Decentralized Certificate Verification Service Based on Hyperledger

Anuj Chaudhari, Nishantkumar Bhalani, Jay Desai, Tejas Panchal
Computer Engineering Department
San José State University (SJSU)
San José, CA, USA
Email: {nishantkumar.bhalani, anuj.chaudhari, jay.desai, tejas.panchal}@sjsu.edu

***Abstract*—A certificate holds sensitive information and is prone to fraud. The certificate verification is an essential step in establishing the authenticity of an applicant for an employment background check, university admission process or any other selection process. Currently, the certificate verification takes weeks to get completed as it requires interaction with the concerned institution. This results in the delay of the entire process. An organization may waste their valuable time and resources on fake candidate profiles until it authenticates the certificates. The project proposes a decentralized solution for certificate verification for academic institutions, employers and/or recruiters. The proposed solution uses Hyperledger Fabric to store educational certificate data secured with cryptography encryption. An end-to-end system is implemented which consists of a decentralized web-application, decentralized back-end as well as decentralized Hyperledger. Authenticated client user interacts with the web-application to upload or verify certificates. The service will help any entity to instantly verify the authenticity of a certificate just by uploading it to the system. This will lead to a significant reduction in manual efforts and processing time from weeks to minutes.**

***Index Terms*—blockchain, certificate, decentralized, fabric, hyperledger, verification**

I. Introduction

Acceptance of fraudulent certificates in the industry or university is the major problem being faced. There have been instances in past where fake certificates were mistakenly accepted and the applicants were offered jobs [1] [2]. Thus, it is necessary to verify and authenticate the certificates. Verifying the authenticity of a certificate is time-consuming and requires manual interventions with the issuer of certificates or the third-party service providers. Moreover, every recruitment or selection process requires a background check to be conducted on the candidate which currently takes about a month to complete. In a case where the candidate turns out to be fraudulent, the organization's time and resources end up being wasted for nothing.

Certificate verification on blockchain has been implemented on different open and public blockchains. Proposed solution uses Hyperledger Fabric [3] which is a permissioned blockchain. Members of the Hyperledger Fabric are trusted members authorized to use network by Membership Service Provider(MSP). This helps to control the access to ledgers and create a more secured environment to store the sensitive certificates. Hyperledger Fabric creates different ledgers for different organizations which makes each organization isolated with others.

II. Project Architecture

The system is comprised of four main components. The first component is the Desktop application that can be used by all actors to interact with the system. The second component is a middleware tier that is primarily used to serve login authentication and tracking user requests coming from the desktop application. The third component is related to database and IPFS[4]. The system stores digitalized version of diploma certificate on IPFS and use a decentralized database of Hyperledger Fabric to store user, university and certificate-related data. The last component is a blockchain network running on the IBM cloud infrastructure that serves as a base for decentralized infrastructure. It also includes core components of Hyperledger fabric like CA, organizations, peers, orderers, and ledger and represents interactions between them.

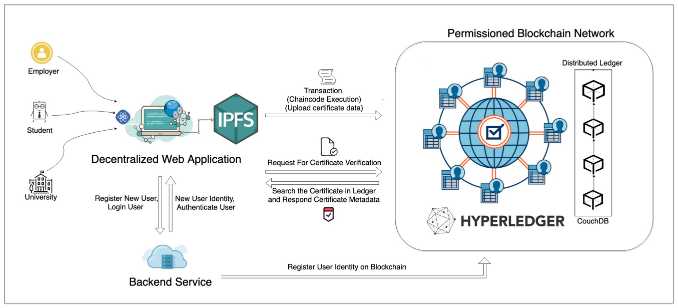

Employer
Student
University
Decentralized Web Application
IPFS
Transaction
(Chaincode Execution)
(Upload certificate data)
Request For Certificate Verification
Search the Certificate in Ledger
and Respond Certificate Metadata
Register New User,
Login User
New User Identity,
Authenticate User
Backend Service
Register User Identity on Blockchain
Permissioned Blockchain Network
Distributed Ledger
HYPERLEDGER
CouchDB

Fig. 1: Architecture Diagram

A. Decentralized Desktop Application

The desktop application is the first entry point of the application. The desktop application has two parts. The first part is a user interface that enables users to interact with the system. It has different dashboards for 3 different actors. Students can use this application to see their uploaded certificates. They can also share these digitalized certificates to employers. University can use this dashboard to register itself to be able to upload verified student documents to the blockchain network. An employer can use this application to verify the diploma document provided by the student. The second part of the application is the Hyperledger fabric SDK that enables the UI to interact with the Hyperledger fabric setup running on the cloud.

B. Middleware tier

Middle tier consists of the centralized back-end service which interacts with the desktop application as well as the mongodb service to manage users and authentication. Users login to the desktop application using the credentials which consumes the back-end service. Back-end service parses the user credentials and interacts with mongodb to authenticate the user. User identity is returned to the desktop application and is verified with network. Back-end service is implemented using spring boot and restful web services. Spring HATEOAS[5] is used to interact with the mongodb service. Back-end service exploits the hyperledger-java sdk[6] to interact with the hyperledger blockchain network to register new users.

C. Database tier

Database layer has mainly 2 types of data certificated related data and digitalized certificate itself. When the university uses the desktop application to upload the certificate, the system performs 2 steps. In the first step, the system uploads the copy of the document to IPFS and in the second step, it creates the entry in the Hyperledgers database. This entry has all the details about the previous transaction like the issuer, student, graduation term, diploma UUID and department.

D. Network tier

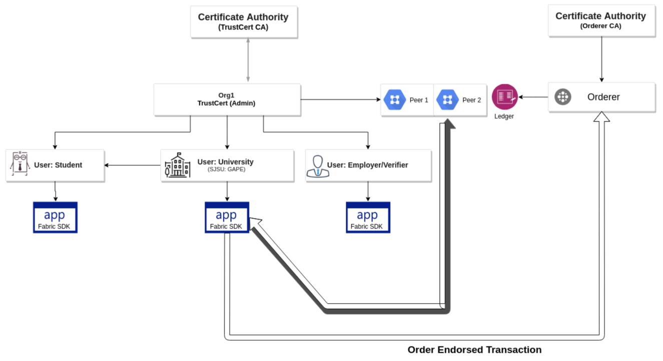


Fig. 2: Network Diagram

A certificate authority provides a capability for organizations to be able to interact with the blockchain network. It issues certificates to organizations with which they are able to interact with the blockchain system. Organizations then creates users like student, university, and employer and provides them an identity. Once users have the identity, they can interact with the blockchain system using SDK. SDK prepares transactions proposed by the user actions and interacts with the peers to confirm them. Once the transaction is confirmed, it is sent to the orderer to commit to the ledger.

E. Hyperledger Blockchain network

The core part of the system is the Hyperledger Blockchain network that is deployed on the IBM cloud infrastructure. To spinoff, the core components like orderer, peer, ledger, and organization IBM uses Kubernetes cluster. It runs these entities as containers and provides APIs to interact with the network. The desktop application is configured to interact with this cloud environment using APIs. When a user tries to interact with the system, the Hyperledger fabric node sdk[7] invokes chain code deployed on this cloud environment.

III. System Design

The system has three types of users. University, Student and Employer or anyone who wants to check the authenticity of the diploma. Below is the flow and verification protocol diagram of the proposed solution which involves how each user interacts with the hyperledger blockchain network to securely store, view and verify the diploma.

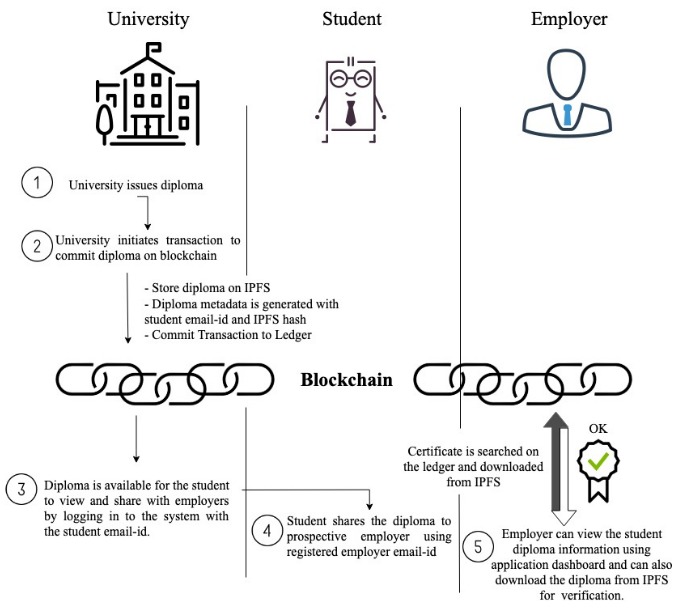

University
Student
Employer
1 University issues diploma
2 University initiates transaction to commit diploma on blockchain
- Store diploma on IPFS
- Diploma metadata is generated with student email-id and IPFS hash
- Commit Transaction to Ledger
Blockchain
3 Diploma is available for the student to view and share with employers by logging in to the system with the student email-id.
4 Student shares the diploma to prospective employer using registered employer email-id
Certificate is searched on the ledger and downloaded from IPFS
OK
5 Employer can view the student diploma information using application dashboard and can also download the diploma from IPFS for verification.

Fig. 3: System Workflow

Below are the steps for university users to upload and view certificates

1) University issues certificate
2) University initiates transaction to commit the certificate on blockchain
3) Certificate is uploaded on IPFS and transaction is committed to ledger with student information and the IPFS hash. University can see the uploaded student certificate on application dashboard.
4) University has the privilege to upload new updated certificate of the student if there is any change in certificate but the older certificate will always be there on the blockchain network and no one can delete that certificate.

Below are the steps for students to view and share certificates

1) Once the certificate is uploaded on blockchain by university, student can create the account with the university email or the email associated with the university
2) Student will be able to view the certificate information on the dashboard and can also download the certificate
3) Student can share the certificate to the employer or anyone who wants to verify the certificate using the email
4) Student do not have any other privileges other then viewing and sharing the certificate, so student can not update or upload new the certificate to the blockchain

Below are the steps for employers to view and verify certificates

1) Employer can create an account with email address which will be sent to student to share the certificates
2) Employers will be able to view information about the certificates and also download the certificates that students have shared

A. Decentralized Desktop Application

The desktop application involves two parts. The front-end server to serve the user interface and the back-end server which communicates with the blockchain network.

1) Front-End Server: The front-end essentially serves user interface for registering, viewing, uploading and sharing the certificate as well as verifying the authenticity of the certificate. This server is developed using ReactJS, a javascript framework which gives responsive component based design. The server is configured with an desktop application framework called Electron which converts web application into desktop application. The front-end server will be run on the user machine and will directly communicates with local back-end server using REST api calls.

2) Back-End Server: The back-end server is the backbone of the entire application which serves data to the front-end, communicates with the hyperledger fabric blockchain network, handles identity management in wallet as well as authentication. Back-end service uses nodejs and fabric sdk for nodejs communicating with IBM hyperledger fabric network. The server also handles the authentication with the blockchain network so only the authenticated users can communicate with the network. This is done based on the user role who is logged in currently. The server is also responsible for uploading the student certificate to the InterPlanetary File System (IPFS) and fetching the certificate back when employee request the certificate. The server uses *js-ipfs* which is nodejs library for communicating with IPFS. The back-end server communicates with the centralized server for registering new users which assigns them a unique identity in the hyperledger fabric blockchain network which user has requested.

B. Centralized Back-End Server

The purpose of the centralized back-end server is to allow different types of user to register with the application which will allow them to view, upload, share and verify the certificates based on their role and identity. A centralized back-end service is implemented which can be consumed by the decentralized desktop application.

Back-end service is developed on the principles of the Restful Web Service. It is implemented in Spring boot serving as an ideal back-end service to use being flexible and offering functionalities and libraries to interact with the database. MongoDB is used as the database service for the back-end service to interact. MongoDB being a NoSQL database offers schema less architecture perfectly suited for the proposed solution.

Back-end service handles functionalities to create and register new employers and universities. Back-end service interacts with the fabric-java-sdk[6] to register the user with the network. When user is registered after filling the registration form, an unique admin identity is generated and registered on the blockchain for the user. This identity is used by the desktop application to communicate directly with the blockchain network. On the registration, email link is sent to the registrant to verify email address and confirm the registration. Once user is verified, user is stored in the database as verified user. Students can register themselves using the application. Once registered, students can add the additional email addresses and retrieve the certificates own by them. Back-end service provides login authentication for the users to interact with application using the login credentials set by user. These credentials are verified against the stored credentials and if verified, a successful login happens.

C. Attribute-Based-Access-Control

Given the sensitivity of the information stored in the network, it is must to control who can access the resources and to what extent. The project implements Attribute-Based-Access-Control[8] based on attribute ”Role” in addition to other security tactics for the purpose of maintaining a secure and immutable diploma ledger.

The system is built on permissioned blockchain network called Hyperledger Fabric which ensures the first level of security by allowing only the authorized entities within the network. Different entities i.e. users within the networks are classified into three different roles: (1) University (2) Student (3) Employer (Verifier). This role information is infused within the certificate as an attribute of the user at the time of registration with the network. Before any interaction with the ledger, requesting user’s role is verified and only permitted services based on the role are granted access to. Below are the roles, their permissions and responsibilities.

- University (Read/Write):
  - Query uploaded diploma
  - Upload diploma and write details to ledger
- Student: Read Only
  - Query the ledger only to view their own diploma
- Employer: Read Only
  - Query the ledger only to view the diploma that are shared with employer

Attempting to disguise as having a different role prevents the user from even logging into the network. This role and certificate based verification is done by the TrustCert chaincode which is installed within each peer. This kind of access control ensures a secure environment where only trusted entities are allowed with controlled actions.

D. TrustCert Chaincode

The chaincode is an important component of the Fabric network and handles the business logic of the system. The chaincode implemented in project ”TrustCert” is written in Golang and is built on Fabric core libraries cid, shim and peer. Any interaction with ledger can only be done through the chaincode installed on the channel which makes it very crucial in preventing any unauthorized access.

The TrustCert chaincode is responsible for following primary tasks:

- Submit transactions to ledger
- Validate invoking request
- Validate requesting user’s authorization
- Indexing ledger entries

The TrustCert chaincode is written to provide the functions to upload the diploma to ledger, to query the diploma from ledger for all three roles, to share student diploma with an employer and to fetch the role of the invoking user. Upon receiving an invoking request from the client application, the chaincode first validates whether the request parameters are in the expected format or not. In order to control the access to the ledger, chaincode also validates the attribute ”Role” of the requesting user identity from the user certificate. The ledger is accessed only if the role permits the user to do so.

To facilitate versatile queries on the ledger, the TrustCert chaincode generates indexes by creating composite keys in the ledger. This enables the user to query the diploma by multiple parameters and not only by the UUID of the diploma. The chaincode submits transactions to the orderer that updates the ledger.

E. Database Design

The TrustCert project maintains two separate databases, a CouchDB database serving as the ledger and a MongoDB database providing user verification services.

The system has couchDB database deployed on all the peers which maintains the state of the ledger on peers. The state of the couchDB database is updated upon submission of each transaction. All the peers must result in the same state after a transaction is submitted by the orderer. The diploma metadata is stored as a key-value pair in base-64 format in the ledger and has the structure as the following:

```
type DiplomaMetadata struct {
        Timestamp     string
        DiplomaUUID   string
        Issuer        string
        Term          string
        Degree        string
        Department    string
        Name          string
        EmailId       string
        IpfsLink      string
}
```

Apart from the diploma metadata, the couchDB also stores the composite-keys created for the purpose of indexing and querying. Composite keys of employer and student email-ids are generated with the diploma UUID to enable queries with different parameters.

MongoDB is deployed on the centralized back-end server. This helps in reducing the overhead as well as performance lag. Back-end service interacts with the MongoDB instance to store information and details on the users. MongoDB is chosen to reap the benefits of the schema-less structure as well as high-availability of the data. Login authentication is performed by querying the relevant repository on the MongoDB instance. On registration, new identity for the user is generated which is stored in the database and retrieved when a user logs in. This identity is verified by chaincode against the network. MongoDB provides with authentication service for the user and functionality to manage the different users.

IV. Evaluation Methodology

Diploma certificates are sensitive and private information for any person. Different evaluations were considered in defining the efficiency and reliability of the application.

Ordering a transcript from the university takes 2 days to be delivered, even for the digital copy or a week for the paper copy. This takes longer for the verifying purposees. The comparison was made based on simplicity, reliability and processing time for the traditional approach and using the proposed application. Other metric was evaluated based on traditional blockchain technologies such as ethereum, bitcoin, and hyperledger fabric. Evaluation was performed on the security aspects of storing the diploma certificate data on the public data store.

To measure the performance of the system, two tests have been performed: (1) Time taken to upload diploma in a batch of 30, 50, 100, 150 and 170 diplomas is measured. This tests how the system handles multiple and concurrent uploads. (2) Time taken to query a diploma as the ledger grows is also measured. This tests how efficiently the system can retrieve the diploma from the ledger.

V. Results

Diploma certificates are considered one most important documents for student and verification for the same is an essential process for either further education or to get into the industry. Manual verification of diploma certificates is very time consuming and it requires manual interventions. Manual verification can be error-prone and after investing so much time, guaranteed success is not a surety. So, we have developed a system that overcame the problems with the legacy approaches and is more secure and efficient.

A. Overcame challenges with manual certificate verification

The system provides simple user interface to upload a certificate to universities. Once the student graduates, the university can securely commit a diploma certificate for the student to the system. The system will store the digital copy of the document on the IPFS and store the related data to the secure and immutable blockchain network. For the verification of the document, an employer or a third party can log in to the system and fetch the record. The system will give the details about the issuer, graduation term, degree information and also the copy of the document itself. With these details, the validity of the record can be easily established.

B. Secure storage

The system leverages the trust and decentralization provided by the blockchain technology. The system verifies the university before allowing it to commit a transaction to the ledger. Once the record is uploaded to the ledger, it is immutable. So, the possibility of changing anything next to impossible.

C. Controlled access to resources

The design of our system uses the permissioned blockchain network. This ensures that only the authorized entities are present in the network. Also, not all the users within the network have the same permissions. The system enforces role-based-access-control based on the credentials of the requesting user before each interaction with the ledger (transaction or query). This ensures that only authorized user accesses the resources in a controlled manner.

D. Reduction in manual efforts and processing time

In the legacy approaches, it takes a few days to finish the diploma verification process as it needs to consult the university for the validation of the diploma records. Also, if the document includes multiple universities then the process may take longer time. In this system, the verifier only needs to register to the system and consent from the student to access the diploma record. Once the verifier has required permission, it can quickly fetch the record and easily check the validity.

In figure 4, it can be observed that it takes approximately 1 second to query a diploma from the ledger. This is irrespective of the number of diplomas uploaded on the network. This shows that the application is quick to verify the certificates as compared to the conventional method. Figure 5 depicts time taken for a university to upload diplomas in batch. The increment in time can be noted as the number of diplomas increases. But this still shows an improvement over existing solutions.

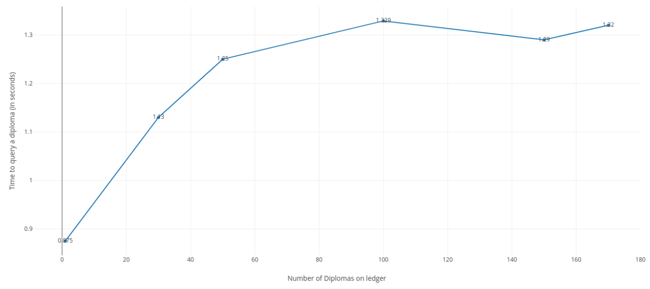


Fig. 4: Time taken to query a Diploma

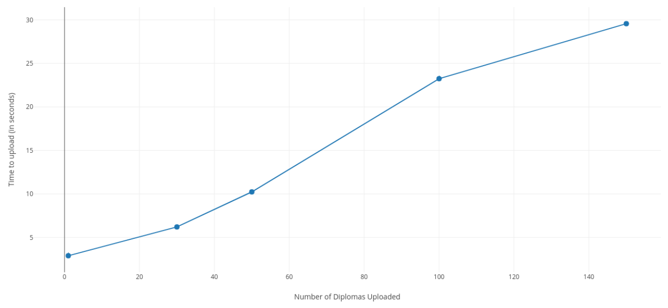


Fig. 5: Time taken to upload Diploma in a batch

VI. Discussion

Diploma certificates being one of the most sensitive document for any student needs to be shared and stored securely. Proposed solution offers integrity, security, flexibility and swiftness over its traditional counterparts. This section talks about all these performance metrics and their comparison with the traditional approaches implemented on ethereum and bitcoin. Current approach to verify diploma certificates is based on the manual verification of the diploma certificates provided by students to employers. Employers, with the help of third-party contractors, verify the diploma certificates with the university and check for its authenticity. This process certainly takes a week to get completed. Proposed solution can verify the authenticity of the certificates within seconds by using the application provided.

In terms of security, most of the traditional solutions does not store the certificates or store in the public data store. Since the diploma certificates are sensitive, these may lead to security issues. To tackle the problem, proposed solution uses Attribute Based Access Control (ABAC) to check different privileges of the user. Since the information stored on the blockchain is immutable, the integrity of the diploma certificates is always intact and employers be rest assured. Proposed solution also offers dashboards to control the different permissible operations based on access which provides the user with flexibility and control over their diploma certificates. Other then the users allowed by the students, no other user is permitted to view the diploma certificate.

VII. Related Works

UZH has proposed a decentralized [9] system using modern Ethereum Blockchain [10] network that involved smart contract development in JavaScript like language called Solidity. It involved techniques like a hash generation from certificates, storing the hash on blockchain and frontend development for the entities to interact with the blockchain network [11]. Implementation involved the development of main components like frontend, backend and smart contract. The HTML5 and JavaScript frontend had two functionalities: issue/verify certificates and had fields that were used to upload PDF. The backend generates hashes from the uploaded PDF using SHA-3 [12] and uses web3.js [13] library to interact with the smart contract. The smart contract is used to develop functionalities like issue certificates and verify certificates.

The other solution team explored is CredenceLedger. CredenceLedger is the existing solution which exploits the permissioned blockchain-based technology to implement the certificate verification [14]. It provides a mobile application for a different group of users as user, admin, miner, issuer or student. Students can access the credentials which can be provided to employers or officials. These credentials can be easily verified using this platform. Data is protected using blockchain security features to provide only need-to-know basis data to users. Three different kinds of permission are provided to users such as Low Risk, Medium Risk and High Risk. These permissions entitle user with the number of operations that can be performed. The solution uses streams leading to the elimination of cryptocurrency. Mining of the blocks by miners is implemented using Mining Diversity scheme, the value of which ranges between 0 and 1. This tends to avoid the monopoly in the mining process as a round-robin scheme is implemented.

Blockcert is another platform for certificate verification. It is a blockchain based open standard for issuing and verifying official certificates such as academic documents, professional documents and other authorized records [15]. Blockcert exploits bitcoin [16] blockchain technology and provides libraries and tools for digitizing paper-based documents and enables trustless verification. Blockcert consists of three main entities - issuer, recipient, verifier. Issuer is responsible for issuing authorized documents to recipient and performs transactions to store the documents on blockchain once confirmed by the recipient. Issuer sends blockchain transaction information and credentials to the recipient which contains bitcoin transaction id, expected merkle root [17], expected hash of recipient's certificate and merkle path from recipient's certificate to merkle root. Recipient sends this information to the verifier. Verifier can check the authenticity of the document by verifying the provided hash with the certificate hash.

Acknowledgment

The authors are deeply indebted to Dr. Younghee Park, Assistant Professor, Department of Computer Engineering, San Jose State University, for her invaluable comments and assistance in the preparation of this study. Authors acknowledge Ikwhan Chang, Founder/Devops, Urhyme, who guided us through the complexities and provided valuable inputs. Authors acknowledge Dr. Divyesh Jadav, Manager in Cloud, IoT and Systems Lifecycle Analytics department at IBM Almaden Research Center, for providing us with the required guidance on hyperledger technology and opportunity to interact with the experts in field.

References

[1] (2014, Aug.) Fifty-eight percent of employers have caught a lie on a resume, according to a new careerbuilder survey. CareerBuilder. [Online]. Available: https://www.careerbuilder.com/share/aboutus/pressreleasesdetail.aspx?sd=8/7/2014&id=pr837&ed=12/31/2014

[2] T. Lewin. (2007, April) Dean at m.i.t. resigns, ending a 28-year lie. The New York Times. [Online]. Available: http://www.nytimes.com/2007/04/27/us/27mit.html

[3] E. Androulaki, A. Barger, V. Bortnikov, C. Cachin, K. Christidis, A. De Caro, D. Enyeart, C. Ferris, G. Laventman, Y. Manevich *et al.*, "Hyperledger fabric: a distributed operating system for permissioned blockchains," in *Proceedings of the Thirteenth EuroSys Conference*. ACM, 2018, p. 30.

[4] J. Benet, "Ipfs-content addressed, versioned, p2p file system," *arXiv preprint arXiv:1407.3561*, 2014.

[5] B. Varanasi and S. Belida, "Hateoas," in *Spring REST*. Springer, 2015, pp. 165–174.

[6] Hyperledger fabric java sdk - api to interact with fabric network implemented in java. [Online]. Available: https://github.com/hyperledger/fabric-sdk-java

[7] Hyperledger fabric node sdk - api to interact with fabric network implemented in node. [Online]. Available: https://github.com/hyperledger/fabric-sdk-node

[8] H. Es-Samaali, A. Outchakoucht, and J. P. Leroy, "A blockchain-based access control for big data," *International Journal of Computer Networks and Communications Security*, vol. 5, no. 7, p. 137, 2017.

[9] J. Gresch, B. Rodrigues, E. Scheid, S. S. Kanhere, and B. Stiller, "The proposal of a blockchain-based architecture for transparent certificate handling," in *International Conference on Business Information Systems*. Springer, 2018, pp. 185–196.

[10] G. Wood, "Ethereum: A secure decentralised generalised transaction ledger," *Ethereum project yellow paper*, vol. 151, pp. 1–32, 2014.

[11] S. Kumar, S. Gil, D. Katabi, and D. Rus, "Accurate indoor localization with zero start-up cost," in *Proceedings of the 20th annual international conference on Mobile computing and networking*. ACM, 2014, pp. 483–494.

[12] M. J. Dworkin, "Sha-3 standard: Permutation-based hash and extendable-output functions," Tech. Rep., 2015.

[13] Web3.js - ethereum javascript api. Release 0.20.5. [Online]. Available: https://github.com/ethereum/web3.js

[14] R. Arenas and P. Fernandez, "Credenceledger: A permissioned blockchain for verifiable academic credentials," in *2018 IEEE International Conference on Engineering, Technology and Innovation (ICE/ITMC)*. IEEE, 2018, pp. 1–6.

[15] P. Schmidt, "Blockcertsan open infrastructure for academic credentials on the blockchain," *MLLearning (24/10/2016)*, 2016.

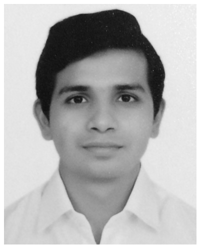
[16] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008.

[17] R. C. Merkle, "A digital signature based on a conventional encryption function," in *Conference on the theory and application of cryptographic techniques*. Springer, 1987, pp. 369–378.

Nishantkumar Bhalani is a deep learning enthusiast, software developer and a MS-CMPE student focused on Data Science at San Jose State University, CA. Nishant looks forward to learning new frameworks and technologies in the field of Machine/Deep Learning, blockchain and enterprise application development.

Anuj Chaudhari is Software Engineering graduate student at San Jose State University. Anuj has a Bachelor of Technology degree in Computer Engineering. His research interests include distributed systems and parallel programming.

Jay Desai is a Software Engineering graduate student at San Jose State University, CA. Jay has a bachelor of engineering degree in Computer Engineering. His research interests include cloud computing and distributed systems.



Tejas Panchal is a Software Engineering graduate student at San Jose State University, CA. He has a bachelor of engineering degree in Electronics & Communications Engineering. His research interests include distributed systems.