An Approach to Recovery of Critical Data of Smart Cities

Using Blockchain

by

Vineet Mishra

A Thesis Presented in Partial Fulfillment
of the Requirements for the Degree
Master of Science

Approved November 2017 by the
Graduate Supervisory Committee:

Sik-Sang Yau, Chair
Kenneth Goul
Dijiang Huang

ARIZONA STATE UNIVERSITY

December 2017

ABSTRACT

Smart cities are the next wave of rapid expansion of Internet of Things (IoT). A smart city is a designation given to a city that incorporates information and communication technologies (ICT) to enhance the quality and performance of urban services, such as energy, transportation, healthcare, communications, entertainments, education, e-commerce, businesses, city management, and utilities, to reduce resource consumption, wastage and overall costs. The overarching aim of a smart city is to enhance the quality of living for its residents and businesses, through technology. In a large ecosystem, like a smart city, many organizations and companies collaborate with the smart city government to improve the smart city. These entities may need to store and share critical data with each other. A smart city has several thousands of smart devices and sensors deployed across the city. Storing critical data in a secure and scalable manner is an important issue in a smart city. While current cloud-based services, like Splunk and ELK (Elasticsearch-Logstash-Kibana), offer a centralized view and control over the IT operations of these smart devices, it is still prone to insider attacks, data tampering, and rogue administrator problems. In this thesis, we present an approach using blockchain to recovering critical data from unauthorized modifications. We use extensive simulations based on complex adaptive system theory, for evaluation of our approach. Through mathematical proof we proved that the approach always detects an unauthorized modification of critical data.

DEDICATION

This thesis is dedicated to my parents, Mr. Vinodkumar Mishra and Dr. Rekha Mishra, and my sibling, Raksha Mishra. I am thankful for their love, blessings and constant support.

ACKNOWLEDGMENTS

I would like to thank Professor Stephen Sik-Sang Yau for providing me motivation, guidance and the opportunity to work on this research project. He is a constant source of inspiration and guidance to me. I would also like to thank my committee members Professors Michael Goul and Dijiang Huang for being the part of my graduate supervisory committee.

The invaluable inputs I have received for my thesis from Yaozhong Song, Tamalika Mukherjee and other members of Professor Yau's research laboratory, in the Information Assurance Centre, are very much appreciated. I would also like to acknowledge the contribution of my colleague Divyesh Dnyanmothe.

I am also thankful to my family and friends, both in India and US, for their support and encouragement.

TABLE OF CONTENTS

Page

CHAPTER Page

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Motivation

With the advancement of internet of things (IoT), many tedious and onerous day to day activities have become easy and manageable. Application of IoT in automating various tasks at home lead to the coining of the term ‘smart home’. Many appliances can be connected to make it convenient for the user to operate them. Now the world is foreseeing a future where things can be automated on a larger scale to provide convenience and services to a bigger segment of the population. Businesses and academia are exploring the concept of ‘smart city’, where there is a larger level of interaction of sensors, devices, services, organizations, etc. To provide better experience to the citizens and the smart city government. A smart city is a designation given to a city that incorporates information and communication technologies (ICT) to enhance the quality and performance of urban services such as energy, transportation and utilities to reduce resource consumption, wastage and overall costs [1] [2] [3] [4]. The overarching aim of a smart city is to enhance the quality of living for its citizens through technology. Many countries are investing in smart city projects and there are many cities across the world that are considered as a smart city. Countries like USA, China, India, UK, etc. have started developing smart cities for the future while modifying the current cities to make them smarter.

This thesis presents the new perspective to approach some of the problems of the smart city. Very few researches are exploring the disruptive power of blockchain and how it can

impact the functioning of a smart city. An approach is being presented on how blockchain can be used on a smart city services platform to enable businesses and companies to participate in the smart city environment. The focus of this thesis is to address a key security challenge of identifying, locating and recovering from unauthorized tampering of critical data in a smart city using a blockchain approach.

Smart cities can be either open or closed in nature depending on the demographics of the city as well as the political and economic conditions of the nation. European based smart cities are open in nature and they encourage investors and business to invest in the smart city services domain. Smart cities like Singapore and Hong Kong are more inclined towards the smart city government owning most of the assets and they encourage unilateral business dealing with tight constraints in the smart city.

In this thesis, the research underwent the following steps. First, the differences between an open and closed smart city (in terms of data ownership and sharing) were simulated and explored [25]. Second, a thorough literature survey was done to identify various challenges in a smart city. Among the various challenges present, we chose the problem of addressing unauthorized modification of critical data. Various potential solutions were considered and based on the problem context we found the blockchain technology to be relevant [8] [12] [13] [14] [15] [16]. Blockchain technology was carefully studied and various aspects were redesigned to make it suitable for the problem. Lastly, we performed some more simulation experiments to compare the communication and network overhead costs between the currently used centralized approach and our blockchain approach.

1.2 Organization of Thesis

This thesis is divided into 8 chapters. The first chapter is the introduction which gives an overview of the research. The second chapter gives a background on some of the topics like smart cities, blockchain and smart contracts. The third chapter describes the problem that we are trying to solve and how the current approach does not fully address it. The fourth chapter goes in to the methodology adopted for approaching this problem. We define the system model, an experimental simulation of smart city and various notation in this chapter. The fifth and sixth chapters describes the blockchain based solutions and covers the detailed aspects of it. We provide mathematical proof along with the simulations to present the advantages and weaknesses of our approach as compared to the centralized approaches. The last chapter is the conclusion which also gives some directions to the future works.

CHAPTER 2

CURRENT STATE OF ART AND BACKGROUND

Most of the smart city initiatives depend on cloud based services and infrastructure to perform smart city related IT operations and functions. These cloud-based IT services are centralized in nature and many use the traditional client-server model of interaction. The on-demand capabilities of cloud coupled with high availability and scalability makes it the default choice for any kind of smart city initiative [5]. While it offers a lot of advantage, there are some weaknesses to this current approach. For example, problems like insider attacks and rogue administrator [22]; unauthorized tampering can be sometimes difficult to detect and mitigate.

In order to overcome some of the crucial weaknesses of the centralized cloud based services, this thesis take a distributed approach which can be coupled with the cloud based services at a different abstraction layer. The approach uses blockchain technology to counter some of the inherent weaknesses of the centralized approaches and tries to address some of the problems in a highly collaborative and volatile environment like smart city. We approached the problem mathematically and then evaluated the simulations results to understand the dynamic nature of a smart city.

2.1 Smart City

Historically, cities have been the center of growth, economics and culture for any civilizations. Due to the advancement of science and technology, we find every field to be

assisted by the technology to increase its impact. Smart cities is the meeting place for traditional concept of city with technological advancements. The ulterior motive that a smart city wants to accomplish is to make the lives of the citizen better and more comfortable. So, a smart city can be defined as the interplay of technology and organizational establishments to improve the quality of people, community, economics, infrastructure, environment, healthcare and governance [1] [2] [3] [4]. There are several cities across the world that can be considered as smart. Current smart cities have limited features deployed in only small geographical areas for experimentation. Example of some of the smart cities are: New York, Vienna, Rio de Janerio, Vancouver, Hong Kong, Yinchuan, Paris, London, Barcelona, Berlin, Los Angeles, Mumbai, Copenhagen, etc. Each smart city has 3 basic components or factors [3] [4] and they are as follows:

1. Technological factors

   These include the physical infrastructure, various computing technologies like mobile technologies, cloud technologies, virtual technologies, etc.

2. Human factors

   These include human infrastructure and social capital of the citizens living in the smart city.

3. Institutional factors

   These include governance, policy factors, rules, directive and regulations imposed or implemented by the smart city government.

The following are some of the security and privacy challenges [6] [7] of a smart city:

- Privacy leakage in data sensing

- Privacy and availability of data storage and processing
- Data sharing and access control
- Trustworthy and dependable control
- False data injection in sensing and control phase
- Scalability of security solutions for data and devices

2.2 Blockchain

Blockchain is a sequence of blocks which store data and these blocks are linked cryptographically. This data structure is distributed in its nature and function. Bitcoin [8] has been one of the most successful application of the blockchain technology. A block in the blockchain consists of a block header and a block body (Figure 1). Each block header contains the cryptographic hash of its previous block which logically forms a link between blocks.

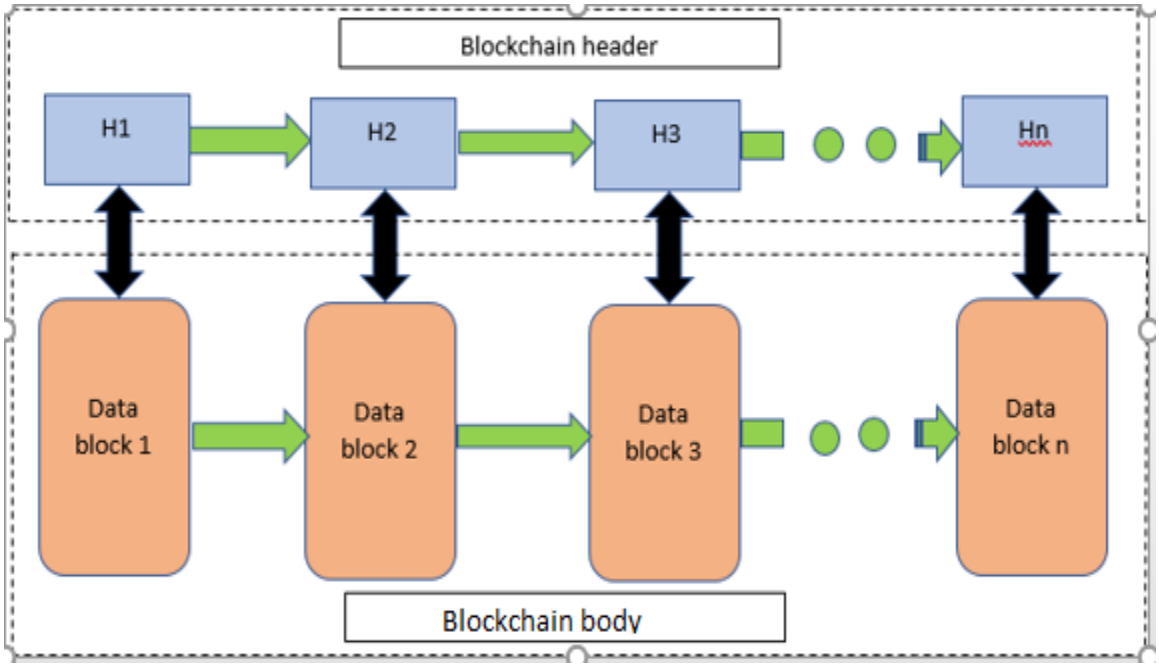

Figure 1 Blockchain Structure

A blockchain header (Figure 2) contains following information [8]:

- Hash: Hash of the entire previous block
- Timestamp (TS): Timestamp of the instance when the block was added to the blockchain
- Root_Hash: Root node of the hash tree of all the data stored in this block.
- Nonce: A random value added to block to make the hash of entire block satisfy the cryptographic challenge condition.

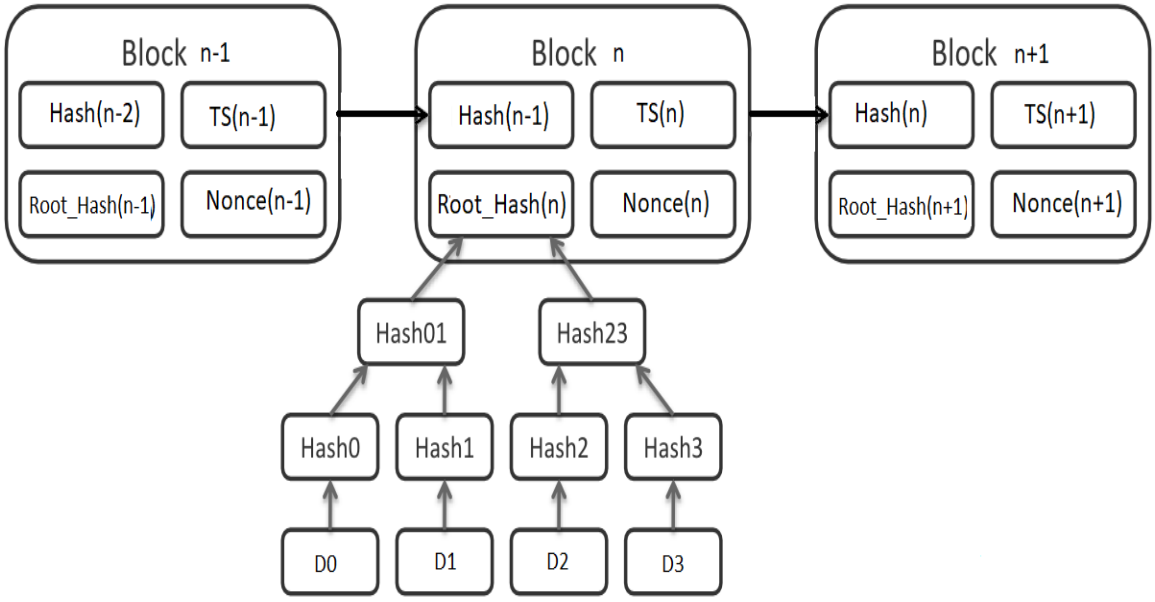

Figure 2 Blockchain Header

In figure 2, let us consider 3 arbitrary consecutive blocks: Block n-1, Block n and Block n+1. The first field in the block header is hash. The hash value of the entire previous block is stored in here. So, Block n-1 will store the hash value of Block n-2, Block n will store the hash value of Block n-1, etc. The second field in the block header is the timestamp

(TS). The time at which the block was added to the blockchain is recorded here. In the figure, Block n-1 has the timestamp TS(n-1), Block n has the timestamp TS(n), and so on. The third field in the blockchain header is the Root_Hash. Every block header has an associated blockchain body which stores the data. In bitcoin application [8], all the transaction information is stored in the blockchain body. In general, any kind of data can be stored here. Let's assume that Block n has 4 data files namely: D0, D1, D2 and D3. These 4 files will be stored in the block body. The hash will be generated for each of the file. So, hash of file D0 is Hash0, D1 is Hash1 and so on. Then we calculate the hash of each of these hashes in pair and we keep on repeating this till we get a single hash value at the root node of the hash tree. This value is stored in the field Root_Hash of the block header. The fourth field is the nonce. Nonce is a random value which is generated to ensure that the hash of the entire block (along with the nonce value) satisfies certain condition. Bitcoin based blockchain requires the nodes, which want to add a block, to perform proof of work. A cryptographic challenge should be solved in order for the node to add a block to the blockchain. An example of a cryptographic challenge: the hash of the entire block should start with seven 0s. Since the data cannot be changed, the nodes change the value of the nonce to find a hash which satisfies the given challenge. Among many nodes which are competing to add a new block, the first node which successfully solves the challenge gets to add the new block to the blockchain.

The cryptographic hash functions that are employed in the blockchain must have the following properties [9]:

1. Pre-image resistance

   Given a value of hash h, it should be difficult to find a message m such that

h= Hash (m).

2. Second pre-image resistance

   Given an input m1 it should be difficult to find another message m2 such that

   Hash (m1) = Hash (m2)

3. Collision resistant

   It should be difficult to find two messages m1 and m2 such that

   Hash (m1) = Hash (m2)

Cryptography is a key unit in the entire blockchain structure. Apart from the hash functions, blockchain employs public key infrastructure in its working. Bitcoin [8] infrastructure assigns a public key private key pair to all its participants. Public key is used as an address to transfer bitcoins and private keys are used as a digital signature for every transaction. This allows for integrity, non-repudiation and confidentiality in the entire bitcoin infrastructure [8].

The data that is added to the blockchain must be verified by the network as a whole. There may exist a divergence of a branch in the blockchain network. There is a need for consensus on a data in a network which might have untrusted nodes. This is an offshoot of the Byzantine Generals Problem (BGP) [10]. In a bitcoin network, a consensus is needed on a transaction when there are several untrusted nodes in the network.

The following are two of the most popular ways to reach a consensus:

- Proof of work (POW)

  This consensus strategy is used commonly in bitcoin network [8]. A node needs to perform computationally expensive operations in order to add a new block to the

blockchain. Such nodes are called miners. A cryptographic challenge is one of the way to ensure that the nodes satisfy this condition. The POW procedure to find the solution of the cryptographic challenge is the process of mining. Each node must find a nonce value which will satisfy the cryptographic challenge. The first node to solve this can add its block to the blockchain. Bitcoin network offers an incentive (a small value of bitcoin) to the node which successfully adds a block. Since there can be several possible block options in the blockchain to build upon, the network considers the longest blockchain as the correct and the most updated one.

- Proof of stake (POS)

  This was proposed as an alternative to POW. This energy efficient method uses ownership to reach consensus. Unlike POW, which asks people to find a nonce for a cryptographic challenge, POS [11] requires people to prove their ownership of currency. The rationale behind this approach is that people with more currency are less likely to attack the network. This approach is unfair as the richest person can have a dominant say in validation process.

One of the key limitation of the blockchain method is that if at least 51% of the network is compromised or if 51% of the nodes pool their resources, they can gain control of the entire network. The incentive provided in POW helps to counter this as mining operation is expensive and pooling the resources can lead to high operations costs for a low value return. The blockchain has following key properties [12] [13] [14]:

1. Decentralization

   Unlike conventional centralized systems, where every validation is done by a trusted party, blockchain offers a peer to peer setup where any node can perform validation without the need of a central trusted party. This helps in avoiding bottlenecks, single point of failure and reduces the server cost which are required in centralized systems.

2. Networked integrity and security

   Security and integrity is encoded in every step of the way using cryptography. The system offers confidentiality, authenticity and non-repudiation to all the activities depending on how cryptographic concepts are used.

3. High availability

   Since the data is replicated and stored in several nodes, the data is highly available and accessible to all nodes in the network.

4. Enforced rights

   Ownership rights are transparent and enforceable. Smart contracts offer a way to ensure this. Due to the public key infrastructure, nodes can maintain anonymity and digital signatures and asymmetric cryptography can help implement access rights.

5. Auditability

   Since all data stored on a blockchain are validated and recorded with a timestamp along with cryptographic hashes, the data can be easily monitored and verified with its previous records across the distributed network.

2.3 Smart Contracts

A Smart Contract is "a computerized transaction protocol that executes the terms of a contract."[14]. Smart contracts are automated scripts which ensure that the conditions put in the contract are always met. Smart contracts are deployed on blockchain and are used by companies and organizations to conduct business and transactions over the distributed blockchain network without the need for an intermediary. Smart contracts can include business contracts in computer program format as well as algorithms and policies translated into a piece of code for enforcing certain conditions on a blockchain. Smart contracts exist in the blockchain and they are implemented autonomously and routinely in a prearranged manner at every node of the network [15] [16]. In our approach, we employ the use of smart contracts for the following reasons:

- To facilitate businesses and organizations to conduct their business relations with each other as well as the smart city government in a secure and transparent manner.
- To enforce the rules and regulations imposed by the smart city government on various participants of the ecosystem.
- To run various protocols and algorithms to ensure smooth functioning of the blockchain infrastructure.

Ethereum [17] is the most commonly used platform for smart contracts, however there are other platforms which are coming up. Ethereum [17] offers a Turing complete language which is the reason why it is so popular among researchers and businesses.

CHAPTER 3

PROBLEM STATEMENT

Smart cities are increasingly using the cloud based services for various smart city related applications. Cloud based IT services provide a centralized view of the entire ecosystem's IT operations and offers a single go-to platform for all the IT operations, activity and information in the smart city. Sensors that are deployed throughout the smart city are connected to a single point cloud application for monitoring, auditing, maintenance, regulatory, etc. reasons [18] [19] [20] [21]. In a smart city, there are many thousands of sensors deployed across the city which are collecting information from its surroundings. These sensors can collect data like temperature, moisture, wind speed, air quality, water quality, congestion on roads, waiting time in traffic, garbage bin levels, videos of citizens, GPS location, etc. A smart city also need to collect and store critical data for its functioning [5] [6] [18] [19] [20]. In an open environment as simulated in case 2 (see Evaluation chapter), there will be a degree of data sharing involved for all kinds of data. If a critical data is shared with several participants by the smart city, then there must be proper mechanisms in place to identify and recover from tampering of this critical data. Past statistics and research [18] [22] show that insider threat can be a serious problem especially if the insider has privileges. This rogue administrator problem [22] can lead to loss of confidentiality and integrity of critical data. Cloud based solutions like Splunk [23] and ELK (Elasticsearch-Logstash-Kibana) [24] provide a centralized view of all kinds of machine generated data. These tools then provide an interface to different kinds of end users to perform different kinds of roles. Privileged users like admin can access most of

the critical data since all the data is stored in a centralized repository. This is a serious problem that we want to address in this thesis. The problem that we want to address in this thesis is how can we identify, locate and recover from any unauthorized modifications of archived critical data of a smart city in a timely manner.

Solving this problem will be a key component in tackling some of the security problem present in the smart city domain. Cities of the future will have a very high degree of inter-connectivity and data sharing [2] [3] [4] [18]. Unauthorized tampering of data for personal, social, financial, political and economic reasons will be a big problem faced by the smart city. We try to address this problem with a narrowed scope using blockchain technology in this thesis.

CHAPTER 4

SMART CITY SYSTEM MODEL

We will be using the system model defined in [25] for our approach. We consider the following entities and capabilities in our smart city service initiative:

- Hosts (H): Any business entity or organization that wants to deploy its analytical models on smart devices and do business.

  A set of hosts (company and business participants)

  $H = \{H_1, H_2, \ldots H_p\}$ or

  $H = \{H_i\}$ (1)

  for i = 0 to p

- Smart Object Host (SOH): Smart objects are associated with a smart object owner or host which provides the smart objects as service and allows provision for pushing predictive models and algorithms to the SOs.

  A set of smart object hosts (sensor cluster owners)

  $SOH = \{SOH_1, SOH_2, \ldots SOH_q\}$ or

  $SOH = \{SOH_i\}$ (2)

  for i = 0 to q

- Smart object (SO): Any end point devices, with limited storage and computing capabilities, that can operate to some extent interactively and autonomously. Example: smart phones, security camera, intelligent kiosk, amazon echo, etc.

  The set of smart objects owned by smart object hosts (actual smart objects with sensors belonging to a sensor cluster owner) (from 2)

$$SO_{SOHi} = \{SO_{SOHi,1}, SO_{SOHi,2}, \ldots SO_{SOHi,m}\}$$

$$SO_{SOHi} = \{SO_{SOHi,j}\} \quad (3)$$

for i=0 to q

for j=0 to m

- A Sense is an input or output of a sensor.

  A set for some smart object, SO_i, has senses S. (from (3))

  $$S = \{S_{SOi,1}, S_{SOi,2}, \ldots S_{SOi,n}\} \quad \text{or}$$

  $$S = \{S_{SOi,j}\} \quad (4)$$

  for i = 0 to r

  for j = 0 to n

- K is a subset of smart object (owned by SOH i) and its corresponding senses

  $$K \subseteq \{SO_{SOHi}, S_j\} \quad (5)$$

  for i=0 to r

  for j=0 to t

- Analytical/predictive model (AM/PM): A predictive model is a function of a subset of the senses that smart object is capable of, and the output of the model is also a sense that smart object is capable of. Any algorithm working on a smart device can be considered as a predictive model.

  A set of predictive models (PM) deployed by some smart object host, SOH_j at the direction of a host H_i or a smart object host, SOH_j, itself to some smart object, $SO_{SOHj,q}$:

Using (4) and (5) we get the following function,

$$PM = [F: K \rightarrow K] \quad (f1)$$

Contract provisions are required to address ownership and other characteristics that can possibly influence the design and operation of a smart city service platform. We used the various contract provisions identified in [25] for data ownership. [25] categorizes the contract provision for data ownership in the following way:

- Data exclusivity vs. Non-exclusivity: This contract provision states whether data is exclusively owned or co-owned. For example, a contract may specify that an soh owns all the data generated by the SOs connected to it. Data is considered non-exclusive when it is co-owned by several entities or when it is open for access.

- Co-mingling vs. No co-mingling: This contract provision states whether the predictive model derived from the co-owned data is exlusively owned or co-cowned. If there is no comingling, the entity that creates the model owns the model exclusively.

- Data post-use vs. No data post-use: In a data post-use contract provision, if a data owner leaves the ecosystem or partnership governed by this contract, then the co-owned data becomes the assets of the remaining owner(s) (if any). If there is no

data post-use contract clause, then the data owned by the exiting entity is deleted from the ecosystem.

CHAPTER 5

OVERALL APPROACH

In our approach, we synergize the smart cities' infrastructure with blockchain to address the problem of unauthorized modification in a smart city. The blockchain which will be deployed in the smart city ecosystem will be a private and permissioned blockchain. That means if a company or an organization wants to enter this ecosystem, they will have to get the smart city government's permission. The blockchain will not be accessible to everyone.

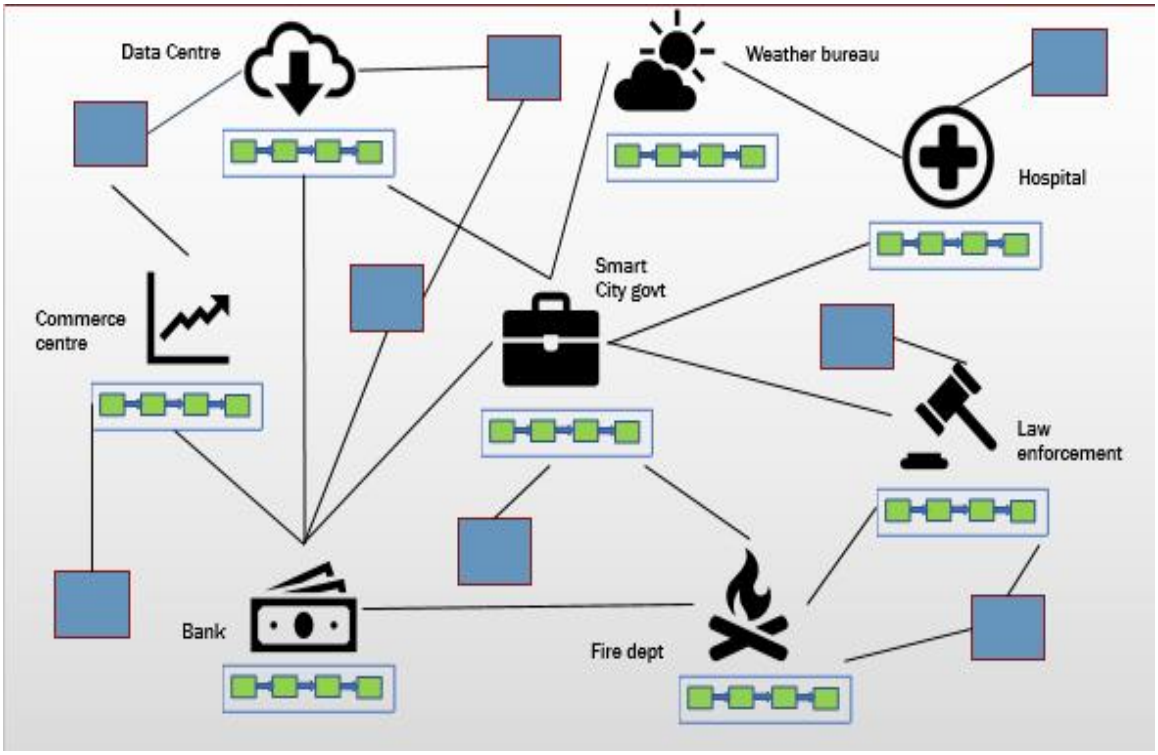

Figure 3 Smart City Blockchain with Major and Minor Nodes

Only the allowed organizations and citizen groups can access the blockchain. In this ecosystem, any node (whether it is belonging to an organization, government or citizen

groups) can be classified as one of the following two types on the basis of computation power and storage capabilities:

1. Major node

   These nodes have enough resources (computation power and storage) to use and maintain blockchain infrastructure. These nodes will be responsible for participating actively in the blockchain infrastructure. All the black nodes in figure 3 are major nodes. Major nodes will perform some crucial function like:

   1.1. Accessing and connecting to the blockchain infrastructure

   1.2. Replicating and downloading the content of the blockchain

   1.3. Listening for incoming request

   1.4. Validating the incoming requests

   1.5. Forwarding or passing on a valid request

   1.6. Listening for new blocks

   1.7. Validating new blocks

   1.8. Forwarding or passing on valid blocks

   1.9. Creation of new blocks

   1.10. Mining the blocks

   Each of the above-mentioned functions are crucial for correct functioning of the blockchain infrastructure.

2. Minor nodes

   These nodes do not have enough resources to maintain and manage the blockchain infrastructure. These nodes usually forward the request to a major node for

processing. The blue blocks in figure 3 are the minor nodes. A minor node will have the following functions in the blockchain infrastructure:

2.1. Accessing and connecting to the blockchain infrastructure

2.2. Listening to incoming request

2.3. Forwarding or passing on a request

2.4. Forwarding or passing on valid blocks

In our approach, we replicate and store critical data at all the major nodes. This helps in recovery from unauthorized modifications as well as provides high availability of critical data across the ecosystem. This critical data will be encrypted and will only be accessible to the authorized groups or individuals.

5.1 Steps to Adding Critical Data to the Blockchain Infrastructure

An important part of our approach is the addition of data in a secure way. Any node (major or minor) can add critical data to the blockchain only if they have the permission to do so, from the smart city government.

The following are the steps to add critical data to the blockchain for a major node:

1. Broadcast the data to be added to the blockchain network.
2. The requested data is validated by the other major nodes of the network using smart contracts and other functioning algorithms (like authentication services) working on the blockchain ecosystem.

3. If the majority of the major nodes validate the data then the data is added to a new block, along with the other validated data, by all the major nodes. If the data is not validated by the majority, then it is discarded and an entry is made into the logs.
4. Once the block size is exhausted with validated data, the major nodes solve a cryptographic challenge.
5. The first major node to solve it will broadcast the solved block (with corresponding timestamp and nonce) as the latest block to the blockchain network.
6. Other major nodes will add this latest block to their blockchain.
7. The nodes which requested to add the data will receive an acknowledgement if their data was successfully added to the block. If their data was invalidated, they will receive a message stating the same. If they don't receive an acknowledgement within the timeout span, they will re-broadcast the data to the blockchain network.

The following are the steps to add critical data to the blockchain for a minor node:

1. The minor node will send the request to add the data to the nearest major node (nearest can be in terms of demographics or network latency)
2. The major node will broadcast the data to be added to the blockchain network.
3. The requested data is validated by the other major nodes of the network using smart contracts and other functioning algorithms (like authentication services) working on the blockchain ecosystem.
4. If the majority of the major nodes validate the data then the data is added to a new block, along with the other validated data, by all the major nodes. If the data is not validated by the majority, then it is discarded and an entry is made into the logs.

5. Once the block size is exhausted with validated data, the major nodes solve a cryptographic challenge.
6. The first major node to solve it will broadcast the solved block (with corresponding timestamp and nonce) as the latest block to the blockchain network.
7. Other major nodes will add this latest block to their blockchain.

The nodes which requested to add the data will receive an acknowledgement if its data was successfully added to the block. If its data was invalidated, they will receive a message stating the same. If they don't receive an acknowledgement within the timeout span, they will resend the data to the nearest major node.

5.2 Steps to Recovering from Unauthorized Modifications of Critical Data

There are three major steps in addressing unauthorized modification of critical data in smart city. The procedure must be executed every time before a new block is added to the blockchain. Figure 4 illustrates the three steps involved in recovering from unauthorized modification.

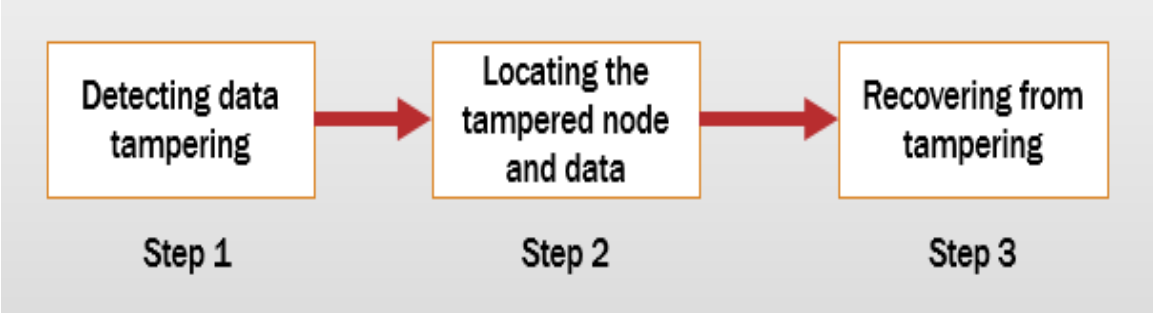

Figure 4 Steps to Recover from Unauthorized Modifications

The steps are as follows:

1. Detection of unauthorized modification in the ecosystem
    - Any change in the data section of the old blocks in the blockchain leads to change in the hash of the data and as a result of that, the hash of the block and the root_hash also changes. The consecutive blocks get invalidated due to change in these hash values.
    - The hash change is identified by the algorithm running on the blockchain infrastructure. This step identifies a list of suspicious nodes where the modification might have happened.
2. Identification of the node where the unauthorized modification occurred
    - From the suspicious nodes identified in step 1, we run an algorithm on each of these suspicious nodes to verify if it was tampered.
    - The node at which the unauthorized modification took place along with the critical data that was modified is identified by our algorithm (section 6.2) by comparing the cryptographic hashes on a suspicious node with a normal node.
    - This step verifies the tampering of the node and identified the block(s) where the unauthorized modifications occurred.
3. Recovery from unauthorized modification
    - Once the block is identified, we overwrite the critical data of that block with the corresponding block of an untampered node.

- The recovery step is complete when the hash values of the overwritten block(s) matches the hash value of the corresponding block(s) of an untampered node.

Section 6 of this thesis details out each of these steps along with a scenario. Since, bitcoin is the most successful application of the blockchain that use pow, we will be drawing analogy from the bitcoin application. A generic CPU uses 6.67KW/GH (i.e. Consumes 6.67 kilo watts per gigahash operation) [26] [27]. A generic GPU (graphic processing unit) uses 0.5KW/GH which is significantly less than mining from a CPU. Bitcoin industry is moving towards ASIC [28] for mining. ASIC stands for application specific integrated cicuits. These are specially designed equipment for performing a singular task. A generic blockchain ASIC [27] uses only 1W/GH which is exponentially less than both CPU and GPU. However, asic costs a lot to manufacture. Antminer S2 is an ASIC which works with an efficiency of 1.1W/GH and has a capacity of 1000GH/sec. This piece of equipment costs around $2259 [27] which is significantly large amount as compared to a CPU or a GPU. Cost of electricity in Arizona for 1 KWH is around 11 cents [29]. This means that the cost of operation will be significantly less but the initial investment of sophisticated hardware will be very large. The complexity of the cryptographic challenge determines the time interval between two blocks as well as the computation power required to solve a cryptographic challenge

CHAPTER 6

DETECTION OF UNAUTHORIZED MODIFICATION IN THE ECOSYSTEM AND IDENTIFICATION OF THE MODIFIED NODE

The following are the notations used in the algorithm:

- M -> Total number of major nodes in the blockchain network
- N -> Total number of blocks in the blockchain
- $B_{i,j}$ -> Block i of Major node j
- T -> List with all the major nodes in the blockchain network, initially contains all the major nodes of the blockchain network.
- t -> Arbitrary major node from T
- t_i -> the i^{th} block of the major node t
- SN -> List of suspicious nodes, initially an empty set
- $SN_{j,i}$ represents i^{th} block of node j in SN
- SN_j means the j^{th} major node in the SN list
- q -> total number of nodes in SN

The following is the pseudo code for the approach which finds unauthorized modifications on a blockchain.

1. For j= 0 to M do, //Check for all the major nodes
2. For i=0 to N do, //Check all the blocks on that major node
3. If i=0 then //Condition for first block
4. continue

5. Else

6. If Hash ($B_{i-1,j}$) ≠ Prev_Hash of $B_{i,j}$ //Check for hash modification

7. Alert()

8. T = T – { major node j}

9. SN = SN U {major node j} //Mark the node as suspicious

10. Else

11. Continue

12. For j=0 to q, //Check all the nodes in SN list

13. For i=0 to N do, //Check all blocks of a specific node

14. If Root_Hash of $SN_{i,j}$ ≠ Root_Hash of t_i

15. While (Root_Hash of $SN_{j,i}$ ≠ Root_Hash of t_i)

16. Overwrite the data in $SN_{j,i}$ with the data in t_i

17. Else

18. Continue

19. SN=SN- {SN_j}

The main condition that we are checking for is: Hash(B_{i-1}) ≠ Prev_Hash of B_i

We synergize the property of blockchain and cryptography hash to help detect unauthorized modification of data. The change in data leads to change in hash. This leads to change in hash of the entire block.

The time complexity of this step is O(m.n) where m is number of major nodes and n is number of blocks in blockchain

6.1 Detection of Unauthorized Modification in the Ecosystem

This is the first step in the procedure for recovering from unauthorized modifications. This step must be executed every time before a new block is added to the blockchain. The details of the steps are as follows:

Step 1: Detection of unauthorized modification in the ecosystem

Input: All the major blocks of the ecosystem.

1.1 Before addition of a new block to the blockchain, run an automated script on every major node

1.2 For i = 0 to n, where B_0 is the first block in the blockchain, B_1 is the second block in the blockchain … B_i is the $(i+1)^{th}$ block in the blockchain … and B_n is the latest/last block in the blockchain, check every consecutive block for the following condition.

If Hash of Block i-1 ≠ Hash(i-1) field of block header Block i,

Alert and add node i-1 to list of suspicious nodes (SN). [Blockchain property 2 in section 2.2]

Else, return null

Output of step 1 is a list of suspicious nodes where the tampering might have occurred.

The time complexity of this step is O(n) where the n is the blockchain length parameter.

6.2 Identification of the Modified Node and its Block

This is the second step in the procedure for recovering from unauthorized modifications. With the list of suspicious nodes identified in step 1, we locate the specific block(s) where the unauthorized modification too place and verify it. The details of step 2 are as follows:

Step 2: Identification of the node and its block where the unauthorized modification occurred

Input: We take the output of step 1, i.e. List of suspicious nodes (SN) SN = $\{SN_j \mid j=[0,m]\}$, along with a normal node T (T is any major node which is not present in SN) as the input to step 2

2.1. For every node SN_j in SN, generate the hash tree for data of each block $(B_i \mid i=[0,n])$ in the block chain stored at the node SN_j .

2.1.1 Compare the root node of this generated hash tree with the Root_Hash of corresponding B_i of a normal node T

2.1.2 If they are not equal, then pass the block number i to step 3 and go to step 3, else continue

2.2. Remove the node SN_j from SN.

Output: we pass the tampered block numbers of suspicious nodes SN_j from (step 2.1.2.) to (step 3). At the end of step 2.2. We get a recovered blockchain at a tampered node.

After step 2, we proceed to step 3 where the recovery takes place. We use an untampered node in the ecosystem and use its block to overwrite the modified block. The specifics of the step 3 are as follows:

Step 3: Recovering from unauthorized modification

Input: Block number i of SN_j from step 2.1.2

3.1.Overwrite the identified block i of suspicious node SN_j with the corresponding block (B_i) of any node T.

3.2. Calculate the hash of the over written block i and compare it with hash of the B_i of a normal node T. (using blockchain property 2 of section 2.2 and cryptographic hash property 2 of section 2.2)

3.3. If the hashes are same then the recovery of the block is complete and continue to step 2.1., else go to step 3.1.

Output: Recovered block of critical data on SN_j

6.3 Mathematical Proof

Let us assume that any three arbitrary consecutive blocks B_{i-1}, B_i and B_{i+1} are the original untampered blocks in a blockchain.

According to the blockchain property 2 (mentioned in section 2.2),

Hash of block B_{i-1} = Hash(B_{i-1}) field of B_i (1)

And

Hash of block B_i = Hash(B_i) field of B_{i+1} (2)

Let us say that a malicious actor modified B_i to Q by changing some critical data.

Data of $B_i \neq$ Data of Q (3)

Let us also assume that the block Q which was tampered was not detected by the above procedure.

If it was not detected by the above procedure that means the condition in step 1.2 was never true. This implies

Hash of block B_{i-1} = Hash(B_{i-1}) field of Q (4)

And

Hash of block Q = Hash(B_i) field of B_{i+1} (5)

From (1), (2), (4) and (5) we get,

Hash of block B_i = Hash of block Q (6)

By the cryptographic hash property 3 (mentioned in section 2.2), we know that it should be difficult to find two messages m1 and m2 such that their hash is same. Current hashing algorithms like SHA 256 and SHA 512 have a very negligible probability of finding two inputs which will lead to a collision. This implies that,

Data of B_i = Data of Q (contradiction with (3))

This proves that our initial assumption was wrong. The algorithm always finds unauthorized modifications in our blockchain based ecosystem.

After completing the above procedure, we are certain that all the unauthorized modifications are detected and recovered.

6.4 An Illustrative Example

Let us look at an example for unauthorized modifications and walk through our approach to fix it. Figure 5 shows a snapshot of the original unaltered blockchain and Figure 6 shows a snapshot of modified blockchain with modified critical data marked in red. Let us walk through the example and see how the approach recovers from unauthorized modifications.

Let us assume that there was a modification at a major node of the smart city. The data D1 of block n was modified to D11 on this major node (Figure 6).

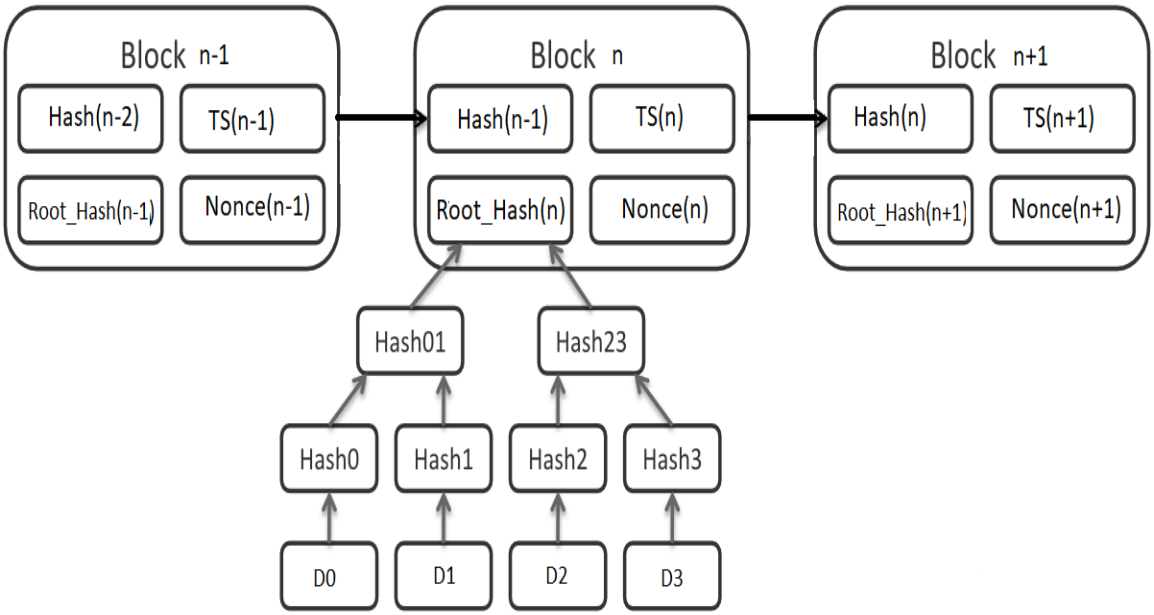

Figure 5 Unaltered Blockchain

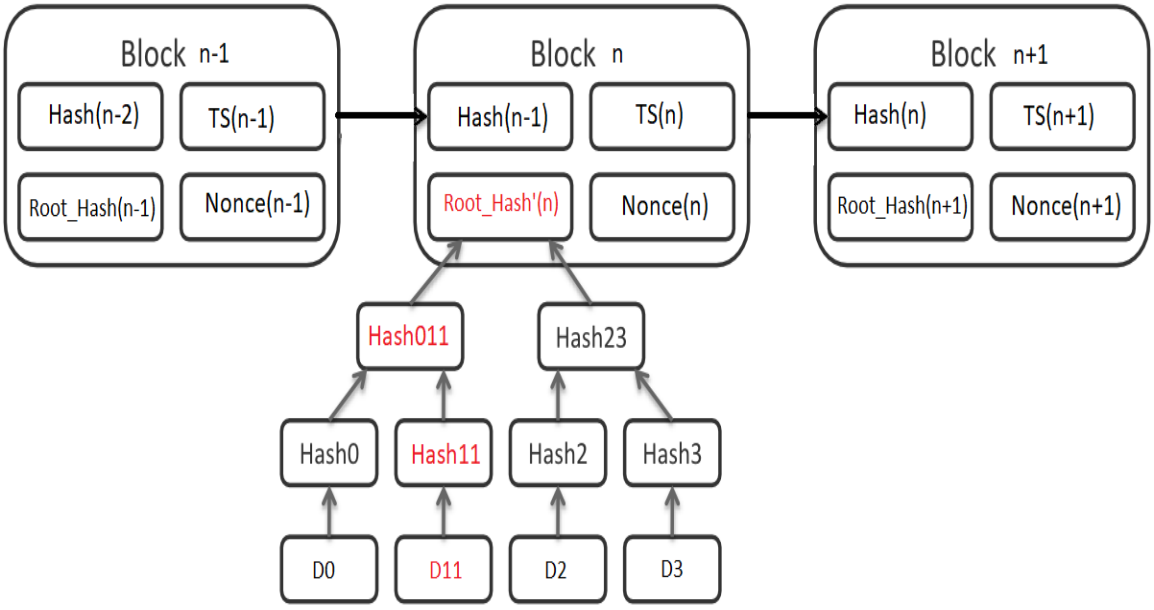

Figure 6 Blockchain after Modifications

Input: All the major nodes of the smart city along with their blockchain.

Step 1: Detection of unauthorized modification in the ecosystem

The goal of this step is to detect if there were any unauthorized modifications in the blockchain of the major nodes. We take all the major nodes as the input to this step.

1.1. We run an automated script before addition of every new block in the ecosystem. This script will be executed on all the major nodes of the smart city.

1.2. At all the major nodes, the script will check for the condition:

$Hash(B_i) \neq$ Prev_Hash of B_{i+1}

As explained earlier in this chapter, this condition will be check the blockchain header for the above condition. From figure 6, we can see that due to the change in D1 to D11 (marked in red), there was a subsequent change in the parent hash node of the merkle tree. This change in leaf nodes led to new intermediate hash value ,'Hash11' (marked in red). This change is further propagated up the merkle tree and the parent node of Hash11 is modified from Hash01 to Hash011 (marked in red). Ultimately, due to this change, the root node of the merkle tree ,i.e. root hash, is changed from Root_Hash(n) to Root_Hash'(n) (marked in red). Changing of this field in the header of the blockchain will lead to change in the hash of the entire block. We will be using this property to identify any unauthorized modifications. Since the hash of the entire block n is now changed (due to change in Root_Hash value), the hash which was stored in the next

block, i.e. block n+1, will no longer match with this modified value. This is check by the condition mentioned above.

Any change in data in block body leads to change in the root_hash field of its block header as shown in figure 7. We can see that now the hash of block n will not be equal to the hash(n) of block n+1 due to the cryptographic hash property 2 (section 2.2). Due to this change, step 1.2 will mark this node as suspicious and raise an alert.

If this condition is true, i.e. the value of Hash(B_{i-1}) is not equal to the Prev_Hash field of B_i , then we know that there was some modification in the ecosystem. We raise an alert, which will be alarm all the other major nodes in the ecosystem (including the smart city government), and we add this major node to the list of suspicious node (SN). This script will run on all the major nodes and the output of this step is a list of suspicious nodes where the unauthorized modification took place.

Output: The list of suspicious nodes SN.

Step 2: Identification of the node and its block where the unauthorized modification took place.

The goal of this step is to pinpoint the block where the unauthorized modification occurred. The input of this step is the SN, list of suspicious nodes.

Input: List of suspicious nodes SN ={SN_j | j=[0,m]} from Step 1. This will have the node where the unauthorized modification occurred in our example.

2.1. Let us assume that there is only one node in SN, which is the node where the unauthorized modification occurred in our example. So in this step we will generate the hash tree (merkle tree) from scratch for each of the blocks on this major node. So we generate the hash tree and calculate the Root_Hash value for all the blocks B_i on this major node.

2.1.1. We take an arbitrary node t in the ecosystem which is not present in SN. For each of the newly generated Root_Hash values on the suspicious node SN, we compare the value of Root_Hash of the block B_i of SN with the value of Root_Hash of the block B_i of t. The aim of doing this comparison is to locate the block where the unauthorized modification took place.

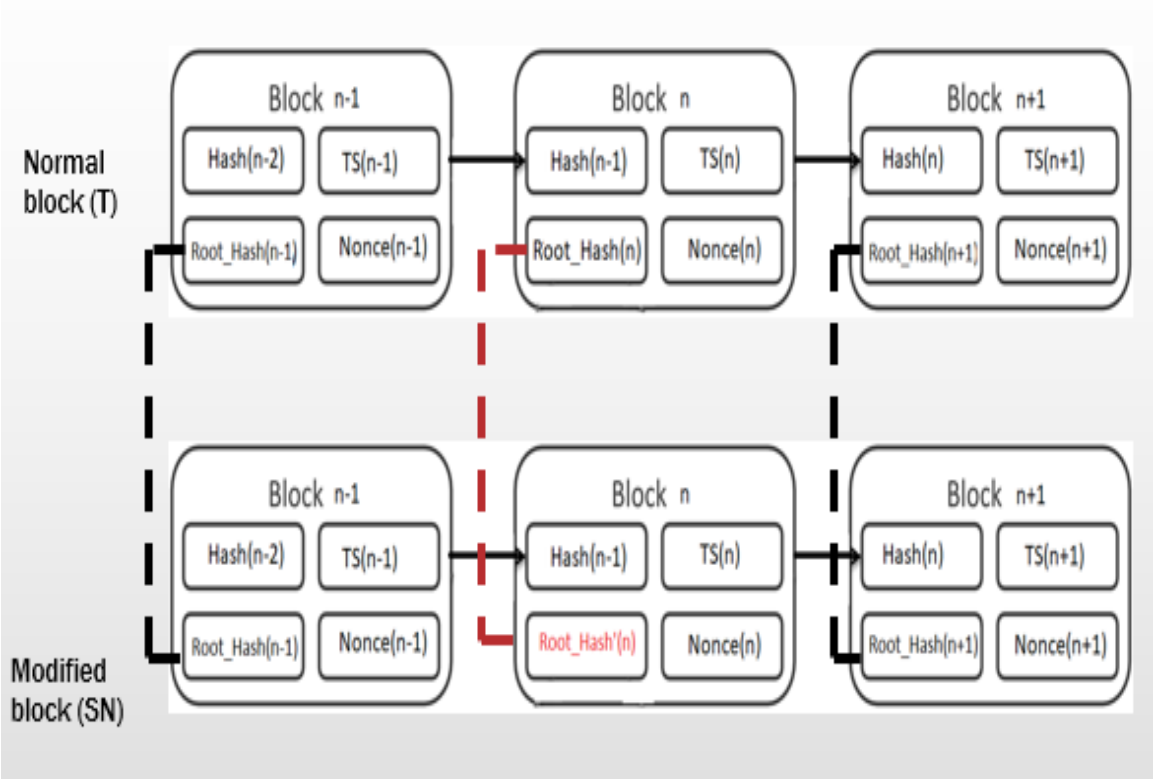

Figure 7 Checking Root_Hash on SN Node

According to the second pre-image resistance property of cryptographic hash function, the probability of finding a different message which lead to a specific hash value should be negligible. Since the hash functions which are used in this method follow the properties of the cryptographic hash functions, we can say that we will locate the unauthorized modification.

2.1.2. In our example, the Root_Hash value of block B_i of SN and block B_i of t won‘t match (figure 7). This means that the unauthorized modification occurred at this block B_i. We flag this block number i of SN and pass it to the next step, step 3.

Output: Block number ‘i’ of suspicious node SN where the unauthorized modification occurred.

Step 3: Recovery from unauthorized modification

The goal of this step is the recovery of the critical data at the node where unauthorized modification occurred. The input of this step is the block number ‘i’ of SN where the unauthorized modification occurred.

Input: Block number I of SN from Step 2.1.2.

3.1. We take the block B_i of t (all the content stored in blockchain based database of block number ‘i’ of t) and overwrite it bit by bit to the blockchain based database of block number ‘I’ of SN. Due to the distributed nature of blockchain, we know that the critical data will be highly available so we use the data of an untampered node and overwrite it to data present on SN.

3.2. We generate the hash tree again on the block B_i of SN after we complete the overwrite process. After we generate the hash tree, we compare the Root_Hash value of the newly generated hash tree and the Root_Hash value present in the block header of block B_i of t.

3.3. We compare these two values mentioned in step 3.2. If the values are equal, then we know that all data was replaced successfully and accurately at SN. This means that the critical data was recovered from unauthorized modification at the major node SN. If the hash values are equal then the recover was successfully completed and we remove that major node from thelist of suspiciois node SN. If the hash values are not equal then we go back to step 3.1 and follow the flow again till we get the recovered block on that major node. In our example, the recover is complete after we overwrite the data correctly from t to SN as shown in figure 8.

Output: The recovered major node.

So we completed the recovery of the critical data D1 after there was an unauthorized modification on a major node.

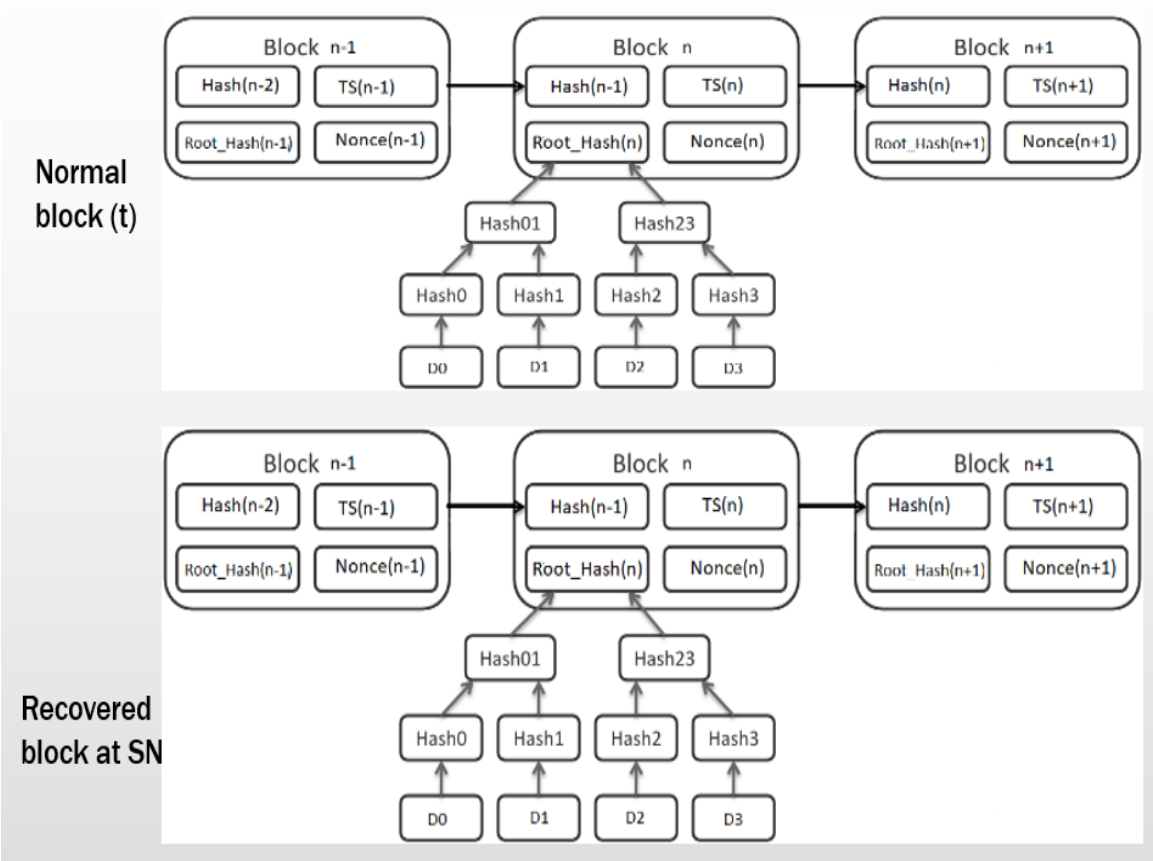

Figure 8 Recovered Block at SN

CHAPTER 7

EVALUATION

This chapter shows the details of the simulation along with the results obtained. We used the metrics, communication cost and coordination cost, described in [25] for evaluation and analysis. Before each run, we set up the different parameters and defined the rules of the environment (data sharing and contractual agreements) and subjected the environment to randomized scripts for several runs. A script is an action/event that occurs in the smart city environment

7.1 Overview of Scripts

Using the definitions and concepts in [25], we simulated the smart city environment in the context of contractual agreements, data sharing and execution of predictive models. We used the parameters like topology (total number of SOHs, Hs and SOs), maximum number of allowed PMs to be added or deleted (to control the volatility of the environment), maximum number of allowed nodes (SOHs, Hs and SOs) to be added or removed (to control the volatility of the environment), number of scripts executed in one simulation run, etc, mentioned in [25] for our simulation of the smart city. We considered the following scripts in our simulation:

- Deploying a predictive model

  This script emulated the activity of pushing a PM from a host to the SOs.

- Deleting a PM

  This script emulated the activity of deleting a PM from the ecosystem.

- Updating a PM

  This script emulated the activity of updating a PM from a host to the SOs.

- SO goes down

  This script emulated the incident when an SO is down/not working.

- SOH goes down

  This script emulated the incident when an SOH is down/not working.

- H goes down

  This script emulated the incident when an H is down/not working.

- Adding a contract

  This script emulated the activity when a new contract is made between two or more entities in the ecosystem under the specified contract provisions.

- Deleting a contract

  This script emulated the incident when a contract is cancelled or made void by the entities involved

- Adding a new H

  This script emulated the incident when a new H is added to the ecosystem.

- Adding a new SOH

  This script emulated the incident when a new SOH is added to the ecosystem.

- Adding a new SO

  This script emulated the incident when a new SO is added to the ecosystem.

We ran the simulation under different script runs of 20,50 and 100 we simulated the smart city environment.

7.2 Simulation for Blockchain Approach vs the Centralized Approach

Along the lines of complex adaptive system theory and simulations used in [25], we simulated a smart city environment again which simulates the behavior of the smart city and various organization which participate in its ecosystem. We ran the simulation for two cases: Case 1 depicts a conventional centralized approach to smart city and Case 2 depicts our approach. The aim of the simulation was to observe the communication and coordination costs of these two cases for a very dynamic and volatile ecosystem where participants are constantly entering and exiting the system. The simulation experiment was done just to observe the communication and network overhead in the two approaches for a volatile environment. The simulation doesn't consider factors like startup investment cost, power consumption, specific operations needed to run the infrastructure (like mining and replication), etc. The underlying assumption for Case 1 is that the SOH interacts with the cloud for updates and changes in the smart contracts. Whereas for Case 2 the SOH will interact with the blockchain infrastructure present at the edge computing layer (closer to SOH in comparison to the cloud) for updates and changes in contracts. In case 1, the SOH contacted the central server after frequent periodic intervals to check for any updates. In Case 2, SOH contacted the blockchain just before addition of a block as well as an infrequent periodic interval (due to the nature of blockchains). After several hundred of simulation runs, we plotted the graph of communication cost vs coordination cost for 3 sets of scripts each for the two cases. Figure 9 shows the graph for Case 1 with script=20, figure 10 shows the graph for case 1 with script=50 and figure 11 shows the graph for Case 1 with script=100.

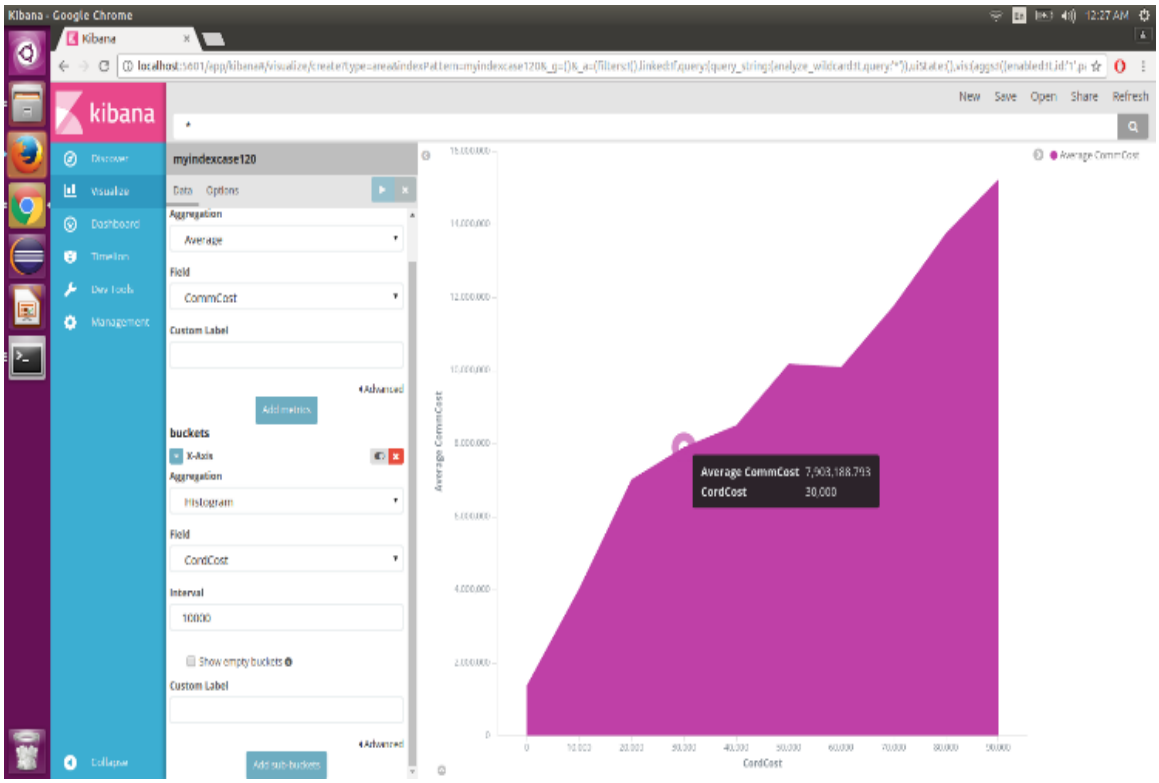

Figure 9 Case 1 Script = 20

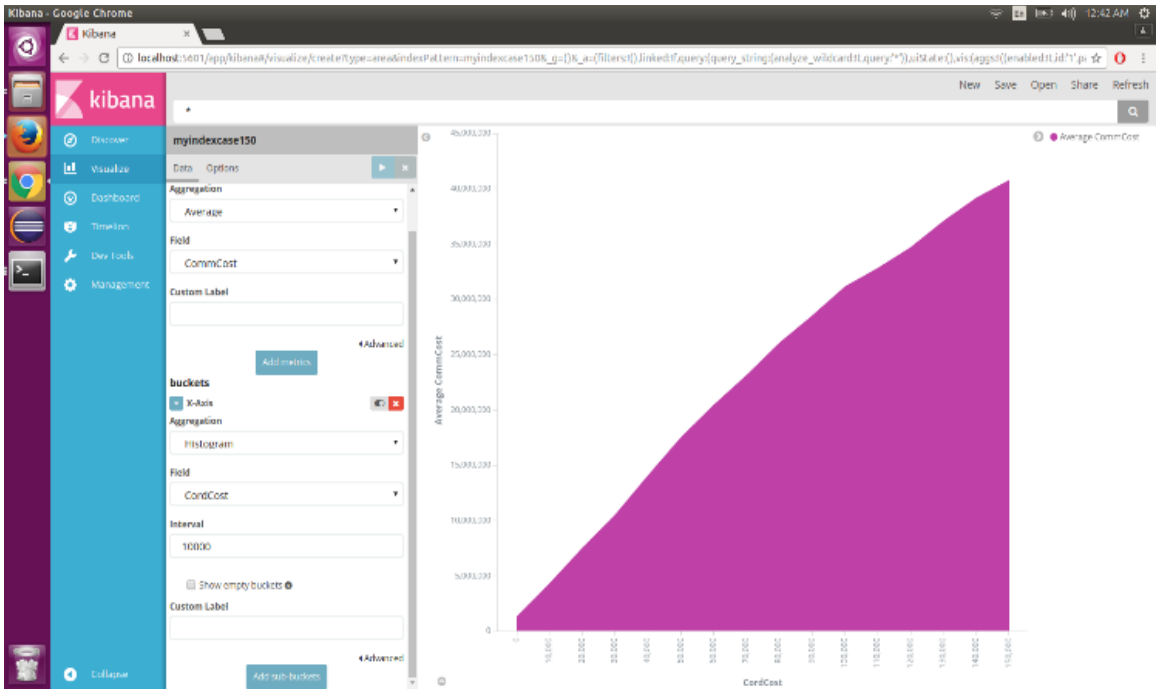

Figure 10 Case 1 Script = 50

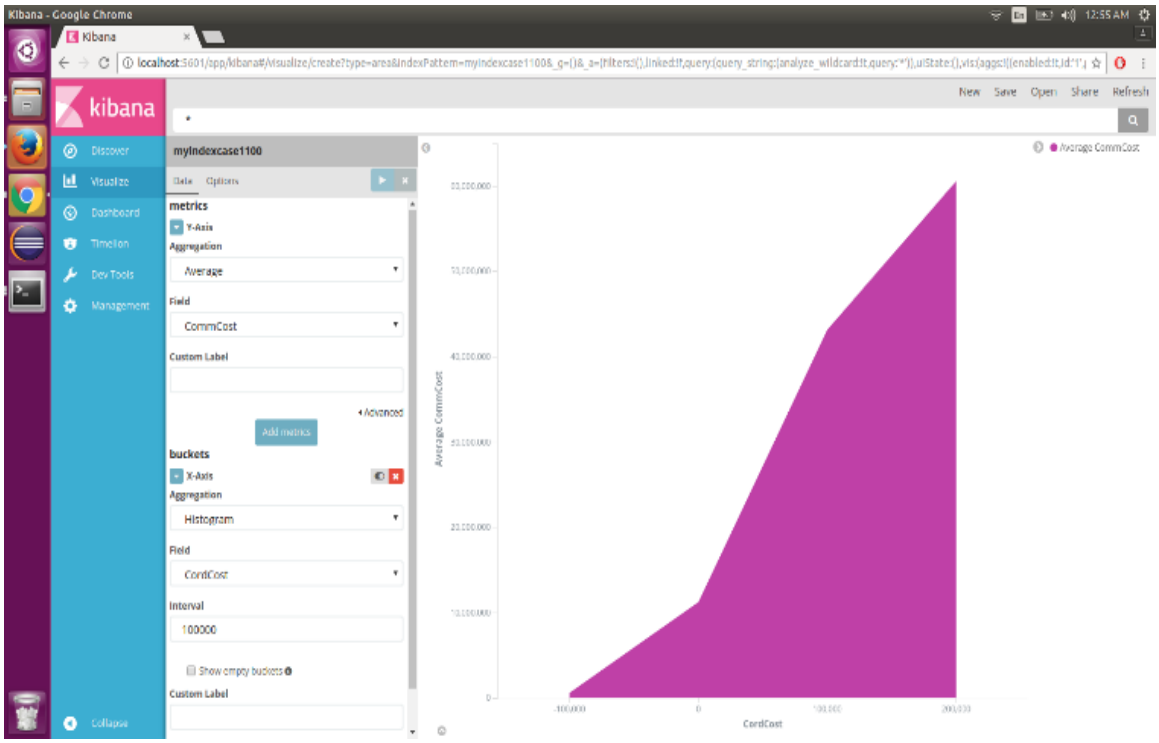

Figure 11 Case 1 Script = 100

figure 12 shows the graph for case 2 with script=20, figure 13 shows the graph for case 2 with script=20 and figure 14 shows the graph for case 2 with script=20.

By comparing case 1 and 2 for script=20 (figure 9 and figure 12), we observed that the communication and coordination cost was much lower for case 2 as compared to case 1. A similar result was observed for script=50 (figure 10 and figure 13) and script=100 (figure 11 and figure 14). We checked for arbitrary points in the graphs to compare the costs. The average communication cost in case 1 for script 20, 50 and 100 were in the order of 10 million, 21 million and 23 million respectively when the coordination cost was set to 60,000. The average communication cost in case 2 for script 20, 50 and 100 were in the order of 748k (748,000), 1 million and 700k respectively when the coordination cost was set to 60,000. This shows that the communication and network overhead is significantly less in our approach as compared to the centralized approach.

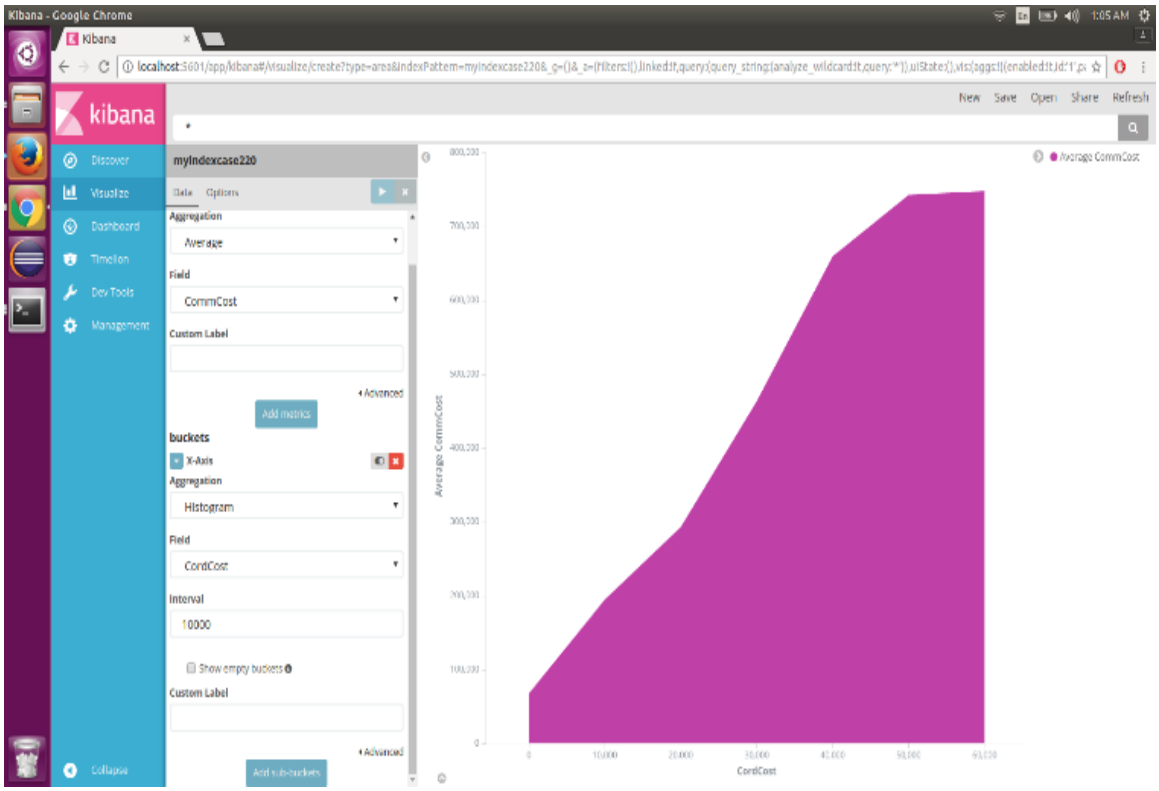

Figure 12 Case 2 Script = 20

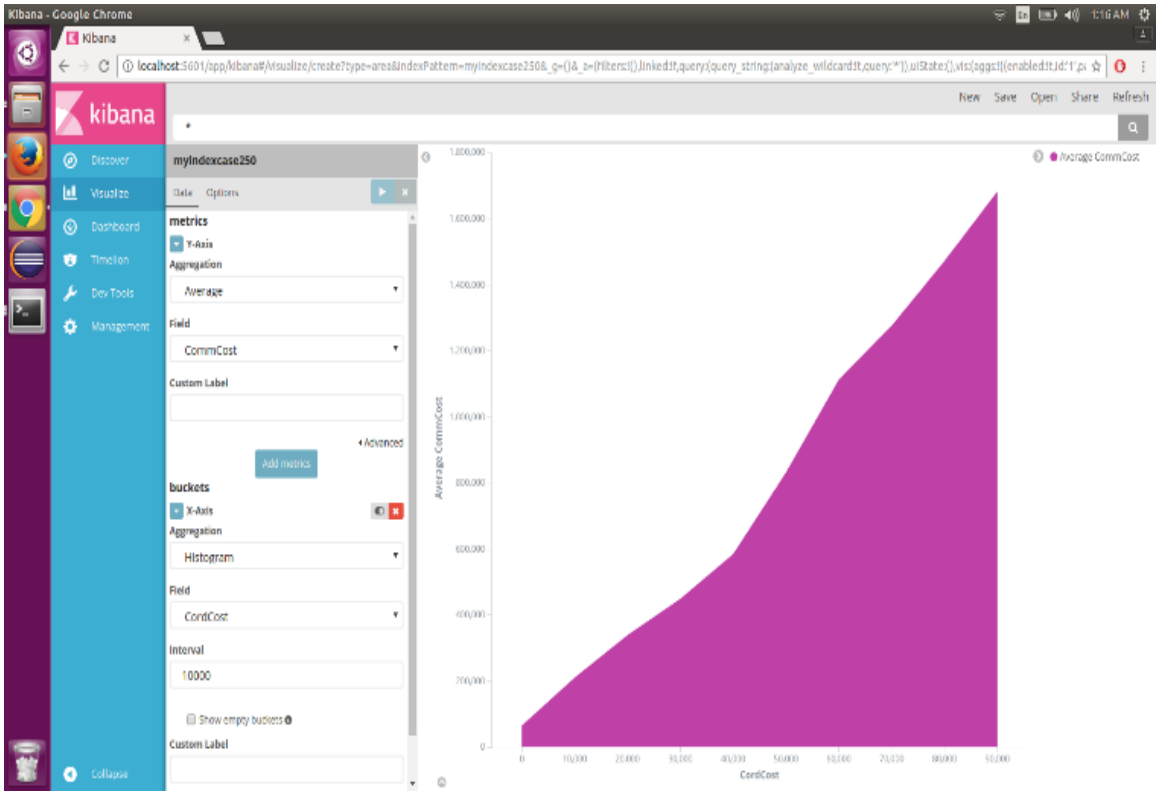

Figure 13 Case 2 Script = 50

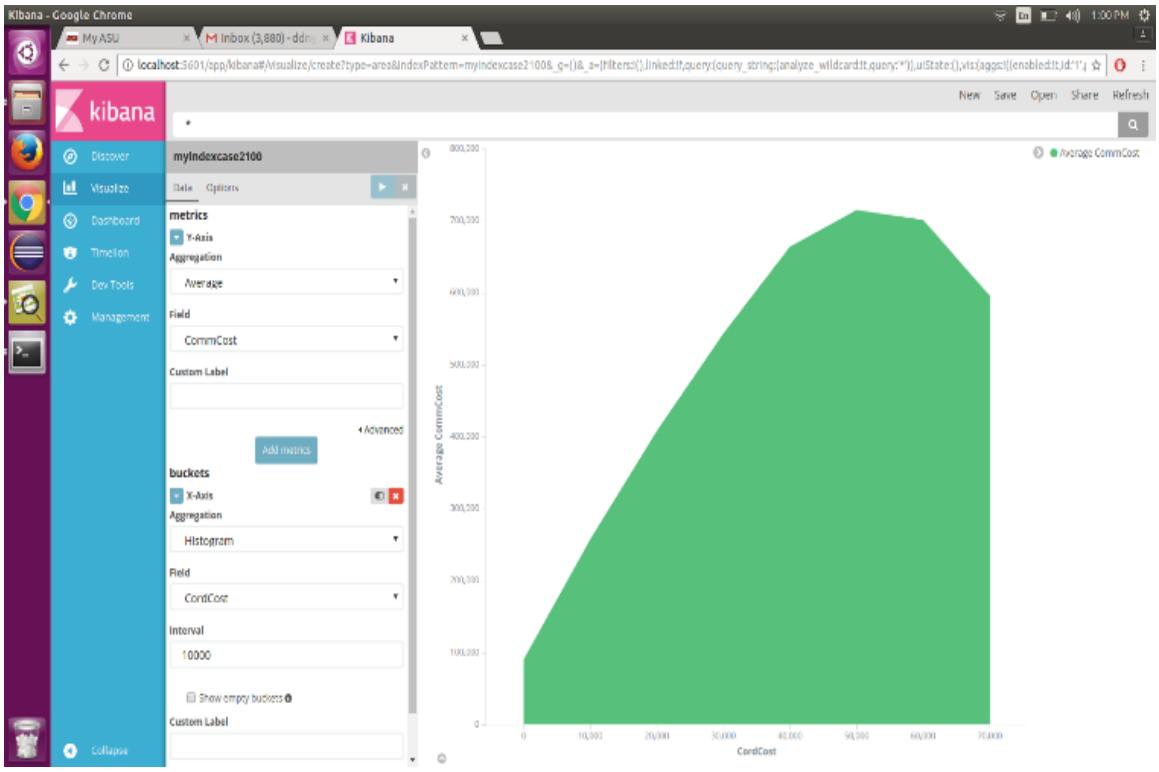

Figure 14 Case 2 Script = 100

CHAPTER 8

CONCLUSION AND FUTURE WORK

In this thesis, our approach for detecting, locating and recovering from an unauthorized modification in a large environment like a smart city was presented. The usage of blockchain in our approach greatly helped in making the entire system more democratized. Our approach generated better results in the simulation as compared to the traditional centralized approaches. We used proof by contradiction to prove mathematically that any unauthorized modifications in the environment was always detected. This approach brings a distributed view of managing a volatile system like a smart city environment.

This approach has several sets of advantage and disadvantage in comparison to the conventional centralized solutions.

The advantages of this approach are as follows:

1. Any unauthorized modification of stored critical data can be detected and located very quickly.
2. The system can recover very quickly from an unauthorized modification.
3. All actions in the ecosystem have authenticity and non-repudiation.
4. Provides an open and trusted platform for corporations to invest in smart city environment as any unauthorized modifications by a malicious actor can be detected.
5. Cost per operation is significantly less as compared to centralized approach.
6. This approach integrates the use of smart contracts which offers a unique way of enforcing smart city regulations and rules.

The limitations of this approach are as follows:

1. This approach is not suitable for non-critical data due to the resource requirement of blockchain.
2. If over 51% of the major nodes in the ecosystem is compromised then the approach fails [12][13][14].
3. Each major node in the ecosystem must have sophisticated hardware for performing computationally expensive operations. This also leads to high consumption of power on these nodes.

Based on the points mentioned above, we can use this approach for practical purposes in a smart city for critical data only. Extending the scope to non-critical data will lead to a huge computational overhead as well as network overhead.

The key points of this approach include better detection of unauthorized modifications from malicious insiders, distributed peer to peer network with no single point of failure, open and trusted environment for organizations and businesses to invest in, lower communication and network overhead, etc. As mentioned earlier, our approach does not fare well if we extend the scope to include non-critical data into the blockchain ecosystem. Also, there is need for large initial capital investment to operate and maintain the blockchain environment. The nodes participating in the process should perform computationally expensive operations in order to maintain the blockchain. Our approach fails when 51% or more of the total nodes are compromised.

Blockchain technology can be employed in different ways to tackle some of the important issues of the smart cities. Future work includes implementing a prototype for this

ecosystem to get a better idea about some of the non-tangible aspects of the environment like the energy consumption of blockchain under different difficulty levels of mining, incorporating blockchain to enhance the access control mechanisms of the smart city entities and devices, and addressing privacy issues. Introduction of a cryptocurrency to a smart city will bring about a fresh way of managing daily lives in a smart city. Another future work would be to see how this blockchain approach can synergize with the cryptocurrency infrastructure for a smart city.

REFERENCES

[1] Pellicer, Soledad, Et Al. "A Global Perspective Of Smart Cities: A Survey." *Innovative Mobile And Internet Services In Ubiquitous Computing (Imis), 2013 Seventh International Conference On*. IEEE, 2013.

[2] Monzon, Andres. "Smart Cities Concept And Challenges: Bases For The Assessment Of Smart City Projects." *International Conference On Smart Cities And Green Ict Systems*. Springer International Publishing, 2015.

[3] Arroub, Ayoub, Et Al. "A Literature Review On Smart Cities: Paradigms, Opportunities And Open Problems." *Wireless Networks And Mobile Communications (Wincom), 2016 International Conference On*. IEEE, 2016.

[4] Nam, Taewoo, And Theresa A. Pardo. "Conceptualizing Smart City With Dimensions Of Technology, People, And Institutions." *Proceedings Of The 12th Annual International Digital Government Research Conference: Digital Government Innovation In Challenging Times*. ACM, 2011.

[5] Khan, Zaheer, Zeeshan Pervez, And Abdul Ghafoor. "Towards Cloud Based Smart Cities Data Security And Privacy Management." *Utility And Cloud Computing (Ucc), 2014 IEEE/ACM 7th International Conference On*. IEEE, 2014.

[6] Ferraz, Felipe Silva, And Carlos André Guimarães Ferraz. "Smart City Security Issues: Depicting Information Security Issues In The Role Of An Urban Environment." *Proceedings Of The 2014 IEEE/ACM 7th International Conference On Utility And Cloud Computing*. IEEE Computer Society, 2014.

[7] Khatoun, Rida, And Sherali Zeadally. "Cybersecurity And Privacy Solutions In Smart Cities." *IEEE Communications Magazine* 55.3 (2017): 51-59.

[8] Nakamoto, Satoshi. "Bitcoin: A Peer-To-Peer Electronic Cash System." (2008): 28.

[9] *Katz, Jonathan; Lindell, Yehuda (2008). Introduction To Modern Cryptography. Chapman & Hall/Crc*

[10] Lamport, Leslie, Robert Shostak, And Marshall Pease. "The Byzantine Generals Problem." *ACM Transactions On Programming Languages And Systems (Toplas)* 4.3 (1982): 382-401.

[11] King, Sunny, And Scott Nadal. "Ppcoin: Peer-To-Peer Crypto-Currency With Proof-Of-Stake." *Self-Published Paper, August* 19 (2012).

[12] Zheng, Zibin, Et Al. "Blockchain Challenges And Opportunities: A Survey." *Work Pap* (2016).

[13] Lin, Iuon-Chang, And Tzu-Chun Liao. "A Survey Of Blockchain Security Issues And Challenges." *Ij Network Security* 19.5 (2017): 653-659.

[14] Tapscott, Don; Tapscott, Alex (May 2016). The Blockchain Revolution: How The Technology Behind Bitcoin Is Changing Money, Business, And The World. Pp. 72, 83, 101, 127. Isbn 978-0670069972

[15] Christidis, Konstantinos, And Michael Devetsikiotis. "Blockchains And Smart Contracts For The Internet Of Things." *IEEE Access* 4 (2016): 2292-2303.

[16] Watanabe, Hiroki, Et Al. "Blockchain Contract: Securing A Blockchain Applied To Smart Contracts." *Consumer Electronics (Icce), 2016 IEEE International Conference On*. IEEE, 2016.

[17] Wood, Gavin. "Ethereum: A Secure Decentralised Generalised Transaction Ledger." *Ethereum Project Yellow Paper* 151 (2014).

[18] Wang, Paul, Amjad Ali, And William Kelly. "Data Security And Threat Modeling For Smart City Infrastructure." *Cyber Security Of Smart Cities, Industrial Control System And Communications (Ssic), 2015 International Conference On*. IEEE, 2015.

[19] Ferraz, Felipe Silva, And Carlos André Guimarães Ferraz. "More Than Meets The Eye In Smart City Information Security: Exploring Security Issues Far Beyond Privacy Concerns." *Ubiquitous Intelligence And Computing, 2014 IEEE 11th Intl Conf On And IEEE 11th Intl Conf On And Autonomic And Trusted Computing, And IEEE 14th Intl Conf On Scalable Computing And Communications And Its Associated Workshops (Utc-Atc-Scalcom)*. IEEE, 2014.

[20] Djigal, Hamza, Feng Jun, And Jiamin Lu. "Secure Framework For Future Smart City." *Cyber Security And Cloud Computing (Cscloud), 2017 IEEE 4th International Conference On*. IEEE, 2017.

[21] Jang-Jaccard, Julian, And Surya Nepal. "A Survey Of Emerging Threats In Cybersecurity." *Journal Of Computer And System Sciences* 80.5 (2014): 973-993.

[22] Claycomb, William R., And Alex Nicoll. "Insider Threats To Cloud Computing: Directions For New Research Challenges." *Computer Software And Applications Conference (Compsac), 2012 IEEE 36th Annual*. IEEE, 2012.

[23] Splunk Website. Https://Www.Splunk.Com/

[24] ELK. Https://Www.Elastic.Co/

[25] Michael Goul, Vineet Mishra, Divyesh Dnyanmothe. “Organizational Data And Analytics Contracting In Smart City Fog Computing Platforms”, Forthcoming In The International Conference On Systems Sciences, January 2018

[26] Bitcoin Mining. Https://Www.Bitcoinmining.Com/Faq/

[27] Bitcoin Mining Hardware Guide. Https://Www.Bitcoinmining.Com/Bitcoin-Mining-Hardware/?Ad=Semd&An=Msn_S&Am=Modifiedbroad&Q=Bitcoin+Mining+Hardware+Fpga&O=29593&Qsrc=999&L=Sem&Askid=4762e9eb-C911-40f7-9519-21a9999a3d16-0-Ab_Msm

[28] Ahamad, Shaikshakeel, Madhusoodhnan Nair, And Biju Varghese. "A Survey On Crypto Currencies." *4th International Conference On Advances In Computer Science, Aetacs*. 2013.

[29] The Price Of Electricity In Your State. “Http://Www.Npr.Org/Sections/Money/2011/10/27/141766341/The-Price-Of-Electricity-In-Your-State”

APPENDIX A

SOURCE CODE FOR SIMULATIONS

The simulation code and its related files is made open source to help the research community. The entire source code can be accessed through the following link:

https://github.com/ddnyanmothe/research_iot

BIOGRAPHICAL SKETCH

Vineet Mishra is an M.S. student in Computer Science specializing in Information Assurance at Arizona State University, and a member of the ASU Information Assurance Centre. He received his B.E. degree in Computer Engineering in 2015 from University of Mumbai at Mumbai, India. His research interests include forensics, cyber threat intelligence, blockchain, and cyber security. He has worked as a security analyst intern at M.U.F.G. Union Bank's Cyber Security Operations Centre. He received a provisional patent for "Systems and Methods for an Intelligent Darkweb Crawling Infrastructure for Cyber Threat Intelligence Gathering" in 2016. This technology was featured in Forbes, MIT technology review, ACM TechNews, Cisco Continuum, etc. His research has been accepted / published in top-tier conferences including, IEEE Intelligence and Security Informatics and International Conference of System Sciences.