International Journal of Scientific Research in Computer Science, Engineering and Information Technology
© 2018 IJSRCSEIT | Volume 3 | Issue 5 | ISSN : 2456-3307

Smart Contract Management Using Blockchain Implementation

Chandrashekhar Singh Rajawat[1], Prof. Hemant Gupta[2]
[1]PG Scholar, Department of Computer Science and Engineering, Lakshmi Narain College of Technology & Science (RIT), Indore, Madhya Pradesh, India
[2]Assistant Professor, Department of Computer Science and Engineering, Lakshmi Narain College of Technology & Science (RIT), Indore, Madhya Pradesh, India

ABSTRACT

Recent interest around blockchains and Emerging smart contract systems over blockchain technology which allows mutually distrustful parties to transact safely without trusted third parties provided requirements good fit for the Smart Contracts sector. In the event of contractual breaches or cancellations, the decentralized blockchain ensures that honest parties get their just compensation. Blockchains permit us to have a distributed peer-to-peer network wherever non-trusting members can interact with each other without depending on an intermediary and be able to verify the transaction.
All transactions, together with the flow of cash are exposed on the blockchain.
A smart contract is used to
1) Facilitates for sharing of services and resources managing to the creation of a marketplace of services between devices and
2) permits us to automate in an exceedingly cryptographically verifiable manner many existing, long workflows
My conclusion is that the smart contract over blockchain combination is powerful and can cause vital transformations across many industries, making new ways for new business models and innovative distributed applications.

Keywords : Smart Contracts Sector, Cryptographically, Peer-To-Peer Digital Payment System, Bitcoin, Blockchain

I. INTRODUCTION

Blockchains have recently attracted the interest of investors across a wide spectrum of businesses from finance, healthcare, utilities, real estate and the government sector.
The reason for this eruption of interest is that with blockchain, applications that could previously run only through a trusted intermediary, can now operate in a decentralized fashion, without the need for a central authority, and achieve the same functionality with the same amount of certainty, which was not possible before.
Blockchain empower trustless networks, because the parties can transactions between parties while in current systems are usually conducted in a centralized form, which requires the involvement of a trusted third party like bank. However, this often means potential security issues and high transaction fees.
Blockchain technology can tackle these issues by allowing untrusted entities to interact with each other in a distributed manner without the involvement of a trusted third party.
Blockchain is a truly a distributed database that records all transactions that have ever occurred in a network. Blockchain was originally introduced for Bitcoin a peer-to-peer digital payment system, but then evolved to be used for developing a wide range of decentralized applications. An enthralling

application that can be deployed on top of blockchain is smart contracts.

What is a Smart Contract?

A smart contract is executable code that runs on the blockchain to facilitate, execute and enforce the terms of an agreement between untrusted parties. It helps act like an expert evidence of as a system that releases digital assets to all or some of the once the pre-defined rules have been met.

Compared to traditional contracts, smart contracts do not rely on a trusted third party to operate, resulting in low transaction costs. There are different blockchain platforms that can be exploit to develop smart contracts, but Ethereum is the most common one.

Smart contracts can be applied to different applications for example smart properties, e-commerce and music rights management. Smart contracts –self-executing scripts that reside on the blockchain– integrate these concepts and allow for proper, distributed, heavily computerized workflows. Blockchains and smart contracts bring a multitude of advantages to the table, but they also come with a sack of disadvantages. This document explains a detailed description of how blockchains and smart contracts work, to identify the pros and cons and highlight the methods of the blockchains and smart contract can be organized.

The structure of this paper is as follows. Next section discusses background information about smart contracts over blockchain technologies and how smart contracts Works. In another section I will show how Smart Contract and blockchains can be used together, and highlight existing smart contract-on-the-blockchain applications.

Smart Contract Blockchains

User-defined assets could be represented with the help of a smart contract on a smart contract blockchain. The contract could store the mapping of the addresses of current holders of the asset to the corresponding balances. These balances could be updated with the help of messages sent to the contract encoding asset transfer or issuance. The contract could use the conventional authorization scheme of the underlying blockchain in order to check transfer and issuance permissions, or could specify new rules for asset transactions. Ethereum is an example of an independent stat smart contract blockchain. Rootstock is a conceptual smart contract blockchain pegged to Bitcoin.

How Blockchains work

A blockchain is a distributed data structure that is replicated and shared among the members of a network. It was introduced with Bitcoin to solve the double-spending problem. As a result of how the nodes on the Bitcoin network (the so-called miners) append validated, mutually agreed-upon transactions to it, the Bitcoin blockchain houses the authoritative ledger of transactions that establishes who owns what. In this section general background information about blockchain and smart contracts technologies is depicted.

Think of the blockchain as a log whose records are batched into time stamped blocks. Each block is identified by its cryptographic hash. Each block references the hash of the block that came before it. This establishes a link between the blocks, thus creating a *chain* of blocks, or *blockchain*

Any node with access to this ordered, back-linked list of blocks can read it and figure out what is the world state of the data that is being exchanged on the network. We get a better understanding of how a blockchain works,

1) Users interact with the blockchain via a pair of private/public keys. They use their private key to sign their own transactions, and they are addressable on the network via their public key. The use of asymmetric cryptography brings authentication, integrity, and nonrepudiation into the network. Every signed transaction is broadcasted by a user's node to its one-hop peers.

2) The neighboring peers make sure this incoming transaction is valid before relaying it any further; invalid transactions are discarded. Eventually this transaction is spread across the entire network.

3) The transactions that have been collected and validated by the network using the process above during an agreed-upon time interval, are ordered and packaged into a time stamped candidate block. This is a process called *mining*. The mining node broadcasts this block back to the network.

4) The nodes verify that the suggested block (a) contains valid transactions, and (b) references via hash the correct previous block on their chain. If that is the case, they add the block to their chain, and apply the transactions it contains to update their world view. If that is not the case, the proposed block is discarded. This marks the end of a round.

II. Implementation Plan

User Registration

User can register in to the application by providing details. Once registered user will be sent an email with link to verify identify and set password. Public and private key will also be need to set at the time of registration. Additional Security features like two factor identification and personalized password encryption algorithm can also be applied however these additional features are not in scope of Block chain technology, hence omitted in the application design.

Public Key – Encrypted #HASH code generated from Login ID of User.
Private Key – Encrypted #HASH code generated on user selected pass phrase.

User Login

User can login in to this Contract Management Application by logging using email and password. Additional 2 factor authentication can also be applied depending on need. However for demonstration purpose we are using simple login with encrypted password using an encryption algorithm.

User Dashboard

After login User is landed on the dashboard. Dashboard will display status of existing contracts.

Initiate Contract

In this section user can initiate an already drafted contract with the other party. For the sack of simplification, a bi-party contract system is developed, however it can be easily be converted in to multiparty contract management system if required.
User in this section will upload draft contract file in the system and select following information.

Type of Contract
Expiration date of Contract
Email id of other party inviting them to the participate in Contract Management and Execution

Once Submitted a HASH code will be generated. This hash code is generated for version of file. Please note that the Contract Document is not physically uploaded or saved in the system anywhere.
Before submitting Contract for other party use, Contract Hash code will be appned by Public Key and Private Key. Public Key will be generated for each contract being initiated by first party and will be used to identify contract negotiation while private key will be appended by sender to indicate that the this contract if originated or forwarded by parties or intermediary node users in the blockchain system.
Status code is the status of current contract so that parties can communicate the status of contract. This status code can only be changed by Contracting parties and not by intermediary.
Public key once generated at the initiation will continue to remain same throughout the contracting process.
PublicKey, StatusCode,Contract Hashcode ,ReceiverKey,PrivateKey,NodeKey

Status Codes

	Draft	**Approved**	**Executed**	**Dispute**	**Request for Contract Termination**	**Contract Closure**	**Contract Terminated**	**Explanation of status**
Draft (D)	Y	Y						While Contract Document is not finalized and agreed up on by both the parties the contract will continue to be in Draft Stage.
Approved (A)	Y		Y					Once a party agrees with the draft of contract it will mark contract as approved from their side.
Executed (E)				Y	Y	Y		Once other party responds to Approved marked by first party, Contract status will be marked as Executed.
Dispute (DP)		Y			Y			Either one of the party may change the status to dispute if one of more of contract terms and conditions are not met and party wants to raise flag for other party.
Request for Contract Termination (RCT)	Y	Y					Y	Once Contract is in dispute by one or both the parties Contract can be raised fro Termination
Contract Closure (CC)								Once the Contract is successfully

								executed, Contract can be closed by one party and once other party also approved contract closure. Contract is finally closed.
Contract Terminated (CT)								Based on the Request of termination the other party confirm by updating the status to Contracted Terminated.

Each time a new version of contract is uploaded during Draft stage, a new HashKey for Contract Document will be generated. However Public Key of Contract will remain same.

User whenever sending out the contract to other other intermediary nodes will send the Next Node Public key of other user in the block chain. User will also send its own Private key as a signature from its own side.

Below is one example depicting entire process.

USER	**PrivateKey**
Party 1	PRVTKYA
Intermediary Node 1	N1
Intermediary Node 2	N2
Intermediary Node 3	N3
Intermediary Node 4	N4
Intermediary Node 5	N5
Party 2	PB

Contract Key	Conxx
Contract HashCode	ContHashyyyy1 (With each revision in the contract document this will continue to change till both the parties agree on a version, in that case it will be frozen)

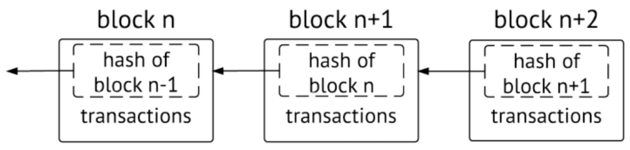

Figure 1. Each block in the chain carries a list of a transactions and a hash to the previous block.

Scenario 1 Party A uploads a contract draft and sends to other party

A Public Key for Contract is generated as PUBCTX

A Contract HashCode is generated for uploaded Document as CONTRACTHASH1

Status of Draft As D is appended to Contract Public Key

Party B email id codified Receiver Key as PARTYB

Party A Private Key is attached as PRVTKYA

STEP 1 – Sender creates an overall hash for blockchain as

PUBCTX,D, CONTRACTHASH1, PARTYB, PRVTKYA, N1

STEP 2 - When Node 1 Receives the HashCode it first checks if PARTYB is same as N1 code,

CONDITION IF same then Contract has reached destination and N1 is the Receiving Party,

it strips the Contract Public Key (PUBCTX), Contract status (D) and Contract Hash (CONTRACTHASH1) from the message and stores the hash In it local

database with the Key of Sender (PRVTKYA). Receiver uploads the Contract Document available at his side and checks if the hashcode matches with the uploaded document.
EXIT
CONDITION ELSE
However if Node 1 is not the recipient then N1 prepares another message with same format above as
PUBCTX,D, CONTRACTHASH1, PARTYB, PRVTKYN1, N2

REPEAT STEP 2
Scenario 2 Party B updates the contract draft and sends to other party
A Public Key and contract HashCode, ReceiverID(Previously Sender ID as PARTYA) along with Node which sent the message last time lets say Node 2.
A Contract HashCode is generated for uploaded Document as CONTRACTHASH2
Status is marked as approved As D is appended to Contract Public Key
Party A email id codified Receiver Key as PARTYA
Party B Private Key is attached as PRVTKYB
STEP 1 – Sender creates an overall hash for blockchain as
PUBCTX,D, CONTRACTHASH2, PARTYA, PRVTKYB, N2
STEP 2 - When Node 2 Receives the HashCode it first checks if PARTYA is same as N2 code,
CONDITION IF same then Contract has reached destination and N2 is the Receiving Party,
it strips the Contract Public Key (PUBCTX), Contract status (A) and Contract Hash (CONTRACTHASH1) from the message and stores the hash In it local database with the Key of Sender (PRVTKYB). Receiver uploads the Contract Document available at his side and checks if the hashcode matches with the uploaded document.
EXIT

CONDITION ELSE
However if Node 2 is not the recipient then N2 prepares another message with same format above as
PUBCTX,A, CONTRACTHASH1, PARTYA, PRVTKYN2, N1 REPEAT STEP 2

Scenario 3 Party B Approved the contract draft and sends to other party
A Public Key and contract HashCode, ReceiverID(Previously Sender ID as PARTYA) along with Node which sent the message last time lets say Node 2.
A Contract HashCode is generated for uploaded Document as CONTRACTHASH1
Status is marked as approved As A is appended to Contract Public Key
Party A email id codified Receiver Key as PARTYA
Party B Private Key is attached as PRVTKYB
STEP 1 – Sender creates an overall hash for blockchain as
PUBCTX,A, CONTRACTHASH1, PARTYA, PRVTKYB, N2
STEP 2 - When Node 2 Receives the HashCode it first checks if PARTYA is same as N2 code,
CONDITION IF same then Contract has reached destination and N2 is the Receiving Party,
it strips the Contract Public Key (PUBCTX), Contract status (A) and Contract Hash (CONTRACTHASH1) from the message and stores the hash In it local database with the Key of Sender (PRVTKYB). Receiver uploads the Contract Document available at his side and checks if the hashcode matches with the uploaded document.
EXIT
CONDITION ELSE
However if Node 2 is not the recipient then N2 prepares another message with same format above as
PUBCTX,A, CONTRACTHASH1, PARTYA, PRVTKYN2, N1 REPEAT STEP 2

III. CONCLUSION

Blockchains could be one of transformative technologies for digital asset management, serving as a specialized platform as a service (PaaS) with significant growth potential. Blockchains could provide for unprecedented levels of counterfeit resistance, openness, transparency, and auditability. Blockchain technology could allow decoupling tasks associated with asset management and transaction processing, therefore providing an attractive alternative to existing centralized asset management platforms for small and medium-sized businesses, third-party application developers and end customers. Internal, algorithmically enforced properties of blockchains and their increased auditability could prove attractive for regulatory bodies. Blockchains give us resilient, truly distributed peer-to-peer systems and the ability to interact with peers in a trustless, auditable manner. Smart contracts allow us to automate complex multi-step processes.

IV. REFERENCES

[1]. Profr3cev.com/blog/2016/6/2/ethereum-platform-review
[2]. etherscan.io/chart/gaslimit
[3]. ethgasstation.info/
[4]. greentechmedia.com/articles/read/the-energy-blockchain-could-bitcoin-be-catalyst-forthe-distributed-grid
[5]. blog.ethereum.org/2015/08/07/on-public-and-private-blockchains.
[6]. rstmonday.org/ojs/index.php/fm/article/view/548/4691
[7]. A Peer-to-Peer Electronic Cash System.
[8]. https://docs.erisindustries.com/blockchains/
[9]. https://www.coindesk.com/information/how-does-blockchain-technology-work/
[10]. http://www.truthcoin.info/blog/wise-contracts/
[11]. https://www.ccn.com/smart-contracts-12-use-cases-for-business-and-beyond/
[12]. https://solidity.readthedocs.io/en/v0.3.1/solidity-in-depth.html
[13]. https://blockgeeks.com/guides/smart-contracts/
[14]. https://medium.com/crypto-currently/build-your-first-smart-contract-fc36a8ff50ca
[15]. https://hackernoon.com/advantages-and-disadvantages-of-smart-contracts-in-financial-blockchain-systems-3a443145ae1c
[16]. http://searchcompliance.techtarget.com/definition/smart-contract