Owning Self Identity and Data with a Decentralized web platform

Aman Singh Bhandari
Departement of Computer Science
Dalhousie University
Halifax, Canada
am727005@dal.ca

Ayush Verma
Departement of Computer Science
Dalhousie University
Halifax, Canada
ayush.verma@dal.ca

Heli Bhavsar
Departement of Computer Science
Dalhousie University
Halifax, Canada
hl751966@dal.ca

***Abstract*—**Proof of Concept for a self-sovereign identity network that allows you to keep a single identity and data in user's control through decentralized web platform. This paper describes three components mainly DID, Verified credentials and DWN. DID is the identity of the user that sits on the blockchain on layer 2 called ION and is a URL that points to DID Document. VC are the verified credential provided by individual, organization, and government of different DID. DWN challenges the centralized way of storing data. It allows the data to be stored on multiple nodes owned by the user making sure that the owner's data is in owner's control. These three components work together to provide a decentralised way of owning the identity and data to the user.

Keywords *– Decentralized Identity, Verified Credentials, token-based governance, DID, Blockchain nodes, W3C Standards*

Introduction

The existing web model doesn't allow user to hold its identity and data. Our identities are the account given by companies where our data is held captive, and its confidentiality and security is on mercy of the company. Users in current web model holds more than one identity of various websites and apps and their data is held with company associated with these multiple identities. The main reason behind this is centralized nature of web.

DID and DID Document

The government, a business, an association, or some other middleman is the owner of the identifiers we are familiar with and use today. For instance, even though they are associated with us, our email addresses and social media handles are under the service providers' ownership and control. We have very little to no control over these companies' ability to block, disable, or delete these identifiers. Therefore, we need decentralized identifiers that users own and control before we can create truly decentralized applications. As a result, we are no longer dependent on centralized organizations to represent and authenticate us.

A W3C-recommended recommendation called Decentralized Identifiers (DIDs) has a standardized structure that essentially links to you and your data [1].

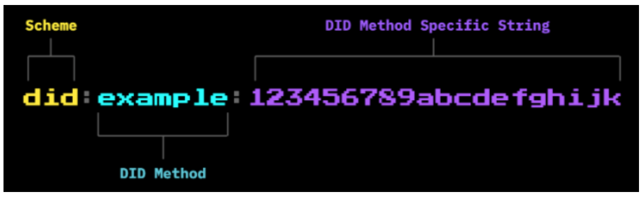


Figure 1:Structure for DID Source

They are a long string of text that consists of three parts [1]:

1) The URI scheme identifier, which it did

2) The identifier for a DID method

3) The DID method-specific identifier

It is recommended to store DIDs on ION (a Layer 2 DID network that runs on top of Bitcoin) when implementing Web 5. Since ION is a decentralized alternative to DNS for identity identifiers, it lacks any centralized authorities, coordinators, tokens, or other bottlenecks. Except for anchoring the keys and endpoints connected to the ID, the only part of Web5 that interacts with a blockchain is DIDs.

However, it is not necessary to anchor DIDs on the Bitcoin blockchain (or any other blockchain). Having standardized formatting for DIDs has the advantage of allowing them to be anchored anywhere or not at all, with varying degrees of decentralization, and still function.

Here are some instances of DIDs from the web, Ethereum, and Bitcoin blockchains. The system, DID method and DID technique identifier are all presented in the same format.

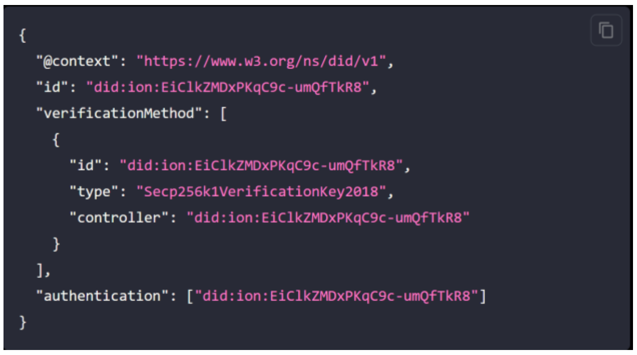
did: btcr: xyv2 - xzpq - q9wa - p7t;

did: ens: some.eth;

did: web: example.com;

Since personal information is not kept on the blockchain, the DID essentially serves as a URI that links the DID subject (the individual, business, or thing being identified) with a DID document that resides off-chain.

The DID subject is described in DID Documents, which are JSON files kept in decentralized storage systems like IPFS. The DID Document includes information on the DID subject's public keys, methods for authentication and verification, and service endpoints that serve as references to the subject's data's storage locations.



```
{
  "@context": "https://www.w3.org/ns/did/v1",
  "id": "did:ion:EiClkZMDxPKqC9c-umQfTkR8",
  "verificationMethod": [
    {
      "id": "did:ion:EiClkZMDxPKqC9c-umQfTkR8",
      "type": "Secp256k1VerificationKey2018",
      "controller": "did:ion:EiClkZMDxPKqC9c-umQfTkR8"
    }
  ],
  "authentication": ["did:ion:EiClkZMDxPKqC9c-umQfTkR8"]
}
```

Figure 2: Web5

Architecture View of DID:

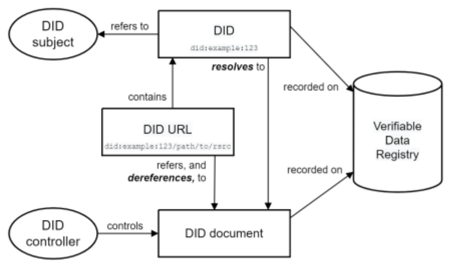


Figure 3:Overview of DID architecture and the relationship of the basic components

DIDs and DID URLs:

A decentralized identifier, also known as a DID, is a URI made up of three components: the scheme did, a method identifier and a special, method-specific identifier provided by the DID method. Documents containing DIDs can be resolved. To find a specific resource—for instance, a cryptographic public key inside a DID document or a resource outside the DID document—a DID URL expands the syntax of a basic DID to include additional standard URI components such as path, query, and fragment.

DID Subjects:

The entity that the DID identifies is, by definition, the subject of the DID. Possible DID controllers include the DID subject.

DID controllers:
The person, business, or autonomous software that has the power to modify a DID document according to a DID method is the controller of a DID. This capability is typically asserted by software acting on behalf of the controller by controlling a set of cryptographic keys, but it may also be asserted through other mechanisms. Keep in mind that a DID may have more than one controller, and the DID subject may be the controller or one of them.

Verifiable data registries:
The majority of the time, DIDs are stored on some sort of underlying system or network in order to be resolved to DID documents. Any system that allows for the recording of DIDs and the retrieval of information required to produce DID documents is referred to as a verifiable data registry, regardless of the technology employed [2]. Databases of any kind, decentralized file systems, peer-to-peer networks, distributed ledgers, and other trusted data storage systems are a few examples.

Verifiable data registries:
Information pertaining to a DID is contained in DID documents. Typically, they describe verification techniques—like cryptographic public keys—and services pertinent to communications with DID subjects.

DID methods:
DID methods refer to the processes used to create, resolve, update, and deactivate a specific type of DID and the associated DID document.

DID resolvers and DID resolution:
A DID resolver is a component of the system that receives a DID as input and outputs a conforming DID document. DID resolution is the name of this procedure [2]. The applicable DID method specification outlines the steps for dealing with a particular type of DID.

DID URL dereference and DID URL dereferencing:
A system component known as a DID URL dereference generates a resource from a DID URL as its input. DID URL dereferencing is the name of this procedure.

First, confirm that you have the correct template for your paper size. This template has been tailored for output on the A4 paper size. If you are using US letter-sized paper, please close this file and download the Microsoft Word, Letter file.

Maintaining the Integrity of the Specifications

The template is used to format your paper and style the text. All margins, column widths, line spaces, and text fonts are prescribed; please do not alter them. You may note peculiarities. For example, the head margin in this template measures proportionately more than is customary. This measurement and others are deliberate, using specifications that anticipate your paper as one part of the entire proceedings, and not as an independent document. Please do not revise any of the current designations.

VERIFIED CREDENTIALS

Public institutions and companies generally employ physical credentials (similar as passports, social security cards, and employee badges) to identify individualities. Individuals can choose where to store their physical credentials, and occasionally, they can decide to whom their credentials are shared with. These familiar privileges inspired a new type of digital credential called a verified credential (VC). Analogous to physical credentials, an individual can store their verifiable credentials in a digital wallet on their mobile phone, on another edge device, or in the cloud, and they can use it for the purpose of identification, authentication, and authorization across various parties including the government organizations [3].

Verifiable credentials and digital wallets offer a convenient, secure, and privacy-oriented alternative to current physical and digital identity management systems used in practice. Quoting a recent example of digitally signed COVID-19 vaccination certificate by government of Nova Scotia, which provides sufficient level of authenticity of vaccination by the government and can be quickly verified at events like a concert or a sports event. This reduces the paperwork work and the ho required to verify the credibility.

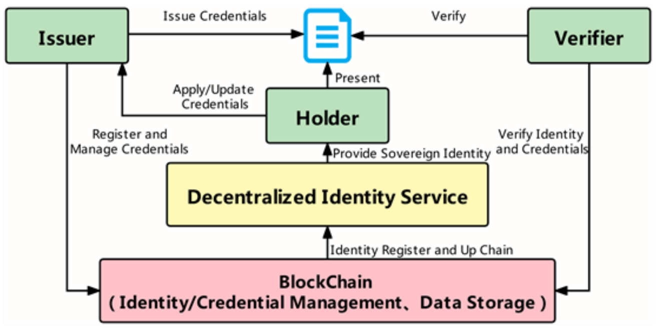
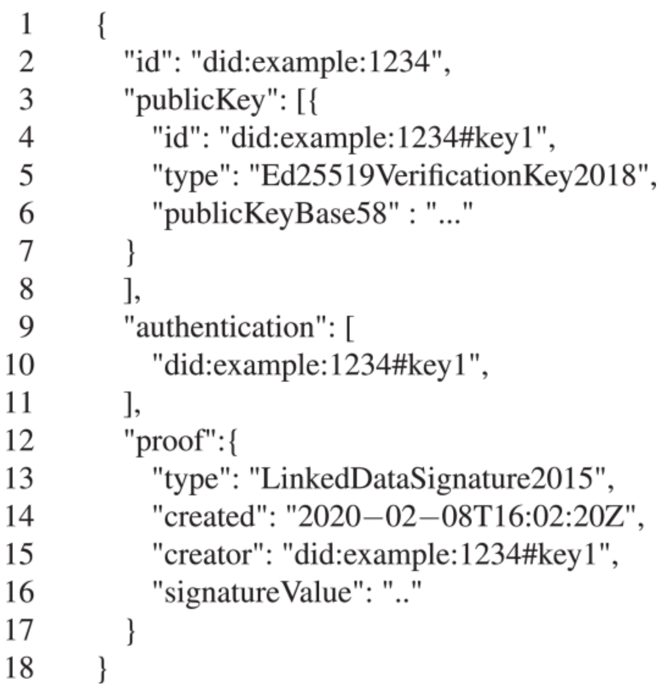

Issue Credentials
Verify
Issuer
Verifier
Present
Apply/Update Credentials
Holder
Register and Manage Credentials
Verify Identity and Credentials
Provide Sovereign Identity
Decentralized Identity Service
Identity Register and Up Chain
BlockChain
(Identity/Credential Management、Data Storage)

Figure 4 :The structure of the decentralized identity authentication system

Verifiable credential creates a strong foundation for the authentication and autonomous identity in a decentralized network. Overall, verifiable credential system consists of three different roles, mainly, Issuer, User, and Verifier. The process and structure followed by an authentication system that uses decentralized identity includes three major processes: identity registration, credential issuance, and credential verification. The issuer checks and confirms legitimacy and sensitive information of the user like personal details, and then signs and issues authentic credentials to the user with a trust endorsement. Now whenever the user needs to verify their authenticity, the verifier will check the signature of the credential issued by the corresponding issuer. If the signature decrypts successfully, the user is a valid user. Once credentials are received, user can store them locally on the mobile or cloud hosting. The credentials contain certain attributes which prove that user is qualified and can access the service with certain requirements. In this way, an user can verify the user's identity. [4].

With respect to decentralized verifiable credentials, there are a lot of ongoing research, out of which, Merkle tree method to implement verifiable credentials is the prominent one. David Bauer, one of the decentralized research scientist, proposed the Merkle tree implementation of the verifiable credentials to minimize the leakage of the information. The proposed method can only be used to verify the credential but the credentials does not include the user's name and other data which in return, could expose the exact identity of the user. The secret or private part of the credential contain the user user's private key and with respect to the Merkle tree, all leaves are micro-claims of the user identity. Users when willing to prove their identity, can prove their identity for different institutions according to the requirements and these structure used to store the credentials are inside the Merkle tree. Using these verifiable content implementation by using the tree, the author in the paper [6], author has implemented a prototype system. By his research, he achieved 200 authentication per second, which is relatively fast approach. The authentication used W3C standard document which is described in the Figure 2 below.



```
{
  "id": "did:example:1234",
  "publicKey": [{
    "id": "did:example:1234#key1",
    "type": "Ed25519VerificationKey2018",
    "publicKeyBase58" : "..."
  }
  ],
  "authentication": [
    "did:example:1234#key1",
  ],
  "proof":{
    "type": "LinkedDataSignature2015",
    "created": "2020–02–08T16:02:20Z",
    "creator": "did:example:1234#key1",
    "signatureValue": ".."
  }
}
```

Figure 5: Approved document structure

A Decentralized Identifier (DID) [13] provides a verifiable and decentralized means for interacting with a DID Subject controlling the DID. A DID can be resolved to a DID Document, which can contain cryptographic material, verification methods, and service endpoints. An example DID is "did:did-name: WRfxPg8dantKVubE3HX8pw", where "did" tells us that it is a DID, "did-name" is the DID Method Name for Sovrin DIDs, and "WRfXPg8dantKVubE3HX8pw" identifies the DID subject.

The International Electrotechnical Commission defines A Decentralized Identifier (DID) [13] provides a verifiable and decentralized means for interacting with a DID Subject controlling the DID objects. A DID can be converted into a DID Document, which in turn contain the cryptographic material, verification methods and few service endpoints. An example would be, with DID "did:sample-did: briu4brbhrfinivniv4hKnjdjododd" identifies the DID subject.

A distributed digital identity includes two parts: a distributed digital identity identifier and a digital identity credential.

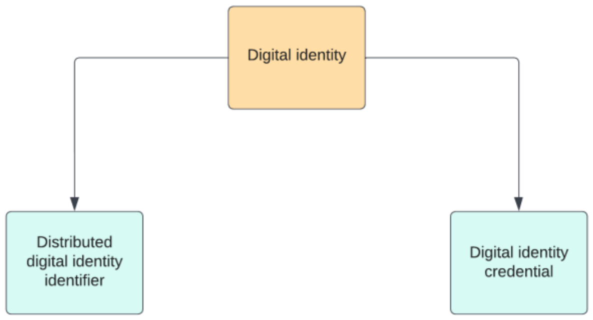

Digital identity
Distributed digital identity identifier
Digital identity credential

Figure 6 : Digital identity bifurcation

“Claims” refer to the attribute which associates the information with the actual identity and asserts the digital identity independently of any system it relies on. Declaration information includes personal information such as first name, last name, email address, age, gender, geographic location etc. Claims are generally issued by the identity owner (could be an individual or an organization) themselves or by any third-party claim issues and are known as verifiable claims when they are verified by the issuer. The end-user submits the claim for the relevant requirement, where the application checks it and the end-service provider can trust the verifiable claim signed by it, just like the issuer. Credentials are collection of multiple claims arranged in an order.

Verifiable Credentials (VC) creates a policy to describe available properties of an entity to achieve the evidence-based trust. Clients holding DID can prove their identity to the other entities that certain attributes of their identity can be verified through verifiable claims. Alone with it, combination of cryptographic technologies such as digital signatures and zero knowledge proofs, the identity can me made extremely reliable by making it more secure and credible, along with maintaining the identity of the client as anonymous.

Issuer is an establishment that owns user data and can issue verifiable credentials based on user data, like governments, banks, schools, and so on. The holder is user, which may apply for a verifiable credential from the issuer, then hold and keep the credential, like in a wallet, and show the credential to the verifier if necessary. Verifier receives the credentials presented by the user and may provide corresponding services to the user according to the credentials. additionally, an identifier registry (Verifiable Data Registry) is additionally required. The identifier registry may be a database that maintains DIDs, like a blockchain or distributed ledger, which may be understood as the example field in the DID. The Verifiable Data Registry is required because the validator needs to validate the credentials, also as the user.

A credential consists of a hash value, a public key list, etc., which are spliced with multiple claims. the knowledge of the credential must include the holder did, proof information (attribute signature information), issue date, issuer information, credential type, etc.

First, issuer will issue a credential for the user, which contains the signature (stored within the proof field), the DID of credential, creation date of credential, signature algorithm and other information, during which the information of the credential is all stored in JSON format. During the presentation stage, the user aggregates the claims within the signatures which include signatures issued by multiple issuers. The BLS aggregate signature is employed for the aggregation here. BLS aggregate signature can't only aggregate the signature information of the claims in different credentials into a new credential, which is named presentation, as shown in Figure 4. When verifying, the verifier only must verify the aggregate signature in the presentation. BLS aggregate signatures can't only aggregate signatures of different claims in different credentials of the same user, but also aggregate signatures in presentations from different users. Verifier only must verify the aggregate signature to verify the presentation of multiple users. it's not necessary to sign all the messages of the credential (such as issuer’s DID, credential type, etc.), signing claims is enough to verify.

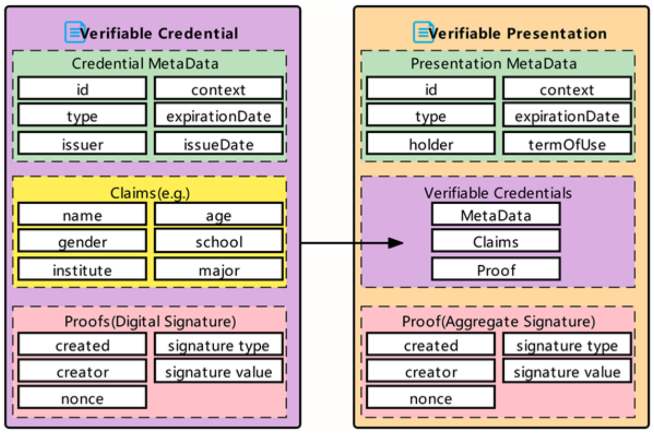


Figure 7: Selective disclosure of VC

To make verifiable credentials highly available throughout the infrastructure, we would recommend applying an additional layer of the caching which allows to quickly access the required credentials from the wallet in the minimum latency. This will make the verification process highly available and scalable.

Decentralized Web node

According to Identity Foundation “A decentralized personal and application data storage and message relay node, as defined in the DIF Decentralized Web Node specification. Users may have multiple Nodes that replicate their data between them.” [1]. Figure 1 showcase where the decentralized web node fits in the topology of communication between Alice and Bob in a decentralized system.

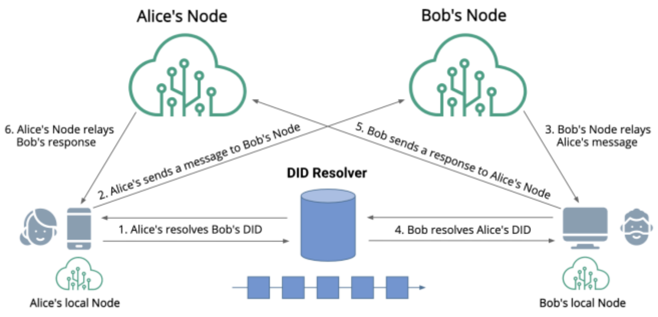

Alice's Node
Bob's Node
6. Alice's Node relays Bob's response
2. Alice's sends a message to Bob's Node
5. Bob sends a response to Alice's Node
3. Bob's Node relays Alice's message
DID Resolver
1. Alice's resolves Bob's DID
4. Bob resolves Alice's DID
Alice's local Node
Bob's local Node

Fig. 8. Topology [1]

Figure 2 shows the different layers of decentralized web node that make sure that multiple nodes of a user can coexist and can work together to act as a single unit.

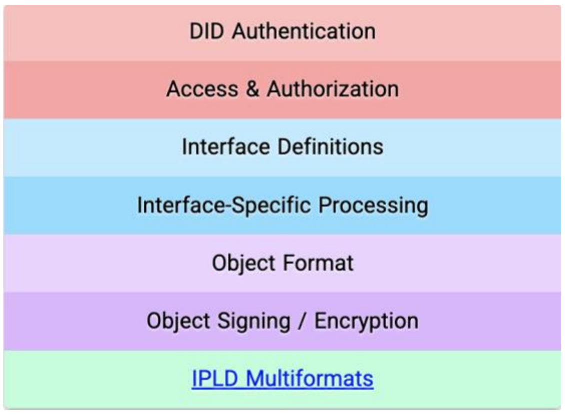
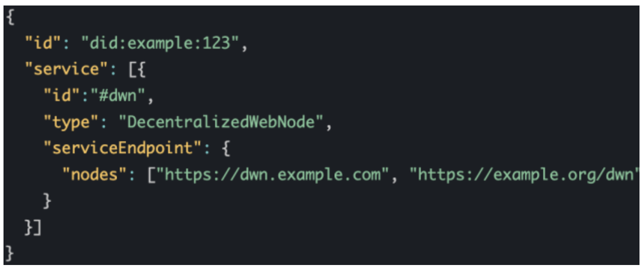


Figure 9:Layers of DWN

The DID document must contain the service endpoint to resolve the location of Decentralized web nodes. As explained in the previous section, t DID documents can be located through DID that exists on the blockchain. Figure 3 contains an example of DID document to better understand such service endpoints of DWNs.



```
{
  "id": "did:example:123",
  "service": [{
    "id":"#dwn",
    "type": "DecentralizedWebNode",
    "serviceEndpoint": {
      "nodes": ["https://dwn.example.com", "https://example.org/dwn"
    }
  }]
}
```

Figure 10:Example of DID document

Addressing

The Decentralized web nodes are associated with a DID and the DID relative URL is used as an address to locate and request DWNs. The DID relative URL is composed of the base URI service parameter and queries. To construct a DID URL, which then later will be used to request the DWN(s), append a service parameter whose value will be 'DecentralizedWebNode', followed by the queries parameter. The queries parameter contains a Base64Url encoded array of JSON stringifies message descriptors. t the end, the DID-relative URL must look like below. [1]

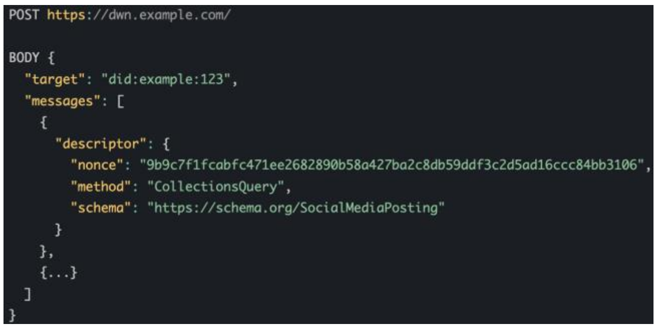
did:example:123 + ?service=DecentralizedWebNode + & queries= + toBase64Url(JSON.stringify([{ DESCRIPTOR_1 }, { DESCRIPTOR_N }])) [1]

An example of DID relative URL will look like below.

did:example:123?service=DecentralizedWebNode&queries =W3sgTUVTU0FHRV8xIH0sIHsgTUVTU0FHRV9OIH1d ... [1]

The above-discussed DID URL will assist to get the messages during sending the request. To get the DWN URL(s), DID document needs to be parsed. First, the DID document is located through the authority portion of the URL. Once the document is located, the second step is to locate the service that has type=DecentralizedWebNode. The service block in JSON is an array of JSON of various services and we are interested in the ones with DWN services. The third step is to read all the nodes under 'serviceEndPoint' that contains the URL pointing to DWNs. [1]

To construct the request object set the target property value to DID base URI. The message property contains an array of all these descriptors. These descriptors are retrieved from decoding the base64 queries string in the relative URL followed by parsing the string to JSON. An example of a POST request is shown in figure 4. [1]



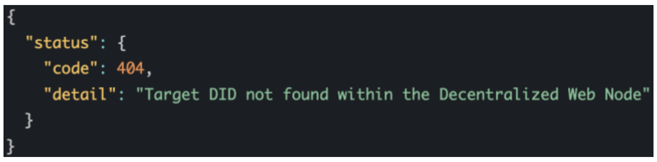
```
POST https://dwn.example.com/

BODY {
  "target": "did:example:123",
  "messages": [
    {
      "descriptor": {
        "nonce": "9b9c7f1fcabfc471ee2682890b58a427ba2c8db59ddf3c2d5ad16ccc84bb3106",
        "method": "CollectionsQuery",
        "schema": "https://schema.org/SocialMediaPosting"
      }
    },
    {...}
  ]
}
```

Figure 11:Example of HTTP POST

Response

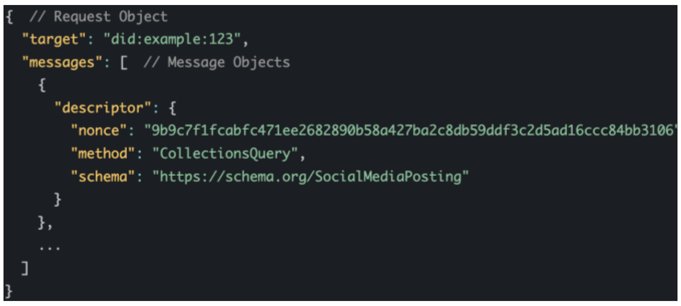
Once the request is made the interface method invocation sends back the responses in a JSON format. If the requested DWN node is not found the response sends back the 404 responses like in figure 5.



```
{
  "status": {
    "code": 404,
    "detail": "Target DID not found within the Decentralized Web Node"
  }
}
```

Figure 12:Example of Response object

In case the messages contained in request are processed fine, the key – entries contain the set of results along with the status code. One example of successful response is depicted in figure 6.

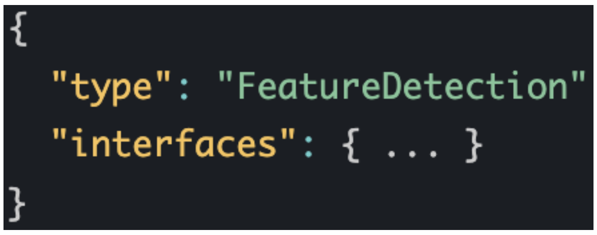


```
{  // Request Object
  "target": "did:example:123",
  "messages": [  // Message Objects
    {
      "descriptor": {
        "nonce": "9b9c7f1fcabfc471ee2682890b58a427ba2c8db59ddf3c2d5ad16ccc84bb3106'
        "method": "CollectionsQuery",
        "schema": "https://schema.org/SocialMediaPosting"
      }
    },
    ...
  ]
}
```

Figure 13:. Successful response

Interfaces

A web node contains multiple features that are exposed through creating interfaces. A decentralized web node must construct the feature detection object as showcased in figure 14.



```
{
  "type": "FeatureDetection"
  "interfaces": { ... }
}
```

Figure 14:Feature Detection Object

The interfaces can contain the objects of three types-

- Collection: Decentralized Web Nodes' Collections interface provides a framework for storing data in relation to shared schemas. By storing the data as defined by the schema, it promotes the storage of data on various devices and platforms. By mentioning the schema or recordId in the request the specific data can be queried from DWN. And similarly, the data and be written in DWN using the collections.
- Threads: Threads are a linked series of communications that are thematically related and are intended to result in activities undertaken by entities participating in the message thread.
- Permissions: Permissions interface allows others to request specific access to the resources in DWN that might be private and encrypted. For example, Facebook wants to access someone's daily visited website list in order to construct advertisement recommendations for the same user. In this case, through the permission interface, Facebook with their DID will be able to request to read these details from user's DWN.

Owning self identity and data

One of the great concerns about digital identity is the centralization of our identities that comes as convenient to use since it is created and maintained by big market players. One downside of companies owning our identity is there are multiple identities of the same person that exists on the internet. Our main purpose here is to allow users to own their identity and data and support their easy portability. A decentralized system could make it possible for a user to hold to its identity and data.

With the three components discussed in this paper DID, verified credentials, and decentralized web node, the ownership of data and identity can be returned to individuals.

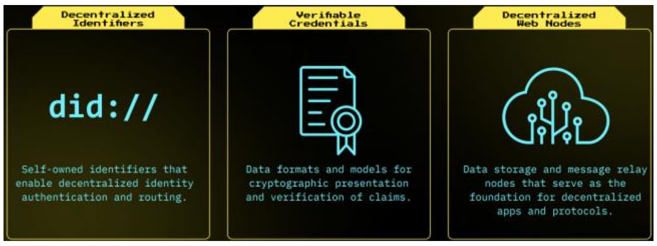

Decentralized Identifiers
did://
Self-owned identifiers that enable decentralized identity authentication and routing.
Verifiable Credentials
Data formats and models for cryptographic presentation and verification of claims.
Decentralized Web Nodes
Data storage and message relay nodes that serve as the foundation for decentralized apps and protocols.

Figure 15:Three pillars of self-sovereign identity and data

DID links the user and its information. DID is the only component that exists on the blockchain and points to DID documents that could exist on IPFS. DID uniquely identifies a user, but the extent of its purpose is to just identify the user. Now, if a user wants to authenticate itself to be related to an organization or a person, it can get a verified credential from that organization. Verified credentials from a trusted source will contain the digital signature of both user and VC provider. For example, a person can be identified with their DID but to be assured that he is a citizen of a country, he will be required to get a VC from the government of that country's DID. In addition, a decentralized web node will store the public and private data of the user. DWN allows users to store and sync data on multiple nodes so that the data is still available in case a node does not exist anymore. The private data will be encrypted and will be only accessible to its owner. Other persons or organizations with different DID will have access to public data, but to access private data they will need to request DWN with permission parameters. This way users will have full authority of their data and can choose to grant or refuse access to a third person. Figure 8 shows the communication between two parties identifying each other and sharing the data securely in a decentralized system.

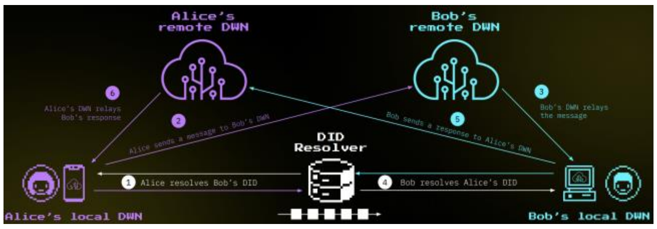

Alice's remote DWN
Bob's remote DWN
6 Alice's DWN relays Bob's response
2 Alice sends a message to Bob's DWN
5 Bob sends a response to Alice's DWN
3 Bob's DWN relays the message
DID Resolver
1 Alice resolves Bob's DID
4 Bob resolves Alice's DID
Alice's local DWN
Bob's local DWN

Figure 16: Secure communication and data sharing though combination of DID, VC and DWN [2]

References

[1] Jones, A. (2022, July 1). What is web5? TBD. Retrieved July 26, 2022, from https://developer.tbd.website/blog/what-is-web5/

[2] Decentralized identifiers (dids) v1.0. (n.d.). Retrieved July 26, 2022, from https://www.w3.org/TR/did-core/

[1] "Together we're building a new identity ecosystem," *DIF - Decentralized Identity Foundation*. [Online]. Available: https://identity.foundation/. [Accessed: 23-Jul-2022].

[2] "Developers," *TBD*. [Online]. Available: https://developer.tbd.website/. [Accessed: 23-Jul-2022].

[3] B. Alzahrani, "An information-centric networking based registry for Decentralized Identifiers and verifiable credentials," IEEE Access, vol. 8, pp. 137198–137208, 2020.

[4] "Intelligent diagramming," Lucidchart. [Online]. Available: https://www.lucidchart.com/pages/. [Accessed: 23-Jul-2022].

[5] Z. Li, "A verifiable credentials system with privacy-preserving based on Blockchain," Journal of Information Security, vol. 13, no. 02, pp. 43–65, 2022.

[6] Z. Li, "A verifiable credentials system with privacy-preserving based on Blockchain," Journal of Information Security, vol. 13, no. 02, pp. 43–65, 2022.