Establishing the authenticity of Educational Certificates using Blockchain

Namita Sham | Sharmeen Salehmohamed

The Fake Certificate Problem

The problem

- Anyone can pay to buy a qualification at any level

The figures

- 85 fake universities closed in the last 5 years in the UK
- 3,300+ unrecognized “degree mills”
- 50000+ fake PhDs bought per year

The impact

- Legitimate educational institutions
  - Negative image on online schools
  - Expenses in intellectual property protection
- Consumers
  - Can be tricked into buying fake qualifications
  - Unfair to legitimately qualified individuals
- National Economy
  - Funding investigation
  - Negative reputation

How Blockchain Can Help

- Publishing certificates on the blockchain to be seen and distributed to employers for verification
- 3 main tasks
  - Identifying Certification Authorities
  - Issuing certificates
  - Verification by employers
- Example
  - Forgery-proof BC Diploma by MANCOSA
  - Encryption + Data Storage on Ethereum BC

The Business Problem

The Fake Certificate Problem

Fake credentials

- Accredited by fake educational institutions
- Created by fake certifying bodies to mimic real educational institutions
- Modified grades from a real institution

The needs for a trustworthy single source of truth which is tamper-resistant

Why Blockchain?

Decentralized

- Restricts unauthorized altering and deletion

An incorruptible digital ledger

- Acting as a public DB storing issuer and holder

Ease of authentication

- No need for a 3rd party

Success Criteria

- Authenticity
- Resilience
- Privacy Preservation
- Real-time online verification
- Third-party verification
- Usability
- Revocation

Technologies and Platforms

Available Technologies

Blockcerts

Open standard for building apps for blockchain-based official records verification

Hyperledger Fabric

Business-ready open blockchain platform whose environment is less complex to develop

Chosen Platform

Oracle Blockchain Cloud Platform

- PaaS
- Quick app development
- Caters for all dependencies of a BC network, easily manageable through Cloud Platform Console
- Support multi-cloud, hybrid blockchain network

Deployment and Integration

Existing Deployment Models

- Encrypted digitally signed certificates issues
- Sent onto the blockchain and consensus achieved
- Authenticity verified through an interface – web or mobile app

Integration with Existing Systems

Certification awarding system

- Remove middlemen
  Track authenticity of progress in online courses
- Upon completion of subsections of online courses

Smart Contracts (Chaincodes) on the Platform

Chaincodes will handle

- Creation of certification credentials
  - Issuer + holder + certificate identifier + credential ID
- Revocation/expiry
- Querying for validation

The Oracle Blockchain Platform updates the blockchain ledger and responds to queries by executing chaincode.

How they work?

1. External applications invoke transactions or run queries through client SDKs or REST API calls, which prompts selected peers to run the chaincodes.
2. Multiple peers digitally sign the results, which are then verified and sent to the ordering service.
3. After consensus is reached on the transaction order, transaction results are grouped into cryptographically secured, tamper-proof data blocks and sent to peer nodes to be validated and appended to the ledger.

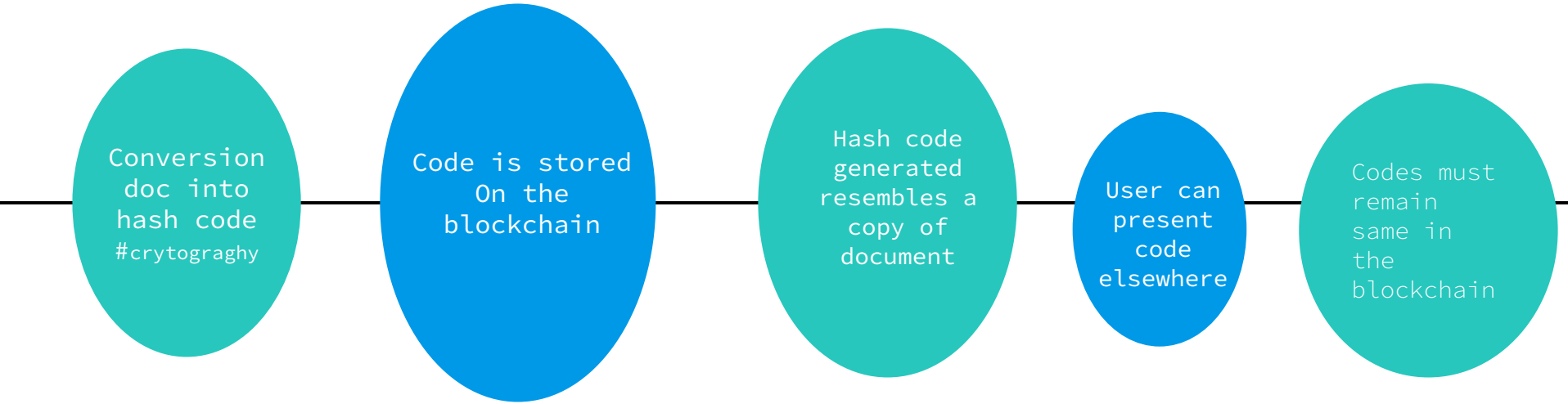
Data Privacy and Sharing

Storing certificates on proposed system

Certificates are placed on a distributed ledger so users can access information anytime, anywhere and any place.

The documents are keep secure allowing only authorized users to access it by the use of private key.

How university can implement blockchain to store the certificates?



But represented as string codes

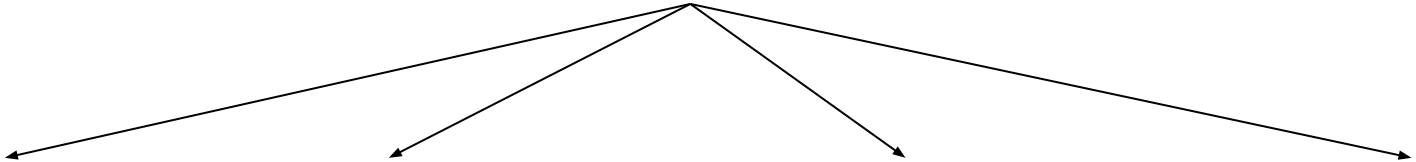
-https://emn178.github.io/online-tools/sha256.html

Details stored to prevent fake online certifications :

- A signed digital certificate containing the owner’s distinguished name
- Owner’s public key
- Certificate authority’s distinguished name
- Signature of the certificate authority over the fields

Types of permissions to use and why?

Types of permissions



Public Blockchain

- Decentralized
- Permissionless
- Operates publicly

Private Blockchain

- Centralized
- Permissioned
- Operates privately

Consortium Blockchain

- Semi-centralized
- Like private blockchain
- But allow multiple organization

Hybrid Blockchain

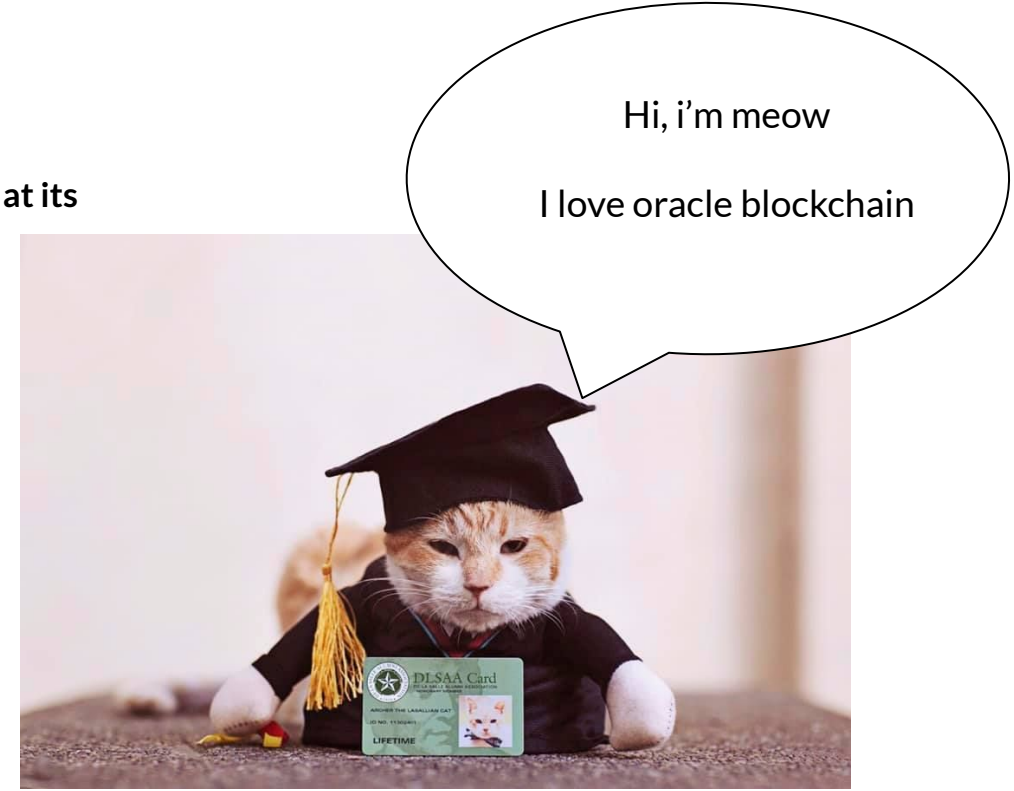
- Combination of private and public
- Allow only a selected section data to be public, keeping the rest private

Public, private, consortium or hybrid?

Consortium Blockchain

-Allow multiple organizations

Assume, meow participated in a small competition at its university in collaboration with oracle.



- **Public blockchain**, no control, difficult to hold accountable

- If we allow **private blockchain**, only one organization has access.

- If we allow **hybrid blockchain**, inability to add multiple organizations

-semi-centralized, allowing security organizations to work together.

Data ownership, Privacy and sharing

1. Data ownership

Personal data can be in a separate database and indexes are given to the blockchain in order to access.

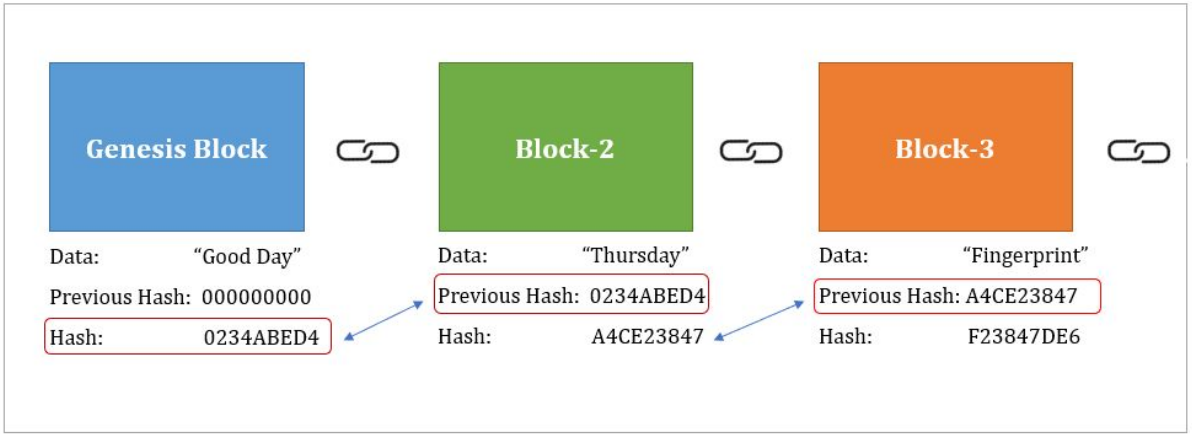
- Institutions would be able to share the blockchain while keeping personal data of students private.

- Students would be allowed to modify and delete from the database

2. Data Privacy

Issuing of certificates can be kept on blockchain in encrypted form using cryptographic hash functions

E.g MD5



3. Data Sharing

Students can publicly share their certifications to different organizations, which will difficult by hackers to attack or manipulate with the certifications.

Data sharing in blockchain would allow students to create profiles, store and share their data to others as per agreement.

How organizations validate certificates for individuals that they are hiring?

- Recruiter requests certificates for verification
- Recruiter can accomplish this by slotting the key generated from hashing document into the institution's portal where it includes database of all their legit graduates.
- If hash matches the details stored on the institution's blockchain. Hence, the certificates are genuine.

Chaincode

REST APIs required to call validation

System chaincode; used for validation system, query system and endorser system

REST API's for the oracle blockchain platform that would get called are :

- **Get chaincode information**: Method GET
  Path `/console/admin/api/v1.1/chaincodes/{chaincodeName}`
- **Get installed chaincode List:** Method GET
  path `/console/admin/api/v1.1/chaincodes`
- **Install chaincode:** Method POST
  path `/console/admin/api/v1.1/chaincodes`
- **Instantiate Chaincode:** Method POST
  path `/console/admin/api/v1.1/chaincodes/{chaincodeName}/instantiate`

Channels that would be created

- Use a corresponding public key for tracking public state
- Chaincode access control
- Sharing private data out of band
- Sharing private data with other collections
- Transferring private data to other collections
- Using private data for transaction approval.
- Keeping transactors private

Algorithm for Chaincode

We must abide by the rules for a node.js chaincode such as:

- package.json must be in the root directory.
- The entry JavaScript file can be located anywhere in the package.
- If "start" : "node <*start*>.js" isn't specified in the package.json, server.js must be in the root directory.

Then, we place the chaincode and package file in a zip file to install it on Oracle Blockchain Platform.

```
{
        "name": "chaincode_example02",
        "version": "1.0.0",
        "description": "chaincode_example02 chaincode
implemented in Node.js",
        "engines": {
                "node": ">=8.4.0",
                "npm": ">=5.3.0"
        },
        "scripts": { "start" : "node chaincode_example02.js" },
        "engine-strict": true,
        "license": "Apache-2.0",
        "dependencies": {
                "fabric-shim": "~1.3.0"
        }
}
```

Challenges

It is hard for novice people

Requires huge initial investment

Privacy and security of data,
As transactions are made transparent

Thank You!