A Survey of Blockchain and Smart Contract Applications

Feiyang Qu[1], Hisham Haddad[1], Hossain Shahriar[2]
[1]Department of Computer Science
[2]Department of Information Technology
Kennesaw State University, USA
fqu@students.kennesaw.edu, {hhaddad, hshahria}@kennesaw.edu

Abstract - Blockchain technology enables sharing of information between "individual computers". The potential of blockchain application is not only in e-currency, but also in secure data sharing and identity management. Ethereum is a representative example of a blockchain-based platform. Created by Vitalik Buterin, Ethereum is an open software construct based on smart contracts. Contract is one kind of account in Ethereum. In a contract, there is data storage and code included for sending a transaction to another. The contract can not only read code and currency in a transaction, but also create another contract for reading code and writing into storage. In this book chapter, we perform an in depth survey of some recent examples and applications of blockchain and smart contracts.

Keywords – Smart Contract, Blockchain

1. INTRODUCTION

Blockchain technology enables sharing of information between individual computers [1]. It can control personal information for better protections against security breaches. For example, in Ethereum wallet (E-wallet), each user has an individual account ID (Address) along with private and public keys. Users who try to use E-wallet to complete transactions use the address and public keys, which does not contain actual personal information.

Ethereum is an example of a blockchain-based platform created by Vitalik Buterin, Ethereum is an open software construct based on smart contracts [9]. Ethereum enables blockchain developers to build and deploy Decentralized Applications (*DApps*). Ethereum provides the foundation upon which many of blockchain applications are written today.

Ethereum based smart contract is one of the most successful blockchain platform. There are two kinds of accounts in Ethereum – externally owned account (EOA) and contract [9]. EOA is similar as a bitcoin account which can be a wallet. User can store and send e-currency to another accounts. Contract is the other kind of account in Ethereum [9]. In a contract, there is the data storage and code included for sending a transaction to another address. The contract can not only read code and quantity of currency in transaction, but also create other contracts for reading codes and writing into storage.

The scripting language of Ethereum is called *Solidity*. User can create a complex smart contract in Solidity for sending transactions to other addresses or writing transactions into storage [9]. However, Ethereum must keep all smart contracts on the Ethereum blockchain, which results in Ethereum network increased storage spaces. Therefore, Ethereum created a concept named *Gas* to count the computational step that an account can execute. By using Gas to limit the code size of smart contract, the computation pressure for running Ethereum will be smaller.

In this chapter, we will focus on smart contracts and related applications. The chapter is organized as follows. We will first introduce backgrounds of blockchain and smart contracts with diagram examples in Sections 2 and 3, respectively. We then compare traditional contracts and smart contract in Section 4. Section 5 describes applications of smart contracts for E-auction while also discussing the advantages of smart contract in E-auction business. Section 6 highlights some applications of blockchain in healthcare, while Sections 7 and 8 provide example applications of blockchain in supply chain and Internet of Things (IoT) area. Section 9 elaborates blockchain for Vehicular Adhoc Network (VANET), while Section 10 shows some recent applications of blockchain in the Cloud and Education area. Section 11 discusses application of blockchain for Ponzi scheme detection. Finally, Section 12 concludes the chapter and discusses some challenges and opportunities.

2. BACKGROUND ON BLOCKCHAIN

Generally, blockchain is a distributed ledger and each block contains several transactions [15]. Blocks can be

linked by using cryptographic hash functions. In a public blockchain, there are two ways to find or mine next block – Proof-of-Work (PoW) and Proof of Authority (PoA). PoW means that the miner or the winner of new block must finish the most amount of computational works than any other participants. Based on the PoW theory, a broker must own more than fifty percent of the whole network computational ability to control the next block. Therefore, it is difficult to control a decentralized Blockchain by one part when the input might be more than the outcome. The PoA is the other way to find the next block. The owner of next block should have the largest amount of resource or the highest technology than any other miners in the network. In this case, the owner who has higher technology or more resources is able to keep blockchain running in a more energy efficient environment. The interaction between blockchain and applications is normally accomplished by smart contracts, which are agreements between two or more parties.

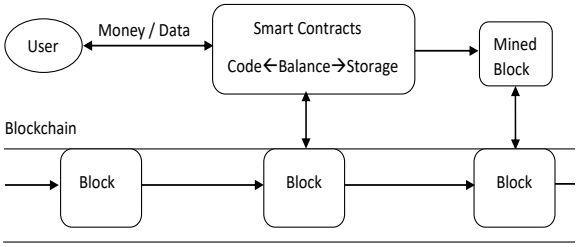

Figure 1: Blockchain and Smart contract Structure Introduction [17]

Figure 1 shows the interaction between blockchain (a collection of blocks as shown in the bottom layer) and smart contracts which result in mined block and continuously being added to the chain. Here, a contract is triggered when exchange of money or data takes plan among users.

3. SMART CONTRACTS

Smart contract is a type of code running on the blockchain, which is executed compulsory by the consensus protocol (e.g. PoW, PoS)[18]. A smart contract can include a group of functions and rules written in Solidity programming language to process resources exchange when certain events take place in the peer-to-peer network. Therefore, smart contract can be exploited in a wide range of applications, including financial instruments and self-enforcing or autonomous governance applications. When user creates and calls a smart contract on the blockchain, it needs to include an address (160-bit unique identifier). Users can use address to send transactions. In the blockchain, all miners will execute smart contracts which are created and uploaded recently after the entire blockchain receives a new transaction and the recipient's address. When all participants on a network agree with a new output block, the next block will be processed using a consensus protocol (e.g. PoW, PoS). Figure 2 shows an example of contract for rewarding a user to solve a puzzle. Lines 2 to 6 define variables (e.g., owner, solution, reward); Lines 8 to 13 define detail of owner and reward (through the constructor method Puzzle); Lines 15 to 31 defines the event to check if a received value is equal to reward and decides if a received message is reward or not.

```
1  contract Puzzle {
2  address public owner ;
3  bool public locked ;
4  uint public reward ;
5  bytes32 public diff ;
6  bytes public solution ;
7  function Puzzle () // constructor {
8     owner = msg.sender;
9   reward = msg.value;
10   locked = false;
11   diff = bytes32 (11111); // pre - defined difficulty
12 }
13 function () {// main code , runs at every invocation
14    if (msg.sender == owner ) {// update reward
15      if (locked)
16           throw;
17       owner.send (reward);
18       reward = msg.value ;
19     }
20    else
21       if ( msg.data.length > 0){ // submit a solution
22          if (locked) throw;
23          if (sha256 (msg .data) < diff){
24             msg.sender.send (reward); //send reward
25             solution = msg.data;
26             locked = true;
27          }
28     }
29  }
```

Figure 2: A contract that rewards users who solve a puzzle [18]

As a common cryptocurrency, Ethereum can store smart contracts on blockchain that code and value can use multiple times [18]. Ethereum has Gas, an internal measurable unit, for keeping the fair compensation for expanded computation effort. Specifically, users need to verify the amount of gas they use to send a transaction to invoke a contract. The price of gas is announced publicly and each instruction in bytecode has a gas limit. If the gas needed in actual processing is more than the gas provided by users, the execution

of transaction will be terminated as the state will be returned to the initial state; meanwhile, the user must pay the actual amount of gas used, which is a way to avoid resource-exhaustion attacks.

A smart contract can also be a self-execution code in blockchain. When a smart contract is created and uploaded into blockchain, it will include a contract address and some amount of Ether [18]. Users create a contract by high-level programming language, Solidity; a contract to be executed will contain a recipient address, which payment for execution and data is needed.

Solidity is a high-level language for smart contract. Programmers write smart contracts by Solidity and compile them to bytecode of the Ethereum virtual machine (EVM), which is a stack-base virtual machine operating on 256-bit words [13]. A contract is executed into a transaction which contains the bytecode and initialization parameters. The smart contract stores into one block and each contract has the unique blockchain address. Users can interact the specific contract by address, contained functions and arguments. The followings are several issues in Solidity when users try to develop smart contracts.

*Security***:** It may be the cause of a hacking user account or contract. One preference in solidity coding is that using *send* instead of *transfer* in transaction process which might cause throwing issue [13]. In Figure 3, the first example suggests users not to use strict balance equality, which hacker can use definition of Gas to mandatorily send Ether from one account to another account. The second example explains that a better way for calling external codes should include a check function, which can avoid to lost Ether on external codes that cannot found. The last example suggests that a conditional statement should not depend on unauthorized external call, which external implementation may throw an exception and cause security issue.

Functional issue: This may cause the un-expected functionality problem or bug. For example, there is not a float or a decimal type in Solidity [13].

Operational issue: This can normally cause run-time problems (e.g., bad performance). For example, it is better to use bytes instead of byte[] to reduce gas consumption [13].

```
if ( this . balance == 42 ether ) { /* ... */} // bad
if ( this . balance >= 42 ether ) { /* ... */} // good
// Balance equality
addr. send (42 ether ) ; // bad
if (!addr. send (42 ether ) ) revert ; // better
addr. transfer (42 ether ) ; // good
//Unchecked external call
function dos( address oracleAddr) public {
        badOracle = Oracle(oracleAddr);
        if (badOracle.answer() < 42) { revert ; }
        // ...
}
//DoS by external contract
```

Figure 3: Security issue examples

```
for ( uint256 i = 0; i < array. length ; i++)
{ costlyF(); }
It will waste Gas which exceeds the Gas limit if
array. length is large enough.
```

Figure 4: Costly loop

Developmental issue: Sometimes the code may be hard to understand and improve. In Solidity, the private modifier will not make a variable invisible, which means Miners can access to all contracts' code and data [13]. Therefore, the lack of privacy is an issue in Ethereum. The following are specific examples/codes for each different types of issue. In figure 5.2, a bad example which not follows Solidity guide about lower and uppercase letter will decrease readability of code (see Figures 5.1 and 5.2).

```
function foo() { /*...*/ } // bad
function foo() public { /*...*/ } // good
function bar() private { /*...*/ } // good
```

Figure 5.1: Implicit visibility level

```
function Foo(); // bad
event logFoo(); // bad
function foo(); // good
event  LogFoo(); // good
```

Figure 5.2: Style guide violation

Base on the function of smart contract included, a smart contract can call the other contract to finish transactions [13]. Costs for transactions happen when users attempt to send values to other addresses. The propose of cost during transaction is to avoid the spamming in smart contracts. Therefore, the Ethereum protocol defines the name of costs during transactions as gas. The gas will be charged based on users expected cost of transaction for each Ethereum Virtual

Machine (EVM) operation. The expected gas to send is usually more than actual gas cost, and the computation will give partial refunds back to user's addresses. If an exception happens, which includes that actual cost is more than expected cost, all state changes are reverted, and the gas will be returned or not base on the function of smart contract. Currently, the unit price of gas is determined by the market.

Ethereum is a "hostile execution environment", which anonymous hackers can send smart contracts contained overload function to get money[13]. Due to the feature of blockchain and smart contracts, developers cannot handle the contract which has already been deployed. "The DAO", a well-known example in June 2016, which hackers steal more than ten million dollars from flawed contracts. A rapid growing in this field will bring a huge blockchain and related support technology market for profit; however, many criminal or similar issues happen due to outdated of many sources.

There are some security challenges presently in Ethereum [13].

Unfamiliar execution environment – Ethereum is not a centralized environment; therefore, developers are not familiar to execute into a "profit-driven" node which is also an anonymous environment.

New software stack – "The Ethereum stack (the Solidity compiler, the EVM, the consensus layer, etc.) is under development, with security vulnerabilities still being discovered."

Very limited ability to patch contracts – Based on the feature of blockchain, a contract cannot be patched or edited after been deployed. Therefore, a contract must be corrected or compiled before uploaded.

Anonymous financially motivated attackers – As mentioned before, hackers use smart contracts to steal money that will be "safer" and gain higher profit; furthermore, they are anonymous on blockchain.

Rapid pace of development – "Blockchain companies strive to release their products fast, often at the expense of security."

Suboptimal high-level language – "Some argue that Solidity itself inclines programmers towards unsafe development practices".

4. CONTRAST BETWEEN SMART AND TRADITIONAL CONTRACTS

Compare with the traditional online bidding system, blockchain based bidding system does not need a third party to help trading [10]. It can save the cost and solve the trust problem. Therefore, people need to create a blockchain based smart contract to finish bidding process. The smart contract contains the address of auctioneer, date information, the current winner and highest price. Base on the feature of blockchain, the smart contract generated via Ethereum will keep the bill in secure, private and inalterability because all transactions are recorded in a decentralized ledger as the same as each node.

The traditional E-auction is popular right now [10]. It doesn't need a bidder to attend and bid together. And E-auction can reduce the cost of transaction and service fee. However, E-auction is still a centralized bidding system which cannot keep customers' personal information in privacy; meanwhile, bidders cannot make sure if the information they receive (bid price) is real.(Figure 6) Figure 6 shows the flowchart of E-auction process where auctioneers share bidding item information. Then bidders bid the prices, and the highest price bidder is notified as the winner. The winner pays the price and an item is sent to buyer.

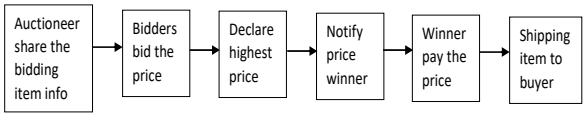

Figure 6: The flowchart of E-auction process

5. BLOCKCHAIN IN E-AUCTION

Ethereum platform has been used to execute E-auction [10]. Before creating smart contracts, there are several announcements in advance: (1) Auctioner: the main address that records the origination contract; (2) AuctionStart: information about auction start time; (3) biddingTime: information about bidding effective time; (4) highest bidder/highest price: records about the highest price and its owner.

There are five functions defined for an auction: (1) AuctionStart(): it is called to check if the start time of bidding actives; (2)AuctionTime(): it is called to check if the auction started and if the effective time expired; (3) blindAuction(): activate the contract by calling this function; (4) Bid(): a bidder can call this function to bid and send to the ideal price to auction which is higher than the current highest price; (5) AuctionEnd(): this function will check if the bidding ended, and send the highest price with owner information to Auctioner; (6) withdraw(): it returns the bidding price back to the bidder who is not successful in the bidding[10].

As blockchain gives organizations more privacy and maintains a complete transcript in blockchain, the problem from the cost of storage and transaction executing speed per second become problem [12]. The Cloud can be a solution that provisioning blockchain as a service on cloud infrastructure even though organizations are looking for solution from third-party protocols on top of existing blockchain.

6. BLOCKCHAIN IN HEALTH APPLICATION

Presently, healthcare organizations are trying to develop blockchain base platforms for cross-institutional sharing of healthcare data. It has the potential to share data on blockchain instead of traditional way of requesting data from different institutions [14]. However, there are still some problems in sharing data between organizations. When some organizations concern about privacy problems or commercial competitions, they might deny the access to share data from others. Also, blockchain technology requires several institutions in a similar technical infrastructure level, which is hard to establish in actual world. Finally, different healthcare institutions sharing data require a standard data prototype. If one organization uploads data by different data types, and the other organizations cannot understand the data, it might cause errors in blockchain.

The problems of sharing healthcare data can be divided into three parts: (i) Security (ii) Infrastructure and (iii) Interoperability[14]. They are discussed below

Security: Security problem is one of the largest problems in data sharing currently not only in the healthcare emphasis. It might cause financial or legal consequences. There are two points to solve the security problem in blockchain based healthcare data sharing. First, each data sharing request must be processed between authorized access because an unauthorized address could expose the commercial advantage of organization or "reveal proprietary practices". The other point is to improve data anonymity of the user/organization in blockchain. This solution requires the user's information be separated from patient records for sharing before the information uploads to blockchain. And, developer can create an extra smart contract to include user's information.

Infrastructure: There are some infrastructure requirements during data sharing, which can be solved by increasing technical consistency of each block miner (data's owner/organization) in future. Data sharing between several organizations needs a centralized data source or "the transmission of bulk data to other institutions". However, centralized data source might request the trust of a single authority, which is the risk of security; and, bulk data transmission requires organizations to monitor, control and re-edit data during transfer data, which will increase quantity of work.

Interoperability: In the healthcare emphasis, the data is more complex due to large number of professional data/prototypes included. Therefore, when some non-healthcare background organizations request access of data sharing, interoperability of healthcare records is hard to solve. There are two types of problem in interoperability: data structure and semantics. Due high volume of professional knowledge included in healthcare data, data structure might be different from normal structures in organizations. Therefore, healthcare organizations need to create a standard data structure which non-healthcare organizations can be recognized and used directly. For semantics problems, healthcare organizations should create a professional digital dictionary or a smart contract for user, which can be called to translate those professional data to data which can be analyzed[14].

Blockchain-based Electronic Medical Records (EMR) information can benefit healthcare providers and physicians because of efficiency [2]. This approach gives researchers access to broad and comprehensive data sets to advance the understanding of diseases, facilitate the development of new drugs, and enhance biomedical discovery.

EMR information can be managed by blockchain-base applications as EMR information is mostly standardized[2]. With blockchain, healthcare activities, such diagnosis, blood work, and X-ray can be created as digital transactions that are then grouped into encrypted blocks with other transactions. Trusted individuates, such as administrators, physicians, and technicians, can access and validate transactions using access keys and then timestamp the transactions. Timestamps for validated blocks create sequences that show the order and procedure for every transaction. This approach improves the accuracy of patient records as transactions cannot be irreversible.

Factom is a software company that uses blockchain technology to store data on a decentralized system[1]. As a feature of *Factom's* technology, healthcare organizations can create smart contracts to develop medical data. The medical data needs to be encoded with a fingerprint of the data. Blockchain uses digital fingerprints to verify processes and time-stamping. Therefore, *Factom* technology can help healthcare

organizations to protect patient's information confidentiality.

There is decentralized record management system for Electronic Medicine Record – *MedRec. MedRec* uses blockchain technology [8]. The system allows patients to manage the log and access their data across providers and treatment sites. Base on the technology of blockchain, *MedRec* has an ability to improve authentication, confidentiality, accountability and data share. Those features are important for handling some sensitive information. The system encourages medical providers, researchers, public health authorities to be the "miner". The propose of this activity will give "miner" access to retrieve aggregate, anonymized data as rewards of mining in return for sustaining and securing the network via Proof of Work.

In the Electronic Medical Record (EMR) environment, it is important to establish trust and continued participation in health care organization [8]. On patients' side, they don't need to doubt anymore confidentiality of their records and they can totally trust the decentralized record management system for managing their records. On researcher's side, the blockchain based EMR system can help scientists to keep track of the accuracy of data. Meanwhile, data on blockchain will be shared more easily.

The decentralized management system *MedRec* records the relationship of patient and provider by smart contracts on Ethereum [8]. The system uses relationship smart contracts to check permissions and data retrieval instructions for external databases use. Under this foundation, providers can add a record for patients; patients can also access the sharing of records between providers.

Basically, the EHR system contains three types of smart contracts: Registrar Contract, Patient-Provider Relationship Contract, Summary Contract. Registrar Contract is used to identify the address of users [8]. Patient-Provider Relationship Contract is used between two nodes on blockchain which stores and manages medical records. Summary Contract is used for holding a list of references, which helps participants to locate their medical record histories.

7. BLOCKCHAIN IN SUPPLY CHAIN

Blockchain and related applications can also be developed as blockchain-enabled composites materials supply chain [4]. Raw materials, semi-finished materials, components, structures and Original Equipment Manufacturers (OEMs) can create several smart contracts based on blockchain. When one process finishes, the next process automatically starts based on the previous result. The transfer of goods can be more efficient and traceable by all owners in the blockchain. This means each owner can estimate the profit from the transaction process; meanwhile, the owners can fully stock the product before their own process. Figure 7 shows blockchain-enabled supply chain systems.

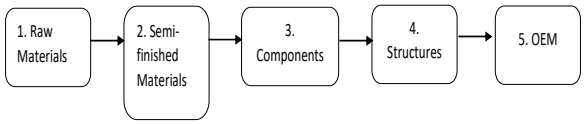

Figure 7: Blockchain-enabled composites materials supply chain

In Figure 7, each stage is a node in blockchain, which contains multiple transfers of records as a distributed ledger: 1. Raw Materials: provide original materials in supply chain (e.g. steel, plastic); 2. Semi-finished Materials: provide materials which finish the last step before produce composite materials; 3. Components: provide the design and technology to develop composite materials for final productions; 4. Structures: provide and collect all materials before assembled; 5. OEM: provides products for selling which is the last stage in supply chain process [4].

Furthermore, composite material suppliers can use blockchain to keep transparent and provenance-tracking of records since it is able to monitor certificates linked to the quality of the component and true origin in the process [4].

Logistics management focuses on the minimization of time and cost of goods sold, so, coordination of each supplier and information sharing efficiently is important [5]. Ethereum uses two different types of accounts, which contain an address of 20 bytes, for different users. An Externally Owner Account (EOA) is an account created with a private key. A Contract Account (CA) is controlled by a code which includes transactions or messages for modifying the storage, calling other contract functions and creating new contracts.

For logistic management process, there are several smart contracts developed [5]. In Figure 8, first part of code shows the identification of address for EOA and CA address; second part of code shows that address needs to be added in the function with the transaction information.

bytes4(keccak256(“setName(strinд)”)) = 0xc47f 0027
--------- (1)
EOAaddress = 0x43f c11,
CAaddress = 0xa8484f 3,
Data[name] = “Paul” ≡ 0x50 0x61 0x75 0x6c
Txdat a = 0xc47f 0027 · · · 5061756c · · · --------(2)
RPCr equest = {“jsonrpc” : “2.0”,
“method” : “eth_sendT ransaction”,
“params” : [{“from” : “0x43f c11”,
“to” : “0xa8484f 3”,
“data” : “0xc47f 0027 · · · 5061756c · · ·”}],
“id” : 1}

Figure 8: Smart Contract

The *Package* contract contains the information of package and its owner’s information. As a package contract being created, the information is updated and shared with each node immediately[5]. Package receiver, supplier and shipper are all parts of nodes. One of functions should be named as Updated function which is for updating uses.

The *User* contract contains public and private information. Public information includes passenger information which can be shared for all parties in tracking the shipment. Private information includes user’s information for verifying proposes [5].

The *Delivery* contract contains information of the sender and recipient of a package. Basically, only the user in Delivery contract can control and update the package status [5].

Compared with traditional logistic management, using blockchain and smart contracts makes shipping transparent as well as timely update notifications [5]. Especially, blockchain-based smart contract can be a better substitute of logistic management for special products like medicine, chemical product, and fresh seafood.

More and more researches appear for using blockchain and smart contract into logistical emphasis. Under the perspective of logistics, IoT is a kind of “network of material flow objects”, and some objects in network can communicate over Internet [19]. In the future, a blockchain based IoT object can improve logistical processes.

A new smart contract interface has been developed onto Ethereum Github repository – Custodian-Client Contrast Standard. Smart contracts are stored on the blockchain, which is decentralized [7]. Users can store, retrieve and exchange data by smart contracts.

A two-level hierarchical architecture is created, which contains two types of smart contracts: custodian contract and client contract [7]. Custodian contract is high-level contract that deploys client contract based on client demand and gives feedback for different using. Client contract is the low-level contract that collects data and sends requests to the custodian. Once the custodian contract receives the request, a new transaction is created and updated to client contract. Several clients can share their information based on blockchain technology (see Figure 9).

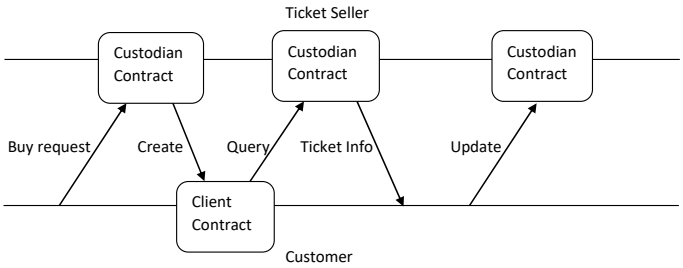

Figure 9 : The process for buying a ticket based on Custodian-Client contract structure

To understand the Custodian-Client contract architecture, an example about using a Custodian-Client contract in ticket selling system shows above[7]. This includes the ticket seller (custodian) and the customer (client). When the customer requests a ticket, a new smart contract is created to include the customer’s address. This client contract is owned by the customer and can be sold to or exchanged with another customer. The seller can update the ticket information and track the owner of ticket. Meanwhile, several custodian contracts are needed to create tickets, track customer’s information, update ticket information.

A custodian contract is the seller of ticket and no other custodian contract is in experiment. The custodian contract has access to retrieve information from the client contract and only update the ticket owner by requests (for exchanging/re-selling tickets).

Compared Custodian-Client contract with another standard interfaces for Ethereum blockchain, the Custodian-Client contract standard can provide a flexible independence of client contract which is traceable by the custodian [7]. Also, the Custodian-Client standard will reduce possible losses when cyber-attacks occur, since the interaction between client contract has been minimized and each client contract is independent of each other.

8. BLOCKCHIAN IN INTERNET OF THINGS (IOT)

There is an IoT data management environment to simulate the application of IoT devices in smart city initiatives. Currently, smart devices can be found everywhere, which is from household appliances to mobile terminal. Therefore, we can consider that all of smart devices make a system of city which is smart city. The experimental environment is based on the Ethereum blockchain and smart contract [6]. Specifically, it needs to explain: how to link different computing capabilities of IoT devices using blockchain technology and smart contracts; and how to store and retrieve data from different IoT devices on blockchain by smart contracts. Figure 10 shows an example environment of IoT simulation. Here, sensors (temperature, humidity, pressure) are connected to proxy node, which can communicate with blockchain network. Figure 10 shows an example environment of IoT simulaton. Here, sensors (temperature, humidity, pressure) are connected to proxy node, which can communicate with blockchain network.

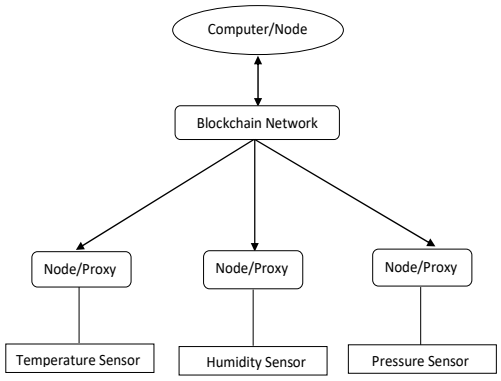

Figure 10: Blockchain based IoT Simulation Structure

Currently, smart devices are widely used. As the usage of smart devices increases, the server/client storing data faces some challenges as to how to efficiently store large amounts of data and securely retrieve such data [6].

Across the technology of blockchain, data from different IoT devices can be stored and retrieved by smart contracts. Therefore, the experiment needs to build up a private Ethereum blockchain which contains a computer and several IoT devices as nodes[6]. After setting up the environment, the project needs to design smart contracts in some using: (1) store data and retrieve data from blockchain; (2) check requests from different IoT devices for sharing information for data analytics; (3) verify the information from users for retrieving data/transaction and send notifications/alarms to users about log-in status.

Finally, there is still an issue of flexibility in using smart contracts from different types of IoT devices. In a private blockchain, IoT devices need to be connected for different proposes. Therefore, it is necessary to develop a high-level language and an operating environment that are both compatible with blockchain based smart contracts which are for IoT devices [6].

9. BLOCKCHAIN INVANET & LOW VOLTAGE ENERGY COMMUNITY

Vehicle Ad-hoc Networks (VANETs) is the supporting structure of vehicle manage and communication tool/network [11]. Main components of VANETs are on-board-units (OBU), application-units (AUs), and road-side-units (RSUs).

RSUs are normally placed on the roadside and provide a short-range communication with other devices in the same range. OBU is used for exchanging information between OBU and RSUs or other OBU devices on vehicles [11]. It requires communication in a short range by wireless or radio technologies. AU is an application that a driver can monitor or communicate by smart phones or other devices. It will be linked to OBU from vehicles and managed by drivers.

VANETs in current technology can help drivers to check the local weather, traffic jam, or speed information [11]. VANETs will contain traffic regulations and rules for notification. Furthermore, OBUs can connect to each other to assist driver's sensors to alert, if the accident happens and needs to change ways.

VANET is an application that developed based on Ethereum to instead of traditional centralized network [11]. Each vehicle/User will have an Ethereum account and share information in blockchain. The application from a user will enforce to add traffic regulations and required components from government, which are mandatory applications. Users can also add optional applications in using for traffic info, platooning. The information exchanged system will be built and run by smart contracts. When different services exchange information, the transaction occurs. Application will use smart contracts to send information to services; meanwhile, smart contract can be used to pay service fee for optional applications. The VANETs application running on blockchain will increase the accuracy of information; at the same time, the cost of application

will be decreased when all transactions process on blockchain.

Some studies suggest that the blockchain technology has the potential development in the energy section [15]. Specifically, blockchain can be an important role in energy trading, automation of processes and new business cases. There are more and more community energy pilot projects and related researches focus on electricity markets, which shows the potential of blockchain technology for "disintermediating aggregators" in energy exchanging environment.

Blockchain technology has also been applied in a local low voltage energy community [15]. In this concept, the whole energy system will be created, stored, exchanged in an internal environment, and try to avoid the external resource using or exchanging. An experiment that the control system for electrical energy storages is running on a private Ethereum blockchain and implementing by smart contracts has been developed. In the experiment, households in an electricity community are represented by using four computers which can simulate electrical system. Each computer will be one node on blockchain, and there is a controller on the blockchain system to manage electricity landscape of four "householders".
Base on the experiment result for testing blockchain based energy community and controller, it is possible to accomplish a self-sufficiency energy environment; meanwhile, the efficiency of energy using improves, and the cost spending decreases [15]. Each "householder" has fully access to store and manage the energy system. The blockchain based energy community will not only benefit in a better trust environment between participants, but also decrease possibility of the extra third party cost. On the other side, the node in energy community can share resources to external users due to the feature of blockchain. During the experiment, the researcher finds that there are several factors influencing the block time and the transaction timing, which is due to the difference between "residual load" and "battery commands". The future development of the blockchain based energy community still need to be run in experiment process. The community system that can interact with other communities or utility organizations in a public blockchain is the next step. For example, the community can have a third party like power plant service to support the community.

10. BLOCKCHAIN IN THE CLOUD AND EDUCATION

The advantage of Cloud has been recognized when people talk about blockchain. Many of worldwide Cloud technology companies have already attempted to use their existing cloud investing to jostle market share of blockchain, which proves the potential of cloud using in blockchain [12].

The Cloud could be the essential of blockchain system. Blockchain organizations and individual users are willing to pay the service fee to store their data instead of spending same storage and more money on storage[12]. The traditional data storage for block will not only require huge space from personal supply, but also need extra technology to manage the block for security propose. The Cloud can solve the problem by third party which has professional team to manage and monitor storage of blockchain.

Blockchain technology has been used into education emphasis, especially in on-line education. On-line education provider can use blockchain to design a new information system to deliver courses, check course progresses and manage students' information [12]. In this context, the blockchain base information system in Cloud enables end users to search information, connects courses materials and sends massages through users' mobile devices.

11. BLOCKCHAIN IN FRAUD SCHEME DETECTION

As blockchain technology becomes more and more popular, some scams could be idenfied with it, particularly Ponzi scheme. In Ethereum, there are more than 400 Ponzi schemes [16].

Ponzi is the name of a notorious fraudster of almost 100 years ago. A Ponzi scheme is an illegal type of investment to attract some investments. The fraudster uses new investments to "fill/reward" the previous investor. As the result, the investor joins in later will lose more when there is no income from investments without previous investments. Presently, some fraudsters find and try to use blockchain technology to make money. Due to the anonymous identity by using blockchain, the Ponzi scheme will act with "low risk" and higher profits; meanwhile, it is worse for government to monitor and investigate a fraud on blockchain than those happen in another environments. Unfortunately, there are more than 7 million USD worth Bitcoin which has been stolen and gathered by scams between September 2, 2013 and September 9, 2014 [16].

Based on the investigation of Ponzi schemes [16], the Ponzi schemes on blockchain will not be terminated or changed after it has been implemented because smart contracts, which execute automatically, cannot be

reversed; at the same time, the sender of transaction is anonymous. The following message is an example provided by article which is a Ponzi Scheme message sending by smart contract:" Hello! My name is Rubixi! I'm new & verified pyramid smart contract on the Ethereum Blockchain. When you send me 1 ether, I will multiply the amount and send it back to your address when the balance is sufficient. My multiplier factor is dynamic (min. x1.2 max. x3), thus my payouts are accelerated and guaranteed for months to come".

After checking the contract transaction, the researcher finds that there are 112 participants in the investment and only 22 investors receive profit. The fraud causes financial damage for most investors, and fraudsters receive money from whom that is one of investors in 22 positive profit investors. As mentioned above, it is difficult to detect Ponzi scheme recently, and the most reasonable way to detect fraud is manually check source code. The fraud detecting is hard and spends lots of time; furthermore, smart contracts can be hidden which makes detecting more difficult, which fraudster only needs several bytecodes to finish transactions [16]. Base on the information from Ethereum, there are more than million smart contracts working right now; however, only about four thousand smart contracts have source codes, which means some Ponzi scheme are still hidden and creators cannot be found. There are still some questions needed to be concerned: "How many smart Ponzi schemes exist on Ethereum? What types of smart Ponzi schemes are there? What are their characteristics? How much is the influence of smart Ponzi schemes?", and a solution need to be found by detecting smart Ponzi schemes without source codes.

Smart contract is written in a high-level language Solidity which is a language running on EVM in the Ethereum platform [16]. The example of key code of smart Ponzi scheme shown consists two parts, functions and data. The function in smart contract will be called during sending messages or executing transaction by any other users or smart contracts; and, the data included in that contract can be covered and over written. In Figure 11, Lines 2 to 11 is the data definition that can measure the current state of the contract. Line 6 defines current profits (*pyramidMultiplier*) to 300. Line 12 defines all investors (*participants*) in order. Line 14 creates the constructor (*Rubixi*) that runs after the contract. Line 17 is the fallback function (*addPayout*) that runs when sending an Ether without data to a contract. Line 19 is defining the function *addPayout*, which is executed when a participant sends money. Lines 21 to 22 record the address and payment of the investor. Lines 23 to 26 reduce profits if more investors enter. Line 28 calculates the fees. Lines 29 to 34 function pays the benefits to previous investors if the balance is enough. Lines 37 to 41 collect fees by calling collectAllFees [16].

```
1   contract Rubixi {
2       uint private balance = 0;
3       uint private colletedFees = 0;
4       uint private feePercent = 10;
5       uint private Order = 0;
6       uint private pyramidMultiplier = 300;
7       address private creator;
8       struct Participant {
9       address etherAddress;
10      unit payout;
11      }
12      Participant [] private participants;
13
14      function Rubixi () {
15              creator = msg.sender;
16      }
17      function () { addPayout (); }
18
19      function addPayout () private {
20              unit fee = feePercent;
21              participants.push( Participant
22              (msg.sender, (mas.value x
    pyramidMultiplier) / 100));
23              if (participants. length == 10)
24              pyramidMultiplier = 200;
25              else if (participants. length == 25)
26              pyramidMultiplier = 150;
27              balance += (msg.value x (100 - fee)) /
    100;
28              colletedfees += (msg.value x fee) /
    100;
29              while (balance > participants [Order].
    payout) {
30                      unit payoutToSend =
    participants [Order]. payout;
31                      participants [Order].
    etherAddress. send(payoutToSend);
32                      balance -= participants
    [Order]. payout;
33                      Order += 1;
34              }
35      }
36
37      function collectAllFees () onlyowner {
38      if (collecetedFees == 0)        throw;
39      creator.send(collecetedFees);
40      collecetedFees = 0;
41      }
42  }
```

Figure 11: Ponzi scheme code example

There are several reasons and benefits for criminal activity which attract people by making smart contracts [17]. First, the exchange between users can

be finished without third-party intermediaries or physical communication, even between distrustful parties. Second, the interaction of criminal is too "light" to detect. Criminal only needs to deploy a smart contract that the rest of transaction will automatically be done by smart contract's functions. Third, smart contracts can use to call an external state and influence transaction through an oracle service, which will broaden the possible of crimes. Finally, each account of Ethereum only has information of address and balance; therefore, it is difficult to identify the person who owns the account.

12. CONCLUSION

Bitcoin and Ethereum, two popular cryptocurrencies, have abilities to hold and manage rules or scripts of transaction during processing [18]. Cryptocurrencies process it transactions on blockchain that blockchain is a decentralized data structure. And users can create a smart contract to process transactions with other users directly; furthermore, smart contracts can call another smart contract to finish transactions.

Decentralized cryptocurrencies have been noticed as Bitcoin was introduced in 2009 [18]. More and more researches show that the potential of cryptocurrencies and blockchain is not only from its new idea of digital currency, but also from its decentralized characteristics that cryptocurrencies are managed by users in that network to instead of a specialized management group. In the other word, users on blockchain don't have to trust any third parties in transactions because the transaction will basically finish just between sender and buyer. And users will run a "consensus protocol" to monitor and maintain the normal operation of a shared ledger of data which is called blockchain.

There are some more features needed for the future development of blockchain related applications [19]. It is necessary to design a user-friendly IoT hardware for blockchain applications. It is also necessary to design a suitable software for blockchain applications. Communication protocols using currently are not enough for blockchain applications, especially IoT blockchain applications. The blockchain based applications should have a technique to formally verify because hacker can use smart contracts in blockchain which will cause a lot of damages. Based on the previous issue in smart contracts, a role of intermediates might be thought about for blockchain. The intermediate role will be able to decide the payment and other details in transactions. It is necessary to improve the scalability of blockchain based applications.

REFERENCES

1. "The non-financial side of Blockchain" - Varshney, Rashi. Express Computer; Mumbai (Aug 11, 2016).
2. "Why Blockchain offers a fresh approach to interoperability" - D'Arcy Guerin Gue. Health Data Management (Online); New York (Mar 21, 2017).
3. Aligning Requirements with HIPAA in the iTrust System
   A. K. Massey, P. N. Otto and A. I. Antón, "Aligning Requirements with HIPAA in the iTrust System," 2008 16th IEEE International Requirements Engineering Conference, Catalunya, 2008, pp. 335-336. doi: 10.1109/RE.2008.53
4. Exploring the applicability of Blockchain technology to enhance manufacturing supply chains in the composite materials industry
   A. E. C. Mondragon, C. E. C. Mondragon and E. S. Coronado, "Exploring the applicability of Blockchain technology to enhance manufacturing supply chains in the composite materials industry," 2018 IEEE International Conference on Applied System Invention (ICASI), Chiba, 2018, pp. 1300-1303.
5. "Smart Contracts based on Blockchain for Logistics Management", ??
   Néstor Álvarez-Díaz, Jordi Herrera-Joancomartí, and Pino Caballero-Gil. 2017. Smart contracts based on Blockchain for logistics management. In Proceedings of the 1st International Conference on Internet of Things and Machine Learning (IML '17). ACM, New York, NY, USA, Article 73, 8 pages. DOI: https://doi.org/10.1145/3109761.3158384
6. "Investigating Blockchain as a Data Management Tool for IoT Devices in Smart City Initiatives"
   Lingjun Fan, J. Ramon Gil-Garcia, Derek Werthmuller, G Brian Burke, and Xuehai Hong. 2018. Investigating Blockchain as a data management tool for IoT devices in smart city initiatives. In Proceedings of the 19th Annual International Conference on Digital Government Research: Governance in the Data Age (dg.o '18), Anneke Zuiderwijk and Charles C. Hinnant (Eds.). ACM, New York, NY, USA, Article 100, 2 pages. DOI: https://doi.org/10.1145/3209281.3209391
7. "Hierarchical interactions between Ethereum smart contracts across Testnets"
   Yao-Chieh Hu, Ting-Ting Lee, Dimitris Chatzopoulos, and Pan Hui. 2018. Hierarchical interactions between Ethereum smart contracts across Testnets. In Proceedings of the 1st Workshop on Cryptocurrencies and Blockchains for Distributed Systems (CryBlock'18). ACM, New York, NY, USA, 7-12. DOI: https://doi.org/10.1145/3211933.3211935
8. "MedRec: Using Blockchain for Medical Data Access and Permission Management"
   A. Azaria, A. Ekblaw, T. Vieira and A. Lippman, "MedRec: Using Blockchain for Medical Data Access and Permission Management," *2016 2nd International Conference on Open and Big Data (OBD)*, Vienna, 2016, pp. 25-30.doi: 10.1109/OBD.2016.11
9. "Blockchain 2.0: Smart Contracts"

Karan Bharadwaj, Blockchain 2.0: Smart Contracts, http://www.linkdapps.com/Blockchain2.0-SmartContracts.pdf, 2016

10. "Blockchain based Smart Contract for Bidding System" Y. Chen, S. Chen and I. Lin, "Blockchain based smart contract for bidding system," *2018 IEEE International Conference on Applied System Invention (ICASI)*, Chiba, 2018, pp. 208-211. doi: 10.1109/ICASI.2018.8394569

11. "Self-managed and Blockchain-based vehicular ad-hoc networks"
Benjamin Leiding, Parisa Memarmoshrefi, Dieter Hogrefe, "Self-managed and Blockchain-based vehicular ad-hoc networks", UbiComp '16 Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct, Pages 137-140
12. "Perspectives of Blockchain Technology, its Relation to the Cloud and its Potential Role in Computer Science Education"
Purdon, Ian & Erturk, Emre. (2017). Perspectives of Blockchain Technology, its Relation to the Cloud and its Potential Role in Computer Science Education. Engineering, Technology and Applied Science Research. 7. 2340-2344.
13. "SmartCheck: Static Analysis of Ethereum Smart Contracts"
S. Tikhomirov, E. Voskresenskaya, I. Ivanitskiy, R. Takhaviev, E. Marchenko and Y. Alexandrov, "SmartCheck: Static Analysis of Ethereum Smart Contracts," *2018 IEEE/ACM 1st International Workshop on Emerging Trends in Software Engineering for Blockchain (WETSEB)*, Gothenburg, Sweden, 2018, pp. 9-16.
14. "A Blockchain-Based Approach to Health Information Exchange Networks"
Peterson, Kevin, Rammohan Deeduvanu, Pradip Kanjamala, and Kelly Boles. "A Blockchain-Based Approach to Health Information Exchange Networks."
15. "ETHome: Open-source Blockchain based energy community controller"
Jonas Schlund, Lorenz Ammon and Reinhard German. 2018. ETHome: Opensource Blockchain based energy community controller. In e-Energy '18: International Conference on Future Energy Systems, June 12–15, 2018, Karlsruhe, Germany. ACM, New York, NY, USA, 5 pages.
16. Detecting Ponzi Schemes on Ethereum: Towards Healthier Blockchain Technology
Chen, Weili & Zheng, Zibin & Cui, Jiahui & Ngai, Edith & Zheng, Peilin & Zhou, Yuren. (2018). Detecting Ponzi Schemes on Ethereum: Towards Healthier Blockchain Technology. 1409-1418. 10.1145/3178876.3186046.
17. The Ring of Gyges: Investigating the Future of Criminal Smart Contracts
Juels, Ari & Kosba, Ahmed & Shi, Elaine. (2016). The Ring of Gyges: Investigating the Future of Criminal Smart Contracts. 283-295. 10.1145/2976749.2978362.
18. Making Smart Contracts Smarter
Luu, Loi, et al. "Making smart contracts smarter." Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. ACM, 2016.
19. Developing a smart storage container for a Blockchain-based supply chain application
J. Hinckeldeyn and K. Jochen, "(Short Paper) Developing a Smart Storage Container for a Blockchain-Based Supply Chain Application," 2018 Crypto Valley Conference on Blockchain Technology (CVCBT), Zug, Switzerland, 2018, pp. 97-100.
doi: 10.1109/CVCBT.2018.00017