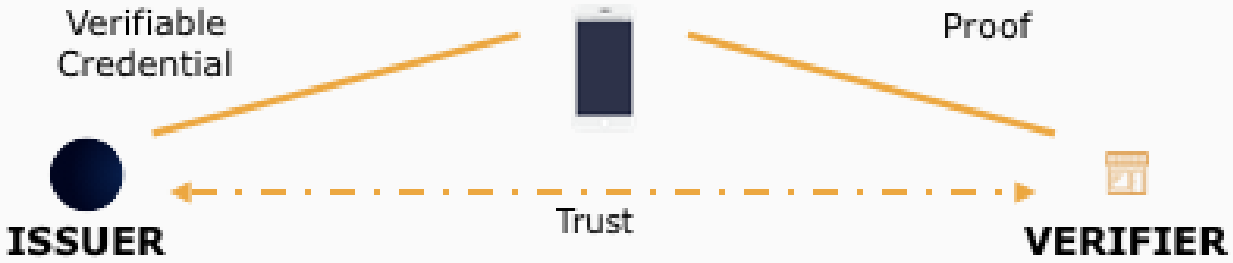
Introduction to Digital Identity with Hyperledger Ursa, Indy, Aries

How does identity work in the physical world

- People hold identification, credit cards, membership passes in wallets (credentials)
- These are presented to relying parties (proof)
- Relying parties verify these
  - Look
  - Scan barcodes
  - Check with blue lights
- Are they not forged and not revoked

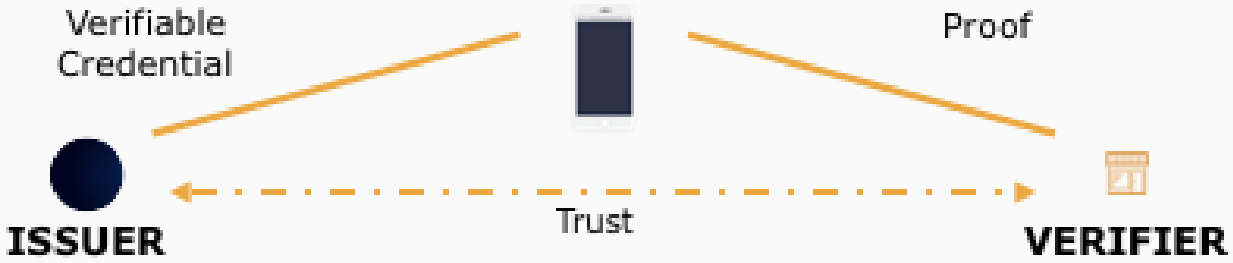
How does identity work in the physical world

- Three parties involved
- Issuers
- Holders
- Verfiers



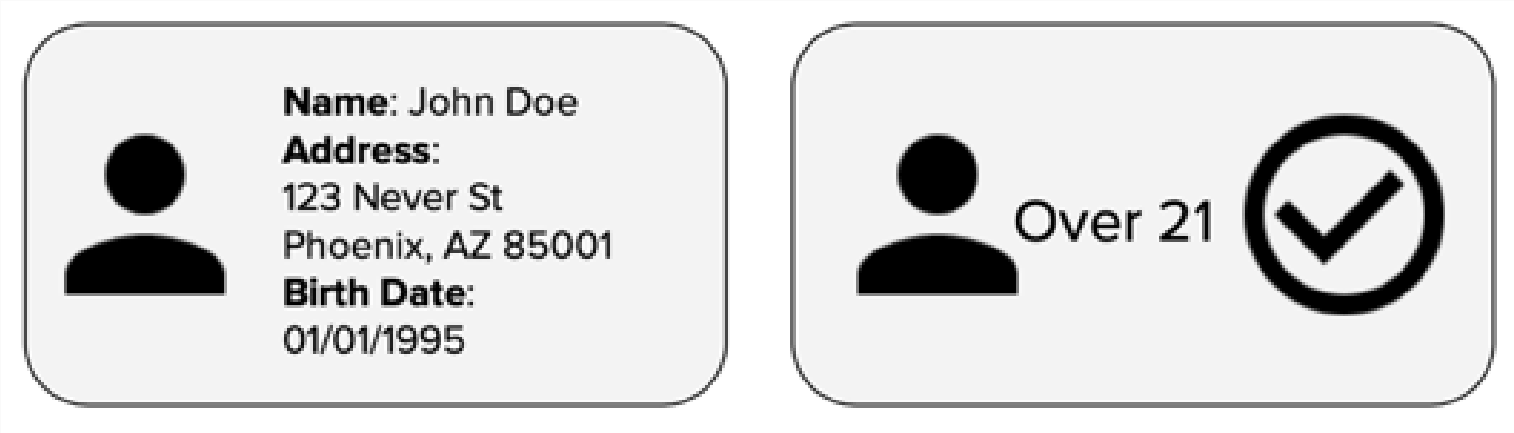
How does identity work in the digital world

- Sovrin (Indy) enables the same structure
- Issuers publish information on Sovrin
- Holders receive credentials from issuers
- Verifiers can check credentials



How can identity work in the digital world

No limit in revealing entire credential



How can identity work in the digital world

- Information hiding techniques
- Correlation reduction
- Doesn’t change validity of data
- Selective disclosure and Zero-knowledge proofs

A zero-knowledge protocol is a method by which one party can prove to another party (the verifier) that something is true, without revealing any information apart from the fact that this specific statement is true

The Knowledge Complexity of Interactive Proof Systems
Shafi Goldwasser, Silvio Micali and Charles Rackoff

What is Hyperledger Indy

- Blockchain for decentralized identity
- For validating verifiable credentials
- Data written by credential issuers
- Read by credential holders and verifiers
- Sovrin: largest deployment of Indy
- Other deployments: Kiva
  - Non-profit for people to lend money to low-income people in over 80 countries
  - Africa, Fiji

https://github.com/hyperledger/indy-sdk
https://github.com/hyperledger/indy-node

What goes on the Hyperledger Indy

- Issuer DID
- Credential Schema (Attribute descriptions)
- Attribute encoding
- Public keys
- Revocation registries
- Presentation descriptions

What is Hyperledger Aries

- Tool kit designed for creating, transmitting and storing verifiable digital credentials
- Anchor to blockchain
- Blockchain agnostic
- Key management
- DIDs
- Messaging

https://github.com/hyperledger/aries

Indy + Aries

- Build Apps for managing Self Sovereign Identity = Identity Wallet
- Messaging
  - Request credentials
  - Present credentials
  - General messages
  - Issue credentials
  - Revoke credentials
  - Actual message like “Hello”

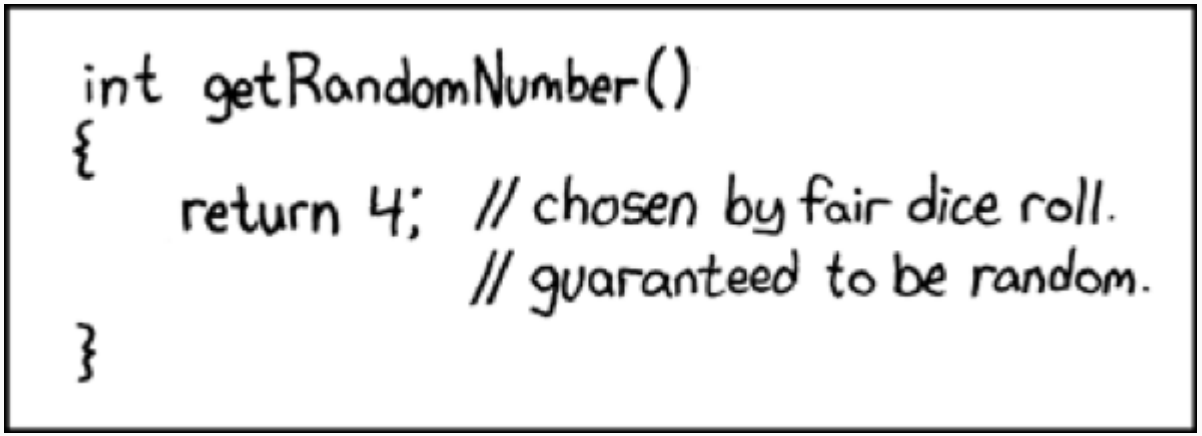
Hyperledger Ursa

Hyperledger Ursa is a shared crypto library that provides Hyperledger projects with safe interfaces to access high-quality implementations of cryptographic primitives and key management functions.

Put simply, Ursa brings high trust and security to users of Hyperledger Frameworks.

https://github.com/hyperledger/ursa

Hyperledger Ursa



Note: monoculture is bad, but own crypto is worse…

Hyperledger Ursa

- Reviewed by cryptographers and crypto engineers
- 2 Libraries: Libursa, Libzmix
- Libursa: Encryption, Hashing, Signatures, Key Exchange, HSM integration
- Libzmix: Zero Knowledge Proof primitives

Zero-knowledge Proof (ZKP) capabilities

20?

Every question in the game 20 questions is an example of how a zero knowledge proof request works. Like the game, if guesser asks good questions, can correlate too much information

Possible ZKPs

Equality (A1 = V1)

Inequality (A2 > 50K, 18 ≤ A3 ≤ 54, A1 ≠ “Name”)

Set Membership (A4 ∈ Set1, A4 ∉ Set2)

Enums, Revocation

Hidden (Don’t reveal A5-A10)

Possible ZKPs

Policies - Authorizations, Access Control

Delegation

Vouchers - Limited # of uses or time (voting, tickets)

Registration - Enrollment

Transferability

Q&A

Demo, Getting started

https://github.com/hyperledger/aries-cloudagent-python/tree/master/demo

https://github.com/hyperledger/aries-cloudagent-python/blob/master/docs/GettingStartedAriesDev/AriesDeveloperDemos.md

https://medium.com/@kctheservant/exploring-hyperledger-indy-through-indy-dev-example-10075d2547ae

Hyperledger Projects

- Core written in Rust
- Apache 2.0 License
- Wrappers written in any language consumes C API

Deep Dive Hyperledger Indy

https://github.com/hyperledger/indy-sdk/blob/master/docs/getting-started/indy-walkthrough.md

https://github.com/hyperledger/indy-sdk/tree/master/docs/design/012-libindy-architecture-overview

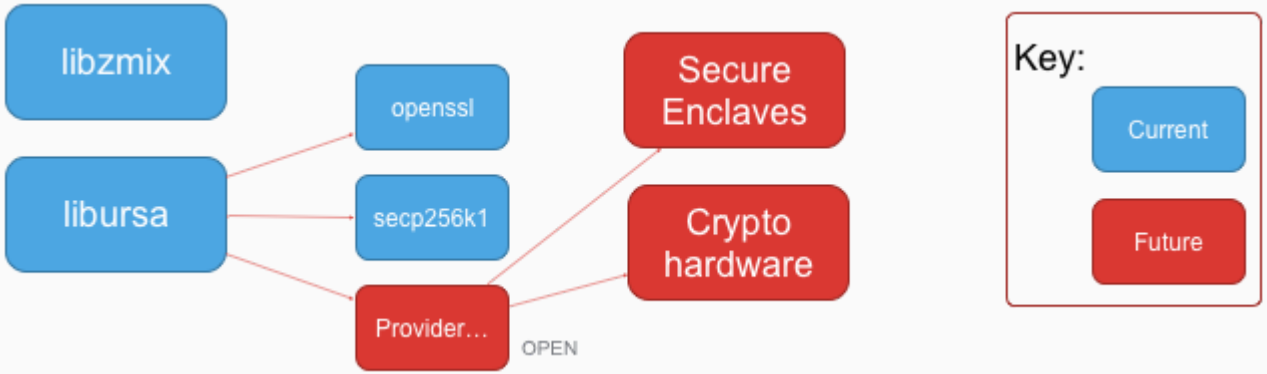
Deep Dive Hyperledger Ursa

- Ed25519 and Secp256k1 ECDH and E(CD)DSA
- BLS signatures using BN254
- PS, BBS+, Delegatable signatures
- Bulletproofs: intervals and set memberships

Deep Dive Hyperledger Ursa

- Developer friendly interface
  - Misuse resistant, Non experts, Hide many details
- Crypto developer interface
  - Flexible, Composable, Descriptive, Experts
- https://github.com/hyperledger/ursa
- https://github.com/hyperledger/ursa-rfcs
- https://github.com/hyperledger/ursa-docs

Deep Dive Hyperledger Ursa



Deep Dive Hyperledger Aries

https://github.com/hyperledger/aries

https://github.com/hyperledger/aries-rfcs

Integration

- Indy - Transaction signatures (BLS/BN254)
- Iroha - Transaction signatures (Ed25519)
- Sawtooth - Transaction signatures (P256k1) (Coming)
- Aries
  - ZKP protocols (BBS+, PS, Bulletproofs, Delegatable signatures)
  - Wallet security (Encryption, Hashing, Signatures)