
W3C TPAC 2022 WITH GOOROOMEE
W3C TPAC 2022
톺아보기
부제 : 웹 서비스를 성장 시키는 HTML5 표준과 활용 바로알기
Gooroomee
W3C®

▶ Session #5.

김근형 교수 W3C DID 표준의 이해와 확장 그리고 VC

Gooroomee

- Why do we need DIDs and VCs?
- What are DIDs and VCs?
  - DIDs
    - Design Goals
    - DID Architecture
    - Components of DIDs
    - DID Methods
  - VCs
    - Components of VCs
    - The roles and Information flows
- Digital Ecosystem with DIDs and VCs
  - Digital Ecosystem
- TPAC 2022 Summary
- Conclusion

Why?

- Why do we need DIDs?
  - traditional globally unique identifiers like telephone numbers, email addresses, usernames on Websites, government ID number s, domain names, etc. are not under our control
    - assigned to us, rented
    - can be taken away
    - can be fraudulently replicated (identity theft)
  - allow the owner to provide cryptographic control over it
  - enable private and secure connections between two parties and can be verified anywhere at any time
- Why do we need VCs?
  - check a lot of boxes when it comes to user privacy requirements
  - address several major issues associated with the current identity management system

What?

Gooroomee

- What is a Decentralized Identifier (DID)?
  - a globally unique identifier made up of a string of letters and numbers
  - can be created and owned by individuals or organizations to represent themselves using systems they trust
  - can be deleted by individuals or organizations
  - come with a private key and a public key that are also made up of a string of letters and numbers
  - can have as many as we want to use in as many different contexts as we desire
  - is authenticated by “proof of control” using cryptographic proof such as signatures

https://www.w3.org/TR/did-core/#introduction

- What is a Verifiable Credential (VC)?
  - can represent all the same information that a physical credential represents
  - tamper-proof credentials that can be verified cryptographically using digital signatures

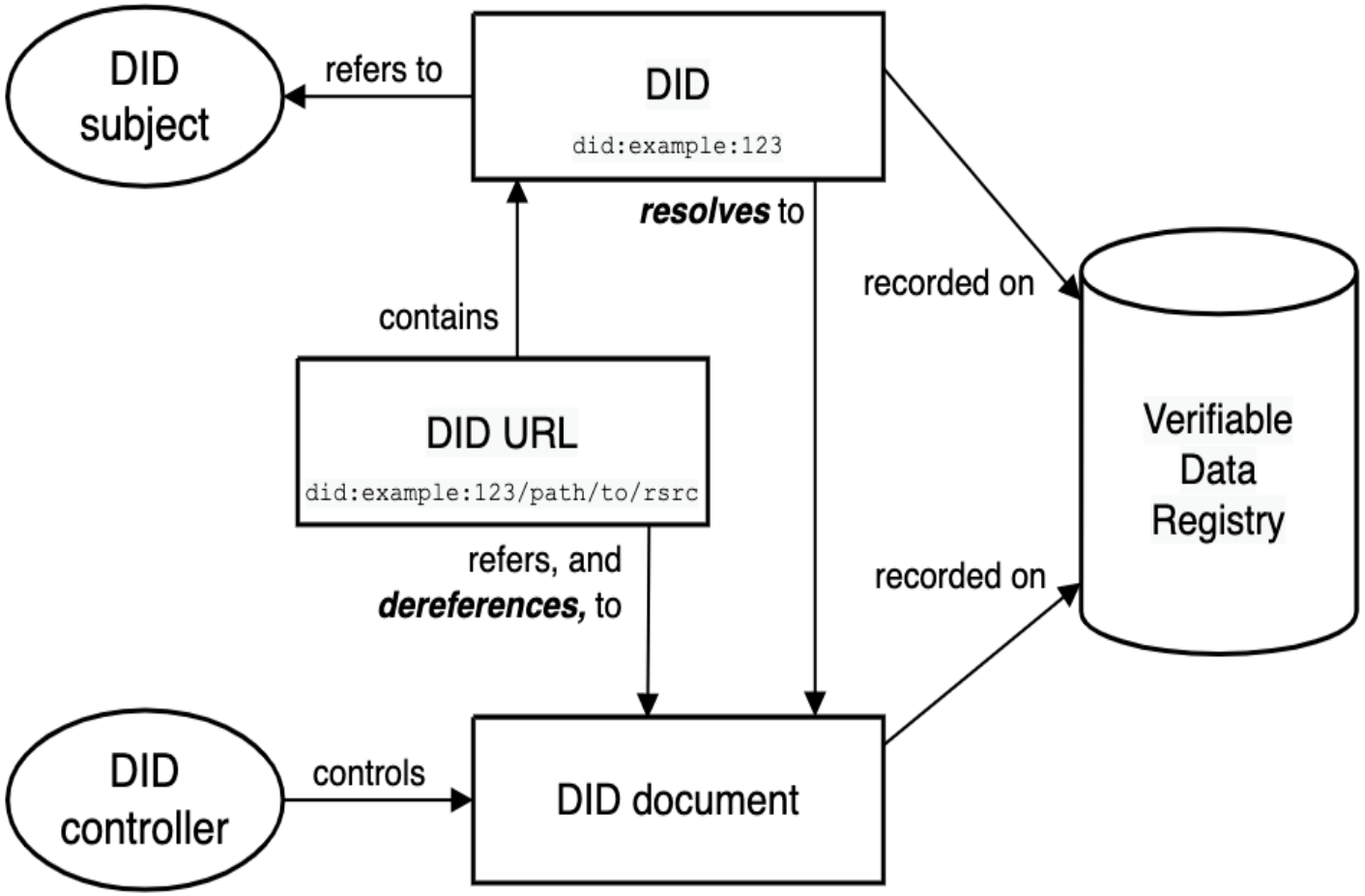
https://www.w3.org/TR/vc-data-model/#introduction

Design Goals

- Decentralization: eliminate the requirement for centralized authorities
- Control: give entities the power to directly control their digital identifiers
- Privacy: enable entities to control the privacy of their information
- Security: enable sufficient security for requesting parties to depend on DID documents for their required level of assurance
- Proof-based: enable controllers to provide cryptographic proof when interacting with other entities
- Discoverability: make it possible for entrees to discover DIDs for other entities
- Interoperability: use interoperable standards so DID infrastructure can make use of existing tools
- Portability: enable entities to use their digital identifiers with any system that supports DIDs and DID methods
- Simplicity: favor a reduced set of simple features to make the technology easier to understand, implement, and deploy
- Extensibility: enable extensibility provided it does not greatly hinder interoperability, portability, or simplicity

DID Architecture

https://www.w3.org/TR/did-core/#architecture-overview



Components of DID Doc.

- Identifiers
  - DID subject / controller: **id, controller**
  - Also Known As: **alsoKnownAs**
- verification methods
  - verificationMethod: **verificationMethod, type**
  - verification Material: **publicKeyjwk, publicKeyMultibase**
- verification relationships
  - authentication: **authentication**
  - assertion: **assertionMethod**
  - key agreement: **keyAgreement**
  - capability invocation / delegation: **capabilityInvocation, capabilityDelegation**
- Services
  - **service, id, type, serviceEndpoint**

How do DIDs work?

Gooroomee

- authentication

```
{
  "@context": [
    "https://www.w3.org/ns/did/v1",
    "https://w3id.org/security/suites/ed25519-2020/v1"
  ]
  "id": "did:example:123456789abcdefghi",
  "authentication": [{
    // used to authenticate as did:...fghi
    "id": "did:example:123456789abcdefghi#keys-1",
    "type": "Ed25519VerificationKey2020",
    "controller": "did:example:123456789abcdefghi",
    "publicKeyMultibase": "zH3C2AVvLMv6gmMNam3uVAjZpfkcJCwDwnZn6z3wXmqPV"
  }]
}
```

https://www.w3.org/TR/did-core/#did-documents

DID example

- A DID is a simple text string consisting of three parts
  - the did URI scheme identifier
  - the identifier for the DID method
  - the DID method-specific identifier

https://www.w3.org/TR/did-core/#a-simple-example

DID example

Gooroomee

- did:web Method
  - a new DID method in conjunction with blockchain-based DIDs that allows them to bootstrap trust using a web domain's existing reputation.

```
did:web:w3c-ccg.github.io

did:web:w3c-ccg.github.io:user:alice

did:web:example.com%3A3000
```

  - the fully qualified domain name that is secured by TLS/SSL certificate with the optional path to the DID document

DID	DID Resolution
did:web:w3-ccg.github.io	https://w3-ccg.github.io/.well-known/did.json
did:web3c-ccg.github.io:user/alice	https://w3-ccg.github.io/user/alice/did.json
did:web:example.com%3A3000:user/aliace	https://example.com:3000/user/alice/did.json

DID example - did:key

- **non-registry based DID method based on expanding a cryptographic key into a DID Document**
- **provide the simplest possible implementation of a DID method that is achieve may but not all, of the benefits of utilizing DIDs**

```
did:key:z6MkhaXgBZDvotDkL5257faiztiGiC2QtKLGpbnnEGta2doK
```

- DID document explains the cryptographic method associated with the DID
- provides the information needed to prove control

https://www.w3c-ccg.github.io/did-method-key

DID example - did:so

- Sovrin DID method specification conforms to the requirements specified in the DID specification currently published by the W3C CCG.

```
did:sov:2wJPyULfLLnYTEFYzByfUR
```

- created on a public blockchain called SOvrin
- read directly from Sovrin and then a DID document is generated
- “proof of control” is based on cryptographic key pairs

https://sovrin-foundation.github.io/sovrin/spec/did-method-spec-template.html

DID Tools

- **DIF Universal Resolver: https://dev.uniresolver.io/**

- **DID actor: https://api.did/actor/**
  - free to create and resolve DIDs,
  - also can create and verify credentials and presentations

https://sovrin-foundation.github.io/sovrin/spec/did-method-spec-template.html

Gooroomee

- Chapter 2 -

Verifiable Credentials

The 3 Components of VCs

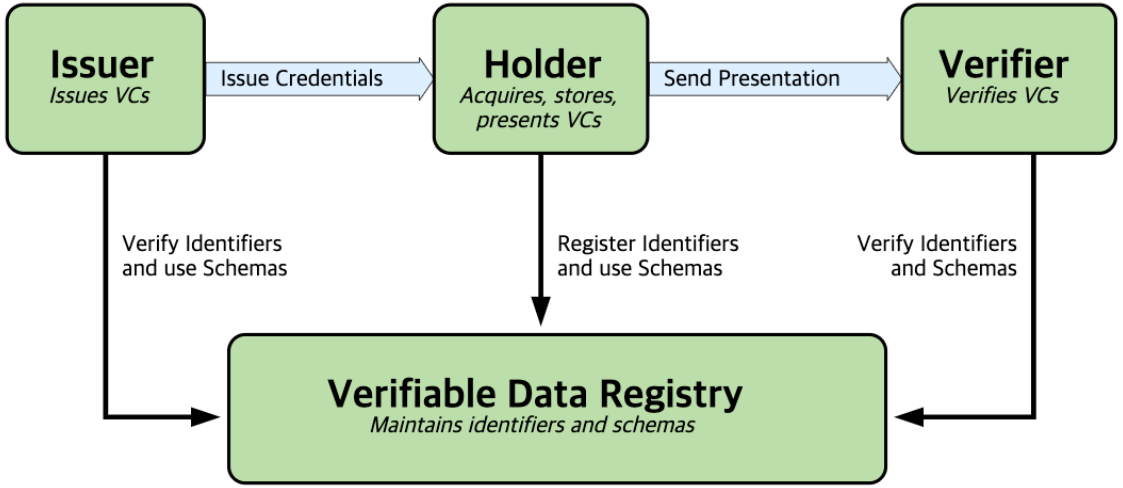
Gooroomee

- Credential metadata
  - might be cryptographically signed by the issuer
  - contain the credential identifier as well as properties about the credentials itself such as expiry date and who the issuer is
- Claims(s)
  - a tamper-proof set of claims made about the credential subject such as someone's employee number and job title
- Proof(s)
  - a cryptographic method that allows people to verify:
    - the source of the data (e.g., who the issuer is)
    - that the data has not been tampered with

https://www.w3.org/TR/vc-data-model/#credentials

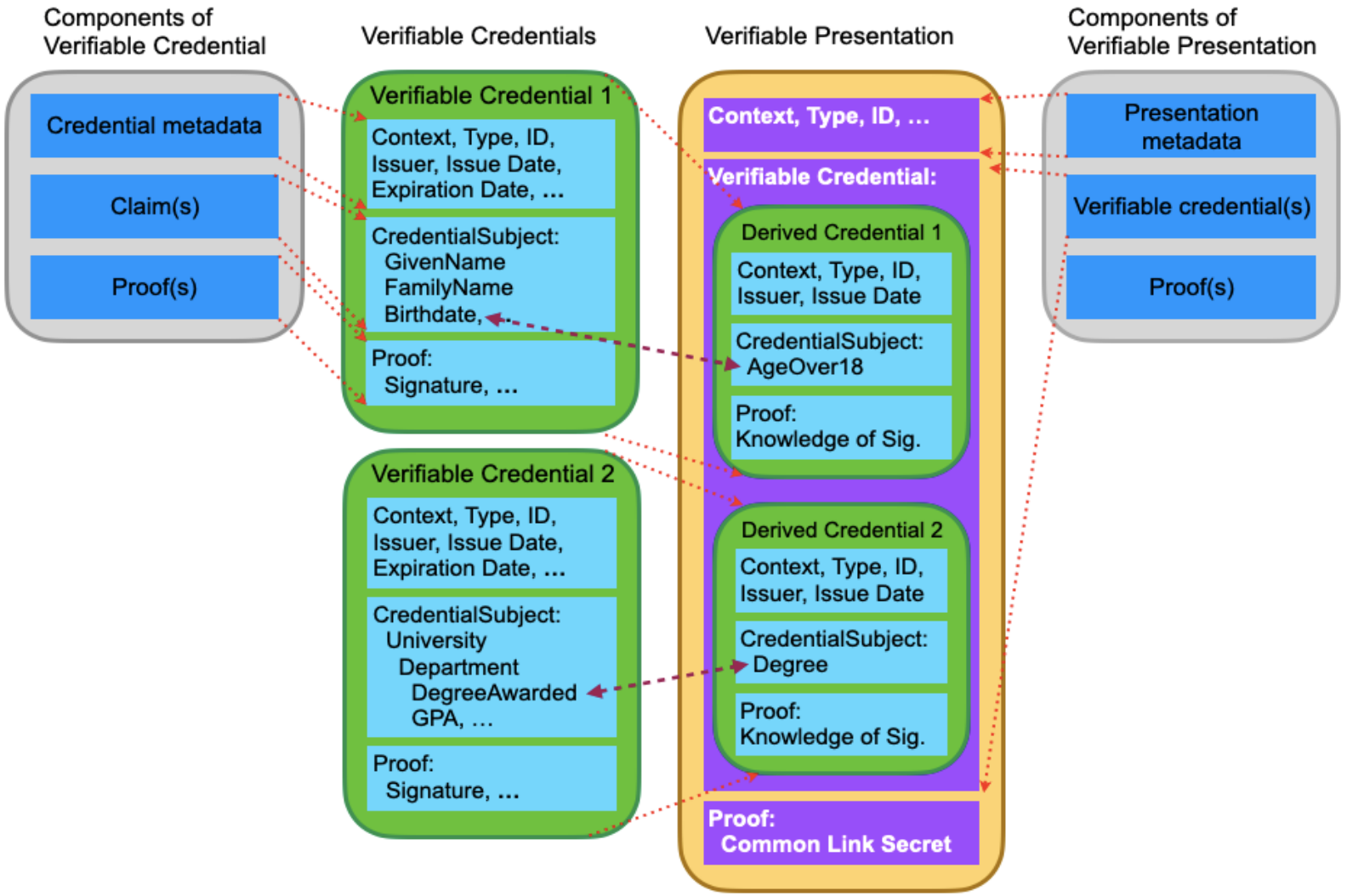
The 3 Components of VCs

- issuer: create a verifiable credential and transmit the verifiable credential to a holder
- verifier: receive one or more verifiable credentials, optionally process, optionally inside a verifiable presentation for processing
- holder: process one or more verifiable credentials and generate verifiable presentations from them

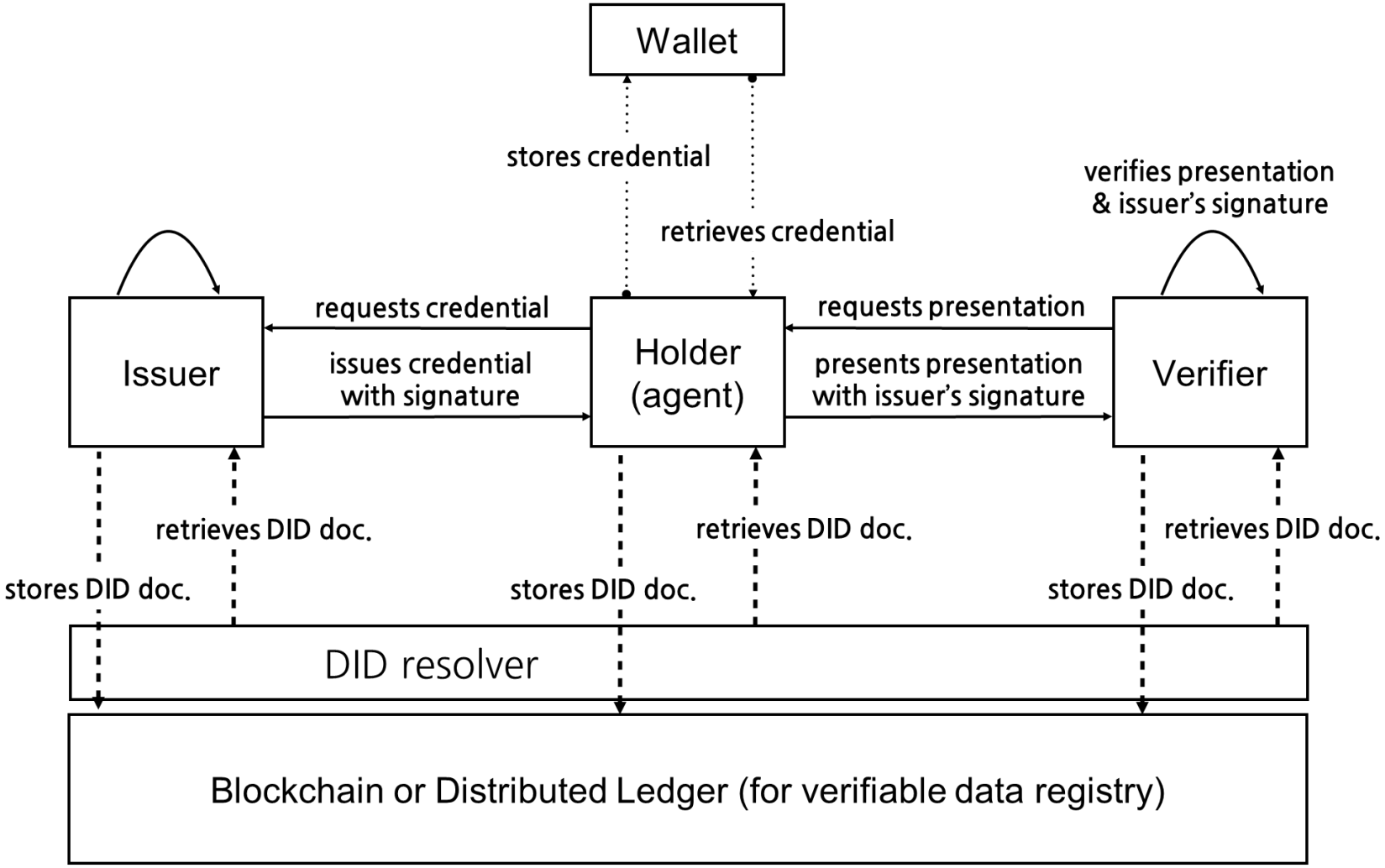


https://www.w3.org/TR/vc-data-model/#ecosystem-overview

VC and VP


Components of Verifiable Credential
Credential metadata
Claim(s)
Proof(s)
Verifiable Credentials
Verifiable Credential 1
Context, Type, ID, Issuer, Issue Date, Expiration Date, ...
CredentialSubject: GivenName FamilyName Birthdate,
Proof: Signature, ...
Verifiable Credential 2
Context, Type, ID, Issuer, Issue Date, Expiration Date, ...
CredentialSubject: University Department DegreeAwarded GPA, ...
Proof: Signature, ...
Verifiable Presentation
Context, Type, ID, ...
Verifiable Credential:
Derived Credential 1
Context, Type, ID, Issuer, Issue Date
CredentialSubject: AgeOver18
Proof: Knowledge of Sig.
Derived Credential 2
Context, Type, ID, Issuer, Issue Date
CredentialSubject: Degree
Proof: Knowledge of Sig.
Proof: Common Link Secret
Components of Verifiable Presentation
Presentation metadata
Verifiable credential(s)
Proof(s)

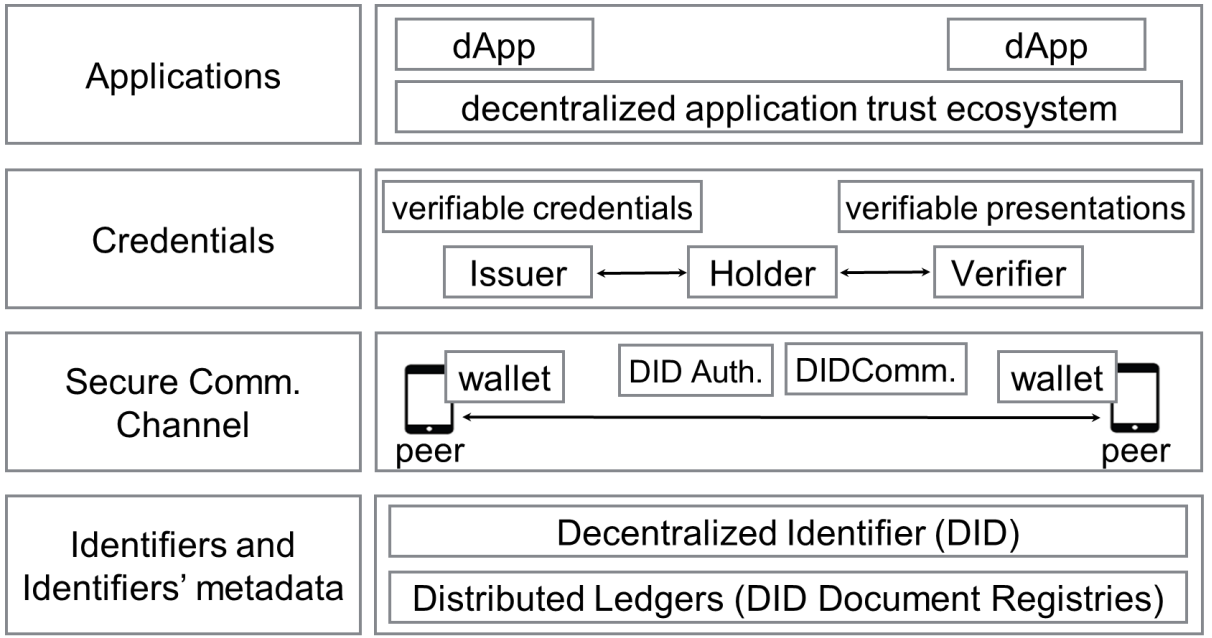
Digital Ecosystems



Layered Architecture

- SSI ecosystem consists of
  - digital wallets, decentralized identifiers, verifiable credentials, verifiable presentations, issuers, holders, and verifiers

- SSI ecosystem can be modeled into four layers according to the functionality.



Gooroomee

- Chapter 3 -

TPAC 2022

Discussion in TPAC DID WG

- Director's Decision on DID 1.0 PR Formal Objections
  - "The DID core specification is approved to advanced by W3C Recommendation" - (2022. 6. 20)
- TPAC 2022 DID WG Meeting (1hr)
  - attendees: 40~50 people
  - goal: to settle on the next Charter
    - work towards agreeing to standardize a few DID methods (like did:key, did:web)
    - start standardizing DID resolutions
- Joint meeting with WOT WG (1hr)
    - discuss applying DIDs to WoT Ecosystem

https://www.w3.org/TR/vc-data-model/#credentials

Discussion in TPAC VC WG

- **Attendees: 40 ~ 50**
- Discuss ongoing activities related to VC Data Model 2.0
  - the core data model with keeping things simple via JSON-LD
  - The streaming Data Integrity crypto suite
  - stayed away from discussing digital wallet protocols
- Current issues
  - standardize the VC-APIs
  - the open wallet APIs
  - multi-signature verifiable credentials

Conclusion

- Requires new credentials and identity to mitigate current problems in the centralized credential system
- Require effective methods for establishing a trustworthy relationship between non-trustworthy subjects in the digital ecosystem
- Dids and vcs would be indispensable to providing credentials in the web 3.0 ecosystem
- Should follow up on activities of other communities along with those of W3C; DIF, wot, IIW

Gooroomee

Q&A

geunkim@deu.ac.kr

Gooroomee

홈페이지 biz.gooroomee.com

도입문의 sales@gooroomee.com

대표전화 1833-9229