Self Sovereign Digital Identity on the Blockchain using Hyperledger Indy

[1]Akshada G .Khedekar & [2] Anuja T. Bhondave
[1]Department of Information Technology, AISSMS Institute of information technology, Pune
[2]Department of Computer Engineering, Pimpri Chinchwad College of Engineering, Ravet

Received: March 08, 2019 Accepted: April 13, 2019

ABSTRACT: : *Digital identity is one of the oldest and hardest problems on the Internet. In a distributed or decentralized domain identity information is decentralized where different Organizations or agencies use different identity attributes to verify you at for getting multiple services. Individuals do not have any control over the identity information. Decentralized identity management while ensuring the self-sovereign identity & distributed trust model is the biggest challenge of today's digital identity management. Blockchain Platform can help to realize a self-sovereign identity - lifetime portable digital identity that does not depend on any central authority and can never be taken away & that puts the user in control of identity information, by enabling a decentralized way to handle public key infrastructure .In the current contribution ,we present the Hyperledger Indy app that utilizes blockchain technology to create a secure protocol for storing encrypted identity information ,as well as sharing verifiable claims about personal information .*

Key Words: *Blockchain, Digital identity, Distributed ledger technology, Hyperledger Indy.*

Introduction

Questions of identity permeate human history. In 2017, Facebook was fined in France and Spain for violating the privacy laws of the users. The Spanish data protection authority (AEPD) said Facebook had been breaking privacy rules on multiple times over the way it uses people's personal data for advertising purposes[1].

In most privacy violation incidents, the identity owners have no idea about the privacy violation, because they can't see what's happening inside the identity management system. Another drawback is the lack of controllability of your own identity data.There is still many problems with contemporary solutions to digital identity as shown by the many hacks that leaked personal identity information.

The present contribution proposes to use blockchain technology which provides tamper-proof storage of identity data using advanced cryptography. Decentralized blockchain technology can increase the security of identity data as single point of trust is not needed. Cryptography gives users direct control over their own identity data with low chances of being hacked.

Related Work

Digital identity is one of the hardest problem on the Internet In the physical world, physical credentials are used to prove identity but there is no an equivalent solution on the internet to prove your identity .every user on the internet have hundreds of username & passwords to manage .Additionally personal details are scattered across a multiple databases guarded by companies who are incapable of keeping it safe[1].

Centralized Identity management systems are controlled by a single authority. Later on federated identity management system used where federated authorities started controlling identity management systems which allowed users to utilize the same identity in multiple places. But again federated identity was centralized at the identity provider's side and sharing identities were not transparent to the identity owner. Then, user-centric identity which allow the identity owners to have complete control of their digital identities, But the identities were bound to the identity register [2].

Public key infrastructure (PKI) is the contemporary solutions for verifying the digital signatures of credential issuers. The fundamental problem with PKI is that it is cumbersome, centralized and costly.
An alternative approach to centrally managed identity is a self sovereign identity, where an individual has full control over their identity information, the individual also controls who has access to their personal information, and security assurances provide the privacy of the information is not compromised.

Blockchain platform is a decentralized root of trust that nobody owns but everybody can use it. blockchain technology allows any user to verify blockchain data & also allows the verification of other data,

such as verifiable claims about identity.Blockchain platform works as a decentralized system for sharing claims about the identity of individuals .Instead of relying on CA's or government to be a cryptographic root of trust, anyone can sign data and choose which data signatories they wish to trust[3]

To enable a self-sovereign digital identity, the W3C has proposed Decentralized Identifiers (DIDs). DIDs are globally unique identifiers that use cryptography to generate proofs of data that can be resolved using the identifiers. The DIDs can provide a standardized format for sharing of verifiable claims about a user's identity[3]

In the present contribution, we will look into an use case of a digital identity management with the help of a Hyperledger Indy platform. Hyperledger Indy is distributed ledger platform for decentralized identity management. The Hyperledger Indy platform allows the user to generate their cryptographic key, send their identity data, encrypt it, and publish salted hashes of their identity data to the blockchain.

Motivational example

Consider a Bob, who uses the Hyperledger Indy app to manage their identity information. Hyperledger Indy is the blockchain platform for private secure & powerful identity management. Hyperledger Indy uses open-source, distributed ledger technology .These ledgers are a form of database provided by a pool of participants. Bob, a graduate of the Ioit College wants to apply for a job at the Yardli software .To get job Bob requires degree certificate from an institute Ioit as proof of his education on the job application.

In this example, The Hyperledger Indy works to realize a self –sovereign digital identity solutions by giving the Bob, control over his own identity data, as he can encrypt identity data & send it to an college themselves.

A. Evaluate a connection Request

When Bob clicks get Degree Certificate, he will download a file that holds an Indy connection request which will allow him to establish a secure channel of communication with Ioit College.

BOB> show Ioit-request.indy

```
{
  "Connection-request":
   "Name": "IOIT College",
   "DID":  "ULtgFQJe6bjiFbs7ke3NJD",
   "Nonce":  "b1134a647eb818069c089e7694f63e6d"
    "sig":" 4QKqkwv9gXmc3Sw7YFkGm2vdF6ViZz9FKZcNJGh6pjnjgBXRqZ17Sk8bUDSb6hsX"
}
```

B. Show connection Request

The connection contains several pieces of information.

BOB> show connection Ioit
Connection<not yet accepted>
Name: Ioit College
DID: not yet assigned
Trust anchor: Ioit College <not yet written to Indy>
Verification key: <empty>
Signing key: <hidden>
Remote: FuN98eH2eZybECWkofW6A9BKJxxnTatBCopfUiNxo6ZB
Remote Verification key: <unknown, waiting for sync
Remote endpoint: <unknown, waiting for sync>
Request nonce: b1134a647eb818069c089e7694f63e6d
Request status: not verified, remote verkey unknown

1. Connection name is Ioit College that Bob has been invited to accept. The name is stored locally and not shared.
2. **Distributed identifier** is a public key of Bob through which the Ioit College can verify Bob. Collection of all DID will constitute the self-sovereign identity of Bob .DID is generated by hyper ledger indy when Bob tries to accept the connection request. DID is shared with Ioit college & used by them to reference Bob in secure interaction. Each connection request on the indy network establishes a pair wise relationship when accepted .A pair wise relationship is a unique relationship

between two identity owners. The relationship between them is not sharable with others. DID contain public key information along with identity information of Bob like college identity number through which the college can verify the identity of Bob.

3. **Trust anchor: Ioit College**
   Trust anchor provide a way for DID's to be added to the ledger. They can be organization or persons as well. Ioit College is a trust anchor & if its connection request is accepted, will write Bob's DID to the ledger.
4. **Verification key: <empty>**
   Bob's verification key allows the ledger; network & college to trust that interaction with Bob are authentically bound as sender or receiver. Indy platform generate this value randomly when it load connection request.
5. **Signing key:<hidden>**
   A different **signing key** is used by Bob to interact with each party on the ledger. A signing key is an asymmetric private key. Indy generates it when it loaded the connection request. Bob will sign messages to Ioit College with this key. College will use the associated verification key to know it's really dealing with Bob.
6. **Remote Verification key: < unknown, waiting for sync>**
   Remote is the unique DID Bob uses to reference Ioit college. Ioit College provided this value in the connection request. Bob can use it to look up Ioit Institute's verification key on the ledger to ensure interactions with Ioit College are authentic.
7. **Request nonce: b1134a647eb818069c089e7694f63e6d**
   Nonce is just a big random number that Institute Ioit generated to track the unique connection request. A nonce is a random arbitrary number that can only be used one time. When a connection request is accepted, Bob digitally signs the nonce such that the Ioit College can match the acceptance with a prior request.

C. Accept a connection Request :

Accepting a connection request takes the nonce that Institute Ioit provided, and signs it with the Bob's signing key. Bob securely transmits the signed data along with the DID and verification key to Institute Ioit. Institute Ioit matches the provided nonce to the record of the nonce it sent to Bob, verifies the signature, then records Bob new pair wise DID into the ledger.

Once the connection is accepted and synchronized, Bob can see the remote verification key and remote endpoint, as well as DID and verification key are updated, which allows him to communicate with Ioit College. He can also see that the identity of the trust anchor was confirmed (from the ledger), and that his connection request has been accepted.

BOB> show connection Ioit
Connection
Name: Ioit College
DID: Z46KqKd1VrNFjXuVFUSY9
Trust anchor: Ioit College(confirmed)
Verification key: ~CoEeFmQtaCRMrTy5SCfLL
Signing key: <hidden>
Remote: FuN98eH2eZybECWkofW6A9BKJxxnTatBCopfUiNxo6ZB
Remote Verification key: <same as above >
Remote endpoint: 10.20.30.101:5555
Request nonce: b1134a647eb818069c089e7694f63e6d
Request status: Accepted
Available Claim(s): Degree Certificate

D. Inspect the claim :

A claim is a piece of information about an identity .claims is offered by an issuer. An issuer may be any identity owner known to ledger who issues a claim about any identity owner it can verify. Bob asks for Degree certificate as a verifiable claim issued by institute Ioit. A schema for degree certificate has been written to the ledger.

BOB> show claim Degree Certificate
Found claim Degree Certificate in connection Ioit College

Status: available (not yet issued)
Name: Degree Certificate
Version: 1.2
Attributes:
Student name
Degree
Year
Status

Bob sees the attributes the Degree certificate contains. These attributes are known because a schema for Degree certificate has been written to the ledger .However, the "not yet issued" means that Degree certificate has not been delivered to Bob in a usable form.

To get the Degree certificate, Bob needs to request it.

BOB> show claim Degree Certificate
Found claim Degree Certificate in connection Ioit College

Status: 2019-05-01 12:32:17.497455
Name: Degree Certificate
Version: 1.2
Attributes:
Student Name: Bob
Degree: Bachelor of Computer Engineering,
Year: 2019
Status: graduated

Now the certificate has been issued

E. Apply for a job

Now Bob would like to work for the company, Yardly software. Indy app ask bob to accept the connection with Yardly Software .Connection request contains a proof request .proof request is made by the party who needs verifiable proof of certain attributes that can be provided by other verified claims..

Yardly software is requesting the Bob to provide a job application .The job application is a document that has a schema defined on the ledger. Bob go through the sequence of commands that establish a new pair wise connection with yardly software.

BOB> show proof request Job-Application
Found proof request "Job-Application" in connection "Yardly Software"
Status: Requested
Name: Job-Application
Version: 0.2
Attributes:
First- Name: string
Last-Name: string
Phone-Number: string
Degree (V): Bachelor of Computer Engineering
Status (V): graduated

The Proof is constructed from the following claims:
Claim (Degree Certificate v1.2 from Ioit College)
Student Name: Bob
Degree: Bachelor of Computer Engineering
Year: 2019
status: graduated

Bob only has one claim that meets proof requirements for this Job Application, so it is associated automatically with the request & this is how some of his attributes are pre-populated. The pre-population doesn't create data leakage though the request is still pending. Bob can edit what he is willing to supply for each requested attribute.

BOB> set First_Name to Bob
BOB > set Last_Name to Bush
BOB > set Phone_Number to 987654321

Bob checks to see what the proof request looks like now.

BOB> show proof request Job-Application
Found proof request "Job-Application" in connection "Yardly Software"
Status: Requested
Name: Job-Application
Version: 0.2
Attributes:
First- Name:Bob
Last_Name:Bush
Phone_Number:987654321
Degree (V): Bachelor of Computer Engineering
Status (V): graduated

Bob decides to submit.

BOB > send proof Job-Application to Yardly Software
Signature accepted.
Response from Yardly Software (15.98 ms):
Your Proof Job-Application 0.2 was received and verified
Response from Yardly Software (23.95 ms):
Available Claim(s): Job-Certificate

The application is accepted, and Bob ends up getting the job. When Bob inspects his connection with Yardly Software a week later, he sees that a new claim is available:

BOB > show connection Yardly Software
Connection
Name: Yardly Software
DID: QANW5P3tjRX8Q8w8iyN9A5
Trust anchor: Yardly Software (confirmed)
Verification key: ~KdJUJwAq6Wj8To8pJgGHqE
Signing key: <hidden>
Remote: CzkavE58zgX7rUMrzSinLr
Remote Verification key: <same as remote>
Remote endpoint: 10.20.30.102:6666
Request nonce: 57fbf9dc8c8e6acde33de98c6d747b28c
Request status: Accepted
Proof Request(s): Job-Application
Available Claim(s): Job-Certificate

Bob goes through a familiar sequence of interactions to get proof of employment from Yardly Software.

Conclusion

We presented the Sora identity system to manage decentralized, self-sovereign identities on the block chain. The sora identity protocol is a practical, production ready method for managing decentralized, verifiable claims about identity using a block chain system.

Our system uses mobile applications to allow users to interact with the, Hyperledger Indy, in order to digitally sign & share proofs of their identity information.

References

1. Sovrin™: "A Protocol and Token for Self- Sovereign Identity and Decentralized Trust",jan 2018.
2. Ayamine Alupotha,"The Rise of Self-Sovereign Identity: Hyperledger Indy",DZone, Jul. 13, 18
3. Makoto Takemiya ,Bohdan Vanieiev," Sora Identity: Secure, Digital Identity on the Blockchain", IEEE International Conference on Computer Software & Applications, 22 June 2018.
4. Hyperledger indy , "https://github.com/hyperledger/indy-sdk", 2018.