



January 2019

## Decision Analytics Using Permissioned Blockchain “Commledger”

Atif Farid Mohammad

[How does access to this work benefit you? Let us know!](#)

Follow this and additional works at: <https://commons.und.edu/theses>

---

### Recommended Citation

Mohammad, Atif Farid, "Decision Analytics Using Permissioned Blockchain “Commledger”" (2019).  
*Theses and Dissertations*. 2476.  
<https://commons.und.edu/theses/2476>

This Dissertation is brought to you for free and open access by the Theses, Dissertations, and Senior Projects at UND Scholarly Commons. It has been accepted for inclusion in Theses and Dissertations by an authorized administrator of UND Scholarly Commons. For more information, please contact [und.common@library.und.edu](mailto:und.common@library.und.edu).

DECISION ANALYTICS USING PERMISSIONED BLOCKCHAIN “COMMLEDGER”

by

Atif Farid Mohammad

Bachelor of Science, University of Karachi, 1992

Master of Computer Science, University of Karachi, 1997

Master of Computer Science, Queen’s University, 2009

Doctor of Philosophy in Information Technology, University of Quebec, 2015

A Dissertation

Submitted to the Graduate Faculty

of the

University of North Dakota

in partial fulfillment of the requirements

for the degree of

Doctor of Philosophy

Grand Forks, North Dakota

May

2019


This dissertation, submitted by Atif Farid Mohammad in partial fulfillment of the requirements for the Degree of Doctor of Philosophy from the University of North Dakota, has been ready by the Faculty Advisory Committee under whom the work has been done and is hereby approved.

  
\_\_\_\_\_  
Dr. Ronald Marsh, Chairperson

  
\_\_\_\_\_  
Dr. Emanuel Grant

  
\_\_\_\_\_  
Dr. Assion Lawson-Body

  
\_\_\_\_\_  
Dr. Yanjun Zuo

  
\_\_\_\_\_  
Dr. Douglas Munsch

This dissertation is being submitted by the appointed advisory committee as having met all of the requirements of the School of Graduate Studies at the University of North Dakota and is hereby approved.

  
\_\_\_\_\_  
Dr. Chris Nelson, Associate Dean  
School of Graduate Studies

  
\_\_\_\_\_  
Date

## PERMISSION

Title            Decision Analytics using Permissioned Blockchain “CommLedger”  
Department    Computer Science  
Degree         Doctor of Philosophy

In presenting this dissertation in partial fulfillment of the requirements for a graduate degree from the University of North Dakota, I agree that the library of this University shall make it freely available for inspection. I further agree that permission for extensive copying for scholarly purposes may be granted by the professor who supervised my dissertation work or, in her absence, by the Chairperson of the department or the dean of the School of Graduate Studies. It is understood that any copying or publication or other use of this dissertation or part thereof for financial gain shall not be allowed without my written permission. It is also understood that due recognition shall be given to me and to the University of North Dakota in any scholarly use which may be made of any material to my dissertation.

Atif Farid Mohammad

April 24, 2019

## TABLE OF CONTENTS

|  |     |
|--|-----|
| LIST OF FIGURES .....                            | vii |
| LIST OF TABLES .....                             | ix  |
| ACKNOWLEDGMENTS .....                            | x   |
| ABSTRACT.....                                    | xi  |
| CHAPTER  |     |
| I.    INTRODUCTION .....                         | 1   |
| Motivation.....                                  | 1   |
| Problem Definition.....                          | 2   |
| Proposition .....                                | 2   |
| Community Ledger Network Approach .....          | 3   |
| Summary .....                                    | 5   |
| II.   STATE OF THE ART WORK.....                 | 6   |
| Blockchain Background.....                       | 6   |
| Concepts of Blockchain .....                     | 6   |
| Public and Private Blockchains .....             | 8   |
| Hyperledger’s Mechanics .....                    | 10  |
| Umbrella Approach for Hyperledger .....          | 11  |
| Hyperledger Tools .....                          | 13  |
| Hyperledger Versus Ethereum.....                 | 15  |
| Cryptocurrency – Permissionless Blockchain ..... | 18  |

|             |  |           |
|-------------|--|-----------|
|             | Distributed Public Ledger .....  | 19        |
|             | Blockchain Analytics .....   | 20        |
|             | Blockchain Privacy .....   | 22        |
|             | Blockchain Access Control.....   | 23        |
|             | Blockchain Approaches .....  | 24        |
|             | CommLedger Risk Management .....                                       | 25        |
|             | Summary .....  | 26        |
| <b>III.</b> | <b>PROPOSITION OF THE COMMLEDGER PERMISSIONED<br/>BLOCKCHAIN .....</b> | <b>27</b> |
|             | CommLedger Technical Description .....                                 | 27        |
|             | Proof of Permission (PoP) and ADAM Block.....                          | 28        |
|             | TALA Key Generation Process .....                                      | 30        |
|             | ADAM Block.....  | 32        |
|             | CommLedger Network .....   | 34        |
|             | CommLedger Potential Use-Case.....                                     | 36        |
|             | Summary .....  | 41        |
| <b>IV.</b>  | <b>IMPLEMENTATION AND RESULTS .....</b>                                | <b>42</b> |
|             | CommLedger Tool Box .....  | 42        |
|             | CommLedger Test Net (CLTN) Environment.....                            | 43        |
|             | CommLedger Test-Net Operational Overview.....                          | 47        |
|             | CommLedger Differentiator .....  | 48        |
|             | CommLedger Account Creation Process.....                               | 51        |
|             | CommLedger Network Initiation (Test Start).....                        | 53        |

|                                     |    |
|-------------------------------------|----|
| Summary .....                       | 57 |
| V. CONCLUSION AND FUTURE WORK ..... | 58 |
| Conclusion .....                    | 58 |
| Future Work .....                   | 61 |
| APPENDIX.....                       | 63 |
| REFERENCES .....                    | 67 |

## LIST OF FIGURES

| Figure   | Page |
|--|------|
| 1. CommLedger ADAM block creation process .....  | 28   |
| 2. Use of Proof of Permission (PoP).....   | 32   |
| 3. The Hash calculated for TALA key generation processing.....                           | 33   |
| 4. Three business entities (Restaurants, Big Data as a Service providers, RnD Firm)..... | 36   |
| 5. Use of SmartContract.....   | 37   |
| 6. The extraction of stored Blockchain for Decision Analytics research .....             | 38   |
| 7. The CommLedger associated miners.....   | 38   |
| 8. Creation of UNDC (UNDCoin) by the resolution of TALA key by miners .....              | 39   |
| 9. A complete CommLedger Network utilization in a nutshell .....                         | 39   |
| 10. Checking the Geth stable version .....   | 43   |
| 11. Ethereum fake node for test net .....  | 44   |
| 12. Object not found .....   | 45   |
| 13. Latest versions of node and NPM.....   | 45   |
| 14. Truffle installation process .....   | 46   |
| 15. Installed Blockchain script editor for CLTN/CLN .....                                | 46   |
| 16. CommLedger Test Network Initiation.....  | 47   |
| 17. CommLedger Successful Initiation.....  | 48   |
| 18. CommLedger Ethereum genesis block created.....                                       | 49   |
| 19. Proof of Permission TALA key gen process to add in CommLedger .JSON.....             | 49   |
| 20. CommLedger.JSON Snapshot .....   | 50   |
| 21. Writing of the custom genesis block with POP-TALA key completed.....                 | 50   |



|     |   |    |
|-----|---|----|
| 22. | CommLedger new account generation .....                       | 51 |
| 23. | Creation of three CommLedger accounts .....                   | 52 |
| 24. | The keystore to obtain CLTN accounts data storage .....       | 52 |
| 25. | Private CommLedger Test-Network with accounts index list..... | 52 |
| 26. | The CLTN Tool Box.....  | 53 |
| 27. | Successful deployment of CLTN.....                            | 54 |
| 28. | CommLedger accounts health check .....                        | 55 |
| 29. | CommLedger Test Network successful transfer of tokens .....   | 56 |
| 30. | Sealing the Block process depiction .....                     | 56 |
| 31. | Shows the account balances of each CLTN user .....            | 57 |

## LIST OF TABLES

| Table  | Page |
|--|------|
| 1. The use of Blockchain in cryptocurrencies and DLT ..... | 59   |

## ACKNOWLEDGEMENTS

Foremost, I would like to express my sincere gratitude to my PhD Committee Chair and Co-Chair, *Professor Dr. Ronald Marsh, Professor Dr. Emanuel Grant* of University of North Dakota for the continuous support of my Ph.D research, for their patience, motivation, enthusiasm, and immense knowledge. Their guidance helped me in all the time of research and writing of this dissertation. I would also like to thank the rest of my doctoral committee: *Professor Dr. Assion Lawson-Body, Professor Dr. Yanjun (Frank) Zuo, Professor Dr. Douglas Munki*, for their wise guidance during my research work and insightful comments, and hard questions. I am also grateful to Dean Hesham El-Rewini, Professor Dr. Brian Tande, the Associate Dean of Computer Science, and Ms. Nancy Nettum and Ms. Jill Schroeder for their on-going administrative support and for their encouragement and help. I am also grateful to Dr. William McGimpsey for his wonderful support as the Dean of the Graduate School alongside our Associate Dean Dr. Chris Nelson, Ms. Stacy Ortiz, and Ms. Laura Look.

Last but not the least; I would like to thank my beloved mother Nasima Tahir and brothers and sister as well as my father-in-law Muhammad Ishaq, my mother-in-Law Hameeda Begum and my sisters and brother in law for their prayers and encouragements.

This doctorate would have never been completed without the immense support of my wife Attia Tun Noor, my son Nauman Ahmad Farid (14 years of age) and my daughter Nymah Farid (9 years of age) at the first place and supporting me in every way throughout my educational life.

## ABSTRACT

The advent of Blockchain has introduced a paradigm shift in the area of Scientific Computing. The decision analytics embodiment in current technology fabric has introduced a need of incorporating Blockchain with industrial technology ecosystem. The utilization of Blockchain has introduced gaps in terms of standard business processes, while the data is being processed using the concept of traditional RDBMS and NoSQL data formats. The lag of permissioned and permissionless Blockchain is the problem area which is dealt with in this doctoral dissertation to provide a Proof of Permission (PoP) protocol for any organization or entity to tailor according to their environmental constraints. There has been a need of an opensource protocol that organizations can customize according to their needs, which is not bound of using only REST interactions. The research presented in this thesis provides such a solution for the industry. The provided propositions are the use of Tiered Asynchronous Locking Algorithm (TALA) to generate a key for securing an Authenticated Data Acceptance Marker (ADAM) block for a permissioned Blockchain Community Ledger (CommLedger).

# **CHAPTER I**

## **INTRODUCTION**

The advent of Blockchain introduced a way to add to organizations' capabilities by increasing the capacity in new business transactional use between two or more industry verticals by using Distributed Ledger Technology (DLT). Blockchain also brought new extensions to in the world of technology education, as well as industry with the resources from the available technology stacks provided by both closed source and technology solution providers. Organizations started adapting either closed or opensource solutions, and in the last few decades, we have seen a tremendous growth in the IT industry as closed source technology providers have captured a large number of organizations to serve; however, in the last decade or so, organizations are starting to move toward opensource technology.

### **Motivation**

The need of a permissioned Blockchain DLT motivated this dissertation. This document provides details on CommLedger (Community Ledger) as a proposed permissioned Blockchain [1] using UNDCoin (UN-Digitization Coin) as a security token offering. Why is DLT is important? That is the integral question. The proposed CommLedger will serve as a tamper proof and incorruptible distributed, digitized ledger proposed to contain all transactions pertaining to something valuable between two or more organizations, CommLedger is a chain of transactional data blocks including detail proceedings that have occurred in the past between two or more departments within one or more organizations. This collection of blocks will be interlinked in

chronological order together with the use of cryptography using SHA256 algorithms and the merger of a Merkle Tree embodying decision trees [2].

A common weakness in the permissioned Blockchain Hyperledger and its associated fabric [3] is the lack of the use of a cryptocurrency. In contrast CommLedger has an associated security token (UNDCoin), which is discussed in detail in Chapters III and IV.

### **Problem Definition**

The Permissioned Blockchain domain has Hyperledger as an umbrella solution. Hyperledger is an opensource solution and does not support cryptocurrency, nor there are any miners associated with it. Other disadvantage with Hyperledger is no discoverability of the transaction.

I have achieved CommLedger (Community Ledger) to solve the problems associated with Hyperledger by making Smartcontract as a Service. CommLedger offers the use of UNDCoin (as a security token for transactional processing) in as a secure way to exchange any kind of goods, services, or transactions, and each member organization within a larger organization will have a node in the proposed Blockchain network with decentralized miners associated for the transactional discoverability. Chapter III provides the CommLedger in detail.

### **Proposition**

This section sheds light on the importance of CommLedger. The associated processing of blocks used in the CommLedger provide the mean for an organization to setup CommLedger nodes for two or more business entities. Each of the mentioned nodes will orchestrate the proprietary data to and from and within these nodes by accessing the latest copy of the encrypted ledger to validate a new transaction. The unique terminologies proposed in this dissertation are as follows:

- CommLedger Network (CLN)
- Proof of Permission (PoP)
- Authenticated Data Acceptance Marker (ADAM) block
- Tiered Asynchronous Locking Algorithm (TALA)

Each CLN node will issue an Authenticated Data Acceptance Marker block (ADAM block). An ADAM block contains four parts: (a) the data pertaining to a transaction, (b) the hash, (c) the previous block hash with a Proof of Permission (PoP) having Byzantine Fault Tolerant (BFT) [4] consensus by both nodes (organizations SmartContract/chaincode), and (d) a key generated by the Tiered Asynchronous Locking Algorithm (TALA key) [5].

The TALA key allows users in the network to validate earlier data for any manipulation before giving consent for the transaction to occur. Furthermore, CommLedger is a research effort producing a solution for open-source utilization for organizations to leverage and customize it according to their needs for Blockchain pragmatic adaptation. Details on the CommLedger, ADAM block and PoP using the TALA key will be provided in detail in Chapter III. Currently, there is an initiative known as Hyperledger available in the space of Distributed Ledger Technology (DLT) [1]. Chapter II presents a detailed discussion on Hyperledger and associated projects to establish the unique need for CommLedger.

### **Community Ledger Network Approach**

The problem of the Hyperledger's permissioned blockchain is solved by the adaptation of CommLedger. This section discusses the proposed approach of DLT in terms of CommLedger built upon a series of networks of Blockchain data stores that allow participants to create, disseminate, and store information in an efficient and secure manner using the TALA key

generation process. The nodes associated within a CommLedger securely operate without the need for any central party or central administrator that every participant knows and trusts.

At the same time, these ADAM blocks within the CommLedger network(s) are constantly available for performing a full audit trail of information history that can be traced back to the moment when a piece of information was initially created. Furthermore, unauthorized changes to the information and its history are very difficult, if not impossible, to make. In other words, operations within CommLedger are designed in such a way that information stored and communicated through the CommLedger networks has a high level of trustworthiness, and every participant in the network can obtain simultaneous access to a common view of the information.

Structurally speaking, a standard Blockchain can be considered as a series of blocks of information that are securely chained together. Any given digital record of an asset, be it a copy of the deed to a bricks-and-mortar property or a virtual commodity, can be stored in a block. New blocks are formed whenever participants create a piece of new information or change an existing piece of information about an asset, for example by entering transaction records, changes of status, new market prices, or new owners. All blocks produced by Bitcoin/Ethereum and such, are newly formed after the first block is securely chained to the previous one, thus ensuring their authenticity and creating a trustworthy audit trail. In fact, one of the earlier uses of DLT was in the area of virtual commodities (e.g., Bitcoin), for which change in the ownership of a commodity is recorded in the Blockchain.

The underlying methodology utilized in CommLedger is the Proof of Permission (PoP) methodology by utilizing decision trees. Such methodologies are used to represent the substitutions accessible to the decision makers in an organizational setting, the ambiguity they



encompass, and evaluation measures representative of how well objectives would be accomplished in the resultant outcome.

The CommLedger's proposed environment allows an organizations' analytics to use feature extraction using supervised and semi-supervised learning. As a decision maker to create the next block within a CommLedger, ADAM block creation will utilize multi-attribute utility/value functions.

### **Summary**

Unlike traditional DLTs, the CommLedger proposition in this dissertation provides a DLT variant Permissioned Blockchain Ledger (PBL) utilizing an associated cryptocurrency (UNDCoin). However, such evolving technology also brings possible risks regarding issues of governance, deployment, risk management, and regulatory compliance, along with the legal implications are discussed in Chapter II, if they are not adequately taken into account.

## **CHAPTER II**

### **STATE OF THE ART WORK**

This chapter details, how several other solutions of Blockchain have been provided for organizations to adopt. The discussions also present the idea of CommLedger in relation to the provisions, unavailable in the current research work.

#### **Blockchain Background**

Businesses deal with variety of contracts and are also bound by the particular country's policies and procedures, Hyperledger allows businesses to construct a Chaincode to be used by several entities to conduct their everyday order of business. Blockchain's popularity is growing quickly and becoming the way we make transactions [1]. The recent trend of changing large enterprises opting to use Blockchain is a sign that Distributed Ledger Technology (DLT) systems are here to stay. These systems are becoming the go-to technology for large industries looking to revolutionize their business models, and even though many applications use traditional, public, and well-established Blockchain platforms, large enterprises don't want to use them as it exposes confidential data to all nodes within the network. This is currently where Hyperledger takes the lead.

#### **Concepts of Blockchain**

The following sections present several concepts frequently used in Blockchain technology and play a vital role in distinguishing the traditional Blockchain architecture when compared with the Hyperledger network. Cryptocurrency is the most common understanding of

what Blockchain is, and the next generation of business-oriented applications now tend to opt for the DLT [2].

### **Distributed Ledger Technology (DLT)**

DLT is a digital system for storing transactions when they are forged in multiple places and even at the same time. These transactions are generally processed in blocks according to the ordering of a Blockchain that results in a distributed ledger [2]. As mentioned previously, Hyperledger is currently filling this gap, however the transactions in Hyperledger are subject to Fiat currencies (such as the US Dollar or Euro) and no embodiment of mining exists within Hyperledger. Mining is the process of adding transaction records to public ledger of past transactions.

Community Ledger (CommLedger) is presented in this dissertation in more detail, which allows the embodiment of a cryptocurrency and/or a security token use to conduct the everyday order of the business between organizational entities using UNDCoin (UN-Digitization Coin).

### **Cryptography**

Blockchain uses a hash-based algorithm to make sure that each entry in the ledger is in a certain order and cryptographically connected to the previous blocks or transactions. The exact order of these entries is transparent and can be verified by all the nodes and participants in the network, thus protecting the entries from being tampered with.

### **Smart Contract (aka smartcontract)**

Smart contracts are a layer above the DLT in the workflow of a Blockchain [2]. They are the code (a computer program) that includes the complex business logic stored and executed in a Blockchain. An apt analogy would be stored procedures updating data in a database and they allow you to transact anything of value without the need for any central authority.

## **Public and Private Blockchains**

Public Blockchains are open source, permission-less, and can be maintained by anyone with enough computing power to solve the cryptic keys associated with the time stamp and previous pointer address do so [5]. They allow full transparency of the information they contain and are not vulnerable to data tampering due to decentralized governance. Private Blockchains sometimes require permission to access them and they operate in a centralized way [5]. In this scenario, unlike public Blockchains, only parties affiliated with the transaction are connected and can consent to complete a deal.

### **Hyperledger**

Hyperledger is software that enables developers across the globe to develop Blockchain-based solutions for their businesses. It is a global collaboration hosted by the Linux Foundation and is the fastest growing project it has ever hosted [6]. Built under technical governance and open collaboration, individual developers, service and solution providers, government associations, corporate members, and end users are invited to participate in the development and promotion of these game-changing technologies [2].

There are two key concerns that an enterprise might have in Blockchain adoption [6].

- Data Privacy: Industries don't want to share their confidential data with everyone.
- Speed of Transactions: Enterprises cannot afford delays while committing a transaction due to business reasons.

Apart from the above, an enterprise might also have other concerns such as mining, legal restrictions on the location of nodes, and the modular design approach, where the 'one model fits all' mindset is not realistic. They might want to use DLT, but they need it customized to

integrate it into the existing processes they are using. Hyperledger is the promising solution that addresses all these concerns [6].

The idea to begin the Hyperledger project was announced at the end of 2015 [7].

Members of the initiative symbolize a diverse group of stakeholders such as Accenture Solutions Private Limited, IBM, Wells Fargo, Intel and J. P. Morgan. Initially the project had a limited number of developers from various sectors including finance, supply, and data management. For the Hyperledger initiative community, the Linux Foundation hosted a collaboration environment, forming neutral ground for conferences, events, and productive deliberations, and provided a structure around the industry and procedural guidance for Hyperledger.

**Vision for Hyperledger:** Blockchain assures diversion from standard business practices [2], and that the perspective of every enterprise using the public Blockchain model will slowly change. Going forward there will be no single Blockchain network that all enterprises will use. Eventually, a huge number of private Blockchain mechanisms will be created for various markets. These will possibly have a diverse consensus algorithm, a smart contract with preferred language and logic, and other new, important features. Hyperledger has a promising future as it seems it will become the go-to and safe Blockchain utilization platform for many industries.

**Goals for Hyperledger:** The Hyperledger community is where open-source teams build diverse approaches for business Blockchain technology systems [2]. Their five goals are listed below:

- 1) Build at an enterprise level, private Blockchain network with all the distributed ledger technologies and business logics to support their industries trades;
- 2) Provide nonaligned, community driven set-ups supported by methodical governance.

- 3) Create developer communities to build Blockchain and distributed ledger frameworks and test scenarios and deployments;
- 4) Edify the public about the scope for Blockchain adoption in the enterprise system.
- 5) Encourage the Hyperledger community to help the public invent more platforms and related frameworks.

### **Hyperledger's Mechanics**

The initial developing Hyperledger team tested the interactions between applications and the secure Blockchain network [8]. During rigorous testing they realized that public Blockchains require each peer to execute every transaction and run the consensus at the same time. But when there is a case of confidentiality and privacy associated with a transaction, it cannot be executed on the public Blockchain, because every ledger on the network would get updated and confidentiality would be lost; therefore, this is the appropriate situation when private and confidential contracts like Hyperledger are needed. Hyperledger doesn't need all the nodes or participants in the Blockchain network to consent to the deal [8]. It only requires concerned members associated with the transaction to be connected. For instance, let's assume there is a deal between node A and node B [8]. Entity A wants to sell something valuable to entity B at a special price. Node A (assigned to entity A) searches his/her application for B's address on the Hyperledger network and the application looks up a membership service and validates B's membership.

The Hyperledger network then connects both parties because they are the only ones directly affiliated with the deal. Both parties generate a result; the results are brought together and then sent to the consensus cloud to be ordered and verified. Once the consensus cloud validates the transaction, node B receives his product, node A receives his/her money, and the

transactions are committed to the ledger. Thus, the Hyperledger mechanism is characterized as a permissioned Blockchain design. For this Hyperledger mechanism to take place, the traditional Blockchain architecture has been tweaked [8]. The peers have been separated into two separate run times and three distinct roles: (a) Endorsers, (b) Committers, and (c) Consenters. Endorser and committer peers are run at the same run time, whereas the consenter is at a different run time. Hyperledger modular architecture allows properties like consensus to be a pluggable feature that in turn allows a high degree of personalization of the network according to the needs of the business.

#### **a) Committers**

The committers only role is to append the validated and ordered transactions to the specific ledger once the consenters returned them. Committers can also work as endorsers, but due to some restrictions in respective networks, that scenario is usually not preferred.

#### **b) Endorsers**

They simulate transactions pertaining to a network and prevent nondeterministic and unstable transactions. All endorsers can act as committers on a Hyperledger network.

#### **c) Consenters**

A consenter is responsible for the network's consensus service, in other words, running the consensus algorithm on the network. The consenter is like the guardian of the network, as it validates every transaction by running the Practical Byzantine Fault Tolerance (PBFT) consensus algorithm and decides whether it should be added to the ledger.

### **Umbrella Approach for Hyperledger**

Like the Linux Foundation, there is also a modular umbrella approach for Hyperledger [2]. On a high level, they provide the infrastructure and the opportunity for the volunteered

developers to come and help the community. This support includes everything related to technical guidance, legal help, and promoting certain aspects of the community. Governed by technical meritocracy, Hyperledger is vendor-neutral. Contributions undergo a rigorous peer review process but are welcome from anyone.

Under Hyperledger's modular umbrella approach, frameworks and their corresponding tools are available to help build the Blockchain network's diverse mechanisms for various enterprises. There are many benefits to this modular approach, such as the flexible modification of any component, extensible code bases, rapid experimentation, common functional modules, defend interfaces, re-use of common building blocks, and a diverse developer community [2]. The Hyperledger project involves teams of volunteer developers, an open building code, managing the groups' own road maps, releasing schedules, being responsible for following Hyperledger policies and requirements, and being encouraged but not required to align their code with other projects [9].

Let's compare CommLedger and the associated UNDCoin with Hyperledger, Bitcoin, and Ethereum [9]:

- UNDCoin and CommLedger provides a comprehensive umbrella to the organizations to use UNDCoin as the inherent cryptocurrency.
- Transfer of UNDCoins takes place when an organizational asset is transferred between two organizational participants to prevent the double-spend problem in terms of asset transactions.
- The proposed CommLedger, uses a security token (UNDCoin) to create the asset transfer in secured blocks.



- CommLedger incorporates a smartcontract that can be used with standard Ethereum ERC20 protocol as well.

### **Hyperledger Tools**

CommLedger is comparable to the available Hyperledger tools in terms of utilizing the following, the use of these tools is detailed in Chapter IV:

- GoEthereum (geth) [10]
- Node JS & NPM [11]
- Atom (Text Editor) [28]

This section sheds light on the tools, available for Hyperledger. The given tools incubate and promotes a range of business Blockchain technologies, including tools and utility libraries [12] [13]. This toolset discussion allows us to understand the inherent complexity of using Hyperledger. For the proposed CommLedger, Chapters III and IV provide the details of utilizing only one environment that CommLedger is built upon to spearhead the potential use of the permissioned Blockchain, rather making comparisons among the various toolsets shown in the following bullet points.

#### **a) Caliper**

This Hyperledger tool measures the performance benchmark for any Blockchain implementation with the help of a set of predefined use cases [12] [13]. The performance reports produced by Caliper give performance indicators such as resource utilization, the latency affiliated with any transaction, and the number of transactions per second. Before Caliper was created, there was no generic tool to measure the performance of diverse Blockchain frameworks based on a set of universally accepted rules [12] [13].

## **b) Cello**

Cello is a Blockchain model toolkit targeted to inculcate the on-demand deployment model to the Blockchain community [12] [13]. The main objective of this Hyperledger tool is to help industries easily and swiftly become accustomed to Blockchain applications by providing programmed ways to build, manage, and remove Blockchains. Cello provides a dashboard for the enterprises to find, for example, the status of their Blockchain environment, statistics related to utilization, performance, and the series of transactions committed. It currently provides support to Fabric [12] [13]; however, it is planned that Cello will soon support Sawtooth and other frameworks.

## **c) Composer**

Composer is a development tool open for everyone to contribute to [12] [13]. It can develop use-cases and deploy a Blockchain network within weeks. This is extremely fast when compared with the small number of other tools in the market that take a few months to accomplish the same thing. Composer also allows users to promptly design a model to replicate their business architecture as well as to aid and integrate their current ecosystem and information with Blockchain applications [12] [13].

## **d) Explorer**

Explorer provides its users with a dashboard to see all the details related to the transactions, participants, and smart contracts [12] [13]. Users can query certain transactions to obtain all the related details. Explorer's main objective is to create a web-based tool that's easy to accumulate and integrate within existing Blockchain frameworks.

### **e) Quilt**

Quilt follows the Inter Ledger protocol (ILP) built in Java [12] [13]. Quilt provides interoperability between the ledger systems using ILP and creates a global namespace for accounts to help make transactions across ledgers; therefore, it allows diverse kinds of ledgers from various enterprise backgrounds to connect with one another in order to perform distributed atomic transactions.

## **Hyperledger Versus Ethereum**

Hyperledger is sometimes preferred over another successful Blockchain technology platform called Ethereum. Below are some key differences and comparisons.

### **a) Target Audience**

Ethereum, which is a decentralized network, enables business logic to be implemented using smart contracts executed on the Ethereum Virtual Machine (EVM) [4]. This is mainly intended for applications diverse in nature and for mass consumption. Hyperledger with its modular approach provides a wide range of what you want to use and what you don't [14]. It's directed at enterprises and industries wanting to simplify their process by integrating the available Blockchain technology into their business.

### **b) Mode of Operation**

Ethereum is a public network without any permissions [14], but with its recent protocols and various versions, it can also serve as a private and permissioned network. Depending on the privacy type, it allows users to participate in the network. Hyperledger is private and permissioned and decides who gets access to participate in the consensus and view the transactions.

### **c) Cryptocurrency**

Ethereum has a network-specific cryptocurrency called Ether [4]. Hyperledger has no specific cryptocurrency, and it gives its developers the freedom to implement the Blockchain technology for their business; if the business needs any cryptocurrency they can always implement one.

### **d) Consensus Mechanisms**

From its inception Ethereum followed Proof of Work (PoW), and recently it started using Proof of Stake (PoS) [15]. Proof of Work is a requirement to define expensive computing, which is also called mining. Mining verifies whether the transactions within each block are authentic, miners must solve a mathematical puzzle known as the proof-of-work problem [15]. The first one to solve the puzzle announces his/her solution and all the other users in the network verify it. That miner is rewarded for the effort. Ethereum miners are usually rewarded with Ether [4]. Proof of Stake (PoS) is a different way to solve the problem. The miner is chosen in a deterministic way, depending on his/her wealth or stake, in other words, whoever holds more currency [15]. There is no block reward in Proof of Stake (PoS); the miners just awarded the transaction fees. Proof of Stake (PoS) is more cost effective than Proof of Work (PoW) as the miners in the latter consumes more energy, which in turn leads to higher electricity costs.

Hyperledger allows its developers to decide whether they need a consensus mechanism. If they decide to use one, then an algorithm called Practical Byzantine Fault Tolerance (PBFT) is used [4]. PBFT is the network's ability to work properly and reach a valid consensus conclusion shared by the honest nodes in the system, despite having some malicious nodes trying to send invalid information to other peers or attempting to make them fail in reaching the valid conclusion [16]. To implement PBFT, each node maintains an internal state regarding the status

of the network, and when the nodes receives any information, they use that message in conjunction with the internal state to run a computation that helps them reach a conclusion regarding the validity of the message [4]. Then the node shares this with all other nodes in the network, and a consensus decision is made based on the total decisions submitted by all the nodes. The key difference between the Proof of Work (PoW) and Practical Byzantine Fault Tolerance (PBFT) is, in PoW the first node who solves the problem shouts out the answer, whereas in PBFT it requires all the participating nodes in the consensus to respond [4].

#### **e) Smartcontract**

Both technology platforms allow the business logic to be implemented with aid of smart contracts. Ethereum smart contracts are generally written in either Ethereum specific scripting language called Solidity or Serpent [4]. Hyperledger smart contracts are written in the GO language.

#### **f) Language of Development**

Ethereum is developed using the GO language and Python [4]. Hyperledger is written in the GO language and Java.

#### **g) Governance**

As a platform, the Ethereum Developer Community maintains Ethereum [4]. Hyperledger comes under The Linux Foundation, and like many other Linux projects, people from all over the world are able to contribute.

Ethereum is an inflationary cryptocurrency, as discussed in Chapter IV. Because Hyperledger does not use a smartcontract, two transacting entities have to establish a chaincode to transact the blocks within. CommLedger allows the entities to embed a smartcontract by

creating ADAM blocks and securing the organizational transactions using UNDCoin, rather than a Fiat currency.

### **Cryptocurrency – Permissionless Blockchain**

Early ideas [17] do encompass decentralized currencies. Drawbacks of these currencies include the requirement for trusted parties and the double-spending problem (where a cryptocurrency can be spent twice, because cryptocurrencies can be reproduced.). A Permissionless Blockchain that is based on cryptographic protocols is resistant to tampering, provides greater security, is driven by decentralized network consensus, and allows data to be transmitted and stored in a peer-to-peer (P2P) fashion. Satoshi Nakamoto [5] proposed a P2P distributed timestamp server that generates computational proof of transactions. In terms of Bitcoin, an electronic coin is a chain of digital signatures. A Bitcoin ledger shows the status of all existing Bitcoins and a state transition function in the form of a transaction. Each output of a transaction can be used only once in the entire Blockchain and thus overcomes the double-spending problem. Demanding PoW from each node verifies a transaction and transactions can be tracked back at any time. Transactions within a block are hashed in the Merkle tree and the root of the leaf nodes is a hash of its children.

A Simplified Payment Verification (SPV) doesn't require nodes to keep a full record of transactions, only a copy of the block headers of the longest chain. The nodes in the network accept the block as valid only if all transactions within it are valid and not already spent. If a block is accepted, the chain continues by creating the next block. Nodes are rewarded with coins and the process of adding new blocks to the Blockchain is called mining. The research conducted by Vujičić, et al [17] also describes the problem of fork (two blocks mined/created at same time) and how Bitcoin resolves it by continuing the network with the longest chain.

Vujičić, et al introduces the concept of SegWit (Segregated Witness) as a solution to the Bitcoin Scalability issue. They later adds on Ethereum and its improvements the Blockchain structure. Their network is based on a modified GHOST protocol intended to tackle issues such as stale blocks and centralization. Each account in Ethereum has a 20-byte address and state transitions. Their research also presents types of accounts and elaborates the fields in an account. The transactions are classified into two types based on their products: (a) message calls, and (b) new account creations.

CommLedger creates Authenticated Data Acceptance Marker (ADAM) blocks. These blocks are mined by UNDCoin miners. Details are provided in Chapter IV where we present the actual ADAM block creation procedure.

### **Distributed Public Ledger**

Blockchain [18] is a distributed public ledger where all users have the same data. Introduced by Satoshi Nakamoto the Bitcoin concept was the first peer-to-peer electronic cash system that made money transfers without a bank acting as trusted third party. However, its design still struggles to ensure some measure of anonymity. The study of security and privacy of Blockchain is hence growing rapidly. Research conducted by Halpin, et al [18] proposes the need for improvements in core cryptographic primitives such as better, more efficient data structures to replace Blockchain while still fulfilling its role as distributed public ledger. The study mentions different signature schemes adopted in Bitcoin. Although Bitcoin offers pseudonymous transactions, pseudonyms may be de-anonymized by determining patterns of usage in Blockchain. Therefore, substantial changes are needed in order to preserve privacy. The article presents two techniques to solve this problem. One is to add anonymization to the existing Blockchain by confidential transfers. The second is to create new Blockchains incompatible with

Bitcoin. A number of companies claim to increase privacy through mixing schemes where Bitcoin transactions from different users are combined together. The cited [18] work also mentions the need for more security and privacy research in smartcontracts. CommLedger permits entities and/or individuals to maintain their privacy by conducting transactions using Proof of Permission (PoP) and securing these transactions by encrypting them by using TALA keys within the UNDCoin smartcontract.

### **Blockchain Analytics**

Blockchain is a transaction database [19] containing information about all transactions and using the Bitcoin protocol. It allows participants on a network to edit the ledger in a secured way and to share its transactions over a distributed network. All nodes present in the network run algorithms to evaluate, verify, and match the transaction information with the Blockchain history to add in an existing chain. The secure hashing algorithm SHA-256 generates a hash to identify individual blocks. Every block contains a hash of the parent block in its header and a sequence of hashes linking the individual block with their parent block. A genesis block is a common ancestor of all the blocks created in 2009 and is encoded by the Bitcoin client software and thus can't be tampered with. The research presented by Singh, et al [19] describes the prerequisites of Bitcoin such as Authentication, Integrity, and Non-Repudiation. There are several advantages to Bitcoin such as it's faster and cheaper, has a decentralized registry, secure transactions, and supports mining.

Bitcoin uses the Elliptic Curve Digital Signature Algorithm (ECDSA) to ensure access of funds to the rightful owners. Each Bitcoin is associated with a public ECDSA key of its current owner. When a Bitcoin is sent, it attaches the new owner's public key with the sender's private key. Any participant can create multiple public and secret key pairs. Bitcoin doesn't require a



third party and Bitcoin miners form a network to maintain the Blockchain. These miners solve cryptographic puzzles that contain data from several transactions. Bitcoin supports single-signature and multisignature transactions. A Bitcoin transaction has a drawback in that it requires a time window before it is confirmed. Singh and Singh [19] introduces techniques such as P2SH transactions, Ethereum, and so on, and also mentions the emergence of Blockchain in the Gartner Hype Cycle, 2016. Moreover, they elaborate on the way Blockchain is currently adopted by several industries such as Nasdaq, Bitfinex, IBM, and Samsung. In conclusion, Blockchain has the potential to reshape the capital market industry and can offer a system that minimizes the amount of trust required with peer-to-peer messaging protocols and distributed data sharing. The transactional data of modern cryptocurrencies (e.g., Bitcoin and Ethereum) [20] is stored in the public Blockchain ledgers within the blocks. This data is immutable. Data analysts analyze this data using traditional and ad hoc methods that are not that efficient considering the volume of data and complexity of the analysis. Hence a framework [20] is provided that seamlessly supports data analytics on Bitcoin and Ethereum as they both store several gigabytes of data related to currency transfers. Their proposed tool consists of two steps: (a) study of the created view containing Blockchain data and other outside data and store these views in a local database for further use, and (b) study of the analyzed view using DBMS query language. Steps (a) and (b) are supported by the Scala standard library. The research work [20] further presents an analysis of Blockchain data without use of external data; the following four types of data were found: (a) the transaction hash, (b) the hash of the enclosing block, (c) the date in which the block was appended to the Blockchain, and (d) the list of transaction inputs and outputs.

Traceability within CommLedger data assets transfers can be traced back to the first block, which is also known as an ADAM block. Miners associated with CommLedger allow

several copies of once confirmed block-transaction, and we can apply regression and time series analytical techniques to conduct advanced decision analytics. Bitcoin miners can only solve the cryptocurrency previously linked addresses in terms of solving the mathematical problems presented to them by the provision of Public Key(s) by the transacting individuals.

### **Blockchain Privacy**

Bahack [21] presents an argument about the privacy guaranteed by Bitcoin in the initial Bitcoin whitepaper by Satoshi Nakamoto in which Satoshi mentioned that the security of the system is maintained only when the attacker's computational power is  $1/2$  or more. However, Bahack argues that the current theoretical limit of an attacker's fraction of total computational power essential for the security of the system is not  $1/2$  but a bit less than  $1/4$ , and Bahack's study also proposes changes to the protocol to raise this limit to come as close to  $1/2$  as possible. Furthermore, the research work analyses the importance of the attacks to the real world applicable to the cryptocurrencies that are economically liquid. The study accomplishes this by finding the equilibrium total hash power. According to the paper, if the attacker's fraction of the hash-power, out of the total hash-power, is a bit less than  $1/4$ , then the attacker gains more reward points and the difficulty level of mining is reduced. However, this may change as there are many different block discarding attacks available.

The study concludes that the two attacks are not practically possible as they are only based on the reasonability to maintain the system. However, the study also mentions there is no risk with the second attack, but the first attack, the "Block Discarding Attack," is limited by the structure of the current Bitcoin network. The research work presentation concludes that theoretically when the attacker moves out all the other miners then he/she could attack the system using the DoS attack. This becomes more problematic as the Block-Discarding attacker

stops mining linearly and all the other miners start returning. CommLedger associated UNDCoin mitigates DoS and DDos attacks, and MITM equally by adding the layer of Proof of Permission using Public, Private and TALA key.

### **Blockchain Access Control**

De Montjoy et al. [22] offers the combination of Blockchain [23] and off-Blockchain storage to construct a personal data storage management to keep privacy as the prime consideration. The privacy problem involved the concerns faced by users using third party services and is addressed with this contribution, primarily for mobile app users. The solution presented in the study revolves around the principles of data ownership, data transparency and auditability. Another principle discussed in the study is fine grained access control.

UNDCoin and CommLedger provides a comprehensive umbrella to the organizations to use UNDCoin as inherent cryptocurrency. The creation or transfer of UNDCoins takes place when an organizational asset is transferred between two organizational participants to prevent double-spend problem in terms of asset transactions. The proposed CommLedger Permissioned Blockchain presented in this dissertation revolves around the following principle of data Ownership, where CommLedger Network nodes act as transacting entities that own and control their own data. CommLedger associates miners maintaining the public ledger, where as an organization is capable of using CommLedger to opt-in for a chaincode as well to maintain the transparency within the transacting entities for the need of data transparency and auditability. The proposed CommLedger has flexibility to incorporate chaincode for inter-organizational asset transactions. The use of Proof of Permission (PoP) protocol allows the organizational user to alter permissions only within the organizational-defined policies and procedures to revoke data collection access previously granted to other transacting entities.

## **Blockchain Approaches**

Federated learning [24] is a machine-learning setting in which a centralized model can be trained even when the training data is distributed over a large number of clients. It eliminates the need for storing the data in the cloud as the data stored on the client devices is used to train a central model. Only updates to the model are sent to the cloud. This increases the privacy as the data remains on the client's device. Clients can immediately use the improved model. Thus, this setting is faster, secure, and efficient, but it has certain drawbacks. The model is located on a central server, so if the server fails, the training could collapse. Besides, there is no reward for devices that contribute positively to the global model training.

The decision analytics utilizing a machine-learning setting within an organization using CommLedger allows the use of a centralized model. This model can be trained even when the training data of transacted assets between partner entities are distributed over a large number of CLN nodes. It eliminates the need for storing the data in the cloud in terms of RDBMS, rather the data of assets transactions has an associated timestamp and is stored in the entity's defined cryptic record wallets to train via this central model within an entity (an organization or partner organizations consortium). The model is located on a central server within participating entities, so if the server fails, the training could collapse; however, because the data assets are secured within the wallet, the restart of the model within the participating organization will resume the decision analytical process.

Mihaylov et al. [25] describes an innovative approach to produce and efficiently distribute renewable resources. The Blockchain-based design allows a conversion of energy to NRGcoins that can be exchanged at any time on the open market. Producers feed energy into the grid and do not rely on batteries to store the energy. Smart meters send the energy production

and consumption information to the Distribution System Operator (DSO) at 15-minute intervals. The information is used to determine the rewards for the prosumers and bill the consumers. Rates of consumption and production can be analyzed to balance the supply and demand. Peaks of production/consumption can be avoided by designing a proper reward system. The decentralized system allows local trading of energy at competitive prices. The prosumers (producers and consumers) are given a reward in the form of NRGcoins.

CommLedger introduces three types of UNDCoin users. These are UNDCoin traders, organizational business user, and CommLedger miners. The miners of UNDCoins are rewarded in the form of UNDCoins. The reward is a function of the amount of transactions validated by the CommLedger miners.

### **CommLedger Risk Management**

Any introduction of new technology inevitably presents new types of risk, and CommLedger is no different. Even when an asset owned by a participant is protected by the participant's digital certificate, and no changes can be made to the information without the correct digital signature (an ADAM block with a TALA key), certain traditional cybersecurity issues still apply to CommLedger. For example, denial of access attacks and other cyber-attacks could still be launched against DLT in an effort to cause its operation to fail.

Due to the anonymous nature of participants in some DLT applications (in particular Bitcoin), DLT is sometimes seen as being associated with issues of money laundering and the sale of illegal goods, and as supporting the ransomware payment model. Although these issues might largely be addressed when DLT is implemented in a "permissioned" network, such as the proposed CommLedger (which only authorized and authenticated participants may join), this kind of solution still needs to be closely examined.

## **Summary**

This chapter presented the “State of the Art” work conducted by researchers on topics related to Blockchain, cryptocurrency, privacy, and security threats related to Blockchain. The first section presented in depth the “DLT” (Distributed Ledger Technology) Hyperledger project. This chapter shed light on the problems with existing methods, which has been addressed by CommLedger using Proof of Permission, where an ADAM block is created and secured by both public, private as well as the TALA key to make it hack proof within a permissioned Blockchain DLT.

In the next chapter, the presented work will discuss the methodologies put to work for CommLedger, PoP, TALA key generation, and the initiation of ADAM block.

## **CHAPTER III**

### **PROPOSITION OF THE COMMLEDGER: PERMISSIONED BLOCKCHAIN**

This chapter details the proposition of the CommLedger and its associated framework with Proof of Permission (PoP) using the Tiered Asynchronous Locking Algorithm (TALA) to generate keys for the CommLedger Network and create the Authenticated Data Acceptance Marker (ADAM) block. In this proposal for a Permissioned Blockchain Ledger (PBL), all members have known identities mentioned in this chapter in terms of set membership.

#### **CommLedger Technical Description**

The CommLedger proposition is built on a sectional architecture that separates contractual CLN Data Block dispensation into the following phases:

- The Community ledger (CommLedger) is the sequenced Block of state change of an Asset within CommLedger Network (CLN).
- Each CLN transaction outcome is in a set of an ADAM block (containing the asset) with the associated TALA key.
- A Cassandra or MongoDB alongside CouchDB can store these Blocks.
- The Modular Programming enables CommLedger Network (CLN) designers to plug in their preferred applications.
- The TALA key generation process can be tailored to the needs of establishing “an organizational procured identity”.
- PoP (Proof of Permission) is proposed for Permissioned Blockchain utilization.

- CommLedger DLT requires a set of common rules by which all participants operate in order to ensure its accuracy and trustworthiness.
- A governance framework that a CommLedger proposes is therefore important for the implementation and sustainable operation of DLT.

### Proof of Permission (PoP) and ADAM Block

Let us say there is a user John, who has an asset, a pen, a house, a car, etc., which has some value for him, and he wants to allow someone else to use this asset. The use must have an associated standard UTXO type of signature to allow the other person to use or possess this asset. UTXO stands for the unspent output from Bitcoin transactions. UTXOs are administered uninterruptedly and are accountable for the commencement and conclusion of each transaction. Validation of the contract results in the UNDCoin monetary transactions of crypto coins from the UTXO database.

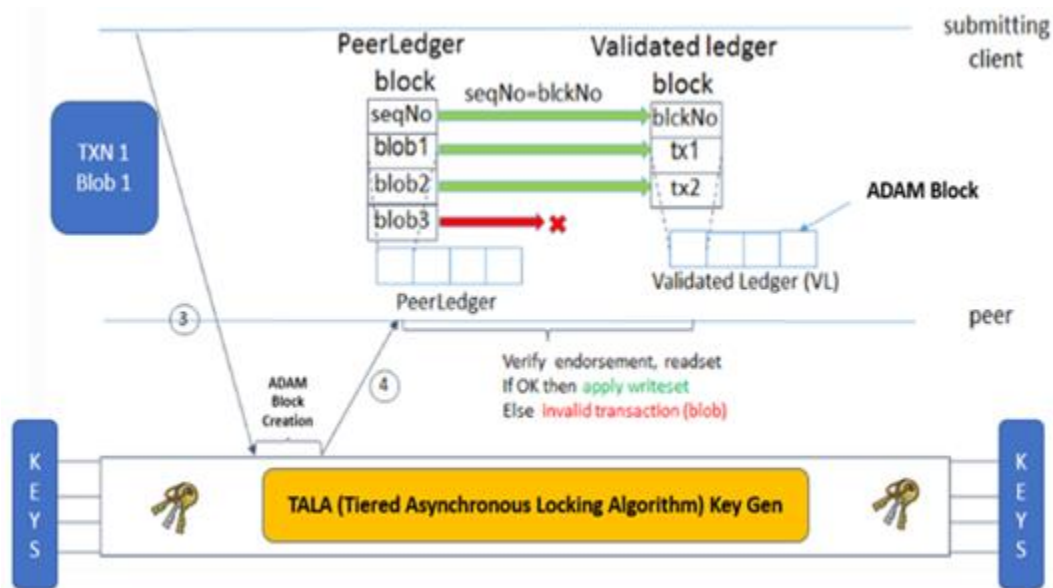


Figure 1: CommLedger ADAM block creation process.



ADAM stands for "Authenticated Data Acceptance Marker", and the creation of an ADAM block is depicted in Figure 1 with the use of TALA keys. In the given use-case of Proof of Permission (PoP), John permits another entity, or a user named David to use the asset by having a PoP Certificate to conduct the successful transaction. The next section explains more details on the ADAM block. PoP initiates an ADAM block by using the TALA key attached to it. Whereas, if John is part of the Community Ledger (CommLedger) and would like to transfer an asset to David, this means that David has permission to use/transfer the asset to another user of CommLedger. To understand this, we take Morris as another potential interested party to this next transaction. Morris will have the asset's ADAM block created (created by John, transferred to David) and issued by making a ledger entry with the TALA key to have a lineage for an Audit Trail available. TALA keys have two associated base keys to work with: (a) The user keys, and (b) the commodity keys.

The user key is associated with more than one commodity key(s) based on the user because a user can own several assets. For example, if a user has multiple commodities and permits someone to use them, then his unique single user key may be associated with all his commodity keys that have their unique IDs. The next section sheds lighter on the TALA key generation process.

In the above use case, if we consider  $J_k$  as John's user key, the commodity keys for the various commodities that John has are as follows:

Pen  $\rightarrow P_{id}C_k$

House  $\rightarrow H_{id}C_k$

Car  $\rightarrow C_{id}C_k$

And because these are John's commodities, we add John's user key to identify these are John's commodities.

For Example, :

Pen  $\rightarrow J_k P_{id} C_k$

Similarly, for John's other commodities we add his user key.

House  $\rightarrow J_k H_{id} C_k$

Car  $\rightarrow J_k C_{id} C_k$

Now, we also have a user David, who does not have any commodities; therefore, David has only his user key:

$D_k$

When John wants to give permission to David to use his commodities, he uses the CommLedger Network to use the TALA key generation and sends it to David to unlock the value of the Asset transferred by John for him to consume. The following describes the base version of the ADAM block's creation for transactions and storage in the CommLedger for an authentic Audit Trail with transaction commencement to completion.

### **TALA Key Generation Process**

The Tiered Asynchronous Locking Algorithm (TALA) generates a key combination for CommLedger permissioned Blockchain nodes to prevent Blockchain Data hacking threats. The compact TALA key uses compressed data structures for a probabilistic illustration of a set-in order to provision membership to an ADAM block at the time of its instantiation. The query is: "Is Block X in (CommLedger) set Y?" This compressed illustration is the payoff in CommLedger Block membership queries; that is, without TALA, queries might erroneously identify a Block already in existence as a member of the CommLedger created earlier for some

TXN (Transaction). This set has a PoP certificate (TALA key) issued by an original Asset Owner for permitting another interested party to either rent or own or rent-to-own the asset within a CommLedger Network Nodes set.

The base concepts of set theory are used to establish a CommLedger Network node(s) membership. ADAM block(s) creation is based on Set theory. This generation of blocks begins with a fundamental binary relation between an object “o” (The Data Element provided by a CommLedger User to be stored in a Blob that later is added in a CommLedger Block) and set “A”, where “A” is a CommLedger Network Nodes set. If “o” is a member (or element) of “A”, the notation “ $o \in A$ ” is used. Because sets are objects, the membership relation can also relate to sets.

A derived binary relation between two sets is the subset relation and is called set inclusion. If all the members of set “A” are also members of set “B”, then “A” is a subset of “B”, denoted “ $A \subseteq B$ ”. For example,  $\{1, 2\}$  is a subset of  $\{1, 2, 3\}$ , and so is  $\{2\}$  but  $\{1, 4\}$  is not; this means ‘1’ and ‘2’ have the PoP Certificate issued to be transacted; however, there is no 4 Block with this CommLedger Network. As insinuated from this definition, a set is a subset of itself. For cases where this possibility is unsuitable or would make sense to be rejected, the term proper subset is defined. A is called a proper subset of B if and only A is a subset of B, but A is not equal to B. Note also that 1, 2, and 3 are members (elements) of the set  $\{1, 2, 3\}$  but are not subsets of it. The given Figure 2 shows the use of PoP. The Blocks architecture of the transactions (TXN – to – TXN) are created by the utilization of an established Service Oriented Architecture (SOA 3.0) [26] [27].

## Proof of Permission (TXN – to – TXN)

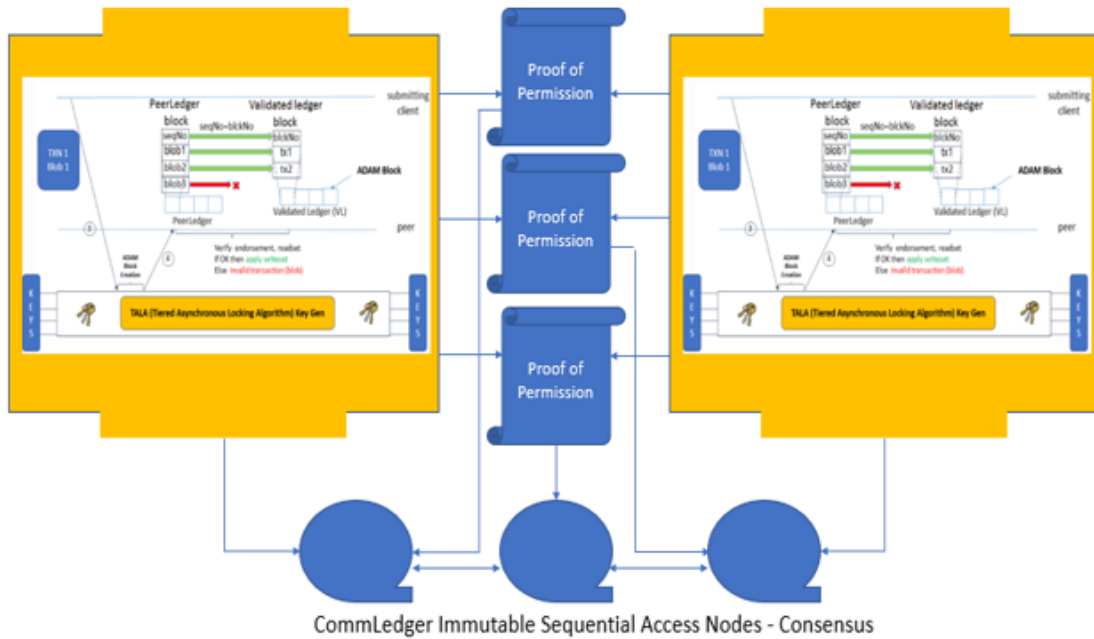


Figure 2. Use of Proof of Permission (PoP).

### ADAM Block

The TALA key generation process works with cached data sets to put the data with the TALA key in a blob form in the CommLedger Block. When all the synopses of these data sets are collected, an ADAM block is then sent to all the nodes in the CommLedger Blockchain Network nodes. To speed up the process of referencing the CommLedger Node(s) record stored within the corresponding differential records, we use the TALA key to convert these differential records individually into a compressed illustration that provides easy access to both records of the CommLedger stored records.

To understand the construction and working of the Tiered Asynchronous Locking Algorithm, consider the following example. Consider CommLedger Network Nodes as a Set  $A = \{a_1, a_2, \dots, a_n\}$  of  $n$  Blocks, where  $a_1$  is an ADAM block. CommLedger gets the PoP by its nodes (organizational users who issue PoP Certificate[s]) that describe the CommLedger

Membership information of A using a bit vector V of length m. For this, k hash functions,  $h_1, h_2, \dots, h_k$  with  $h_i : X \rightarrow \{1..m\}$ , are used as described below.

To summarize and verify large datasets in the ADAM block blobs, an efficient data structure called a binary hash tree (also known as a Merkle tree) is used for the TALA key generation processing. Furthermore, it is a tree structure in which each leaf node is a hash of a block of data, and each non-leaf node is a hash of its children and used in disseminated data classifications for data authentication. The application of this structure is one of the efficient ways it uses hashes instead of full files.

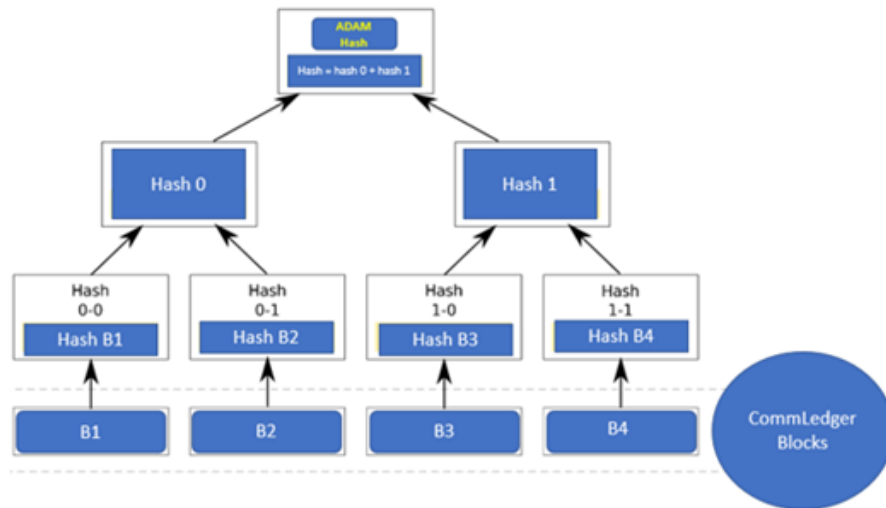


Figure 3. The Hash calculated for TALA key generation processing.

In Figure 3, we have Blocks B1-B4 that are then hashed, and these hashed files are combined. The decision analytics associated with CommLedger can trace the time of the transactions within an organizational CLN in a secure way, which they can embed in their decision support systems. The application of timeseries in analyzing both cyclical and seasonal trends can allow an organization using CLN to create Robotic Process Automation for their advanced analytics needs.

## CommLedger Network

Let us consider an example of having a large database and the data in the database is stored in the form of a Merkle tree. Also consider that the root of the Merkle tree is publicly known and trusted. If a CLN user wants to perform a key value lookup on the CommLedger Stored data sets, then the CLN user can ask for Merkle proof and on receiving the proof, he/she can identify whether the lookup is correct or not. It also allows a mechanism for validating a small number of Blocks in a CLN, like a hash, to be extended to also validate large databases of potentially unbounded size.

The initial application of these Merkle trees is in the Bitcoin introduced by Satoshi Nakamoto [28]. With the help of Merkle trees and Merkle proofs instead of downloading every transaction and every block, it allows the light client to download the chain of 80n byte block headers and contains the five most important characteristics of a block: (a) a hash of the previous header, (b) a timestamp, (b) mining difficulty value, (c) a PoW nonce, and (d) a root hash of the Merkle tree.

Although a Merkle tree has merits, it also has limitations. One major drawback is that though the user can prove the inclusion of transactions, he/she cannot prove anything about the current state, for example digital asset holdings, name registrations, and the status of financial contracts.

The following procedure builds an  $m$  bits TALA key that, corresponds to a CommLedger Network Nodes set  $A$  and uses  $h_1, h_2, \dots, h_k$  hash functions, as depicted in Figure 3.3:

*Procedure TALA\_Key\_Gen(set  $A$ , hash\_functions, integer  $m$ )*

*returns key*

*key = allocate  $m$  bits initialized to 0*

```

foreach  $a_i$  in  $A$ :
    foreach hash function  $h_j$ :
         $key[h_j(a_i)] = 1$ 
    end foreach
end foreach

return key

```

Therefore, if  $a_i$  Block is present in the a CommLedger and is a member of a CommLedger Network Nodes set  $A$ , in the resulting TALA key  $V$  all bits attained conforming to the hashed values of  $a_i$  are set to 1. Testing for CommLedger (CL) Network Node membership of an ADAM block ADAM is equivalent to testing that all corresponding bits of  $V$  are set:

```

Procedure CL_Node_MembershipTest (ADAM, key, hash_functions)
returns yes/no

foreach hash function  $h_j$ :
    if  $key[h_j(ADAM)] \neq 1$  return No
end foreach

return Yes

```

The primary features of the TALA key\_Gen process are that they grow incrementally when a new Block is added to the set. Combining two different sets is done by performing a simple OR operation to the TALA key generation process of the two different sets. The TALA key allows organizations to use the Public Key for the CommLedger associated miners to resolve and lock the transaction within the CommLedger. These miners can be either bind with a transaction-associated fee, set by the organization(s) or the miners can generate an UNDCoin as they find the resolve of the TALA key using their computer.

The math behind the TALA key generation process allows us to observe that the probability of finding a bit is “0” after inserting “n” keys into a TALA key of size “m” using “k” hash function remains as:

$$p_0 = \left(1 - \frac{1}{m}\right)^{kn} \approx 1 - e^{-\frac{kn}{m}}.$$

The probability of a false positive is given by

$$p_{err} = (1 - p_0)^k = \left(1 - \left(1 - \frac{1}{m}\right)^{kn}\right)^k \approx \left(1 - e^{-\frac{kn}{m}}\right)^k$$

$p_{err}$  is minimized when  $k = \frac{m}{n} \ln 2$  hash functions.

Due to the uniformity in the computational overhead of each supplementary function and the reduction in the incremental advantage of each additional hash function after a certain threshold, only a few numbers of hash functions are used, and this forms the basis for generating a TALA key.

### CommLedger Potential Use-Case

The following is a test use-case, that elaborates the use of CommLedger in an industrial setting.

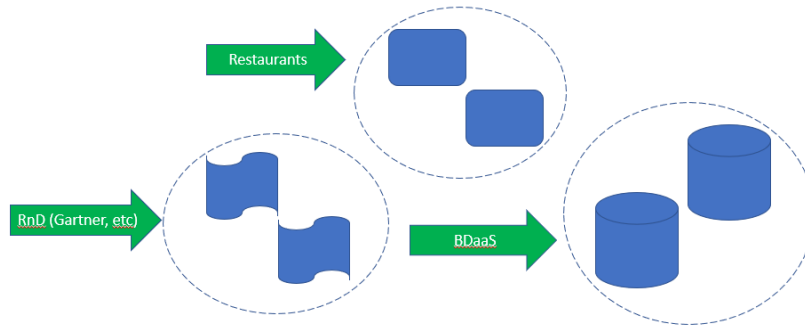


Figure 4. Three business entities (Restaurants, Big Data as a Service providers, RnD Firm).



As depicted in the Figure 4, there are three diverse sets of organizations using the CommLedger PBL (Permissioned Blockchain Ledger). Next Figure 5 illustrates the use of SmartContract between entities. As it is a known fact that Blockchain is immutable, our depicted restaurants are to store their everyday transactions using CommLedger and the ADAM block using TALA keys to store their assets in the Big Data as a Service (BDaaS) provisioning organizational storage clusters.

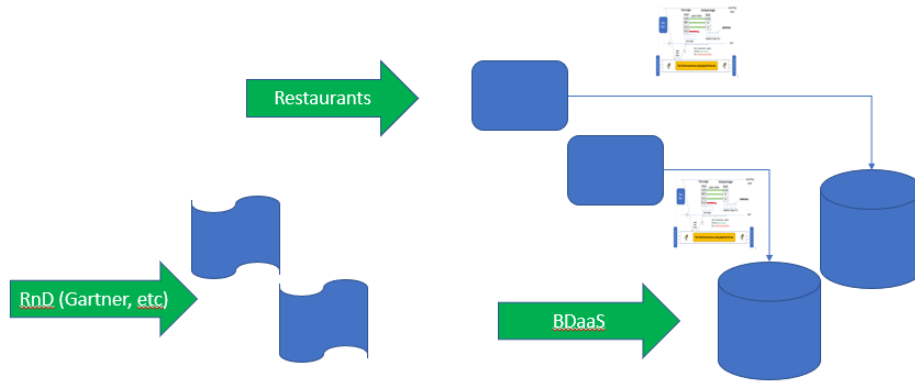


Figure 5. Use of SmartContract.

CommLedger permissioned Blockchain utilizes Proof of Permission and permits multi-chain utilization by other organizations. We can see in Figure 6, that the RnD firm uses a SmartContract for extracting the stored blocks to conduct Decision Analytics to create RnD reports to provide their users within this cryptonomy.

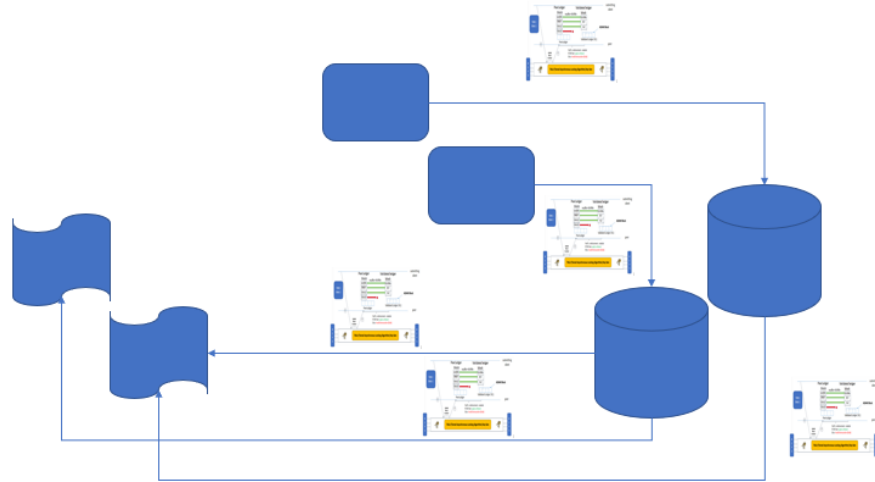


Figure 6. The extraction of stored Blockchain for Decision Analytics research.

In the following Figure 7, smartcontract’s execution process is depicted by the CommLedger’s associated miners by utilizing the resolution of TALA keys to instantiate ADAM blocks.

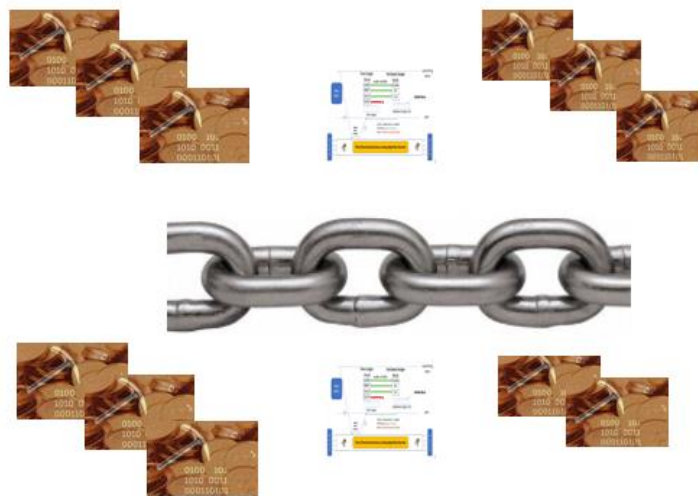


Figure 7. The CommLedger associated miners.

The generation of UNDCoin is presented in Figure 8, this is used by the organizations as transaction validation payment for the miners.

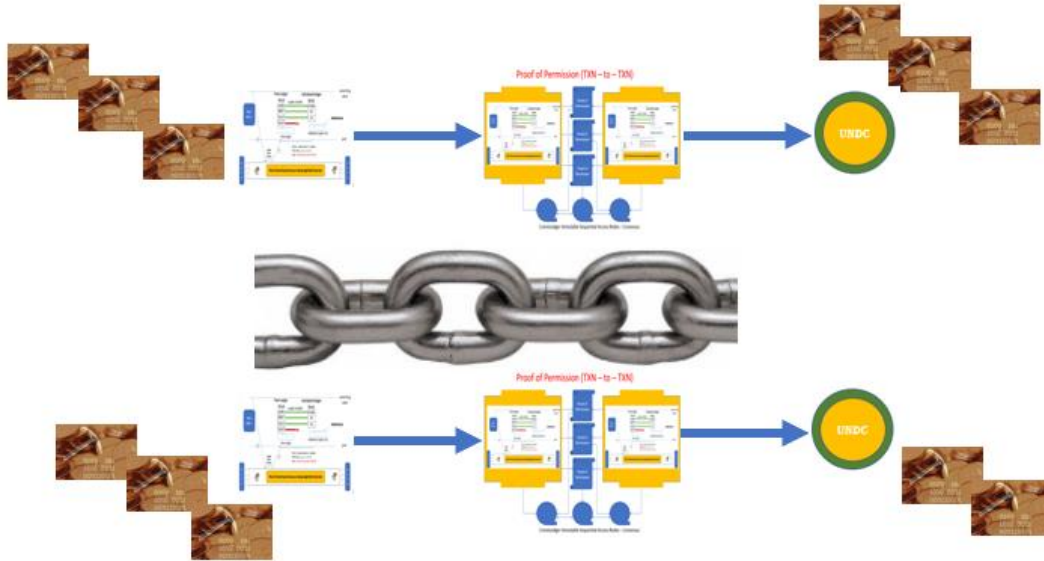


Figure 8. Creation of UNDC (UNDCoin) by the resolution of TALA key by miners

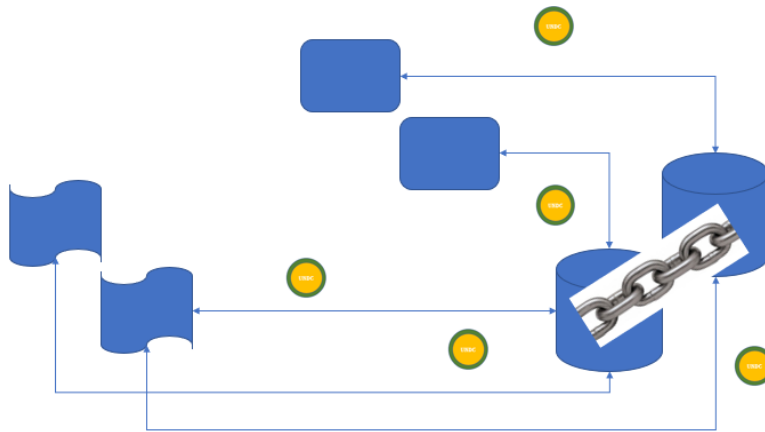


Figure 9. A complete CommLedger Network utilization in a nutshell.

The unique differentiator of CommLedger over available Hyperledger and associated fabric is the use of Proof of Permission protocol for a consensus establishment using Decentralized Miners for the Distributed Ledger Technology users using TALA key generation and an ADAM block.

UNDCoin has no cap associated and CommLedger provides a comprehensive umbrella to the organizations to use UNDCoin as inherent cryptocurrency. The creation or transfer of UNDCoins takes place when an organizational asset is transferred between two organizational participants to prevent double-spend problem in terms of asset transactions. The proposed CommLedger Permissioned Blockchain presented in this dissertation revolves around the following principles:

**Data Ownership:** CommLedger Network nodes act as Transacting entities that own and control their own data.

**Data Transparency and Auditability:** CommLedger has miners maintaining the public ledger, where as an organization is capable of using CommLedger to opt-in for a chaincode as well to maintain the transparency within the transacting entities for the need of data transparency and auditability.

**Fine Grained Access Control:** As mentioned previously the proposed CommLedger has flexibility to incorporate chaincode for inter-organizational asset transactions. The use of Proof of Permission (PoP) protocol allows the organizational user to alter permissions only within the organizational-defined policies and procedures to revoke data collection access previously granted to other transacting entities. In case of a trade dispute or contract termination within the entities. CommLedger's differentiating factors from Hyperledger and associated fabric are:

- 1) **Proof of Permission:** Miners wait for a given entity's permission by receiving the TALA key to be part of the miners' consortium to create ADAM blocks.
- 2) **CommLedger Decentralized Consensus:** Miners' use of the TALA key resolution attempts to solve the problem of agreement within transacting entities to authenticate the transaction's validity.

- 3) **CommLedger Network Fault Tolerance:** The provenance of CommLedger transactional finality can tolerate Byzantine faults. CommLedger utilizes UNDCoin to mitigate a double spend problem by having a Proof of Permission consensus.

### **Summary**

This chapter has detailed the proposition of a CommLedger a permissioned Blockchain as an alternative to the available Permissioned Blockchain Ledger (PBL) and associated technologies. The next chapter provides details on CLTN deployment, and how UNDCoin is instantiated by the adaptation of available open-source tools.

## CHAPTER IV

### IMPLEMENTATION AND RESULTS

This chapter discusses the detailed implementation of the Proof of Permission (PoP), Tiered Asynchronous Locking Algorithm (TALA) key generation process and creation of an ADAM block within a CLTN. Deployment of the CLTN tool box with a running example of CommLedger KeyStore, accounts creation, as well as successful locking of generated blocks is also provided with completed transactions between two accounts.

#### CommLedger Tool Box

Deployment of CommLedger can have its own cryptocurrency, or we can use the standard Ethereum smartcontract to establish a CLTN. We use the following opensource tools to establish this deployment. The entire work of tooling our CLTN will be conducted using Windows PowerShell.

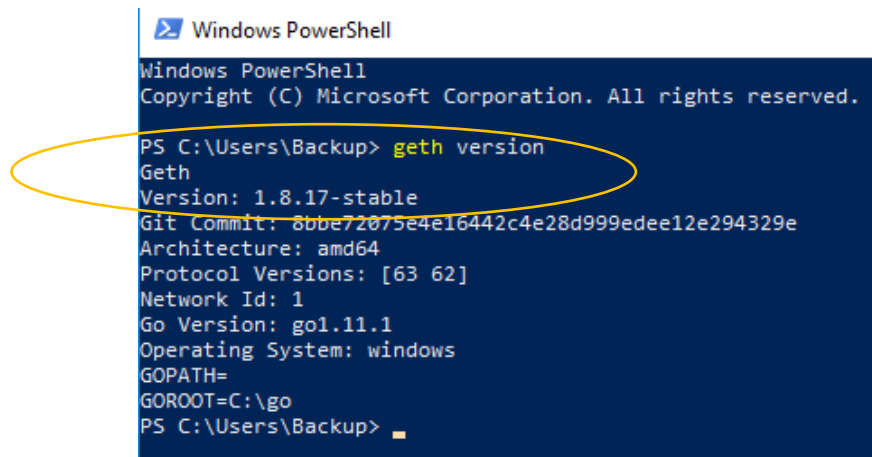
1. **Geth:** Geth is a Go-Ethereum [10] platform and allows a user to run an Ethereum node. It is implemented in the programming language Go to mine blocks by generating Ether, to inspect the Block history. We also create user accounts using Geth that allows the user to connect with both public and private Ethereum networks.
2. **NodeJS & NPM:** NodeJS [11] is the server-side Java script program used to code the applications that interact with Ethereum nodes. We also install Node Package Manager (NPM) that allows us to manage the libraries needed to program for Ethereum.

3. **Atom:** Atom is a free and opensource text and source code editor for most utilized operating systems. It has built-in support for plug-ins written in Node.js and is embedded in Git control developed by GitHub.

### CommLedger Test Net (CLTN) Environment

Let us look into the toolbox mentioned earlier on the deployment for CLTN/CLN. This chapter details the Windows installation processes that need either Google Chrome or FireFox.

- <https://ethereum.github.io/go-ethereum/downloads>
- Download latest Windows version: To date it is V1.8.17
- Start Windows PowerShell
  - Enter command: > geth version



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\Backup> geth version
Geth
Version: 1.8.17-stable
Git Commit: 8bbe72075e4e16442c4e28d999edee12e294329e
Architecture: amd64
Protocol Versions: [63 62]
Network Id: 1
Go Version: go1.11.1
Operating System: windows
GOPATH=
GOROOT=C:\go
PS C:\Users\Backup>
```

Figure 10. Checking the Geth stable version.

- ❖ <https://truffleframework.com/ganache>

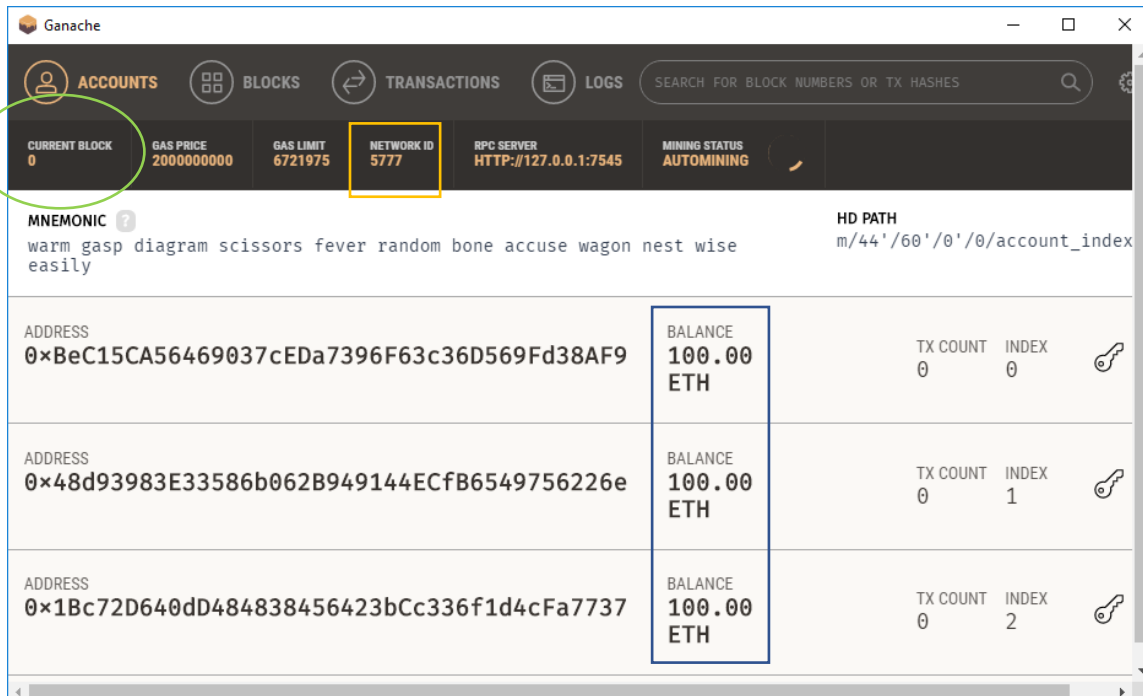


Figure 11. Ethereum fake node for test net.

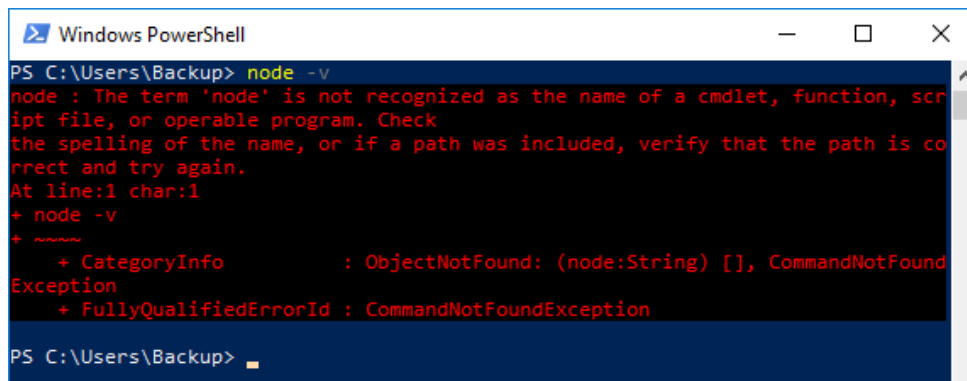
- ❖ This is a test network presented for a visual understanding of a base Ethereum fake node with a GUI display for standard Ethereum transactions.
- ❖ Ganache starts a new Ethereum node in memory. It has the network identifier 5777 as shown in the yellow lined box.
- ❖ This fake Ethereum node has:
  - 10 accounts with 100.00 ETH as shown in the balance in the blue lined box.
  - The current block is zero (0) as seen in the picture in the green circle.

CommLedger utilizes a cryptocurrency smartcontract of any sort, as the differentiator from the standard Hyperledger instead of using chaincode. The other most important information that the fake node as shown in Figure 11 shares for development purposes is MNEMONIC; it is the seed, Ganache uses to generate the first account. We establish CommLedger using the given



development setup, and CommLedger mimics the behavior of cryptocurrency transactions by using the later given setup.

- Open Windows PowerShell and run:
  - > node -v

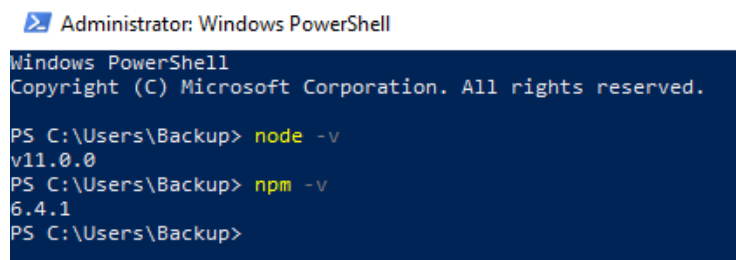


```
Windows PowerShell
PS C:\Users\Backup> node -v
node : The term 'node' is not recognized as the name of a cmdlet, function, scri
pt file, or operable program. Check
the spelling of the name, or if a path was included, verify that the path is co
rrect and try again.
At line:1 char:1
+ node -v
+ ~~~~~
+ CategoryInfo          : ObjectNotFound: (node:String) [], CommandNotFound
Exception
+ FullyQualifiedErrorId : CommandNotFoundException

PS C:\Users\Backup>
```

Figure 12. Object not found.

- The same result we will get for > npm -v
- Now we download
  - <https://nodejs.org>
    - 11.0.0 is the latest as of October of 2018



```
Administrator: Windows PowerShell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\Backup> node -v
v11.0.0
PS C:\Users\Backup> npm -v
6.4.1
PS C:\Users\Backup>
```

Figure 13. Latest versions of node and NPM.

- Now download the Truffle Framework
  - <https://truffleframework.com>

```
Administrator: Windows PowerShell
PS C:\Users\Backup>
PS C:\Users\Backup> npm uninstall -g truffle
up to date in 0.528s
PS C:\Users\Backup> npm install -g truffle@4.0.4
C:\Users\Backup\AppData\Roaming\npm\truffle -> C:\Users\Backup\AppData\Roaming\npm\node_modules\truffle\build\cli.bundle
d.js
+ truffle@4.0.4
added 92 packages from 318 contributors in 36.187s
PS C:\Users\Backup> truffle version
Truffle v4.0.4 (core: 4.0.4)
Solidity v0.4.18 (solc-js)
PS C:\Users\Backup>
```

Figure 14. Truffle installation process.

Now we will install a program script editor. We are using the opensource editor Atom for the CommLedger.

- <https://github.com/atom/atom/releases/tag/v1.23.2>
- Download AtomSetup-x64.exe

```
Administrator: Windows PowerShell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\Backup> atom -v
Atom : 1.23.2
Electron: 1.6.15
Chrome : 56.0.2924.87
Node : 7.4.0
PS C:\Users\Backup> apm -v
npm 1.10.12
npm 1.10.10
atom 1.23.2
python 2.7.15
visual studio
PS C:\Users\Backup>
```

Figure 15. Installed Blockchain script editor for CLTN/CLN.

Because we are using the base cryptocurrency Ethereum for our CommLedger, we will install the language packages for it using the Atom Package Manager (APM).

- > apm install language-ethereum

Now we install Git for version control of our CommLedger Network programming.

- <https://git-scm.com/download/win>

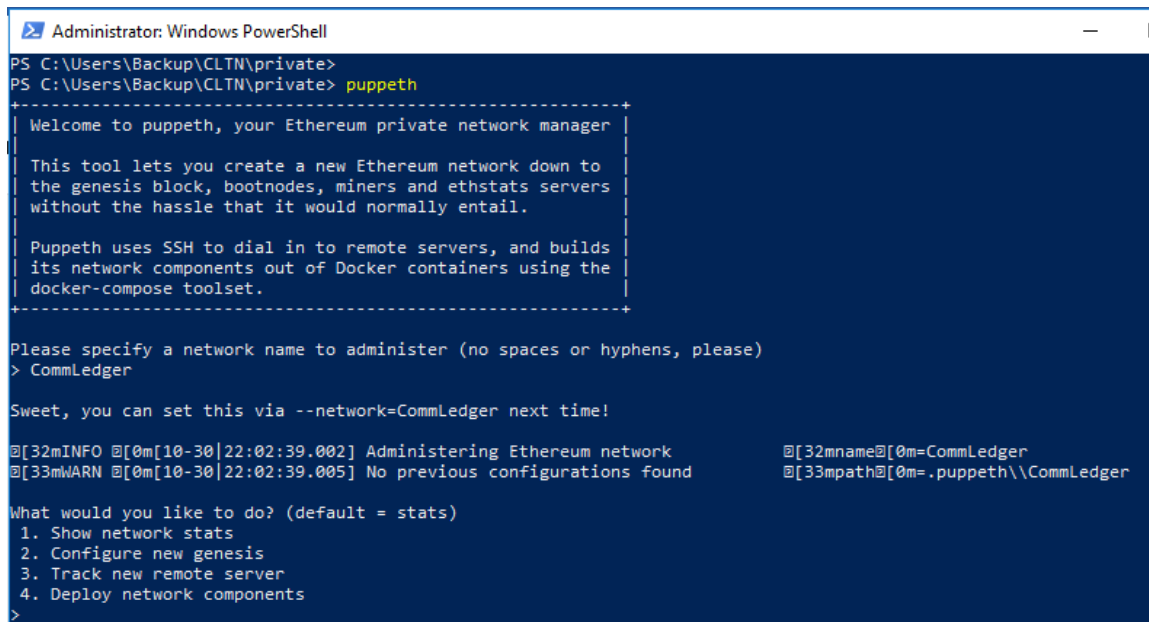
This is where our CommLedger TestNet (CLTN) is now completed. Next we will start the CLTN private node using Ethereum as the base Cryptocurrency. The use of CommLedger with a Crypto is the prime differentiator from Hyperledger.

## CommLedger Test-Net Operational Overview

Now let us look at a running CLTN Ethereum based network. The commands are applied using Windows PowerShell alongside Atom to view the ADAM block creation as the genesis block of Ethereum Blockchain for asset transfer between more than one user.

- Open Windows Power Shell
  - > mkdir -p CLTN/private
  - > puppeth

Because we are using Ethereum as the base cryptocurrency for our CLTN to transact assets to and from a sender-receiver base, we use Geth, the tool we have installed, which includes “Puppeth” to generate the genesis block. Because we are implementing CommLedger, we will name our test network for development purpose, as: CommLedger



```
Administrator: Windows PowerShell
PS C:\Users\Backup\CLTN\private>
PS C:\Users\Backup\CLTN\private> puppeth
-----+-----
| Welcome to puppeth, your Ethereum private network manager |
|                                                             |
| This tool lets you create a new Ethereum network down to  |
| the genesis block, bootnodes, miners and ethstats servers |
| without the hassle that it would normally entail.          |
|                                                             |
| Puppeth uses SSH to dial in to remote servers, and builds |
| its network components out of Docker containers using the  |
| docker-compose toolset.                                    |
|-----+-----|
Please specify a network name to administer (no spaces or hyphens, please)
> CommLedger

Sweet, you can set this via --network=CommLedger next time!

@[32mINFO @[0m[10-30|22:02:39.002] Administering Ethereum network          @[32mname@[0m=CommLedger
@[33mWARN @[0m[10-30|22:02:39.005] No previous configurations found        @[33mpath@[0m=.puppeth\CommLedger

What would you like to do? (default = stats)
1. Show network stats
2. Configure new genesis
3. Track new remote server
4. Deploy network components
>
```

Figure 16. CommLedger Test Network Initiation.

## CommLedger Differentiator

As Hyperledger does not work with Proof of Work or Proof of Stake, this is where CommLedger comes into the picture by using Proof of Work and later by adding Proof of Permission to transact assets between two organizations. The CLTN can be configured using Network ID 4224, which is the Kovan faucet. The Kovan network faucets provide Kovan Ether to assist CommLedger in deploying and testing smartcontracts on the Kovan network. This allows CommLedger to add the Proof of Permission TALA key that is generated separately by the CLTN/CLN permitted users to use the faucets for preventing malicious actors from obtaining large amounts of Ether in case CommLedger users are using Ethereum as the base cryptocurrency.

```

[32mINFO [0m[10-30|22:02:39.002] Administering Ethereum network [32mname[0m=CommLedger
[33mWARN [0m[10-30|22:02:39.005] No previous configurations found [33mpath[0m=.puppeth\CommLedger

What would you like to do? (default = stats)
 1. Show network stats
 2. Configure new genesis
 3. Track new remote server
 4. Deploy network components
> 2

Which consensus engine to use? (default = clique)
 1. Ethash - proof-of-work
 2. Clique - proof-of-authority
> 1

Which accounts should be pre-funded? (advisable at least one)
> 0x

Specify your chain/network ID if you want an explicit one (default = random)
> 4224
[32mINFO [0m[10-30|22:20:47.699] Configured new genesis block

What would you like to do? (default = stats)
 1. Show network stats
 2. Manage existing genesis
 3. Track new remote server
 4. Deploy network components
>

```

Figure 17. CommLedger Successful Initiation.

```

> 2
Which file to save the genesis into? (default = CommLedger.json)
>
@[32mINFO @[0m[10-30|22:24:13.901] Exported existing genesis block

What would you like to do? (default = stats)
1. Show network stats
2. Manage existing genesis
3. Track new remote server
4. Deploy network components
> █

```

Figure 18. CommLedger Ethereum genesis block created.

- Now Press CTRL + C to exit the Windows Power Shell

Each CLTN User is provided a “Proof of Permission Tiered Asynchronous Locking Algorithm” to generate the (POP-TALA) key. The program is provided in the addendum. The CLTN User needs to run “POP-TALA.exe” within their organizational domain (e.g., VPN) to generate the TALA key to add in CommLedger.JSON,

```

C:\Users\Backup\AppData\Local\Temp\Temp1_PoPTala.zip\PoPTala.exe

***** Welcome to the TALA Key Generator App *****

Please Enter the Input:

***** Welcome to the TALA Key Generator App *****

Please Enter the Input: CommLedger use for TALA Key Gen.

The SHA256 hash of "CommLedger use for TALA Key Gen." is: f8cf6036e75084ec2d17db6d1e83cda18e9f501f3b54394ec74a0f354bfa70e3.

Do you want to save this hash value into a file?
Please Enter your decision [Y/N] : █

Do you want to save this hash value into a file?
Please Enter your decision [Y/N] : y

You can provide a Location in your device to save the Hash value into a Text file.
Sample Location Input - E:\Folder1 (or) D:\Folder1\Fodler2
Note: If there is already a file with name "HashFile" in the location, it will be overwritten.

Please Enter the location: C:\Users\Backup\CLTN\private

Saving Hash File to C:\Users\Backup\CLTN\private\HashFile.txt

***** Hash File has been saved *****

Press any key to close this application.

```

Figure 19. Proof of Permission TALA key gen process to add in CommLedger.JSON.

- The given Figure 20 provides a snapshot view of CommLedger.JSON. The complete JSON file is available in the Addendum.

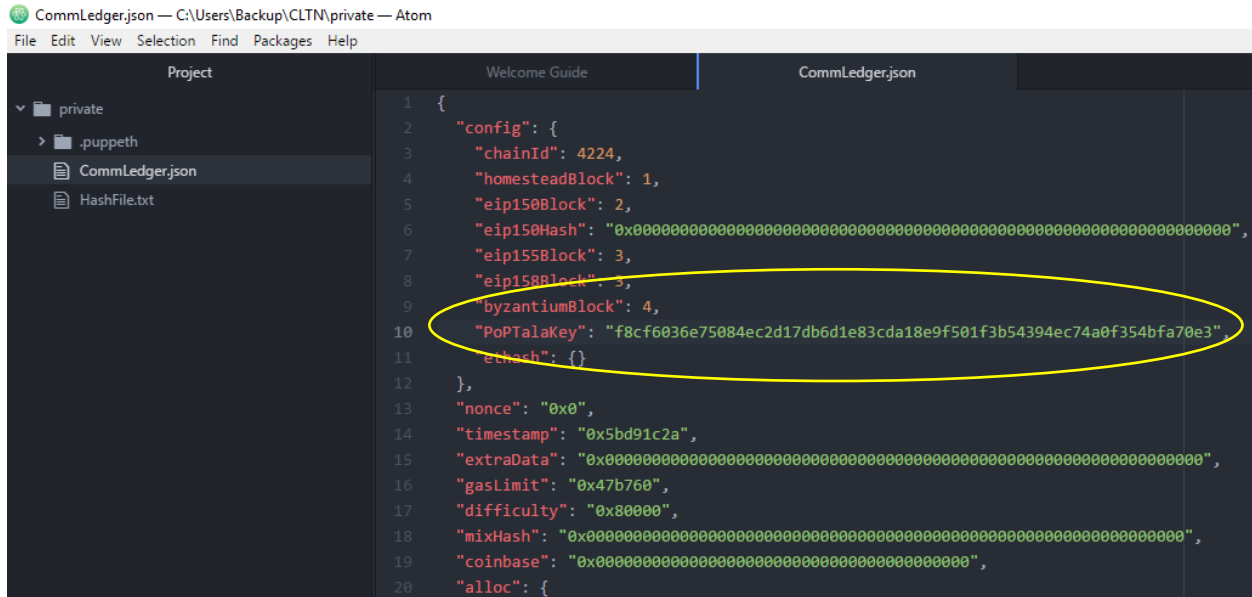


Figure 20. CommLedger.JSON Snapshot.

- Now we open Windows Power Shell and run the following command with POP-TALA key within the config of CommLedger.JSON
- `geth --datadir . init .\CommLedger.json`

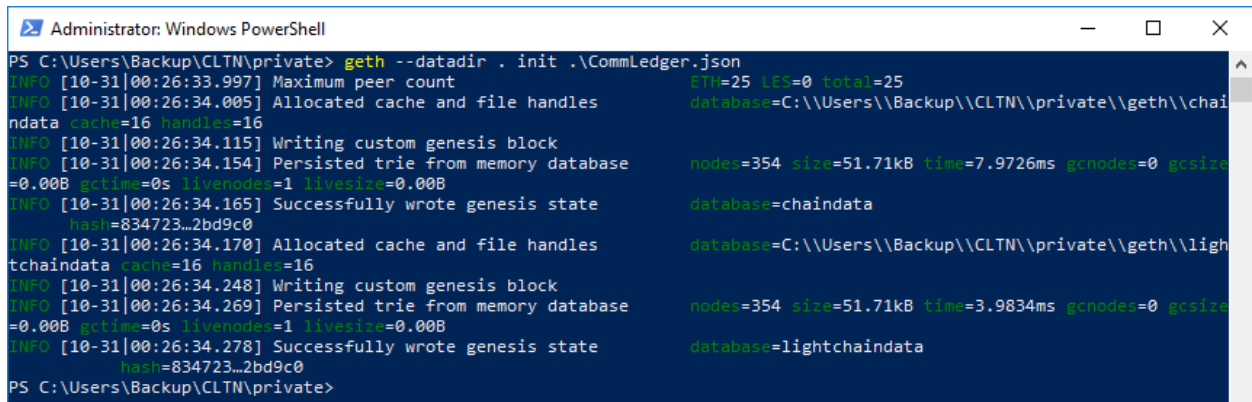


Figure 21. Writing of the custom genesis block with POP-TALA key completed.

This completes the process of starting a permissioned Blockchain using the CLTN used for transactions between two organizations. The section provides detail of the creation of organizational users, as well as miners. This is the differentiating point that CommLedger has associated miners, whereas Hyperledger does not have any associated Miners. This genesis block contains a POP-TALA key and within CLTN/CLN it is used as an ADAM block.

### CommLedger Account Creation Process

Because we are using the standard Geth command structure with the Proof of Permission protocol using the Tiered Asynchronous Locking Algorithm we now need to have two accounts named “Sender” and “Receiver” as well as “Miner” to mine the blocks from the aforementioned ADAM block at the time of the CommLedger.JSON configuration file.

- Open Windows Power Shell:
  - > geth --datadir . account new
  - Each new account created in this CommLedger will have a private key in addition to the POP-TALA key that the CLTN user already has as an extra Blockchain security layer. This private key is also protected by a Passphrase.
  - Passphrase: cltn1234

```
PS C:\Users\Backup\CLTN\private> geth --datadir . account new
INFO [10-31|13:48:03.396] Maximum peer count          ETH=25 LES=0 total=25
Your new account is locked with a password. Please give a password. Do not forget this password.
Passphrase:
Repeat passphrase:
Address: {be3802142f492326d4514c8d9860daa8ae96ced8}
PS C:\Users\Backup\CLTN\private> █
```

Figure 22. CommLedger new account generation.

- We will run this account twice more to create two more accounts.

```

PS C:\Users\Backup\CLTN\private> geth --datadir . account new
INFO [10-31|13:48:03.396] Maximum peer count ETH=25 LES=0 total=25
Your new account is locked with a password. Please give a password. Do not forget this password.
Passphrase:
Repeat passphrase:
Address: {be3802142f492326d4514c8d9860daa8ae96ced8}
PS C:\Users\Backup\CLTN\private> geth --datadir . account new
INFO [10-31|13:57:15.694] Maximum peer count ETH=25 LES=0 total=25
Your new account is locked with a password. Please give a password. Do not forget this password.
Passphrase:
Repeat passphrase:
Address: {cd8135ce27c215406a5909abee2ff465574dab0b}
PS C:\Users\Backup\CLTN\private> geth --datadir . account new
INFO [10-31|13:57:33.852] Maximum peer count ETH=25 LES=0 total=25
Your new account is locked with a password. Please give a password. Do not forget this password.
Passphrase:
Repeat passphrase:
Address: {ac601e0b459f6c2dce31c99d6ffc1751f3dc2245}
PS C:\Users\Backup\CLTN\private>

```

Figure 23. Creation of three CommLedger accounts.

- Test network for CommLedger now has three accounts created, as seen in Figure 4.14.

```

Administrator: Windows PowerShell
PS C:\Users\Backup\CLTN\private> ls .\keystore\

Directory: C:\Users\Backup\CLTN\private\keystore

Mode                LastWriteTime         Length Name
----                -
-a----            10/31/2018   1:54 PM             491 UTC--2018-10-31T18-54-49.806523300Z--be3802142f492326d4514c8d9860daa8ae96ced8
-a----            10/31/2018   1:57 PM             491 UTC--2018-10-31T18-57-25.667581400Z--cd8135ce27c215406a5909abee2ff465574dab0b
-a----            10/31/2018   1:57 PM             491 UTC--2018-10-31T18-57-45.509489100Z--ac601e0b459f6c2dce31c99d6ffc1751f3dc2245

```

Figure 24. The keystore to obtain CLTN accounts data storage.

- Now let's look at the account index, where account 0 (Zero) will get rewards in terms of reward permitted in the allocation section of CommLedger.JSON
  - > geth --datadir . account list

```

Administrator: Windows PowerShell
PS C:\Users\Backup\CLTN\private> geth --datadir . account list
INFO [10-31|14:33:36.158] Maximum peer count ETH=25 LES=0 total=25
Account #0: {be3802142f492326d4514c8d9860daa8ae96ced8} keystore://C:\Users\Backup\CLTN\private\keystore\UTC--2018-10-31T18-54-49.806523300Z--be3802142f492326d4514c8d9860daa8ae96ced8
Account #1: {cd8135ce27c215406a5909abee2ff465574dab0b} keystore://C:\Users\Backup\CLTN\private\keystore\UTC--2018-10-31T18-57-25.667581400Z--cd8135ce27c215406a5909abee2ff465574dab0b
Account #2: {ac601e0b459f6c2dce31c99d6ffc1751f3dc2245} keystore://C:\Users\Backup\CLTN\private\keystore\UTC--2018-10-31T18-57-45.509489100Z--ac601e0b459f6c2dce31c99d6ffc1751f3dc2245
PS C:\Users\Backup\CLTN\private>

```

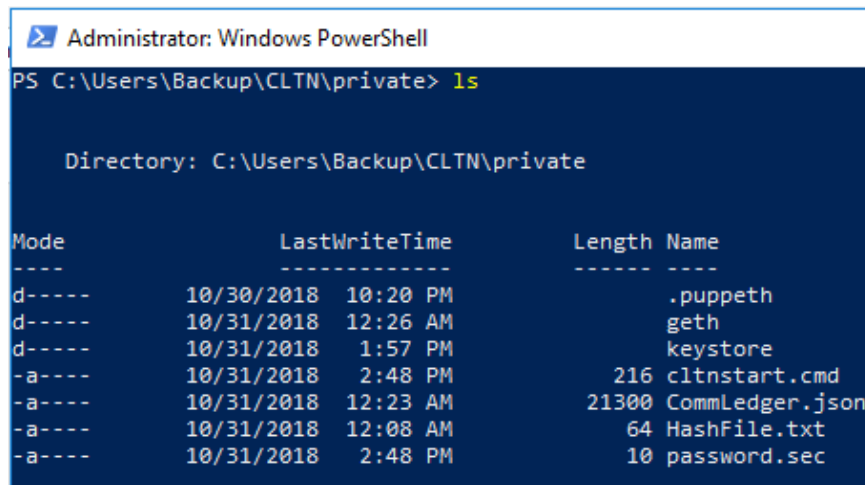
Figure 25. Private CommLedger Test-Network with accounts index list



## CommLedger Network Initiation (Test Start)

We have successfully completed all related processes of a CLTN environment. Now we will write a script using our opensource text editor Atom to get the CLTN started.

- Open Windows Power Shell
  - > atom cltnstart.cmd
- The entire structure of the following commands should be in one straight line, otherwise all the next lines will be ignored by the CommLedger Test-Network
  - `geth --networkid 4224 --mine --minerthreads 1 --datadir "./" --nodiscover --rpc --rpcport "8545" --port "30303" --rpccorsdomain "*" --nat "any" --rpcapi eth,web3,personal,net --unlock 0 --password ./password.sec`
  - Now create a new file using the Atom menu and name it “password.sec”, where we have to enter the passphrase we set for our Miner Node: cltn1234
  - Save this file and open Windows PowerShell, type “ls” for the list of files.



```
Administrator: Windows PowerShell
PS C:\Users\Backup\CLTN\private> ls

Directory: C:\Users\Backup\CLTN\private

Mode                LastWriteTime         Length Name
----                -
d-----           10/30/2018  10:20 PM             .puppeth
d-----           10/31/2018  12:26 AM             geth
d-----           10/31/2018   1:57 PM             keystore
-a-----           10/31/2018   2:48 PM             216 cltnstart.cmd
-a-----           10/31/2018  12:23 AM            21300 CommLedger.json
-a-----           10/31/2018  12:08 AM             64 HashFile.txt
-a-----           10/31/2018   2:48 PM             10 password.sec
```

Figure 26. The CLTN Tool Box.

Now let’s start our CLTN to generate the blocks for Blockchain. At this step the CLTNTest Network needs to be restarted and our miner starts mining blocks for any potential

Blockchain Transactions that the CLTN users can do. Now we look at the process of attaching the Java Script console to our CommLedger Blockchain. These processes are beyond the scope of this research and are only provided here for reviewers' future test needs.

```

Administrator: Windows PowerShell
PS C:\Users\Backup\CLTN\private> atom
PS C:\Users\Backup\CLTN\private> .\cltnstart.cmd

C:\Users\Backup\CLTN\private>geth --networkid 4224 --mine --minerthreads 1 --datadir "." --nodiscover --rpc --rpcport "
8545" --port "30303" --rpccorsdomain "*" --nat "any" --rpcapi eth,web3,personal,net --unlock 0 --password ./password.sec

INFO [10-31|15:24:55.200] Maximum peer count                               ETH=25 LFS=0 total=25
INFO [10-31|15:24:55.468] Starting peer-to-peer node                               instance=Geth/v1.8.17-stable-8bbe7207/windows-amd64/g
o1.11.1
INFO [10-31|15:24:55.473] Allocated cache and file handles                       database=C:\Users\Backup\CLTN\private\geth\chai
ndata cache=768 handles=1024
INFO [10-31|15:24:55.919] Initialised chain configuration                            config="{ChainID: 4224 Homestead: 1 DAO: <nil> DAOSup
port: false EIP150: 2 EIP155: 3 EIP158: 3 Byzantium: 4 Constantinople: <nil> Engine: ethash}"
INFO [10-31|15:24:55.937] Disk storage enabled for ethash caches                    dir=C:\Users\Backup\CLTN\private\geth\ethash ca
unt=3
INFO [10-31|15:24:55.954] Disk storage enabled for ethash DAGs                      dir=C:\Users\Backup\AppData\Ethash             co
unt=2
INFO [10-31|15:24:55.973] Initialising Ethereum protocol                            versions="[63 62]" network=4224
INFO [10-31|15:24:56.010] Loaded most recent local header                          number=0 hash=834723...2bd9c0 id=524288 age=17h18m38s
INFO [10-31|15:24:56.015] Loaded most recent local full block                     number=0 hash=834723...2bd9c0 id=524288 age=17h18m38s
INFO [10-31|15:24:56.018] Loaded most recent local fast block                     number=0 hash=834723...2bd9c0 id=524288 age=17h18m38s
INFO [10-31|15:24:56.055] Regenerated local transaction journal                   transactions=0 account=0
INFO [10-31|15:24:56.129] Starting P2P networking
INFO [10-31|15:24:56.376] RLPx listener up                                       self="enode://2cf436891e5b78225852466973efcbc2eb683a4
d9463b1f487895e1ef5e0c75628fd18d3e7f6b01e5045b159c237487c867f472100916e3cc031b9b6b94f47f7@140.186.60.217:30303?discport=
0"
INFO [10-31|15:24:56.406] IPC endpoint opened                                       uri=\\\\.\\pipe\geth.ipc
INFO [10-31|15:24:56.444] HTTP endpoint opened                                     uri=http://127.0.0.1:8545 cors=* vhosts=localhost
INFO [10-31|15:24:56.455] Mapped network port                                     proto=tcp extport=30303 intport=30303 interface=NAT-P
MP(192.168.0.1)
WARN [10-31|15:24:56.477] -----
WARN [10-31|15:24:56.479] Referring to accounts by order in the keystore folder is dangerous!
WARN [10-31|15:24:56.481] This functionality is deprecated and will be removed in the future!
WARN [10-31|15:24:56.485] Please use explicit addresses! (can search via `geth account list`)
WARN [10-31|15:24:56.488] -----
INFO [10-31|15:24:57.971] Unlocked account                                       address=0xBE3802142F492326d4514c8d9860Daa8AE96CeD8
INFO [10-31|15:24:57.975] Transaction pool price threshold updated                price=1000000000
INFO [10-31|15:24:57.977] Updated mining threads                                  threads=1
INFO [10-31|15:24:57.979] Transaction pool price threshold updated                price=1000000000
INFO [10-31|15:24:57.982] Etherbase automatically configured                    address=0xBE3802142F492326d4514c8d9860Daa8AE96CeD8
INFO [10-31|15:24:58.021] Commit new mining work                                  number=1 sealhash=9a9ed3...e9af6e uncles=0 txs=0 gas=0
Fee=0 elapsed=36.994ms
INFO [10-31|15:24:58.425] Got interrupt, shutting down...
INFO [10-31|15:24:58.472] HTTP endpoint closed                                     uri=http://127.0.0.1:8545
INFO [10-31|15:24:58.475] IPC endpoint closed                                     endpoint=\\\\.\\pipe\geth.ipc
INFO [10-31|15:24:58.478] Blockchain manager stopped
INFO [10-31|15:24:58.481] Stopping Ethereum protocol
INFO [10-31|15:24:58.483] Ethereum protocol stopped

```

Figure 27. Successful deployment of CLTN.

- Open a new Windows PowerShell by right clicking on the PowerShell icon.
  - > geth attach ipc:\\.pipe\geth.ipc



- The default duration to open the account using the Java Script console will be 10 minutes.

```

Administrator: Windows PowerShell
PS C:\Users\Backup> geth attach ipc:\\.\pipe\geth.ipc
Welcome to the Geth JavaScript console!

instance: Geth/v1.8.17-stable-8bbe7207/windows-amd64/go1.11.1
coinbase: 0xbe3802142f492326d4514c8d9860daa8ae96ced8
at block: 273 (Wed, 31 Oct 2018 20:43:21 CDT)
datadir: C:\Users\Backup\CLTN\private
modules: admin:1.0 debug:1.0 eth:1.0 ethash:1.0 miner:1.0 net:1.0 personal:1.0 rpc:1.0 txpool:1.0 web3:1.0

> eth.sendTransaction({from:eth.coinbase, to:eth.accounts[2], value:web3.toWei(20,"ether")})
"0x1f6e98118f99c01c307c152ba46ce3d68c7558ac10c187f205eeb3bd3a8c8f57"
>

```

Figure 29. CommLedger Test Network successful transfer of tokens.

As shown in Figure 4.20, we can see that 20 Ethers have successfully been transferred from the Coinbase using CLTN cryptocurrency, which we have coined the term UNDCoin (UN-Digitization Coin) from the Coinbase where the blocks are being mined and Ether is submitted to the Coinbase account to be used when needed for a Blockchain transaction.

```

INFO [10-31|20:44:09.543] Successfully sealed new block          number=275 sealhash=e24ab8...e4df77 hash=70bf89...f5ccdd
elapsed=7.208s
INFO [10-31|20:44:09.550] @@ block reached canonical chain          number=268 hash=f93496...0b6646
INFO [10-31|20:44:09.554] @@ mined potential block                  number=275 hash=70bf89...f5ccdd
INFO [10-31|20:44:09.558] Commit new mining work                    number=276 sealhash=0e6307...f1db82 uncles=0 txs=0 gas=
0 fee=0 elapsed=7.999ms
INFO [10-31|20:44:14.468] Successfully sealed new block          number=276 sealhash=0e6307...f1db82 hash=993a41...83108f
elapsed=4.918s
INFO [10-31|20:44:14.473] @@ block reached canonical chain          number=269 hash=0b77a6...0d45db
INFO [10-31|20:44:14.477] @@ mined potential block                  number=276 hash=993a41...83108f
INFO [10-31|20:44:14.481] Commit new mining work                    number=277 sealhash=de332b...7e97cd uncles=0 txs=0 gas=
0 fee=0 elapsed=7.000ms
INFO [10-31|20:44:14.532] Submitted transaction                    fullhash=0x1f6e98118f99c01c307c152ba46ce3d68c7558ac10
c187f205eeb3bd3a8c8f57 recipient=0xac601e0b459f6C2dCE31c99d6fFC1751f30C2245
INFO [10-31|20:44:17.474] Commit new mining work                    number=277 sealhash=2a3138...82a97e uncles=0 txs=1 gas=
21000 fee=2.1e-05 elapsed=0s
INFO [10-31|20:44:26.035] Successfully sealed new block          number=277 sealhash=2a3138...82a97e hash=3198b6...d41244
elapsed=8.561s
INFO [10-31|20:44:26.042] @@ block reached canonical chain          number=270 hash=6e9861...deb8c6
INFO [10-31|20:44:26.047] @@ mined potential block                  number=277 hash=3198b6...d41244
INFO [10-31|20:44:26.051] Commit new mining work                    number=278 sealhash=fd85d6...c45d0f uncles=0 txs=0 gas=
0 fee=0 elapsed=8.997ms

```

Figure 30. Sealing the Block process depiction.

In this figure, we can see a submitted transaction and a successfully sealed block in the Ledger. Because we performed two transactions, the next figure shows the balances in each account.



## **CHAPTER V**

### **CONCLUSION AND FUTURE WORK**

As described in this dissertation, though there are advantages in using Blockchain and Distributed Ledger Technology (DLT) based systems, there are yet many realistic problems which must be solved such as organizational transactional structures to align according to key-value pair structures and transaction transitioning methodologies for both within and without the organization enterprise system architectures. This type of transactional transitions with several Blockchains needs a faster cryptic key generation, as data blocks are rapidly moving between sender and receiving entities. This research sheds light on a few of the issues and the proposed CommLedger as a solution dealing with DLT oriented business solutions.

#### **Conclusion**

CommLedger a permissioned Blockchain distributed ledger technology has been proposed in this doctoral dissertation. The proposition is a combination of several techniques for the potential use of CommLedger that, incorporates smartcontract as a service in an organizational virtual private cloud. This is a distinctiveness of seamlessly transmitting secured and immutable data within permissioned Blockchain. Once the proof of concept within the organization is completed, it is deployed as a CLN. The exponential growth of Blockchain has provided clear evidence of the establishment of decentralized cryptocurrencies and the adaptation of permissioned distributed ledger technology This dissertation also presents novel

propositions of CLN (CommLedger) using PoP (Proof of Permission), ADAM (Authenticated Data Acceptance Marker) block and TALA (Tiered Asynchronous Locking Algorithm) key.

### CommLedger Differentiating Factors

Table 1 provides a comparison between two famous cryptocurrencies and two DLTs based on Blockchain. The information sheds light on the aspects these artifacts utilization by individual users and industrial adaptation.

Table 1. The use of Blockchain in cryptocurrencies and DLT.

| Type   | Bitcoin   | Ethereum   | Hyperledger  | CommLedger  |
|--|---|--|--|---|
| <b>Language Characteristics and Network System</b> | Bitcoin is written in C++. It is decentralized and is a jurisdiction-less entity. It has a peer-to-peer network without the need for intermediaries.                        | Ethereum is written in the Turing complete language that includes several different programming languages (C++, Ruby, Rust, Go, Python, Java, JavaScript, Solidity). | Hyperledger uses "Chaincode". A Chaincode typically handles business logic agreed to by members of the network, so it may be considered as a smart contract. These Chaincodes are written in Golang, a programming language created by Google. It is decentralized | CommLedger uses a "SmartContract" for its Cryptocurrency UNDCoin, and a "Chaincode" can be incorporated to conduct business asset transfer. The languages used are C++, JavaScript. |
| <b>Basic Build</b>                                 | BTC   | Ether (ETC, ETH)   |  | UNDCoin   |
| <b>Blockchain Network</b>                          | It has peer-to-peer network without the need for intermediaries. It is anti-bank - rather than sending funds via their bank, BTC holders can send transfers to one another. | Ethereum Blockchain not only stores the transaction list of the Blockchain but is also the most recent state of the network  | Hyperledger is a software for people to develop their own personalized blockchains tending to the needs of their businesses.   | Proof of Permission (PoP), TALA key generation and the creation of an ADAM block are required for both permissioned and un-permissioned Blockchain adaptations.                     |
| <b>Supply Style</b>                                | They have a fixed supply and are deflationary.  | Inflationary.  |  | Permissioned Blockchain will bind organizations to have a Hard Cap. Un-permissioned crypto tokens will have a Hard Cap of 28 Million.   |

Table 1 cont.

| Type                  | Bitcoin   | Ethereum  | Hyperledger   | CommLedger  |
|-----------------------|---|---|---|---|
| <b>Consensus</b>      | Proof of Work (PoW)<br>Reward is given to Miners  | Proof of Work (PoW) & Proof of Stake (PoS)<br>In PoS there is a Transaction Fee given to Miners   | No Consensus or the use of Agreement Protocol using Practical Byzantine Fault Tolerance (PBFT)  | Proof of Permission (PoP), Transaction Fee for Miners   |
| <b>Scaling</b>        | Bitcoin's blocks contain the transactions on the Bitcoin network. The on-chain transaction processing capacity of the Bitcoin network is limited by the average block creation time of 10 minutes and the block size limit. These jointly constrain the network's throughput. The transaction processing capacity maximum is estimated between 3.3 and 7 transactions per second. | 15 Transactions per second.   | Hyperledger has a fine-grained control over consensus and restricted access to transactions which results in improved performance scalability and privacy.  | CommLedger allows the Permissioned Blockchain to construct their own Transaction Processing time. UNDCoin however has a speed of Mining for the Block Authentication throughput between (in CLTN) 2 to 5 per second.    |
| <b>Liquidity</b>      | Bitcoin is one of the most liquid cryptocurrencies with 42.51% dominance in the overall crypto market.  | It is on more than 400 Crypto-Exchanges and is quite liquid.  | Hyperledger Fabric enables real-time visibility on the liquidity of Nostro accounts, easing reconciliation and allowing liquidity savings.  | UNDCoin will be quite liquid.   |
| <b>Implementation</b> | Bitcoin Core considered Bitcoin's reference implementation. Bitcoin Core serves as a bitcoin node (the set of which form the bitcoin network) and provides a bitcoin wallet which fully verifies payments.  | The Ethereum Virtual Machine (EVM) is the runtime environment for smart contracts in Ethereum. Smartcontracts are high-level programming abstractions compiled down to an EVM bytecode. | Hyperledger solves performance scalability and privacy issues by application of permissioned mode of operation and fine-grained access control. Furthermore, the modular architecture allows Hyperledger to be customized to a multitude of applications, analogous to a toolbox. | CommLedger Permissioned Blockchain allows RBAC, DBAC fine-grained control. Un-permissioned cryptocurrency will allow complete autonomous control for entities using UNDCoin for their everyday Crypto-Exchange trading. |



## **Future Work**

We will be looking into DLT evidence theory for our future work, which is so far unheard of within the Blockchain for business community. There have been several advances in computational performance in terms of both traditional, and parallel programming aspects in last three decades. the latest advent of Permissioned Blockchain has reintroduced Solidity and Go programming to bring us the design and development of Blockchain computing simulation tools using Ganesh and Hyperledger.

This doctoral dissertation illuminates on how an organization can step into the era of permissioned Blockchain using CommLedger and associated novel methodologies, by having the right strategy using the provided TALA. Any strategy, which is good in nature and well thought out, while in the phase of designing, cannot succeed, if poorly executed. It is vital that the organizational leadership has a sense of how to execute the strategy. The very beginning for any organization is to figure out collectively, what the strategic permissioned Blockchain challenges are that they can face, and which challenge is the biggest. In other words, they don't implement CommLedger, the permissioned Blockchain strategy in this dissertation, the leaders will need to determine what factors critically impact the organization, in terms of marketplace competitiveness.

Inarguably, we must account for uncertainty even in Blockchain streams, when corrupted data can also stream, or the chains can get broken due to a fork, such as an Ethereum fork resulting in ETH and ETC as two parallel cryptocurrencies. Traditionally we have seen that evidence theory has been utilized to measure uncertainty in terms of the uncertain measures of belief and plausibility based on whatever data we have. It is also evidenced in the emerging distributed ledger technology that uses computing communities where Cloud computing must

provide a flexible and scalable infrastructure for Blockchain miners and users to grow beyond contemporary borders not only for organizations, but also the users' everyday experience with a Blockchain.

The challenges of the present Blocknomics climate can be resolved by Blockchain designers, developers and decision makers, by aligning business needs in terms of using a crypto token either by launching a STO (Security Token Offering), or using a digital coin, such as UNDCoin (UN-Digitization Coin) as a service component to improve service to their peer organizations and/or consumers.

The decision analytics based on Big Data analytics [30] can also be associated with CommLedger is secured by using machine learning and evidence theory algorithms that can give an organization the uniqueness of its ideas, which they can embed in their decision support systems. Decision analytics recommended for the organizations adopting CommLedger are the use of machine learning and cloud computing.

## APPENDIX

### POP-TALA keyGen Algorithm

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Security.Cryptography;
using System.Text;
using System.Threading.Tasks;
namespace PoPTala
{
    public class Program
    {
        static void Main(string[] args)
        {
            try
            {
                Console.WriteLine("\n***** Atif Farid Mohammad CommLedger
                Welcome to the TALA key Generator App, each CLTN User is to download this
                application in their Laptop/Computer to Generate their POP-TALA_Key
                *****");
                Console.Write("\nPlease Enter the Input: ");
                var source = Console.ReadLine();
                string hash = string.Empty;
                using (SHA256 sha256Hash = SHA256.Create())
                {
                    hash = GetHash(sha256Hash, source);
                    Console.WriteLine();
                    Console.WriteLine($"The SHA256 hash of \"{source}\" is: {hash}.\n");
                }
                Console.WriteLine("Do you want to save this hash value into a file?");
                Console.Write("Please Enter your decision [Y/N] : ");
                string saveDecision = Console.ReadLine();
                if (saveDecision.ToUpper() == "Y")
                {
                    Console.WriteLine("\nYou can provide a Location in your device to save the
                    Hash value into a Text file.");
                    Console.WriteLine("Sample Location Input - E:\\Folder1 (or)
                    D:\\Folder1\\Fodler2 ");
                }
            }
        }
    }
}
```

```

        Console.WriteLine("Note: If there is already a file with name \"HashFile\" in the
        location, it will be overwritten.");
        Console.Write("\nPlease Enter the location: ");
        string location = Console.ReadLine(); //"E:\\Videos\\Dance Videos";

        if(location == string.Empty || location == "")
        {
            Console.WriteLine("\nLooks Like you haven't provided any location.");
            location = "C:\\";
        }
        string fileName = "HashFile.txt";
        string path = Path.Combine(location, fileName);
        File.WriteAllText(path, hash);
        Console.WriteLine("\nSaving Hash File to {0}", path);
        Console.WriteLine("\n\n***** Hash File has been saved
*****");
        Console.WriteLine("\n\nPress any key to close this application.");
        Console.ReadKey();
    }
    else if(saveDecision.ToUpper() == "N")
    {
        Console.WriteLine("\n\nPress any key to close this application.");
        Console.ReadKey();
    }
    else
    {
        Console.WriteLine("\n\nLooks Like you have selected a different option");
        Console.WriteLine("\nThanks for using the app");
        Console.WriteLine("\n\nPress any key to close this application.");
        Console.ReadKey();
    }
}
catch(Exception Ex)
{
    Console.WriteLine("\n***** Expection Occurred *****\n");
    Console.WriteLine("Exception: {0}", Ex.Message);
    Console.WriteLine("\n\nPress any key to close this application and try again by
providing correct values");
    Console.ReadKey();
}
}
private static string GetHash(HashAlgorithm hashAlgorithm, string input)
{
    // Convert the input string to a byte array and compute the hash.
    byte[] data = hashAlgorithm.ComputeHash(Encoding.UTF8.GetBytes(input));

```

```

// Create a new StringBuilder to collect the bytes
// and create a string.
var sBuilder = new StringBuilder();

// Loop through each byte of the hashed data
// and format each one as a hexadecimal string.
for (int i = 0; i < data.Length; i++)
{
    sBuilder.Append(data[i].ToString("x2"));
}

// Return the hexadecimal string.
return sBuilder.ToString();
}
}
}

```

### CommLedger JSON

```

{
  "config": {
    "chainId": 4224,
    "homesteadBlock": 1,
    "eip150Block": 2,
    "eip150Hash":
"0x0000000000000000000000000000000000000000000000000000000000000000",
    "eip155Block": 3,
    "eip158Block": 3,
    "byzantiumBlock": 4,
    "PoPTalaKey": "f8cf6036e75084ec2d17db6d1e83cda18e9f501f3b54394ec74a0f354bfa70e3",
    "ethash": {}
  },
  "nonce": "0x0",
  "timestamp": "0x5bd91c2a",
  "extraData":
"0x0000000000000000000000000000000000000000000000000000000000000000",
  "gasLimit": "0x47b760",
  "difficulty": "0x80000",
  "mixHash":
"0x0000000000000000000000000000000000000000000000000000000000000000",
  "coinbase": "0x000000000000000000000000000000000000000000000000",
  "alloc": {
    "0000000000000000000000000000000000000000000000000000000000000000": {
      "balance": "0x1"
    }
  }
}

```



## REFERENCES

- [1] Marko Vukolić. Rethinking Permissioned Blockchains. BCC '17 Proceedings of the ACM Workshop on Blockchain, Cryptocurrencies and Contracts Pages 3-7. 2017
- [2] G DeCandia, D Hastorun, M Jampani. Dynamo: Amazon's highly available key-value store. dl.acm.org ACM SIGOPS operating, 2007
- [3] Androulaki, E., Barger, A., Bortnikov, V. Hyperledger fabric: a distributed operating system for permissioned blockchains. EuroSys '18 Proceedings of the Thirteenth EuroSys Conference, April 2018
- [4] Kaiwen Zhang, Roman Vitenberg. Deconstructing Blockchains: Concepts, Systems, and Insights. Proceedings of the 12th ACM International Conference on Distributed and Event-based Systems. Pages 187-190. New Zealand, June 2018
- [5] Karanveer Singh Jhala, Rajvardhan Oak, Mrunmayee Khare, "Smart Collaboration Mechanism Using Blockchain Technology", Cyber Security and Cloud Computing (CSCloud)/2018 4th IEEE International Conference on Edge Computing and Scalable Cloud (EdgeCom) 2018 5th IEEE International Conference on, pp. 117-121, 2018.
- [6] Fabrice Benhamouda, Shai Halevi, Tzipora Halevi. Supporting Private Data on Hyperledger Fabric with Secure Multiparty Computation. IEEE International Conference on Cloud Engineering (IC2E). Orlando, FL, USA, May 2018
- [7] Linux Foundation Announces Hyperledger [online] Available: <https://www.Hyperledger.org/announcements/2016/02/09/linux-foundations-Hyperledger-project-announces-30-founding-members-and-code-proposals-to-advance-Blockchain-technology>
- [8] Shivam Bajpayi, Pedro Moreno-Sanchez, Donghang Lu, and Sihao Yin. Exploring Confidentiality Issues in Hyperledger Fabric Business Applications. The Summer Undergraduate Research Fellowship (SURF) Symposium, Aug 2018
- [9] Harish Sukhwani, José M. Martínez, Xiaolin Chang, Kishor S. Trivedi, Andy Rindos. Performance Modeling of PBFT Consensus Process for Permissioned Blockchain Network (Hyperledger Fabric). IEEE 36th Symposium on Reliable Distributed Systems (SRDS). October 2017
- [10] Chen.T,Li. X, Wang. Y, Chen. J, Li. Z, Luo. X, Ho. M, Zhang. A. An Adaptive Gas Cost Mechanism for Ethereum to Defend Against Under-Priced DoS Attacks. Information Security Practice and Experience. pp 3-24, ISPEC 2017

- [11] Ojamaa. A, Düüna.K. Assessing the security of Node.js platform. 2012 International Conference for Internet Technology and Secured Transactions. Dec, 2012
- [12] Benhamouda. F, Halevi. S, Halevi.T .Supporting Private Data on Hyperledger Fabric with Secure Multiparty Computation. 2018 IEEE International Conference on Cloud Engineering (IC2E), April 2018
- [13] A. E. Kosba, A. Miller, E. Shi, Z. Wen, C. Papamanthou, "Hawk: The blockchain model of cryptography and privacy-preserving smart contracts", 2016 IEEE Symposium on Security and Privacy, pp. 839-858, May 2016
- [14] Hyperledger Blockchain vs Ethereum Blockchain [online] Available: <https://etherworld.co/2017/10/01/Hyperledger-Blockchain-vs-Ethereum-Blockchain/>
- [15] Proof of Work vs Proof of Stake: Basic Mining Guide [online] Available: <https://blockgeeks.com/guides/proof-of-work-vs-proof-of-stake/>
- [16] What is Practical Byzantine Fault Tolerance? [online] Available: <https://blockonomi.com/practical-byzantine-fault-tolerance/>
- [17] D. Vujičić, D. Jagodić, & S. Randić, "Blockchain technology, bitcoin, and Ethereum: A brief overview," 2018 17th International Symposium INFOTEH-JAHORINA (INFOTEH), East Sarajevo, pp. 1-6, 2018.
- [18] H. Halpin, & M. Piekarska. Introduction to Security and Privacy on the Blockchain. Euro S&P 2017 - 2nd IEEE European Symposium on Security and Privacy, Workshops, Apr 2017, Paris, France. IEEE, Security and Privacy Workshops (EuroS&PW), 2017 IEEE European Symposium, pp.1-3, 2017.
- [19] S. Singh & N. Singh, "Blockchain: Future of financial and cyber security," 2016 2nd International Conference on Contemporary Computing and Informatics (IC3I), Noida, pp. 463-467, 2016.
- [20] M. Bartoletti, S. Lande, L. Pompianu, & A. Bracciali. A general framework for Blockchain analytics. SERIAL '17 Proceedings of the 1st Workshop on Scalable and Resilient Infrastructures for Distributed Ledgers. Article No. 7, 2017.
- [21] L. Bahack, Theoretical Bitcoin Attacks with less than Half of the Computational Power. IACR Cryptology ePrint Archive, 868. 2013.
- [22] Y-A. de Montjoye, E. Shmueli, S. S. Wang, & A. S. Pentland, "openpds: Protecting the privacy of metadata through safeanswers", PloS one, vol. 9, no. 7, pp. e98790, 2014.
- [23] Atif Farid Mohammad. Pre-Blockchain Semi/Un-Structured Data Storage. GSTF Journal on Computing (JoC), [S.I.], v. 6, n. 2, p. 6. ISSN 2010-2283, 2018



- [24] Z. Bao, W. Shi, D. He, & K.R Choo, IoTChain: A Three-Tier Blockchain-based IoT Security Architecture. CoRR, abs/1806.02008. 2018
- [25] M. Mihaylov, I. Rázo-Zapata, & A. Nowé, "NRGcoin - A Blockchain-based Reward Mechanism for Both Production and Consumption of Renewable Energy," in Transforming Climate Finance and Green Investment with Blockchains, ISBN: 978-0128144473, Elsevier, 2018.
- [26] Atif Farid Mohammad, Emanuel S Grant. Cloud Computing, SaaS, and SOA 3.0: A New Frontier. Cloud Computing and Virtualization 2010 International Conference, Singapore May 2010
- [27] Atif Mohammad, Emanuel Grant. Use of SOA 3.0 in Private Cloud Security Gateway Service Design: In the Era of Big Data. Proceedings of International Conference on Computer Games, Multimedia & Allied Technology (CGAT). 2014
- [28] Sumangali, K., Borra, L., Mishra, A.S. A Comprehensive review on the open source hackable text editor-ATOM. IOP Conf. Ser.: Mater. Sci. Eng. 263 042061, 2017
- [29] E. Ben Hamida, K. L. Brousmiche, H. Levard, & E. Thea. Blockchain for Enterprise: Overview, Opportunities and Challenges. The Thirteenth International Conference on Wireless and Mobile Communications (ICWMC 2017), France. 2017.
- [30] Hamid Mcheick, Atif Farid Mohammad. The Evident use of Evidence Theory in Big Data Analytics using Cloud Computing. 2014 IEEE 27th Canadian Conference on Electrical and Computer Engineering (CCECE). DOI: 10.1109/CCECE.2014.6901158