Cloud computing interactions with the blockchain

1st Dawid Pionk   
*South East Technological University*Carlow, Republic Of Ireland  
Dawidpionk01@gmail.com

*Abstract*—This paper explores the possible uses for cloud computing with the blockchains and vice versa. It aims at trying to connect decentralized blockchain with centralized off-chain databases through the use of smart contracts. This project was conducted using several technologies. Aws for cloud storage. Node.js using several libraries for event listening for changes on the blockchain. Solidity smart contracts previously deployed on the blockchain. Infuria was used to connect to Ethereum nodes. Foundry was used for testing and casting onto the blockchain. Then the system was evaluated to see how it performed against set out goals. Then the systems limitations were examined. The findings in this paper suggest that this type of hybrid system would be right for some system.

# Introduction

In this paper, we will present a review of how cloud computing can be interacted with on the blockchain. More specifically we explored the area of Product as a service. This brought several limitations such as security etc into play. We think this is a valuable field of research since it could offer more security and cheaper ways of storing data. Some other interesting parts include decentralized access to data and some real word use case research. With this paper we contribute to the field by researching hybrid systems and developing a prototype for a cloud to blockchain application.

# Literature Review

## Technology Review

First, confirm that you have the correct template for your paper size. This template has been tailored for output on the A4 paper size. If you are using US letter-sized paper, please close this file and download the Microsoft Word, Letter file.

## Similar Projects

The template is used to format your paper and style the text. All margins, column widths, line spaces, and text fonts are prescribed; please do not alter them. You may note peculiarities. For example, the head margin in this template measures proportionately more than is customary. This measurement and others are deliberate, using specifications that anticipate your paper as one part of the entire proceedings, and not as an independent document. Please do not revise any of the current designations.

# System Architecture

## High level overview

So how cloud computing interacts can be bit complicated but has many applications. In this report I decided to focus how decentralized blockchain could interact with centralized cloud data centers. This process can be complicated since blockchains don’t have a direct way to interact with cloud data centers.

## Components

Below I explain how each component interacts with the overall system:

* Cloud Server: This is where the data can be stored or retrieved from. Examples of these include AWS or Azure
* Blockchain: Here is the system where the underlying Ethereum is.
* Smart Contracts: They are hosted on the blockchain, these are programs where the users can interact with the blockchain or different data.
* Web3 programs: Here the user can use as a point to interact with blockchain from.
* UI: Here is the actual interface the user interacts with
* Lamba function: This listens out for the changes on the blockchain.

## Architecture diagram

# Experiment setup and performance evaluation

After the text edit has been completed, the paper is ready for the template. Duplicate the template file by using the Save As command, and use the naming convention prescribed by your conference for the name of your paper. In this newly created file, highlight all of the contents and import your prepared text file. You are now ready to style your paper; use the scroll down window on the left of the MS Word Formatting toolbar.

## Environmental setup

How this project was setup was through several steps:

1. Created a database on the aws servers where the data be stored
2. Created a table on the said database.
3. Created a smart contract using solidity that deals with creating vaults.
4. Created an project key using infuria so I would be able to interact with the sepolia testnet.
5. Deployed smart contract onto the testnet.
6. Created an event listener using javascript and node.js that listens out for any events emitted by the smart contracts address.
7. Tested it out by creating a vault and seeing if the table was updated.

## Use cases

The primary use case for this specific experiment would be users who want an easy way to view and store data produced on the blockchain. This system shows how a proposed hybrid blockchain to cloud data storage system could work. When a vault is created the data is stored both on the blockchain and on a AWS database. This has benefits such as users having easy access to view data, as well as data stored on servers being easy to audit since the blockchain can’t be altered. This system could be expanded into areas that need to be easy to access and couldn’t be altered such as secure document storage or encrypted medical data.

## Evaluation

We managed to achieve several functionality goals:

* The smart contract was successfully deployed onto the blockchain.
* The event listener successfully detects addVault() events.
* The table on the AWS database was successfully updated after an added vault.
* The system doesn’t miss any events when it is actively listening.
* The system takes under 6 seconds to go from emitted event to updated table.

##### Acknowledgment

I would like to express my thanks to my supervisor Shei Lei for their guidance through this project as well as the staff at UNUM for their classes that enhanced my understanding of cloud computing,

##### References

1. G. Eason, B. Noble, and I. N. Sneddon, “On certain integrals of Lipschitz-Hankel type involving products of Bessel functions,” Phil. Trans. Roy. Soc. London, vol. A247, pp. 529–551, April 1955. *(references)*
2. J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.
3. I. S. Jacobs and C. P. Bean, “Fine particles, thin films and exchange anisotropy,” in Magnetism, vol. III, G. T. Rado and H. Suhl, Eds. New York: Academic, 1963, pp. 271–350.
4. K. Elissa, “Title of paper if known,” unpublished.
5. R. Nicole, “Title of paper with only first word capitalized,” J. Name Stand. Abbrev., in press.
6. Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, “Electron spectroscopy studies on magneto-optical media and plastic substrate interface,” IEEE Transl. J. Magn. Japan, vol. 2, pp. 740–741, August 1987 [Digests 9th Annual Conf. Magnetics Japan, p. 301, 1982].
7. M. Young, The Technical Writer’s Handbook. Mill Valley, CA: University Science, 1989.

##### aPPENDIX

Code Listings

Listing 1: listener.js

const { ethers } = require("ethers");

const AWS = require("aws-sdk");

// AWS Regon setup

AWS.config.update({ region: "eu-north-1" });

// DynamoDB client

const dynamoDb = new AWS.DynamoDB.DocumentClient();

// Ethereum provider (Sepolia via Infura)

const provider = new ethers.JsonRpcProvider("XXXXXXXX");

// Contract address and ABI

const contractAddress = "0x9d58134Dd3fba0B3dB67264E73195186E28BafA7";

const abi = [

    "event VaultAdded(uint256 vaultId, address owner)"

];

// This section listens for sepolia vaultAdded events

const contract = new ethers.Contract(contractAddress, abi, provider);

console.log("Searching for events");

contract.on("VaultAdded", async (vaultId, owner) => {

    console.log(`VaultAdded event: Vault ID ${vaultId.toString()}, Owner ${owner}`);

    const params = {

        TableName: "UserVaults",

        Item: {

            vaultId: vaultId.toString(),

            ownerAddress: owner

        }

    };

    // Store the event in DynamoDB or error handling

    try {

        await dynamoDb.put(params).promise();

        console.log("Successfully stored event in DynamoDB:", params.Item);

    } catch (err) {

        console.error("Error with storing data", err);

    }

});