Decentralized DigiCerts

Ankita Patil
ankitaprabhakar.patil@sjsu.edu

Charit Upadhyay
charit.upadhyay@sjsu.edu

Devika Jadhav
devika.jadhav@sjsu.edu

Mustafa Bandukwala
mustafa.bandukwala@sjsu.edu

Vivek Maran
vivek.maran@sjsu.edu

ABSTRACT

The problem of direct exchange of digitally signed documents without an intermediary like notary requiring to authorize the signer can be solved using Blockchain Technology. Blockchain technology offers new tools for authentication and authorization in the digital world that preclude the need for many centralized administrators.

1 INTRODUCTION

When we lived in small tight-knit communities, people knew whom they could turn to when they needed an expert (and whom to avoid). However, as we started moving around more and our lives and networks grew, we needed to come up with portable ways to signal our expertise to new acquaintances. Some of these original systems are still in place. For example, in Germany many carpenters still do an apprenticeship tour that lasts no less than three years and one day. They carry a small book in which they collect stamps and references from the master carpenters with whom they work along the way. The carpenter's traditional outfit, the book of stamps they carry, and—if all goes well—the certificate of acceptance into the carpenter guild are proof that here is a man or woman you can trust to build your house. Ideally, we should be in charge of our own credentials, similar to the journeymen carpenters who carry around their books of stamps and references, but most of the time we have to rely on third parties, such as universities or employers to store, verify, and validate our credentials. Job seekers have to request official transcripts from their alma maters (and typically pay a small fee), and employers still need to call the university if they want to be sure that a transcript wasn't faked. It's a slow and complicated process. To overcome this hassle, we propose the solution 'Decentralized Digicerts'. This system allows institutions to issue blockchain notarized digital credentials which the recipient can directly share it to other institutions (without any intermediaries), employers etc.. who then can verify that credential for integrity and authenticity through blockchain.

2 INSPIRATION

The trail of credentials and achievements that we generate throughout our lives says something about who we are, and it can open doors that allow us to become who we want to be. Some credentials, such as university degrees, are more important than others. But at the end of the day, all of these credentials represent experiences that are part of our lives.

There are many advantages for recipients to have more control over the certificates they earn. Being in control doesn't mean it would be easy to lie. Similar to the carpenter's book of references, it should not be possible to just rip out a few pages without anyone noticing. But being in control means having a way to save credentials, to carry them around with us, and to share them with an employer if we chose to do so (without having to pay, or ask for the issuer's permission or cooperation).

3 WORKFLOW

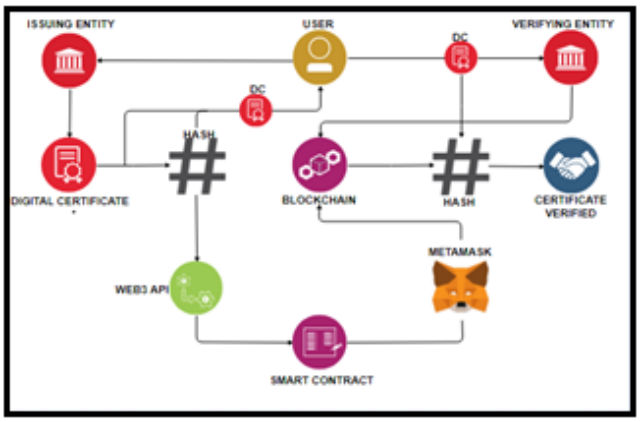

Figure 1: *Decentralized digicerts system*

The system involves three parties: credential issuer, recipient, and the credential verifier.

3.1 Credential Recipient

Credential recipient is the one who has earned the credential. In the Decentralized Digicerts system the recipients are associated with unique public keys.

3.2 Credential Issuer

Credential issuer is the institution (or) organization which issues the credential. The issuer uses the Decentralized Digicerts server software to issue the certificate which performs the following sequence of steps:

1. Creates a certificate with issuer provided details
2. Computes SHA3 of the certificate and signs it using issuer's private key.
3. Adds the mapping of the recipients public key and the signed hash of the recipients certificate to the Ethereum blockchain.
4. Sends the certificate to the recipient.

3.3 Credential Verifier

The party who receives a certificate from the owner of the credential and would like to verify the authenticity and integrity of the certificate. The certificates generated by the system has an embedded functionality to verify the integrity and authenticity of the certificate. The steps involved in credential verification are as follows:

1. The hash of the certificate element is computed.
2. The hash from Ethereum blockchain is retrieved (using the blockchain reference present in the certificate)
3. The local hash and blockchain hash are compared.
4. If they are equal, then the certificate is considered to be a valid.

4 ETHEREUM BLOCKCHAIN AND SMART CONTRACTS FOR DIGICERTS

4.1 Overview of Blockchain And Smart Contracts

A blockchain is a digitized, decentralized immutable database, constantly growing as 'completed' blocks (the most recent transactions) are recorded and added to it in chronological order. While blockchains were initially targeted storing financial transactions, in the recent past Ethereum blockchain network has provided the technology for storing software code, called smart contracts, into the blockchain and this has opened up a window for developing a new range of blockchain based applications.

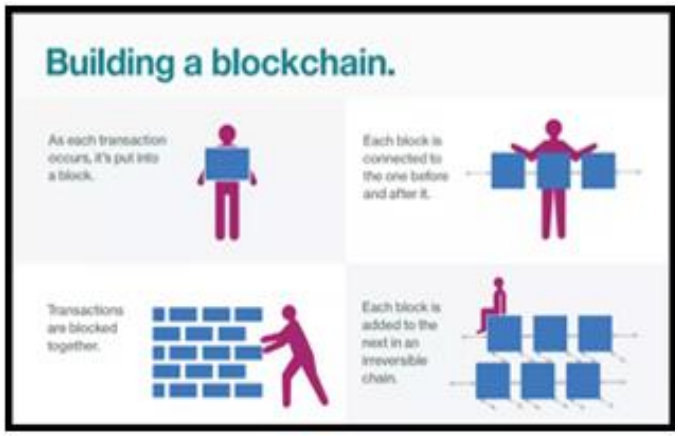

Figure 2: *How Blockchain works. Source: IBM, "Blockchain 101 Infographic."*

Digicerts system harnesses this Smart Contract feature for issuing and verifying secure digital credentials

4.2 Deploying And Interacting With Smart Contracts

Smart contracts are simple programs that are written in Solidity and stored on the blockchain. Applications interacting with Ethereum blockchain, can access the API's in the solidity contract and also listen events from the contract. Following picture depicts on how an application can deploy a contract and interact with the contract present in the blockchain.

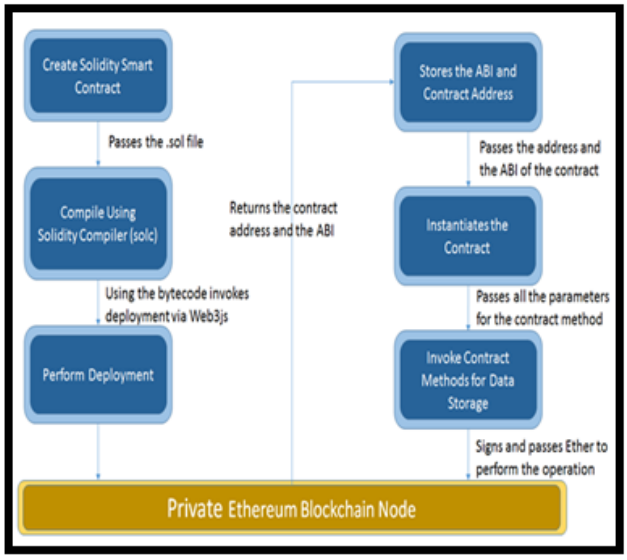

Figure 3: *Contract Deployment workflow. Source: "https://medium.com/cryptolinks/ethereum-smart-contract-development-610718d0629".*

4.3 Smart Contracts for Decentralized

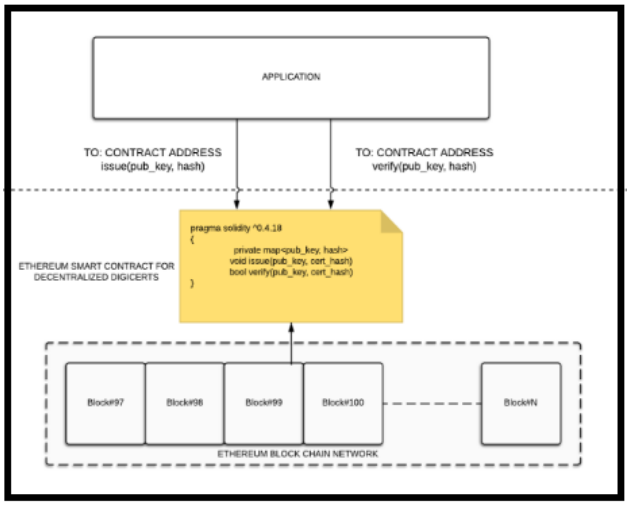

Figure-4: *Smart contracts for Digicerts*

When the issuing institution registers with the Decentralized Digicerts server software, a contract is deployed(currently pre-deployed contract are used) for the issuing institution as shown in the Figure-3. The contract consists of the following entities:

1. API to issue certificate.
2. API to verify certificate.
3. Constructs to maintain the mapping of certificate holders' public key and their certificate hashes.

Henceforth whenever the issuing institution issues a certificate, the institution's contract state is updated to have an additional {public key: certificate hash} mapping by invoking the 'issue' method present in the contract, and by this way the contract records the list of all the issued certificate hashes along with their owners.

When the credential verifier triggers a verify, the 'verify' method present in the contract is invoked with the public key of the credential owner and the hash of the local certificate as arguments to check the integrity and authenticity of the credential.

5 SOFTWARE ARCHITECTURE

5.1 Software Architecture Diagram

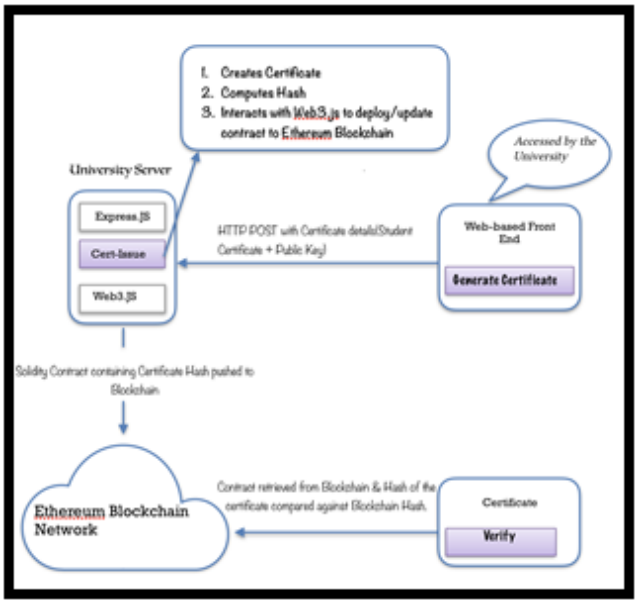

Figure 5: *Software architecture*

5.2 Software Architecture Description

Decentralized DigiCerts system has a web based front end which is used by the universities to issue the digital certificates. The requests generated from this web app are directed to our 'Digicerts' (NodeJS) server. The implementation of the server is as follows:

1. Top most layer of the stack is the 'Express.Js' which is HTTP server framework.
2. The lower most layer of the stack is the 'web3.js' framework which is responsible for interacting with the Ethereum blockchain.

The middle layer 'Cert.js' is where the core functionality of the 'Digicerts' app is written. It handles the request provided by express JS and drives the 'web3.js' for blockchain interaction to create a blockchain notarized certificate with embedded functionality to verify the authenticity and integrity.

6 CONCLUSIONS

The overall design of the certification architecture is simple. In summary, a certificate issuer signs a well-structured digital certificate and stores its hash within a blockchain transaction. A transaction output is assigned to the recipient. Recipient then sends the certificate to verifying entity and verifying entity validates the authenticity of the certificate by comparing the hash values.

7 ACKNOWLEDGMENTS

This work was supported by Dr. Saldamli Gokay, Assistant Professor in Computer Engineering Department at San Jose State University.

8 REFERENCES

[1] https://en.wikipedia.org/wiki/Blockchain.

[2] https://www.investopedia.com/terms/b/blockchain.asp.

[3] https://medium.com/mit-media-lab/certificates-reputation-and-the-blockchain-aee03622426f.

[4] https://github.com/blockchain-certificates/cert-issuer.

[5] https://github.com/blockchain-certificates/cert-verifier.

[6] https://wiki.mozilla.org/images/b/b1/OpenBadges-Working-Paper_092011.pdf.

[7] https://www.faoglobal.com/is-the-blockchain-a-game-changer-for-global-trade/.

A HEADINGS IN APPENDICES

The rules about hierarchical headings discussed above for the body of the article are deferent in the appendices. In the appendix environment, the command section is used to indicate the start of each Appendix, with alphabetic order designation (i.e., the first is A, the second B, etc.) and a title (if you include one). So, if you need hierarchical structure within an Appendix, start with subsection as the highest level. Here is an outline of the body of this document in Appendix-appropriate form:

A.1 INTRODUCTION

A.2 INSPIRATION

A.3 WORKFLOW

A.3.1 Credential Recipient

A.3.2 Credential Issuer

A.3.3 Credential Verifier

A.4 ETHEREUM BLOCKCHAIN AND SMART CONTRACTS FOR DIGICERTS

A.4.1 Overview of Blockchain And Smart Contracts

A.4.2 Deploying And Interacting With Smart Contracts

A.4.3 Smart Contracts for Decentralized

A.5 SOFTWARE ARCHITECTURE

A.5.1 Software Architecture Diagram

A.5.2 Software Architecture Description

A.6 CONCLUSIONS

A.7 ACKNOWLEDGMENTS

A.8 REFERENCES