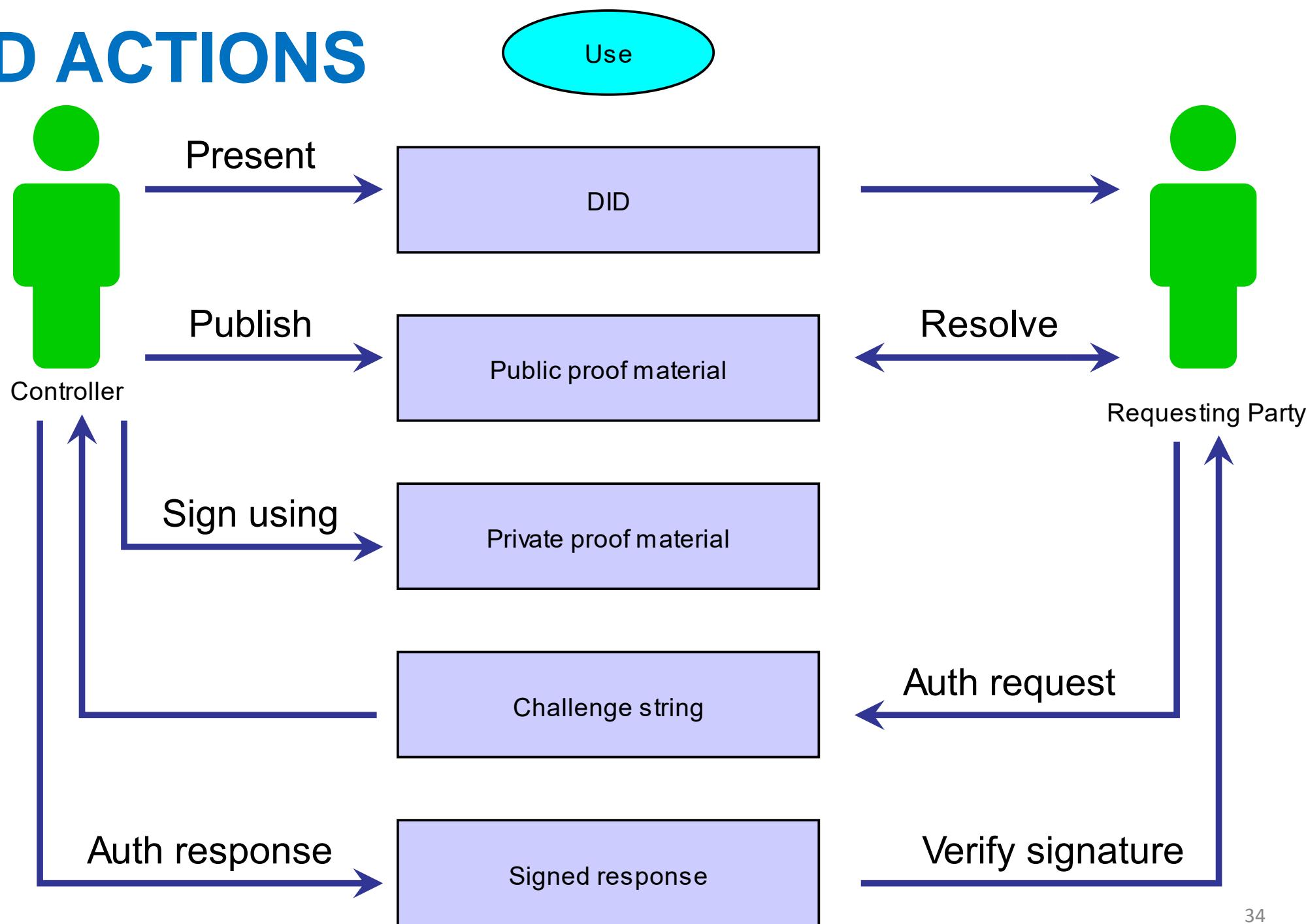
- Read

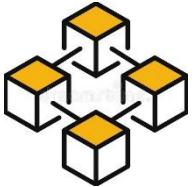




4.6. DID ACTIONS

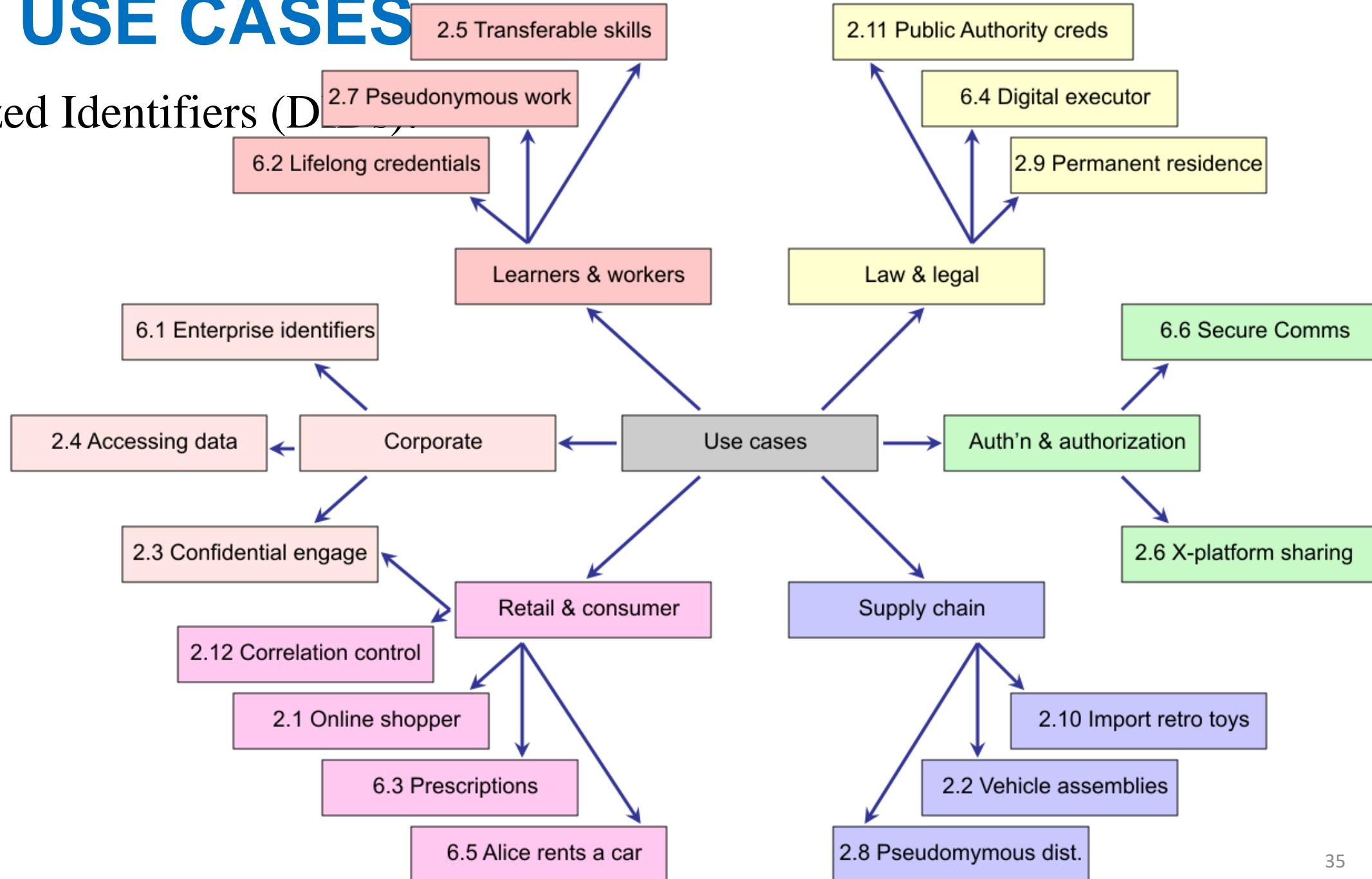
- Use

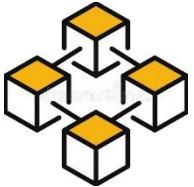




4.7. USE CASES

Decentralized Identifiers (D)





5. VERIFIABLE CREDENTIALS

5.1. INTRODUCTIONS

5.2. WORKFLOW OF VC

5.3. COMPONENTS OF VC

5.4. USER TASKS OF VC

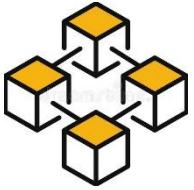
5.5. VERIFIABLE PRESENTATION

5.6. PRESENTATIONS USING DERIVED CREDENTIALS

5.7. USE CASES

Verifiable Credentials

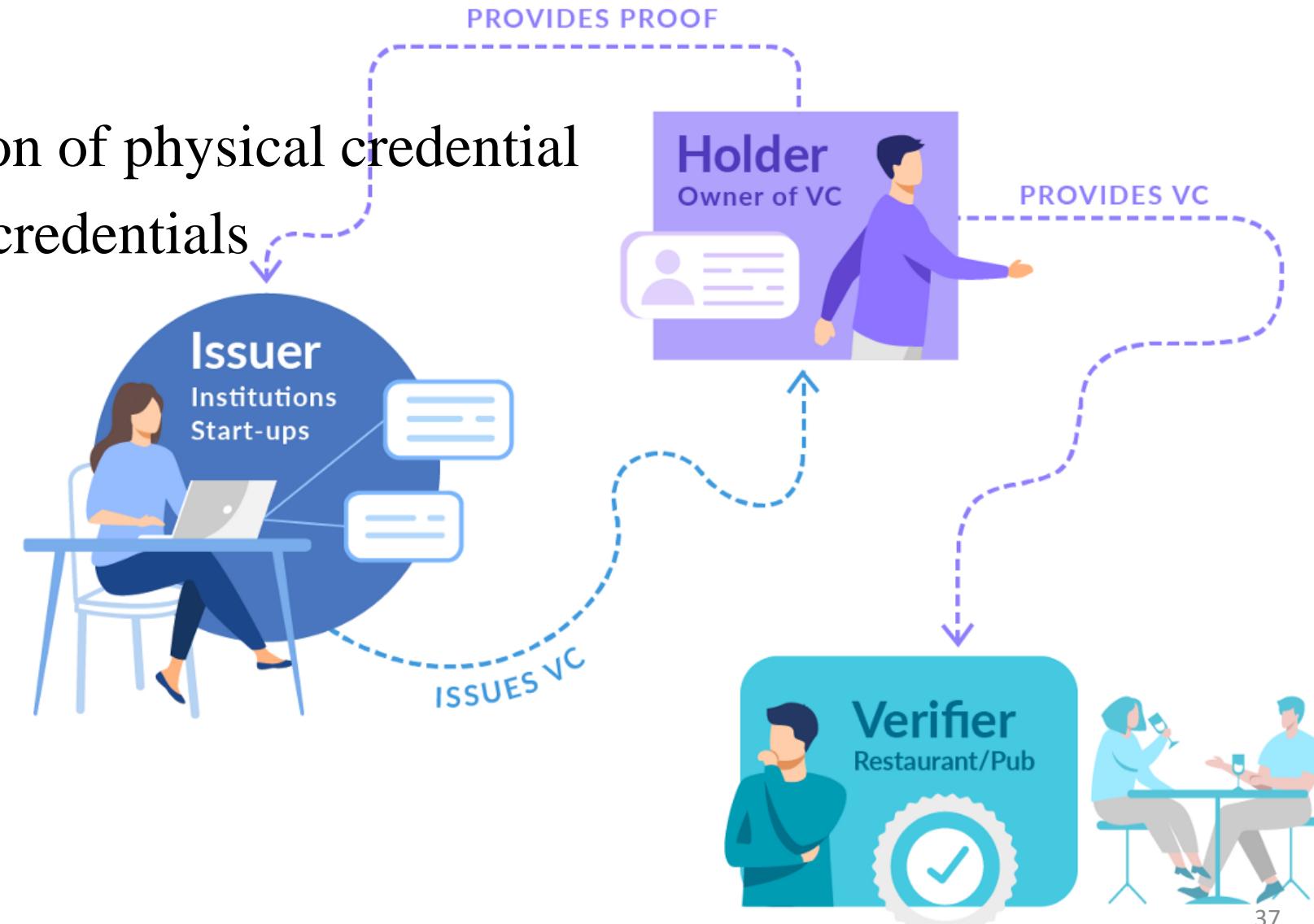


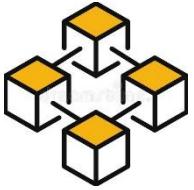


5.1. INTRODUCTIONS

Verifiable Credentials (VC)

- are a digital credentials
- can represent all information of physical credential
- cryptographically secured credentials
- present for verification
- created by Holder



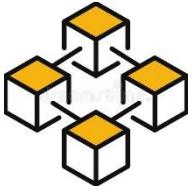


5.1. INTRODUCTIONS

Verifiable Credentials properties:

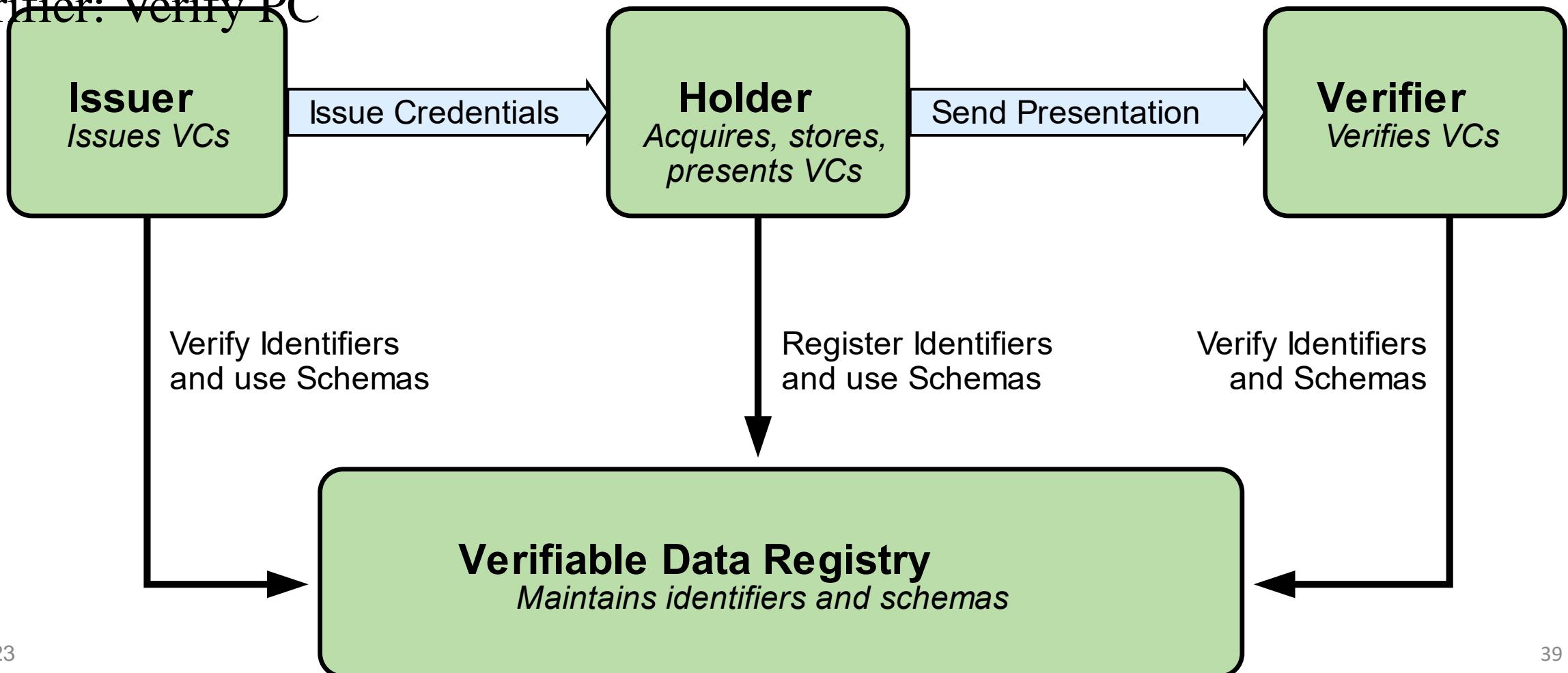
- Portable: send the credential online, move wallets between different providers
- Verifiable: issuer, holder, information, revoked.
- Private: zero-knowledge proofs choose information to share.
- Tamperproof: can't be altered or forged.

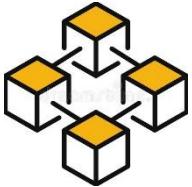




5.2. VERIFIABLE CREDENTIALS WORKFLOW

- Issuer: Create VC - Put subject's claims
- Holder: Create PC - Use one/more VCs (data)
- Verifier: Verify PC

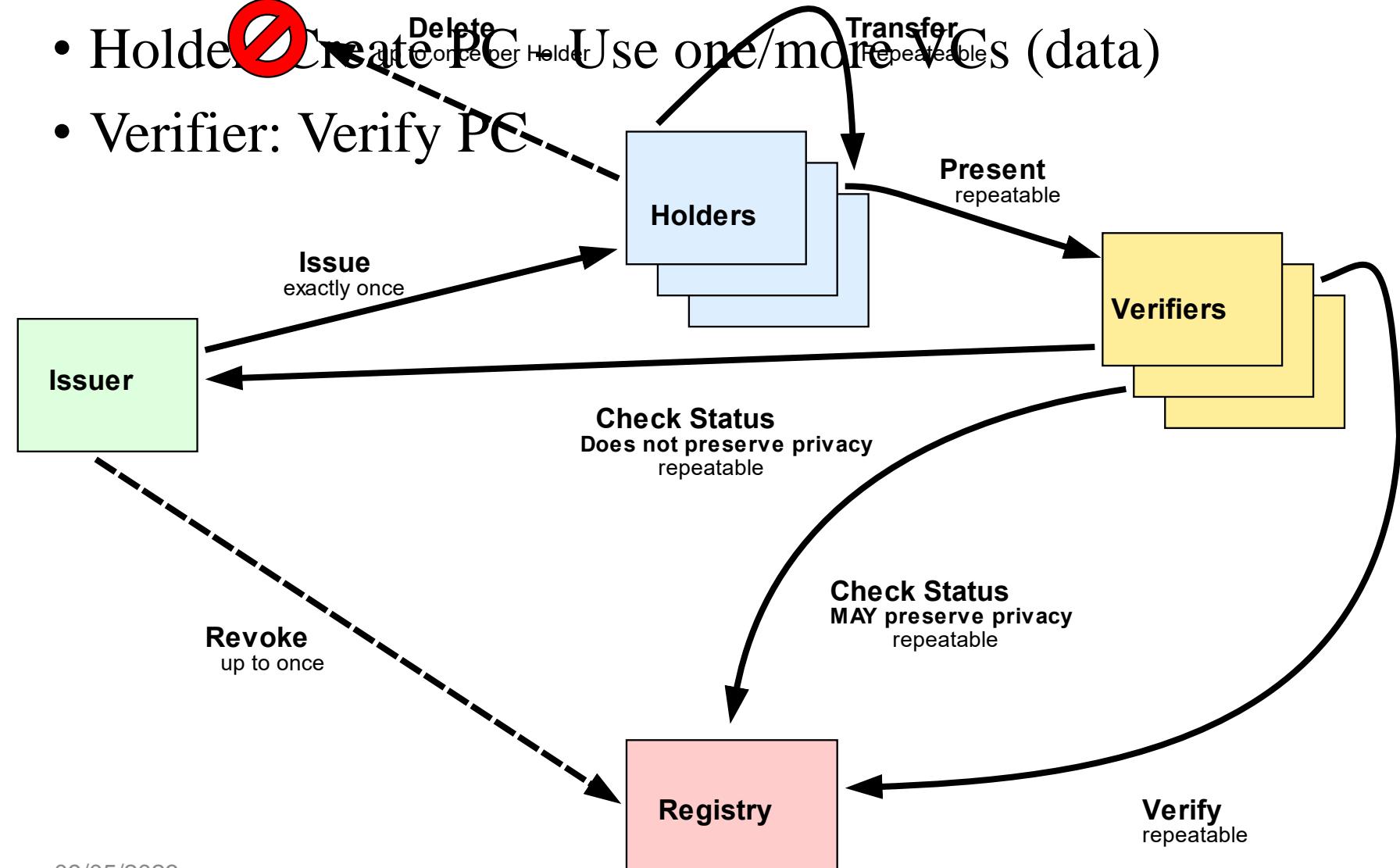


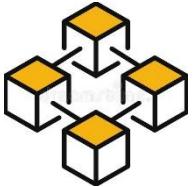


5.2. VERIFIABLE CREDENTIALS WORKFLOW

Life of a single Verifiable Credential

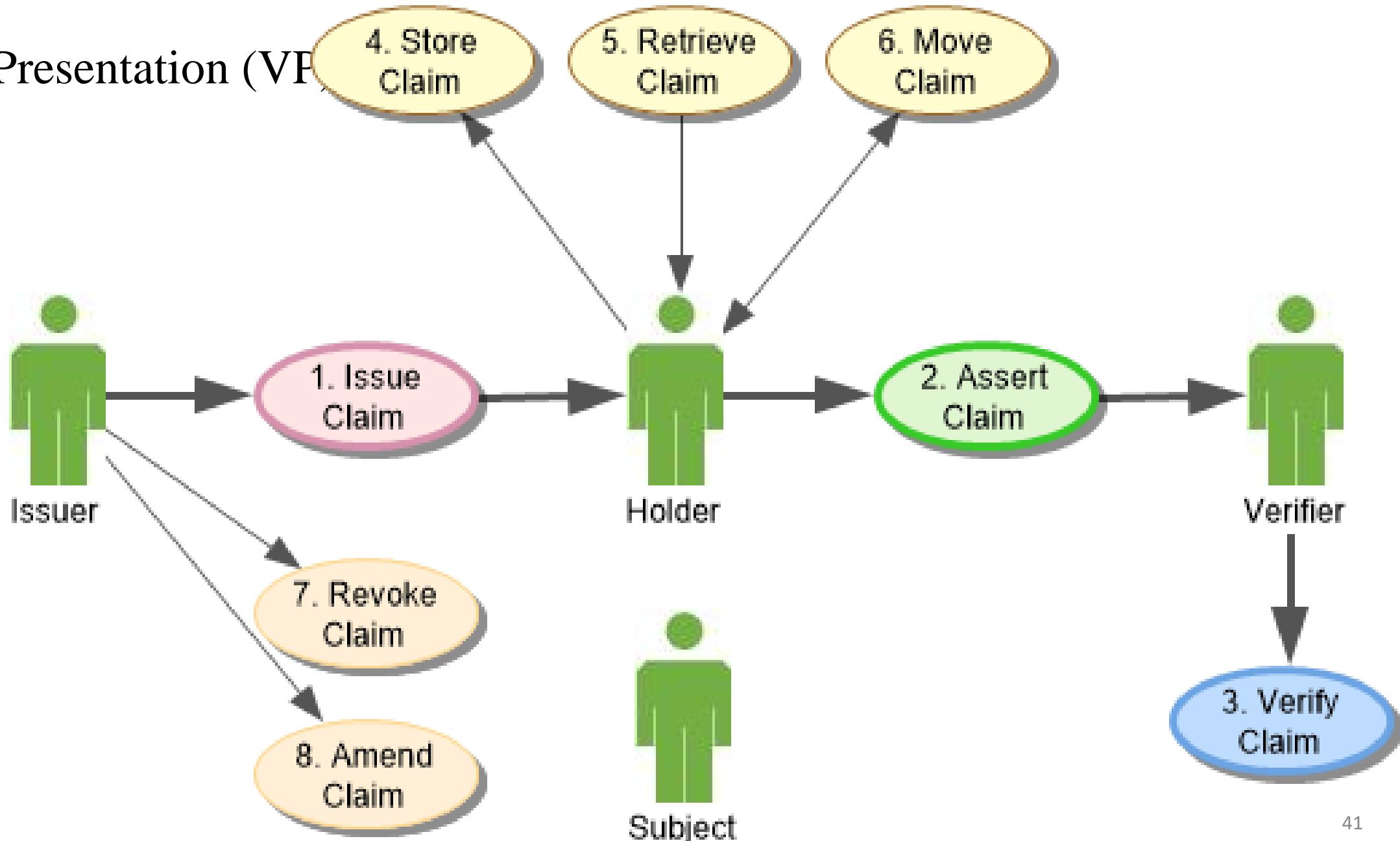
- Issuer: Create VC - Put subject's claims
- Holder ~~Create VC~~ Delete, Reconcile Holder Use one/more VCs (data)
- Verifier: Verify VC

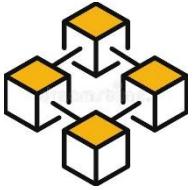




5.2. VERIFIABLE CREDENTIALS WORKFLOW

Verifiable Presentation (VP)





5.3. VERIFIABLE CREDENTIAL'S COMPONENTS

Verifiable Credential components:

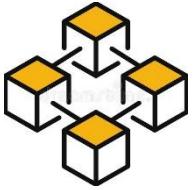
- set of tamper-evident (sign) claims and metadata.
- Proof: cryptographically prove who issued it.

Verifiable Credential

Credential Metadata

Claim(s)

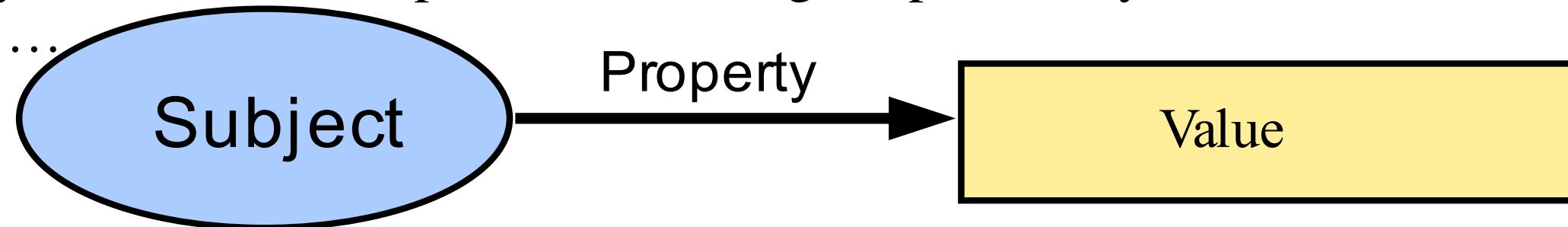
Proof(s)



5.3. VERIFIABLE CREDENTIAL'S COMPONENTS

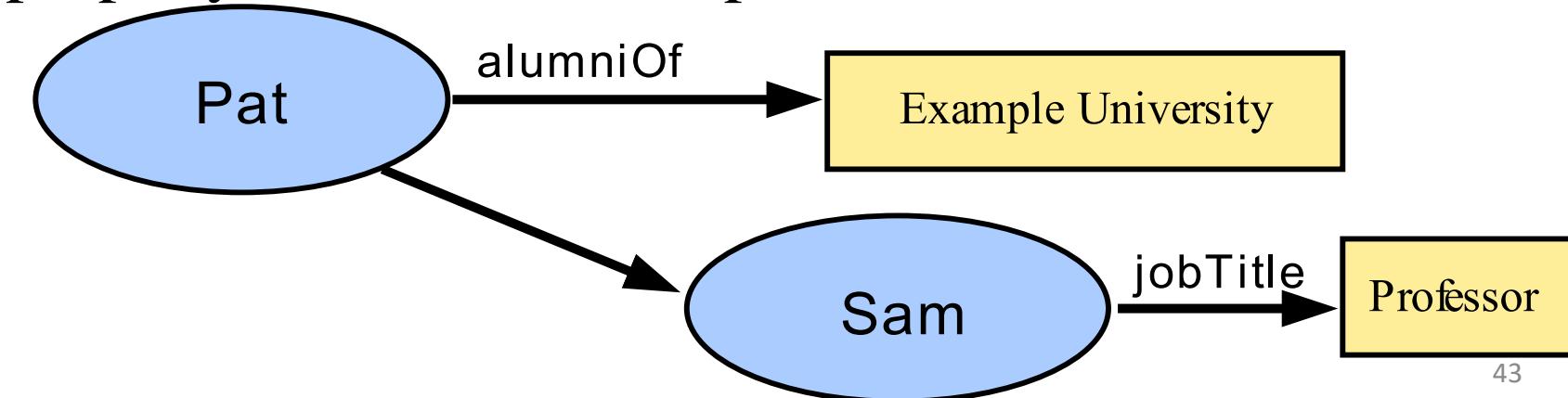
Metadata:

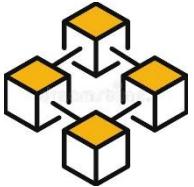
- describe properties of the credential
- issuer, expiry date and time, representative image, a public key, revocation mechanism, ...



Claim:

- statement about a subject
- expressed using subject- property-value relationships





5.3. VERIFIABLE CREDENTIAL'S COMPONENTS

Credential Graph

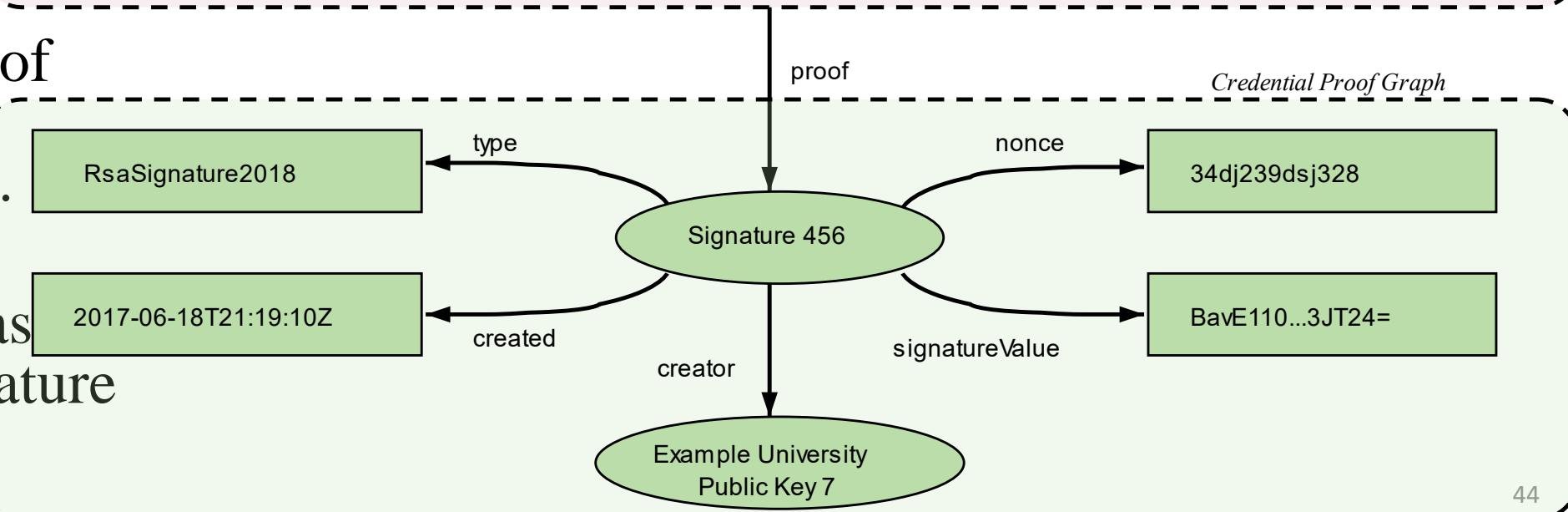
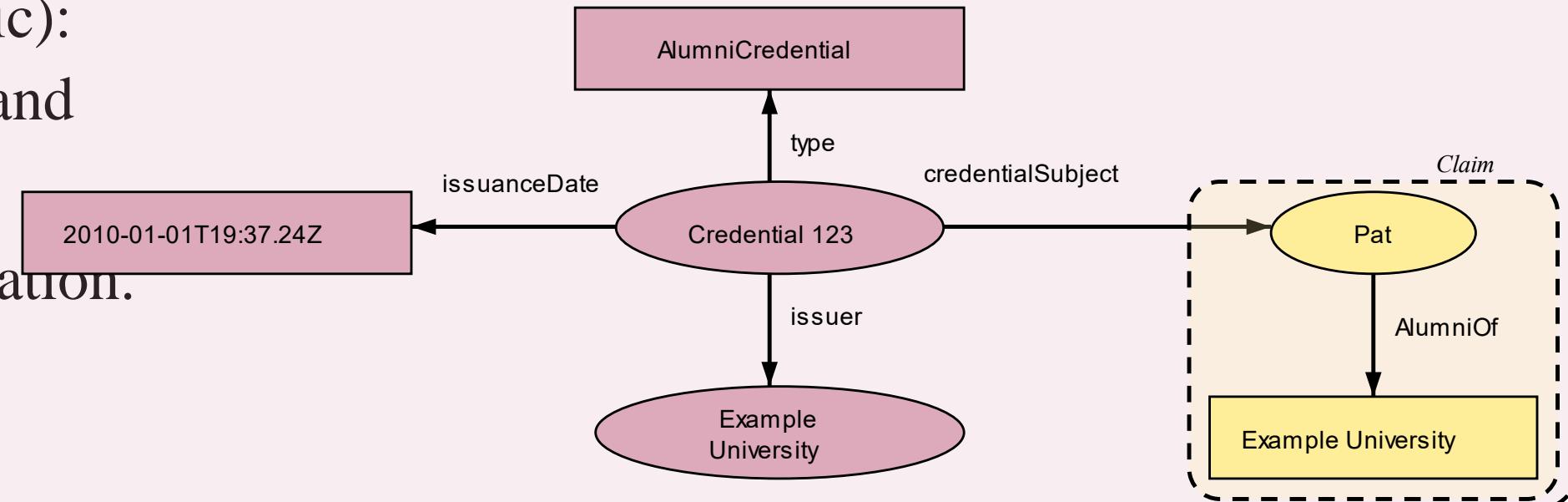
Proof (cryptographic):

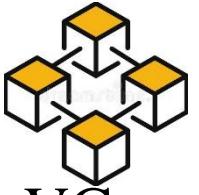
- detect tampering and verify the authorship of credential/presentation.

Classes of proof mechanisms:

- External proof: wraps expression of data model as JSON Web Token.

- Embedded proof: included in data as Linked Data Signature





5.3. VERIFIABLE CREDENTIAL'S COMPONENTS

Context

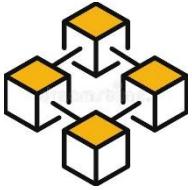
VC content:

- @context: URI, exchange data
 - "id": "0892f680-6aeb-11eb-9bcf-f10d8993fde7",
"type": [
 "VerifiableCredential",
 "UniversityDegreeCredential"
],
"issuer": {
 "id": "did:example:76e12ec712ebc6f1c221ebfeb1f",
 "name": "Acme University"
},
"issuanceDate": "2021-05-11T23:09:06.803Z",
"credentialSubject": {
 "id": "did:example:ebfeb1f712ebc6f1c276e12ec21",
 "degree": {
 "type": "BachelorDegree",
 "name": "Bachelor of Science"
 }
},
"proof": {
 "type": "Ed25519Signature2018",
 "created": "2021-05-17T15:25:26Z",
 "jws": "eyJhbGciOiJFZERTQYjY0Ii19..nlcAA",
 "proofPurpose": "assertionMethod",
 "verificationMethod": "https://pathToIssuerPublicKey"
}
- id: URI, refer to object
- type: URI/map to context
- credentialSubject: claims one/more subjects
- issuer: claim issuer, URI/id
- issuanceDate:
- expiration:
- status: information (id, type)
- proofs (Signatures): detect tampering, verify the authorship

Metadata

Claims

Proofs



5.4. VERIFIABLE PRESENTATION

Verifiable Presentation (VP):

- expresses data from one/more verifiable credentials
- is packaged so that the authorship of the data is verifiable.

Presentation's data:

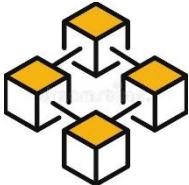
- is about the same subject
- be issued by multiple issuers
- information aggregation

Verifiable Presentation

Presentation Metadata

Verifiable Credential(s)

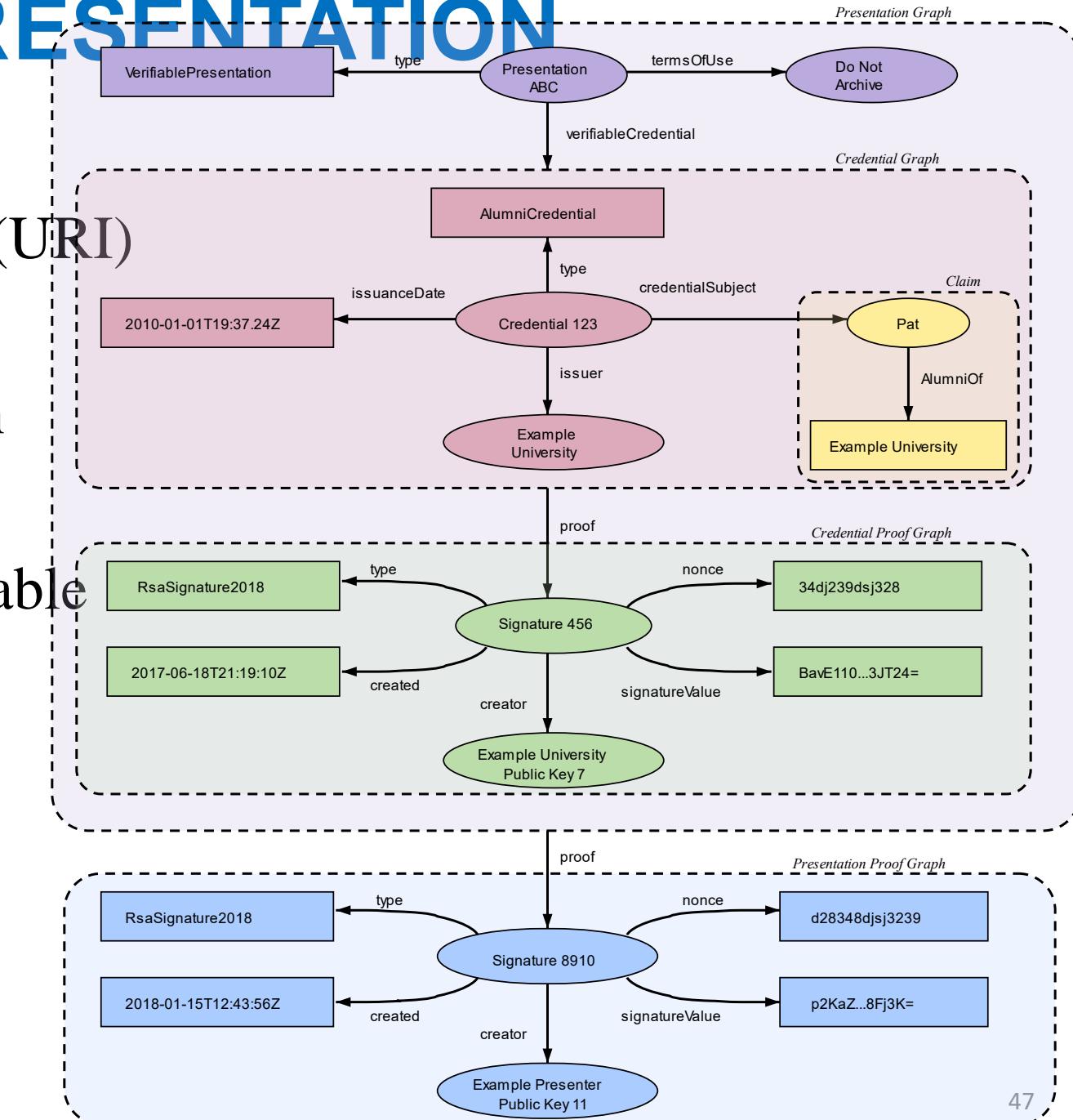
Proof(s)

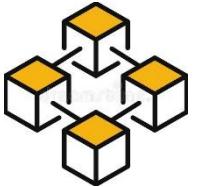


5.4. VERIFIABLE PRESENTATION

Verifiable Presentation content:

- id: unique identifier for presentation (URI)
- type: type of presentation, VP
- holder: URI for subject's presentation
- verifiableCredential: one/more verifiable credentials, or data derived from verifiable credentials
- proof: value of the proof

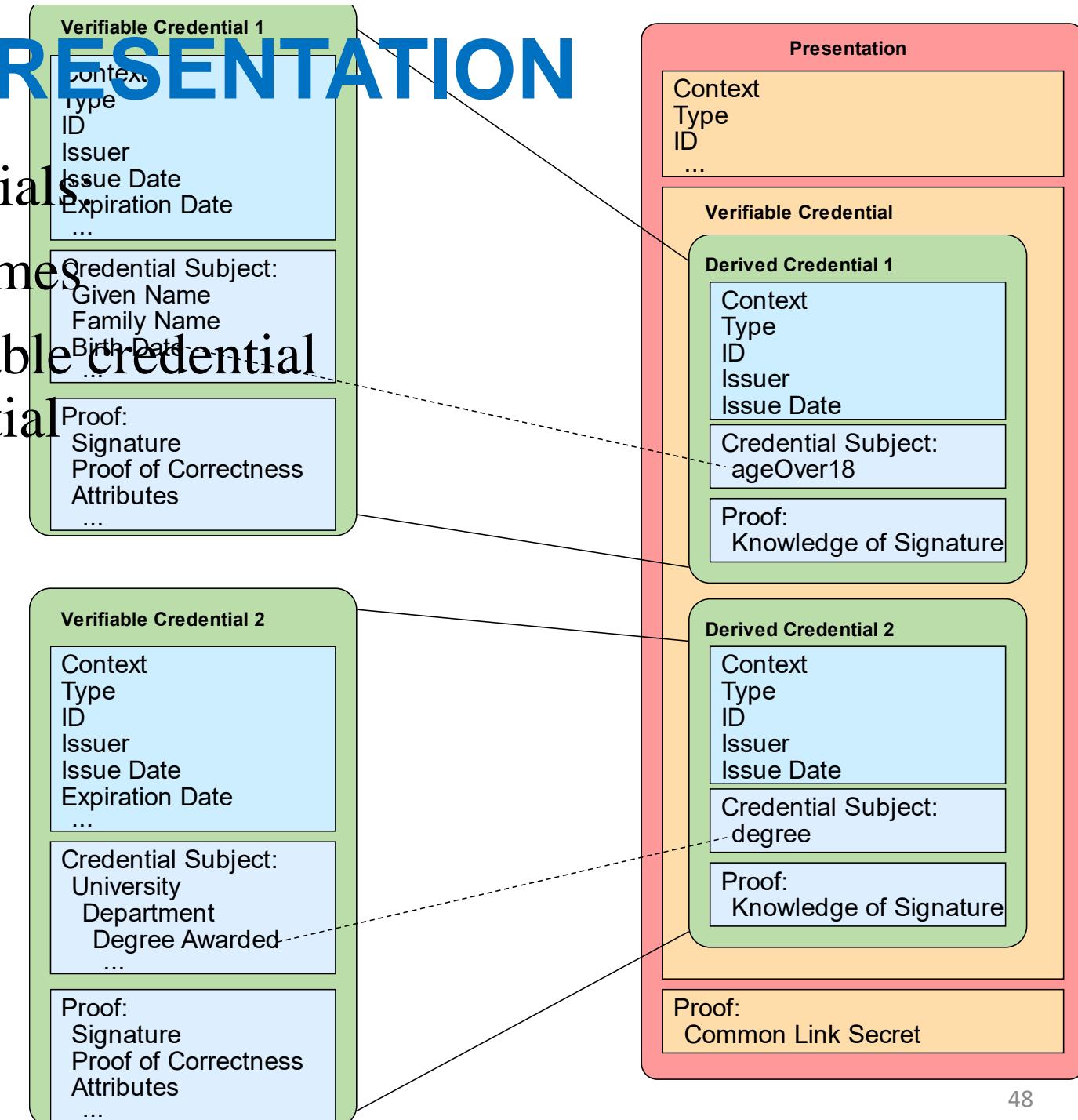


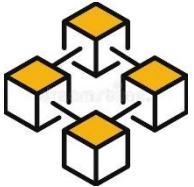


5.5. VERIFIABLE PRESENTATION

Presentations Using Derived Credentials

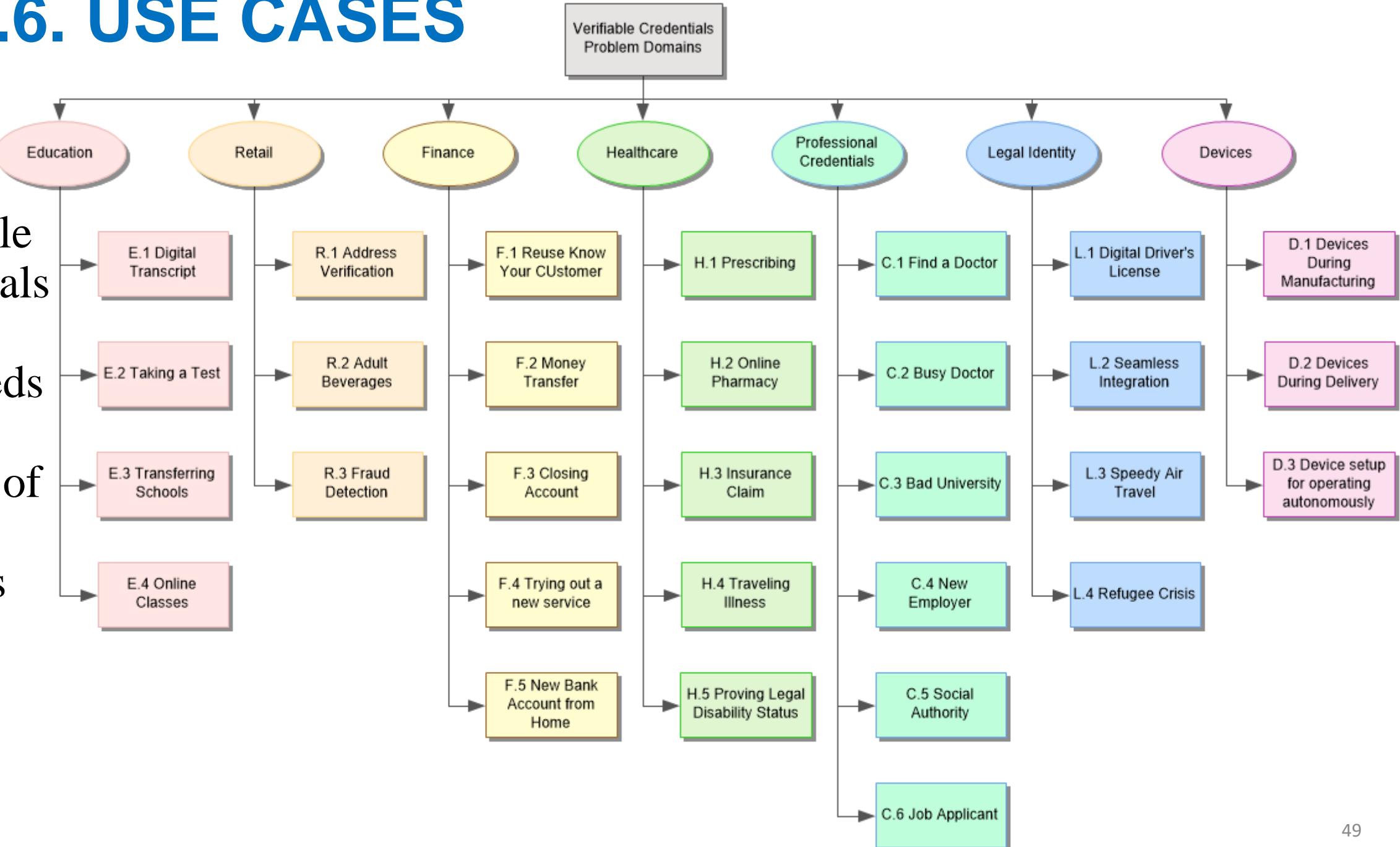
- zero-knowledge cryptography schemes
- indirectly prove claims from verifiable credential without revealing verifiable credential
- Selective disclosure schemes using zero-knowledge proofs

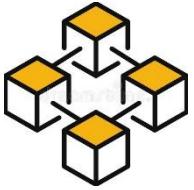




5.6. USE CASES

Verifiable credentials address user needs in a number of key domains





6. SSI BLOCKCHAIN PLATFORMS

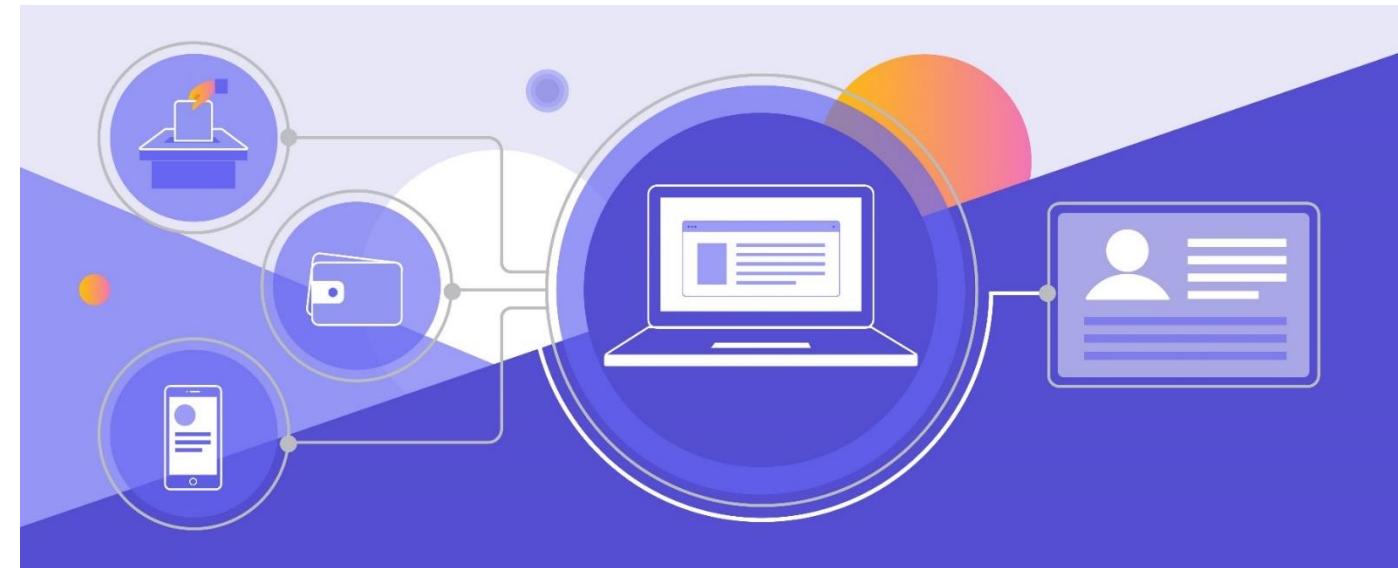
6.1. SSI BLOCKCHAIN PLATFORMS

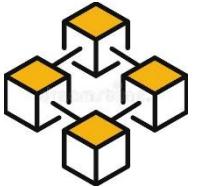
6.2. SSI HYPERLEDGER

6.3. HYPERLEDGER URSA

6.4. HYPERLEDGER INDY

6.5. HYPERLEDGER ARIES





6.1. SSI BLOCKCHAIN PLATFORMS

Utilities with permissioned write access:

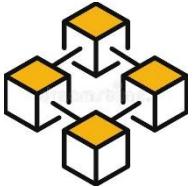
- Sovrin, Alastria
- in planning: SSI4DE (Lissi), Findy, eSSIF.



Utilities with permissionless write access:

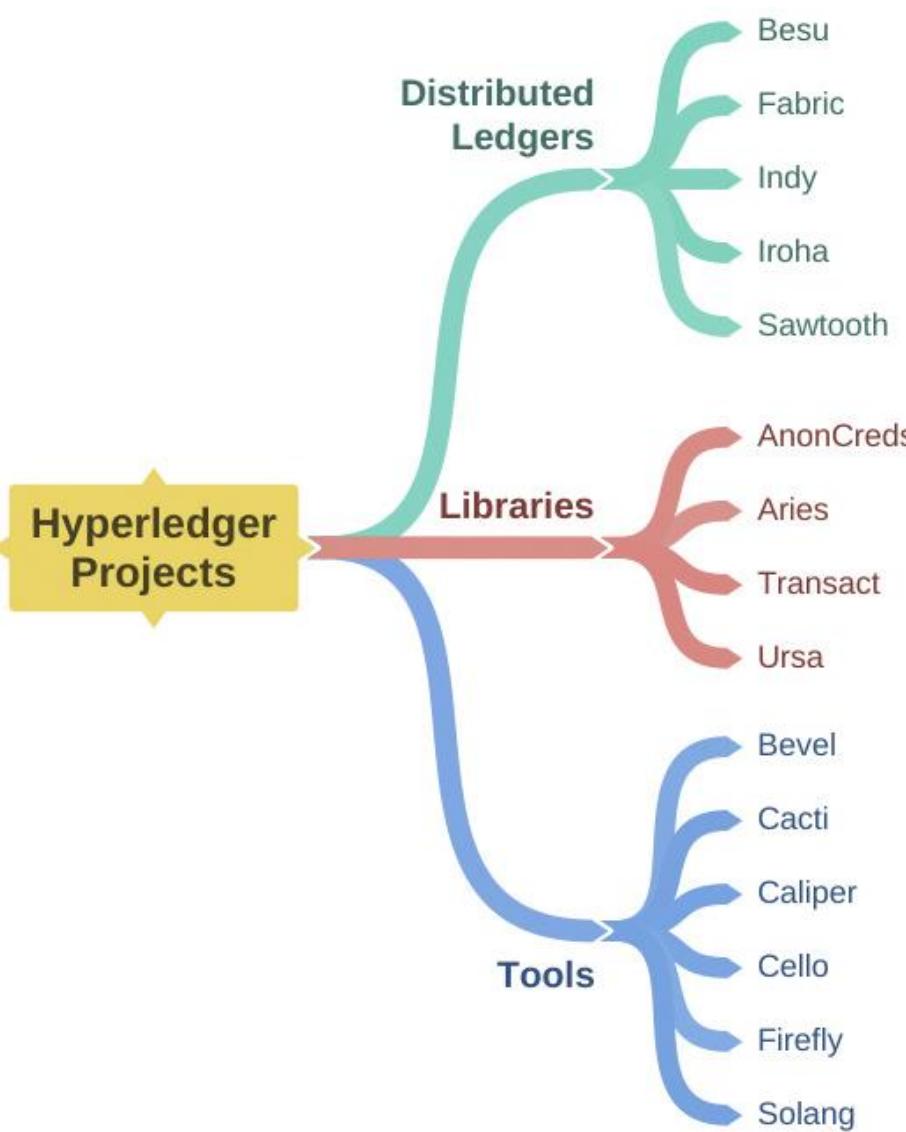
- ION on Bitcoin, uPort, Jolocom, Selfkey,
- Elements, Digital Bazaar, more on Ethereum.



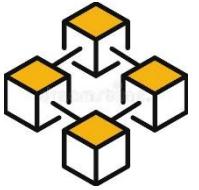


6.2. SSI HYPERLEDGER

Hyperledger Projects:



DISTRIBUTED LEDGERS	
	HYPERLEDGER FABRIC Status: Active Proposal: 14 April 2016 Acceptance: 14 April 2016 Active: 18 May, 2017
	HYPERLEDGER BURROW Status: Incubation Proposal: 28 March, 2017 Acceptance: 6 April, 2017
	HYPERLEDGER SAWTOOTH Status: Active Proposal: 14 April 2016 Acceptance: 14 April 2016 Active: 18 May, 2017
LIBRARIES	
	HYPERLEDGER URSA Status: Incubation Proposal: No date Acceptance: Nov 2018
	HYPERLEDGER ARIES Status: Incubation Proposal: No date Acceptance: Feb 2019
	HYPERLEDGER QUILT Status: Incubation Proposal: No date Acceptance: Oct 2017
TOOLS	
	HYPERLEDGER CELLO Status: Incubation Proposal: Jan 2017 Acceptance: Jan 2017
	HYPERLEDGER EXPLORER Status: Incubation Proposal: No date Acceptance: Aug 2016
	HYPERLEDGER CALIPER Status: Incubation Proposal: No date Acceptance: Mar 2018
HYPERLEDGER IROHA	
	Status: Active Proposal: 26 September 2016 Acceptance: 13 October 2017 Active: 18 May, 2017
HYPERLEDGER BESU	
	Status: Incubation Proposal: No date Currently on v.1.3
HYPERLEDGER TRANSACT	
	Status: Incubation Proposal: April 2019 Acceptance: May 2019
HYPERLEDGER AVALON	
	Status: Incubation Proposal: June 2018 Acceptance: Oct 2018

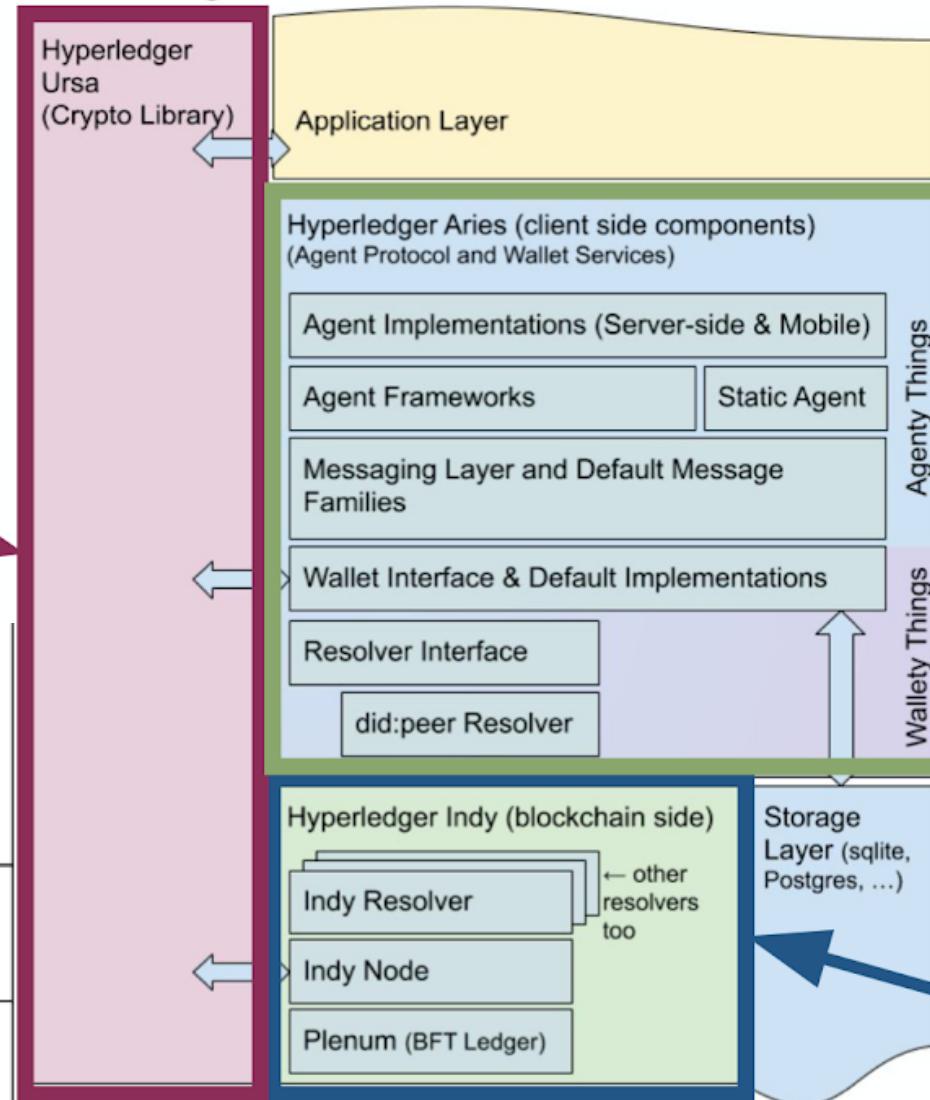


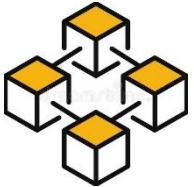
6.2. SSI HYPERLEDGER

- Aries
- Indy
- Ursa



Hyperledger as a Verifiable Information Exchange Platform





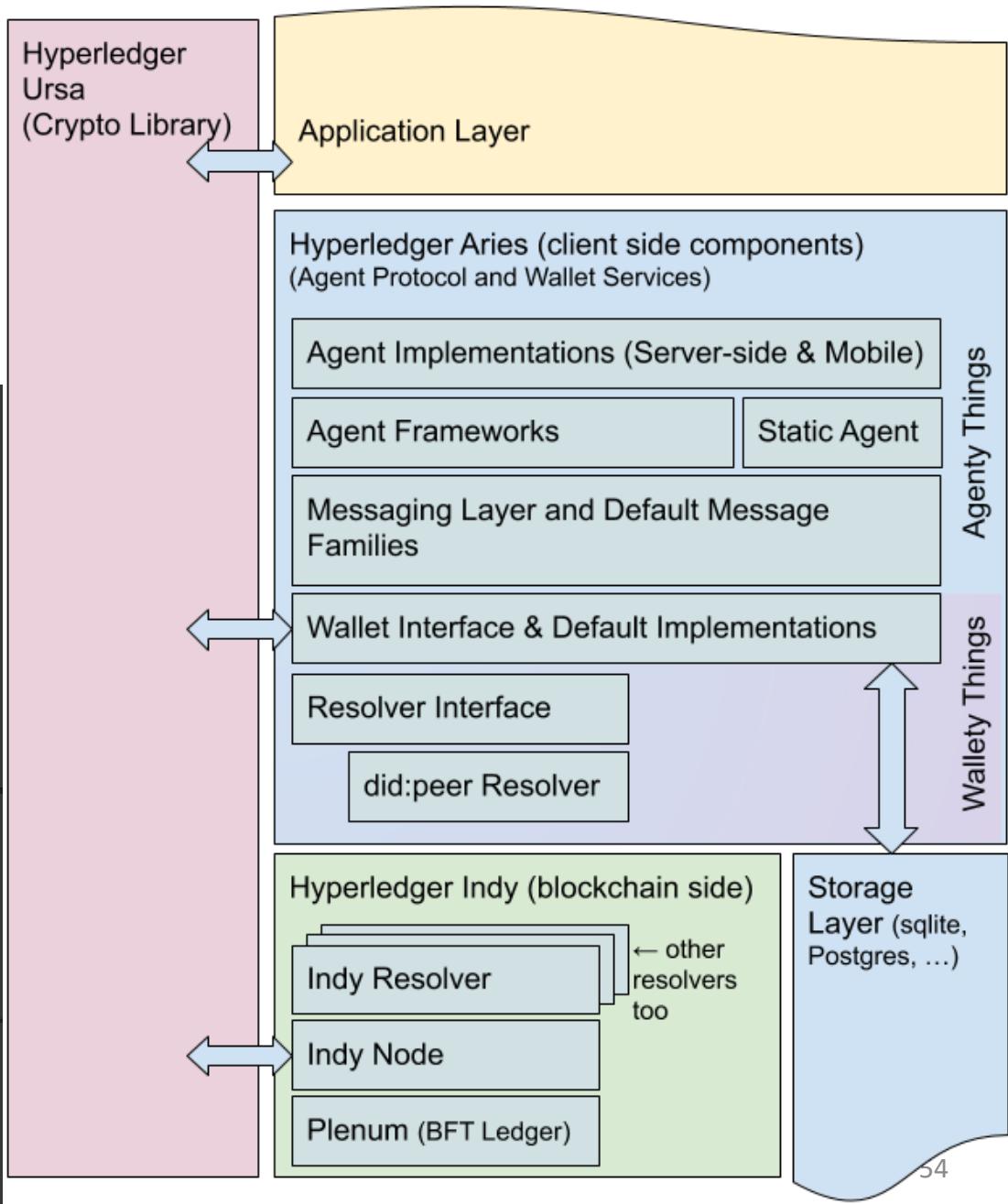
6.2. SSI HYPERLEDGER

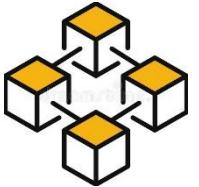
- Aries
- Indy
- Ursula



09/05/2023

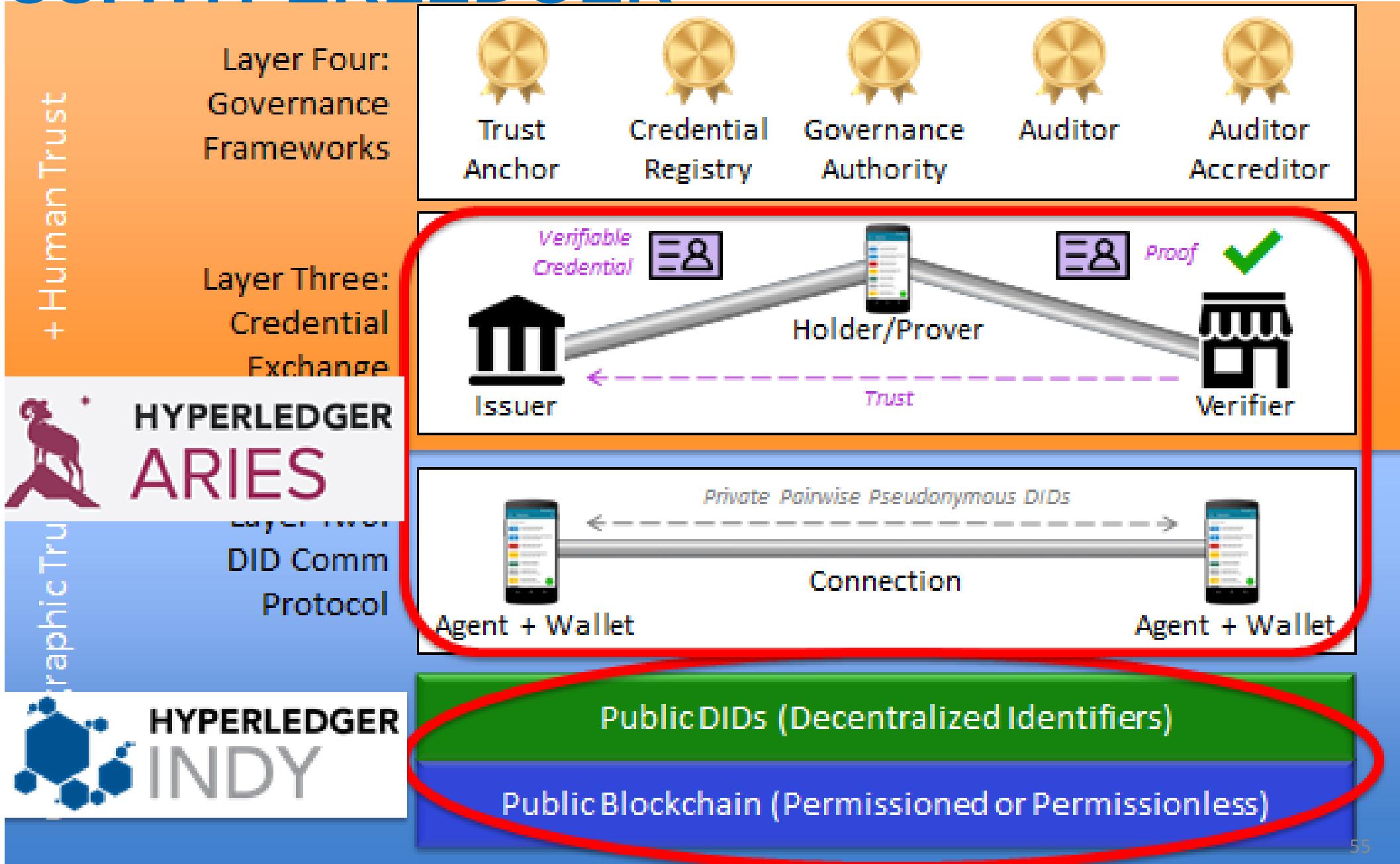
Hyperledger as a Verifiable Information Exchange Platform

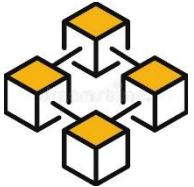




6.2. SSI HYPERLEDGER

- Layer 4:
Aries
Controller,
Ursa
- Layer 3:
Aries, Ursa
- Layer 2:
Aries, Ursa
- Layer 1:
Indy, Ursa





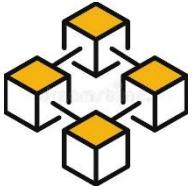
6.3. HYPERLEDGER URSA

Hyperledger Ursu:

- shared cryptographic library (avoid duplicating code)
- opt-in repository (for Hyperledger and non Hyperledger projects).
- C-callable library interface
- Rust crate
- Libursa: cryptographic primitives:
 - digital signatures
 - encryption schemes
 - and key exchange.
- Libzmix:
 - create zero-knowledge proofs, proving statements about cryptographic building blocks, containing signatures, commitments, verifiable encryption.
 - uses blocks in Libursa.



HYPERLEDGER
URSA



6.4. HYPERLEDGER INDY

Provides tools, libraries, reusable components for providing digital identities rooted on blockchains or other distributed ledgers

WHAT IS HYPERLEDGER INDY?

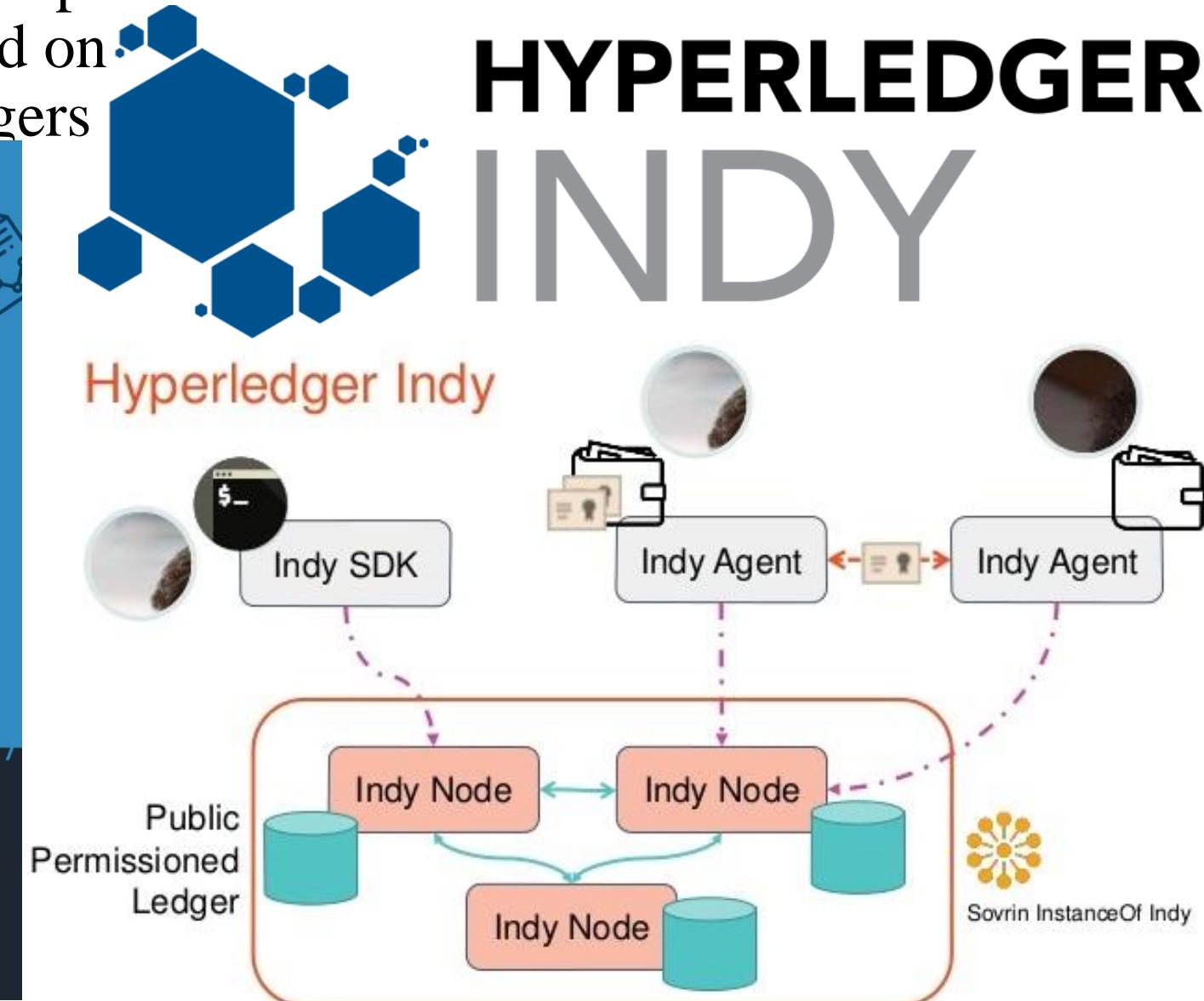
Hyperledger Indy is a project that is made for decentralized identity. It offers lots of libraries, tools, and reusable components for creating decentralized identities.

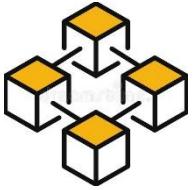
KEY BENEFITS FOR USERS

 Users will have full control over their identities.	 Anything on the ledger is visible to all.	 Users can use their identities on any network that allows them.	 Only the user will have access to his/her data.	 A user can delete or make changes to his/her ID.
 Other parties will need consent from the user.	 Disclosure of any documentation is limited.	 The platform will protect user rights all the time.	 Users will have full independent existence on the ledger.	 Users can also transport their IDs to other devices.

HYPERLEDGER INDY USE CASES

 Digital Documents	 Password-less Authentication	 Software Vulnerability Alert
 End of Spam	 Document Provenance	 Employment Verification
 Decentralized Membership Management	 Global Access with Single ID	 Age Restriction For Services





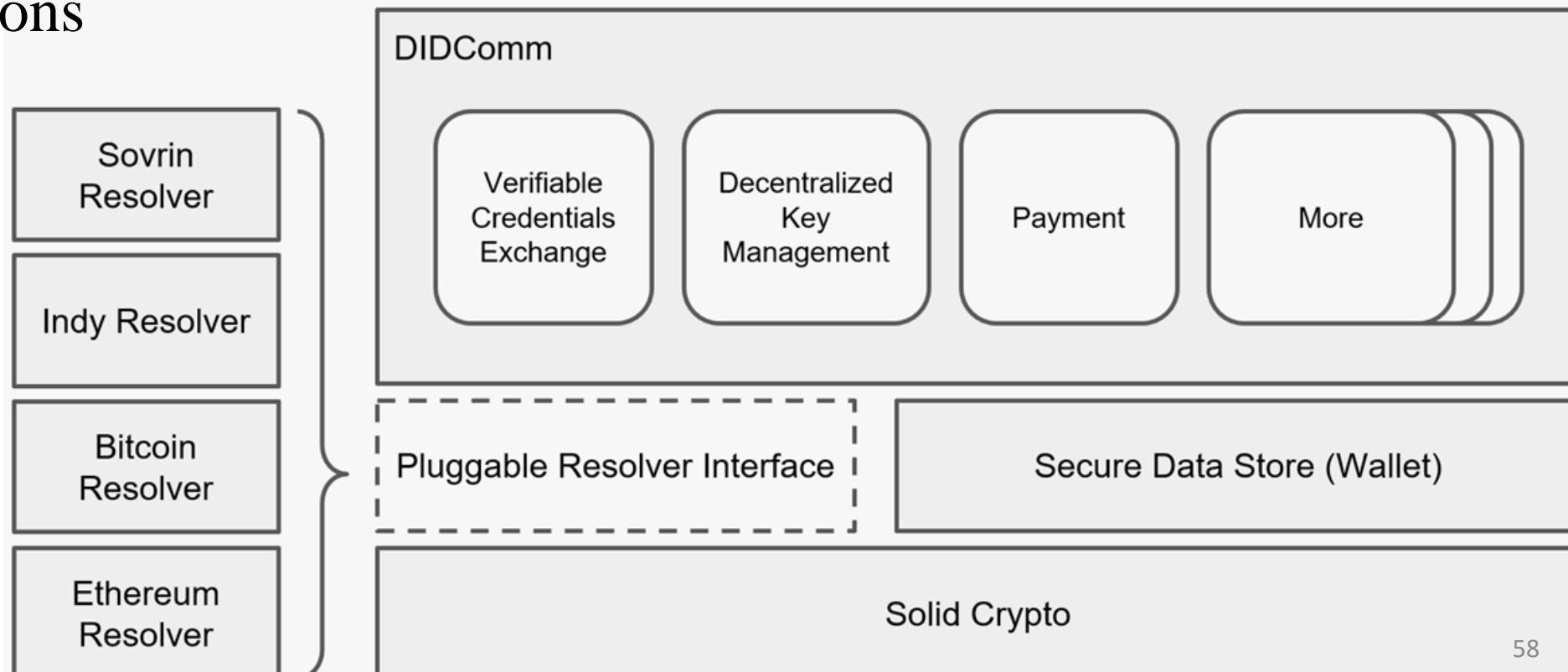
6.5. HYPERLEDGER ARIES

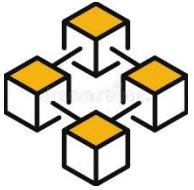
allows trusted online peer-to-peer interactions based on decentralized identities and verifiable credentials.

key components of Aries:

- agents
- DID communications
- Protocols
- key management.

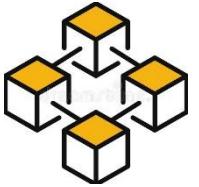
Aries = Agent that can be rooted to any DLT



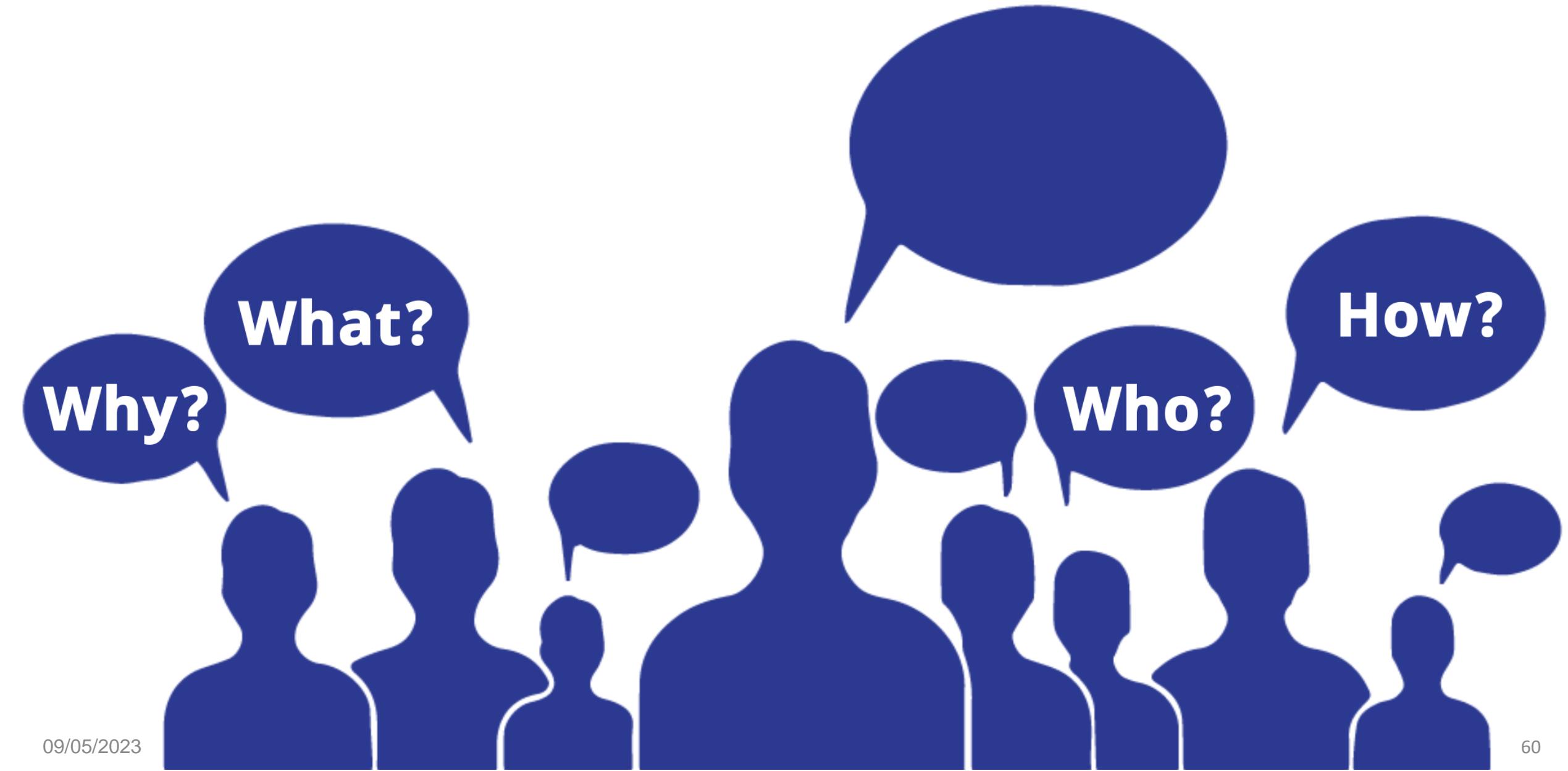


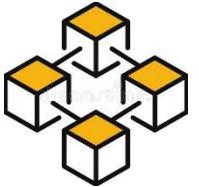
7. SUMMARY

- Digital Identity: Identity digital world
- Self-Sovereign Identity: Individuals control identity data
- Decentralized Identifier: globally unique identifierenables refers to subject, associate a DID subject with a DID document
- Verifiable Credential: are a digital credentials, cryptographically secured
- SSI Blockchain platforms: Hyperledger Projects (Indy, Aries, Ursa)



8. DISCUSSION





FINISH

Thank You