Hierarchical Design and Execution of Smart Contracts in Blockchain

Srinidhi Srinivasan, Rubasri Sundar, Sam Joy Herald Immanuel, Ramesh Belvadi, Mithileysh Sathiyanarayanan
Department of Computer Science and Engineering, Information Technology
Sri Sairam Engineering College, Chennai, India
MIT Square, London, United Kingdom
srinidhisrinivasan2001@gmail.com srubasri1322001@gmail.com samjoy2616@gmail.com

***Abstract*— In light of the multiple legal issues, compliance and disruptions caused by the pandemic, organisations searching for new solutions need to know what smart contracts are and how they would function under the legal doctrine of force majeure in light of COVID-19. The Blockchains which use Bitcoin type of scripts have been popular as payment solutions, but it is less used as smart contracts. In the case of multi-level games and incremental project payments, there is a high potential to use Bitcoin type of scripts, but it is not being used currently. Interestingly, there have been attempts to associate smart contract mainly using Ethereum Blockchain but not with Bitcoin type of scripts. This article intends to demonstrate the novelty of designing smart contracts using Bitcoin type of scripts for hierarchical execution of smart contracts. An attempt is done to show its application in two use cases (multi-level reward games payment and incremental project payment). An evaluation is done with three methods each having a combination of pros and cons based on the requirements which aids in understanding for transparency and control over funds through Blockchain.**

Keywords—Blockchain, bitcoin, smart contracts, Ethereum, Hyperledger, hierarchical design, payment, escrows, casino game

I. Introduction

Bitcoin was invented in the year 2008 which introduced the concept of a distributed ledger. This solved the problem of double spending, p2p transfer of funds through. The distributed ledger was different from the centralized ledger and an improvement over double-entry ledger that was popular. This distributed ledger is also called a Bitcoin Blockchain. From the existing research, Bitcoin used cryptography concepts in creating identity of users. The identity is created by a Public-key/Private-key paid created by asymmetric cryptography. The public-key was used to create the Bitcoin address, which established identity. The holder of private-key can exercise his rights on the assets help in Blockchain by the feature of signing with private-key. The network of ledger keepers called miners was introduced to verify the identity of the users of Blockchain and write the records of their activity in Blockchain.

The Bitcoin also introduced two methods of writing and verifying Blockchain activity. One based on a signature and another based on a secret. In the signature method, the user asserts his rights by signing with private-key this was called the P2PKH method. In another method, the user asserts his rights by a secret he holds this was called the P2SH method. The multi-signature usage of P2SH is described in [3]. This provides an option to make the spending address depend on two or more parties to sign the transaction. The Bitcoin Blockchain also had the concept of smart contracts which was possible using the P2SH method. The smart contract is also called custom contracts. This made possible to many new applications for escrow, lottery using smart contracts. However, custom contracts were also called non-standard contracts. The miners found it very risky to support custom contracts as the OPCODE sequence is not thoroughly tested. Hence, it did not become popular. An improvement over the custom contract was introduced in patent application [2] to use standard OPCODES sequence to achieve new applications by having composite-key. The composite-key method can be used to design execution on payment in a hierarchy. This has application in incremental project payment [7] and hierarchical game reward payment [6]. The advantage of using the composite-key method compared to Ethereum and Hyperledger is it can be implemented in the application layer, as Blockchain can be used as an enforcement and control mechanism. We will be using three methods and five core components and two use cases (incremental project payment and hierarchical game reward payment) to demonstrate our work.

Three Methods used for the Design of Hierarchy in Blockchain:

- Multi-signature method (P2SH technique)
- Single-user signature with hierarchy of control codes method (P2PKH technique)
- Hierarchy of control codes method

Five Core Components used for the Design of Hierarchy in Blockchain:

- Hierarchical design solution on Blockchain with escrows
- Locking funds in escrow on Blockchain
- Creating controls to release funds from escrows
- Application design to determine type of locking in escrows
- Application control of funds based on states of application

Two Use Cases used for the Design of Hierarchy in Blockchain:

- Multi-level game design & reward payment
- Multi-level project management & payment

II. Methods of Design of Hierarchy in Blockchain Escrows

Escrow address can be created based on one of the three methods. The method-1 uses multi-signature technique (P2SH), the method-2 uses P2PKH along with composite-key method, the method-3 uses composite-key method.

978-1-7281-7213-2/20/$31.00 ©2020 IEEE

Method 1: Multi-signature (P2SH):

The escrow address is created using multi-signature method using [3] to mention the number of parties to sign for spending the funds in escrow address. In the hierarchical multi-level game (as shown in Table 1), each level has different codes and once the codes are found, the agreed funds in that escrow is released as reward by both parties signing. In this method, the game player and game manager are two parties who may need to sign to spend the funds. The agreement to spend is done in external methods.

In incremental project payment, the hierarchy of funds are encoded in escrow addresses. Upon completion of project the corresponding agreed amount of funds will be released by both parties signing (as shown in Table 2). In this method, the project manager and worker are two parties who may need to sign to spend the funds. Similarly, the agreement to spend is done in external methods.

TABLE 1. Different Game Levels in the Multi-signature P2SH Method.

Game levels	Both verify codes found in game and sign.	Reward	Comments
Level-1	Code1	100$	Both parties sign to withdraw
Level-2	Code1 + Code2	500$	Both parties sign to withdraw
Level-3	Code1+Code2+Code3	1000$	Both parties sign to withdraw

We consider all the five Core Components used for the design of hierarchy in Blockchain:

- Hierarchical design solution on Blockchain with escrows
- Locking funds in escrow on Blockchain
- Creating controls to release funds from escrows
- Application design to determine type of locking in escrows
- Application control of funds based on states of application

TABLE 2. Different Project Levels in the Multi-signature P2SH Method.

Project levels	Both verify completion and sign for payment.	Payment	Comments
Level-1 (advance)	Manual verification	100$	Both parties sigh to withdraw
Level-2 (after 2 months)	Manual verification	500$	Both parties sigh to withdraw
Level-3 (after completion)	Manual verification	1000$	Both parties sigh to withdraw

Method 2: Single-user signature with hierarchy of control codes (P2PKH)

The escrow address created using Fig-2C of [2], Sheet 20 there is a drawing that mentions of custom contract being a P2PKH and being used in the composite-key method. Here the spending depends on hierarchical codes and signature for P2PKH for spending the funds in escrow address. In multi-level game, each level has different codes and once the codes are found, the funds in that escrow is released as reward (as shown in Table 3). The manager has to sign along with the codes to release funds. In this method, the game player will find the codes by playing the game. The game manager will sign after accepting the code provided by the game player. The funds will be released as reward.

TABLE 3. Different Game Levels in the Single-user signature with hierarchy of control codes.

Game levels	Hierarchy of codes	Reward	Comments
Level-1	Code1	100$	Manager signs along with code
Level-2	Code1 + Code2	500$	Manager signs along with code
Level-3	Code1+Code2+Code3	1000$	Manager signs along with code

In Project payment, the hierarchy of funds are encoded in escrow addresses. Upon completion of project the corresponding code is entered and the manager also signs to release the funds. In this method, the project worker will get the codes from validator upon completion (as shown in Table 4). The project manager may need to sign after accepting the code provided by worker. The funds will be released as payment.

TABLE 4. Different Project Levels in the Single-user signature with hierarchy of control codes.

Project levels	Completion codes	Payment	Comments
Level-1 (advance)	Code1	100$	Manager signs along with code
Level-2 (after 2 months)	Code1 + Code2	500$	Manager signs along with code

Level-3 (after completion)	Code1+Code 2+Code3	1000 $	Manager signs along with code

We again consider all the five Core Components used for the design of hierarchy in Blockchain:

- Hierarchical design solution on Blockchain with escrows
- Locking funds in escrow on Blockchain
- Creating controls to release funds from escrows
- Application design to determine type of locking in escrows
- Application control of funds based on states of application.

Method 3: Hierarchy of control codes

The escrow address created using Fig-2C of [2], Sheet 20 there is a drawing that mentions custom contract has one or more codes being used in the composite-key method. Here spending depends on hierarchy codes for spending the funds in escrow address. In the multi-level game, each level has different codes. These codes are used to create escrow address. Upon using these codes, the funds in that escrow is released as reward (as shown in Table 5). In this the game player will be get codes upon winning. It can be part of the game. The funds are released by entering correct codes.

TABLE 5. CORRELATION TABLE FOR FACTOR 5

Game levels	Hierarchy of codes	Reward	Comments
Level-1	Code1	100$	The codes are sufficient to withdraw
Level-2	Code1 + Code2	500$	The codes are sufficient to withdraw
Level-3	Code1+Code 2+Code3	1000$	The codes are sufficient to withdraw

In Project payment, the hierarchy of funds are encoded in escrow addresses. Upon completion of project the corresponding code is required to release the funds (as shown in Table 6). In this method, the project worker will get codes from validator upon completion. Upon using these codes, the funds in that escrow will be released as payment.

TABLE 6. CORRELATION TABLE FOR FACTOR 6

Project levels	Completion codes	Payment	Comments
Level-1 (advance)	Code1	100$	The codes are sufficient to withdraw
Level-2 (after 2 months)	Code1 + Code2	500$	The codes are sufficient to withdraw
Level-3 (after completion)	Code1+Cod e2+Code3	1000$	The codes are sufficient to withdraw

We again consider all the five Core Components used for the design of hierarchy in Blockchain:

- Hierarchical design solution on Blockchain with escrows
- Locking funds in escrow on Blockchain
- Creating controls to release funds from escrows
- Application design to determine type of locking in escrows
- Application control of funds based on states of application

III. IMPLEMENTATION & EVALUATION OF HIERARCHY DESIGN IN BLOCKCHAIN ESCROWS

A. Design:

The complete design for both the multi-level game reward and incremental project payment design are mentioned below:

Multi-level Game Reward Design

- Design of hierarchy game design consists of the following activities
- Define the levels of games
- Define the players, managers, controllers
- Create random codes for each level and arrange in hierarchy
- Use chosen method to create escrow address
- Create process for player to play and obtain the code
- Create process for encashing the funds in escrow
- Create process to withdraw unused funds in escrow

Incremental project payment design:

- Design of hierarchy for project payment consists of the following activities
- Define the relationship between parties in the project
- Payment at each level of the project for parties
- Creation of smart contracts for payment at each level
- Identifying validators for validating progress
- Create random codes for each level and arrange in hierarchy
- Using chosen method create escrow address
- Create process for workers to work and obtain the code
- Create process for workers to encash the funds in escrow
- Create process to withdraw unused funds in escrow

B. Typical Architecture of Solutions

Below architecture of solution design for hierarchical games and incremental payments is provided. This depends on type of method chosen to implement the solution.

Method-1 uses multi-signature technique. Method-2 uses single-user signature with composite-key method, Method-3 uses only composite-key method.

Method 1: Multi-signature (P2SH)

The method of creating escrow address using the first method is shown below. In [3], Pay-to-Script-Hash section an example for the multi signature is provided. In that example, the script requiring both parties to sign for multilevel game and incremental project payment will be as below

OP_2 <pubKey1> <pubKey2> OP_2 OP_CHECKMULTISIG

The pubKey1, pubKey2 correspond to the two parties requiring signing.

The Escrow address is created based on above script, in P2SH method.

The corresponding <input to script> is as below

0 <sig1> <sig2>

The 0 in above corresponds to a workaround for a Bug mentioned in [4]. [5] with OP_CHECKMULTISIG

a) for multi-level games & reward

In this method, both parties will sign to withdraw funds after the player wins the game.

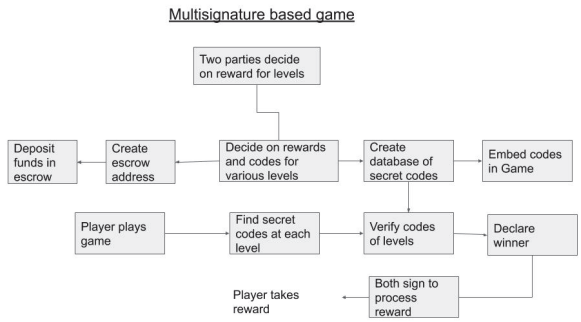

Fig. 1. Architecture of Multi-signature (for game reward)

b) for incremental project & payment

In this method, both parties will sign to withdraw funds after the worker has performed the work.

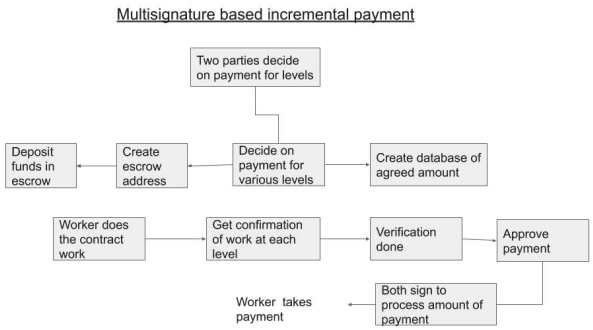

Fig. 2. Architecture of Multi-signature (for project payment)

Method 2: Single-user signature with hierarchy of control codes (P2PKH)

In Fig-2C of [2], Sheet 20 there is a drawing that mentions of custom contract being used in the composite-key method. The custom contract for multilevel game and incremental project payment will be as below

Custom contract needed is

OP_DUP OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG

When pubKeyHash is the multilevel game managers or project managers public-key hash

The composite-key needed can be created as in drawing in Fig-4B of [2], Sheet 12 using JSON data

Var jsondata = {

Level1: 'code1'

};

The example of converting JSON data to composite-key is shown in sheet3 of FIG-3A. The corresponding <input to script> will be

<sig> <pubKey> <compositeKey>

This ensures the multilevel game manager or project manager signs for spending funds. It also requires the necessity of providing code.

(a) for multi-level games & reward

In this method, game manager will sign to withdraw after the player finds the code and wins the game.

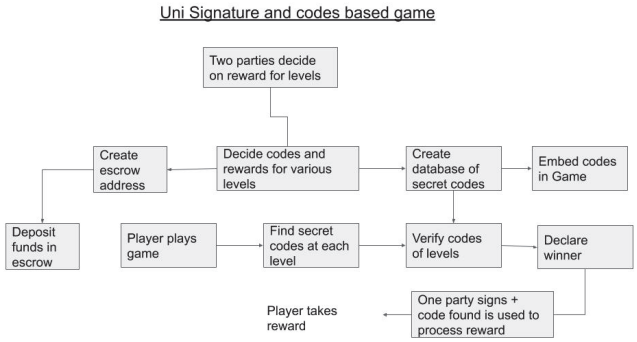

Fig. 3. Architecture of Single-user signature with hierarchy of control codes (for game reward)

b) for incremental project & payment

In this method, project manager will sign to withdraw after the worker gets the code from validator after performing the work.

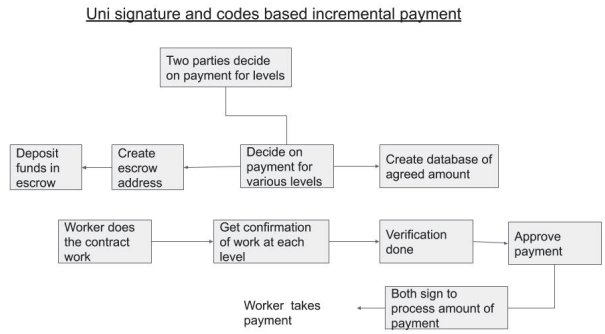

Fig. 4. Architecture of Single-user signature with hierarchy of control codes (for project payment)

Method 3: Hierarchy of control codes

In Fig-4B of [2], Sheet 12 there is a drawing that mentions the use of composite-key method. The composite-key is created as JSON data

Var jsondata = {

Level1: 'code1',

Level2: 'code2'

};

The example of converting JSON data to composite-key is shown in sheet3 of FIG-3A. The corresponding <input to script> will be the <compositeKey>

This ensures the both parties provide keys and also the codes for levels to execute the payment.

(a) for multi-level games

In this method, the player finds the code by playing and use that to withdraw funds.

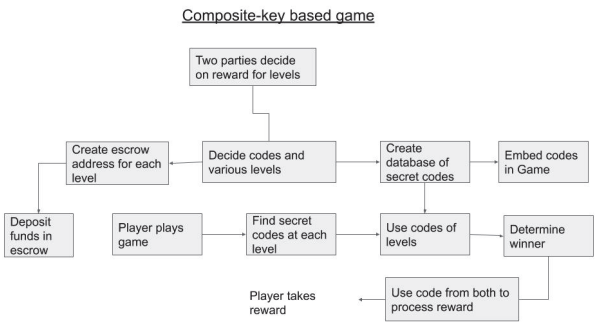

Fig. 5. Architecture of Hierarchy of control codes (for game reward)

(b) for incremental payment

In this method, the worker gets the code from validator after performing the work and uses that to withdraw funds.

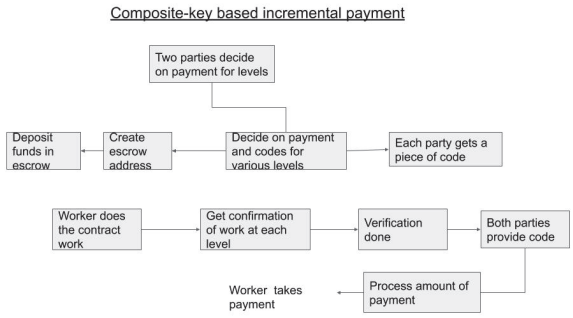

Fig. 5. Architecture of Hierarchy of control codes (for project payment)

IV. Results & Discussion

In design of smart contracts on Blockchain in industry, Ethereum and Hyperledger smart contracts are created to perform multilevel gaming or incremental project payment solutions. In this article three methods are provided that can be used in Blockchains which uses Bitcoin type of scripts.

Method 1 (Multi-signature) is suitable where enforcement is done outside the Blockchain. Where in both parties see the amount to be spent from escrow is "as mutually agreed". If either party does not sign, the amount cannot be spent. The amount to be spent can be kept cumulatively in one escrow address for each level of payment to be disbursed.

Method 2 (Single-user signature with hierarchy of control codes) is suitable where a manager wants to sign for every transaction. The enforcement is done inside the Blockchain. However, it also requires a manager to sign for every transaction. For each level of payment, a separate escrow address is created. There is a necessity to provide the codes applicable for the level along with signature for the manager to spend. The other party can confirm that the money has kept in escrow before playing the game or working for the contract. The codes needed for spending has to be kept secret from manager for some duration of contract to prevent him from emptying the contract voluntarily. In the context of game, the code may be required to be found in the game. In the incremental payment the validator may provide the code upon verifying the work done.

Method 3 (Hierarchy of control codes) is suitable where either party does not want to sign for transaction. The enforcement is done inside the Blockchain. However, it may require codes at each level for every transaction. For each level of payment, a separate escrow address is created. There is a necessity to provide the codes applicable for the level to spend. Each party can confirm that the money is kept in escrow before playing the game or working for the contract. The codes needed for spending has to be kept secret for some duration of contract or game. In the multi-level game, the player may have to find the code of the level. In the case of incremental payment, the validator may provide the code upon verifying the work done.

V. Conlusion & future works

The bitcoin type of public blockchains can be used in a variety of applications including construction management [8] requiring execution of smart-contract based on a signature of an owner or an owner of a secret-key. They can also be structured to work together as shown in this article. The hierarchical execution of smart contracts will be useful in funding the management of projects by incremental payments. It can also be used in gaming applications where multi-level rewards need to be issued. This method allows for transparency and also control over funds through Blockchain [9] [10].

References

[1] S. Nakamoto, “Bitcoin: A peer-to-peer electronic cash system,” 2008.

[2] Belavadi Ngarajaswamy Ramesh, System and Method for composite key based Blockchain device control, US patent application 16/101,452

[3] https://en.bitcoin.it/wiki/Transaction

[4] The combined validation script begin with 0, a work around for CHECKMULTISIG bug https://blockgeeks.com/guides/bitcoin-script-guide-part-2/

[5] Pay to Script hash https://www.oreilly.com/library/view/programming-bitcoin/9781492031482/ch08.html

[6] S. Jagdeep, "Syscoin: A peer-to-peer electronic cash system with blockchain-based services for e-business." 2017 26th international conference on computer communication and networks (ICCCN). IEEE, 2017.

[7] L. Eunhee, and Y. Yoon. "Project management model based on consistency strategy for blockchain platform." 2019 IEEE 17th International Conference on Software Engineering Research, Management and Applications (SERA). IEEE, 2019.

[8] T. Žiga, and R. Klinc. "Potentials of blockchain technology for construction management." Procedia engineering 196 (2017): 638-645.

[9] M. Sathiyanarayanan, S. Sokkanarayanan. Understanding the Emergence and Importance of Blockchain-based Cyber-physical Social Machines: A Proposal. In2019 International Conference on contemporary Computing and Informatics (IC3I) 2019 Dec 12 (pp. 142-147). IEEE.

[10] G. Tripathi, MA. Ahad, M. Sathiyanarayanan. The Role of Blockchain in Internet of Vehicles (IoV): Issues, Challenges and Opportunities. In2019 International Conference on contemporary Computing and Informatics (IC3I) 2019 Dec 12 (pp. 26-31). IEEE.