Blockchain for Securing Decentralized Cloud Storage

COMP 8045 - Major Project

Richard W. Gosse - A00246425
2018-12-07

Table of Contents

Table of Contents 1
1. Introduction 3
1.1. Student Background 3
1.2. Project Description 5
2. Body 7
2.1. Background 7
2.2. Project Statement 8
2.3. Possible Alternative Solutions 8
2.4. Chosen Solution 9
2.5. Details of Design and Development 10
2.5.1. Structure 10
2.5.2. Client Layer 11
2.5.3. Blockchain Layer 15
2.5.4. Storage Layer 18
2.6. User Guide / Setup & Installation 20
2.7. Testing Details and Results 22
2.7.1. Test Cases / Functional Requirements 22
2.7.2. Test Results 23
2.8. Implications of Implementation / Non-Functional Requirements 43
2.8.1. Extensibility, Coupling & Scope 43
2.8.2. Security 43
2.8.3. Reliability, Performance, Performance and Concurrency 43
2.9. Innovation 44
2.10. Complexity 45
2.10.1. Block chain technology 45
2.10.2. Distributed storage 45
2.10.3. Data and Privacy Protection 45
2.11. Future Enhancements 46
2.11.1. Blockchain Visualizer: 46
2.11.2. Transaction Batching: 46
2.11.3. Storage Minion Heartbeat and Load Balancer: 46
2.11.4. Automatic List of Peers 46

2.12. Timeline, Methodology and Milestones 47
2.12.1. Proposed Timeline 47
2.12.2. Actual Timeline 48
2.12.3. Timeline Progression and Adjustments 49
3. Conclusion and Lessons 51
4. Appendix 52
4.1. List of Figures 52
4.2. Typical Run Through 53
4.2.1. Client 53
4.2.2. Master Node 54
4.2.3. Minion Node 55
5. References 59
6. Change Log 61

1. Introduction

1.1. Student Background

PAGE REMOVED

PAGE REMOVED

1.2.Project Description

This project was created for the 9-credit practicum requiring approximately 405 hours of work. I worked alone on this project.

This project's overall goal was to create a working model for a decentralized, distributed block chain based secure 'cloud' file storage application. The addition of block chain methodology to network storage transactions will allow for network meta data, such as URLs and access rights, to be secured. Combining this with a focus on decentralization of the storage locations is a direction that many cloud providers are looking at. Recent laws such as those passed in the European Union create new challenges for those cloud providers as they seek to be in compliance and convince clients that they truly do have their best interests in mind. I was made aware of the degree of focus that many international companies are making into this area. The challenges that may be faced were explored by way of completing this project.

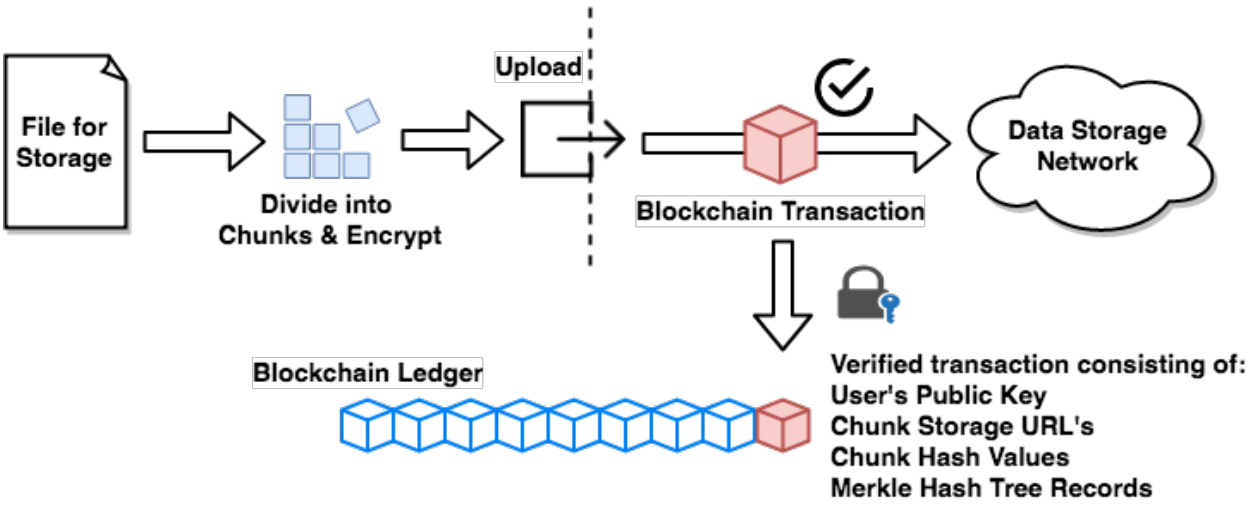


Figure 1 - high level design

These ideas were brought together to envision the system above. A Client would have a file that they wish to store remotely. First, as is important in modern design, an element of security must be applied. This is achieved by dividing the file into chunks which are then encrypted. These chunks are then uploaded across the distributed data storage network. That transaction is then verified by the network through the creation of a Block and adding that block to the Blockchain Ledger. These Blocks could then be subsequently examined by a security minded person to verify the integrity of data chunks.

Three main components were required by design for this project:

Client Side Module:

A client wishing to interact with the system will do so through a front end that will be well-documented. Any human <-> computer interactions will be designed in way as to feel as familiar to the user possible. Meaning through command line interface or a simple text UI. Transactions possible for the client will be based on, but not necessarily limited to, 'CRUD' database principles. (Create, Retrieve, Update and Delete.) Files that the user intends to be uploaded into the system will first be chunked into blocks of fixed volume, then encrypted and finally uploaded. No complete files nor unencrypted chunks of data are transmitted. This will grant end to end security of the client's data and therefore provide a reasonably safe transmission over any unsecure channels.

Additional commands were created for manual control of synchronizing the Blockchain, mining new blocks and running integrity checks on existing blocks and locally stored chunks.

Data Storage Network:

The Data Storage network will provide for the decentralized, distributed storage of actual data blocks across several connected nodes. Each node being their own computer. It is the project's aim to allow for a reasonable enough degree of elasticity in the size of the network as to permit an adequate demonstration of capability. In actualization, the storage network consists of a first point of contact which provides the client with a number of what could be described as empty, pre-addressed and labeled containers. The client then fills those containers with encrypted data before sending them to the storage addresses as labeled. Once a node receives a chunk, that node may then forward that chunk to other nodes. The more storage nodes available, the better for data redundancy. In retrospect, this is similar in basic practice to the master/minion operation of a Hadoop Distributed File System.

Block Chain Transactions & Ledger:

Through the verifiable process of the pseudo-random distribution of data chunks through-out the storage network, the storage locations of every data chunk will be saved as part of a verified block chain transaction and then stored in the block chain ledger. Meta data used in ensuring the confidentiality and integrity of every block of data will also be saved in the block transaction as coded hash values. This results in the ledger acting as a secure, private and reliable record of how each individual client's data blocks are being stored across the network. No actual blocks of data are saved within the block chain nor does simply having knowledge of the Blockchain permit access to files.

2. Body

2.1. Background

Generally speaking I'm very skeptical of new ideas and fads. I cringe when buzzwords like "Blockchain" and "Cloud" are thrown around. Still, these are not make believe ideas. Therefore, in an effort to dispel my misgivings, I'd like to explore the uses of Blockchain beyond the most prevalent use case of cryptocurrency. The high financial value placed upon data makes it difficult for companies to sufficiently trust third parties with that data. The solution that block chain technology brings to the problem issue of trust is interesting. To be able to algorithmically establish a decent degree of trust in an otherwise unknown person or computer for the purposes of completing some transaction is admittedly rather appealing.

When I was younger my family and I would run SETI@home on all of our computers, offering our unused computing power to search through masses of radio telescope data looking for aliens as part of a distributed 'cloud' before there was such a modern public notion of distributed computing. But that is someone else giving me their data. I don't like the idea of trusting my potentially private information to a computer that I cannot throw out a window. Originally that belief probably considered the computer's size and its proximity, but it really meant don't fully trust a computer that doesn't belong to you and isn't fully under your control.

That idea has matured with the modern times to mean that I'd prefer to store my data across many smaller location rather than one large central location if I can. This ensures that no one location has the full contents of my data. A verifiable transaction of the kind block chain will allow should permit for the building of a decentralized and trust worthy network of computers for the purpose of distributed data storage.

2.2. Project Statement

The problem that this project hopes to solve is that of the general need for businesses to comply with data privacy and protection laws while trusting mission critical data to an outsourced vendor who may have little if any accountability. Centralized storage locations provide a point of weakness in terms of security and reliability putting data at risk of exposure. Data integrity must be guaranteed against system fault or malicious intent. A record of change and access is need to verify origin in case of error, idea or property theft. The ability to hold responsible parties accountable for loss is highly desirable.

The general use case for this project is that:

- The user wants to store data at remote locations to achieve data redundancy.
- The user cannot rely upon any single source or vendor for their storage needs.
- The user needs to verify data integrity to protect intellectual property from theft.
- The user needs to hold accountable those who make mistakes or unauthorized changes to data.

2.3. Possible Alternative Solutions

There are numerous existing solutions to remote storage and Blockchain needs or both for that matter. They each currently fall short of satisfying the project's underlying use case in some fashion.

- **Amazon Web Services S3** fails to provide for the verification of data integrity. AWS is occasionally subject to unexpected downtime and for mission critical systems, depending on one provider for storage needs is not recommended.

- **Storj.io** provides block-level encrypted storage and allows for the third-party provisional of storage capacity but, as with AWS S3, it does not provide for data integrity verification.

- **Sia.tech** offers block-chain secured, enterprise grade, distributed and encrypted storage. The company is young and they are still developing their product. While the services offered to end users seem promising, the company's financial backers, goals and their values are unclear. Their services are highly monetized and are all heavily involved in cryptocurrency. A user's desire for privacy or data protection may not be a priority here, and those users may not wish to be involved with such a complex business model.

- **Hadoop Distributed File System (HDFS™)**: A distributed file system that provides high-throughput access to application data.

2.4.Chosen Solution

I have decided for the purpose of educating myself to not exploit any existing system for this project. There are several services such as Etherium that allow users to create their own Blockchain systems by entering transaction variables into their API. I did consider building just the storage network and client module, and then harnessing their existing Blockchain system to my purposes. While using such a service would give me a jump start to create perhaps a more capable system, I may miss out on a good learning opportunity. Subsequently, the design and coding of block mining and managing the ledger, has become a large component of my project.

Scope

The minimum scope of this project is to create an application that allows for the secure remote distributed storage of files for multiple users across multiple devices. Users shall at least be able to upload, download, update and Delete files on the network.

User Characteristics

The user is assumed to have some IT experience although that experience does not have to have been extensive. They should be able to operate a Linux or similar terminal interface with well documented commands and/or a simple GUI depending on the stage of development.

Proposal Limitations

Visual Style: Although I do have some experience making graphical user interfaces, my artistic skills are somewhat wanting. The priority of this project is not artistic. It may be necessary to leave room for improvement in the deliverables final visual appearance. Any intended visuals, while they will be functional, they may also be crude.

Time Constraints: As with all things, life may interfere with development. There is a limited time period in which to complete this project. Unforeseen risks may call for an adjustment in the project's overall scope in a manner that is acceptable to the project requirements. Sprints and milestones will be used to judge overall progress and to remain on track.

Resources: Real world projects tackling this and similar issues are often taken on by large companies with many skilled employees and deep financial resources. While this projects aspirations may be grand, resources will ultimately limit the final products quality and capabilities.

Revenue and Profits: This project is being carried out to meet the requirements of the program. No revenue or profit is expected or sought.

2.5. Details of Design and Development

2.5.1. Structure

The project had been divided into 3 layers, with each layer being a focus of the first 3 design and then implementation sprints. The application is written in Python 3.6 with the intent of having it run on Linux systems such as those located in the BCIT Datacomm lab.

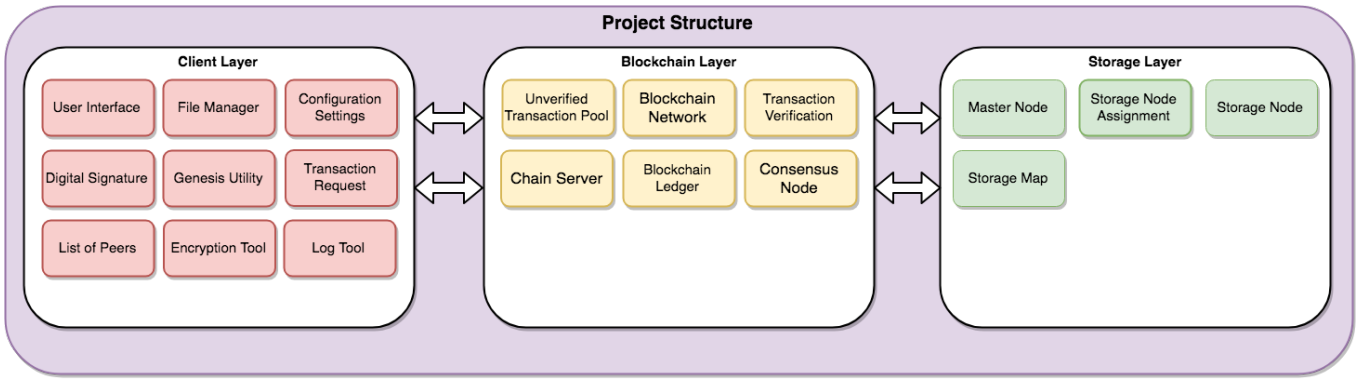


Figure 2 - Project Structure

Objects as shown in the above diagram are detailed below. Process flow diagrams have been added where appropriate.

2.5.2. Client Layer

User Interface:

Provide a means for the user to interact with the system by means of a command line. This interface is intentionally simple. Most functions involving the Blockchain are performed automatically as needed, although the user may force a Blockchain request manually. All variable settings are managed through a mandatory settings configuration file. Once their proper settings have been set in the configuration file, end users of the system should only need be concerned with uploading and downloading of their own files. The following flow chart details the design for the client node interactions, namely get(), put() and delete(). Delete was completed later during the integration sprint. Commands for checking block integrity were also added later.

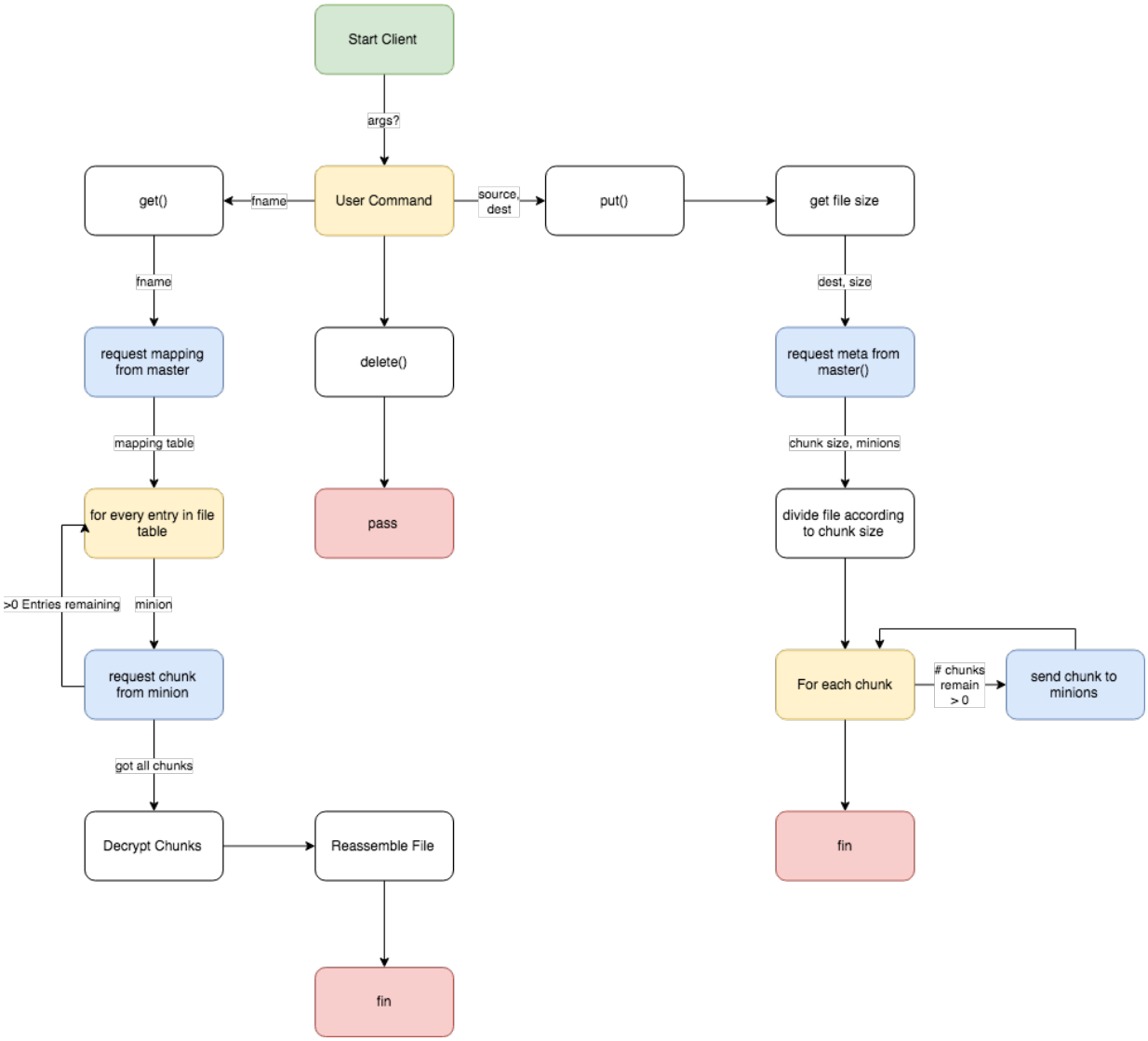


Figure 3 - client node design work flow

File Manager:

The client layer will require a means to access the users file system. Files are chunked into smaller components as per a set of instructions received from a storage network master node. Those chunks are then encrypted and uploaded to the dictated minion storage locations. In some cases for transmitting more complex data structures like blocks, the python pickle library was perfect for packaging payloads. Likewise, those files would be unpickled and then reassembled into complete files once retrieved and decrypted. The File manager is used by those devices who are flagged as client active in the settings file. The file manager allows for basic CRUD operations. (Create, Retrieve, Update, and Delete)

Configuration Settings:

All adjustable settings are loaded from an included settings configuration file. This file is to be edited manually from a text editor. Settings control which components are active such as if the device will operate as a master, minion, miner, client or, for whatever the purpose, all four. Example shown.

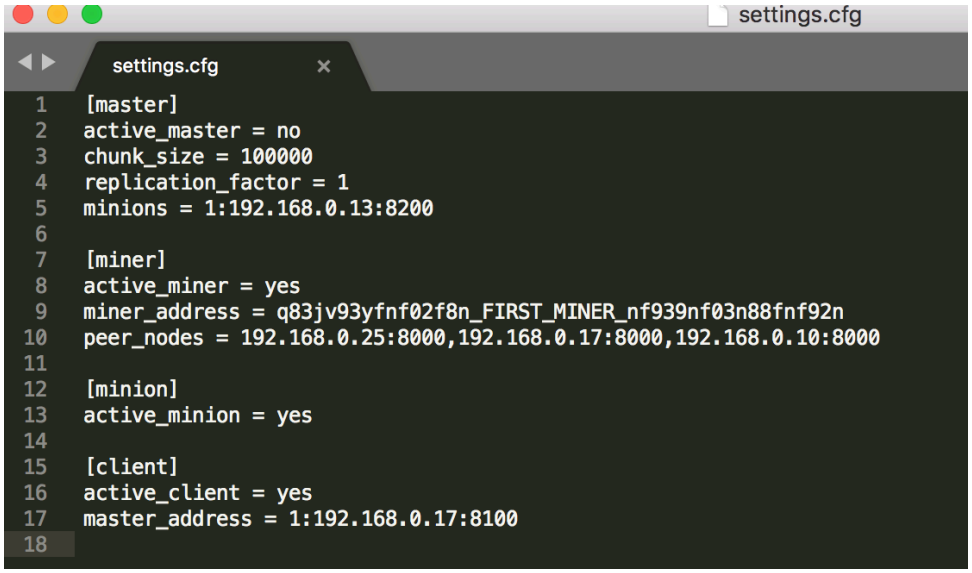


Figure 4 - settings.cfg - configuration file

Digital Signature:

Each individual user shall be uniquely identified by the system. This identification is used in the Blockchain process to identify block miners and also to control access permissions in the storage network.

Genesis Utility:

A tool by which new block chains are created. Creates a genesis block which is the first piece of a chain. This utility is run when no chains can be obtained from any network peers. The blocks fields are made up of 'junk' or meaningless data. The block is not identified with any data chunk location or users. The mathematical proof field is occupied by a look-alike, starting with multiple zeros as is required for the proof but it indicates that this is the genesis block.

```
block_data = {}
block_data['index'] = 0
block_data['version'] = VERSION
block_data['timestamp'] = date.datetime.now()
block_data['previous_hash'] = "0"
block_data['user_data'] = "none"
block_data['data_hash'] = "none"
block_data["proof"] = '00000048_GENESIS_BLOCKb16e9ac6cb'
block_data['data_url'] = "0"
first_block = Block(block_data)
```

Figure 5 - genesis block data

Transaction Request:

The client layer will have to send requests, file chunks and associated data to the network. All connections between client and nodes are handled using the TCP/IP protocol. For larger packages such as data chunks and block chains, pythons pickle utility is used to convert those packages into a format more easily managed by pythons network socket library.

List of Peers:

Local list of other known clients on the network. This list is edited manually in the included settings file. Although (perhaps regrettably) outside of current scope, a future consideration is to have this list updated and maintained automatically.

Encryption Tool:

No file chunks are to be uploaded to the network without prior encryption. Chunks downloaded will need to be decrypted. The tool may also generate keys and perform hash functions. The tool uses the Python3 pycrypto Advanced Encryption Standard library. This module is easily modified to utilize other modes of encryption and should prove droppable into other future applications.

Log Tool:

A portion of the client layer involved the creation of a logging tool to document all development, tests and fixes. Copies of the logs are included for completeness sake in the project deliverables. Most attached log entries will refer to the devices in the map below.

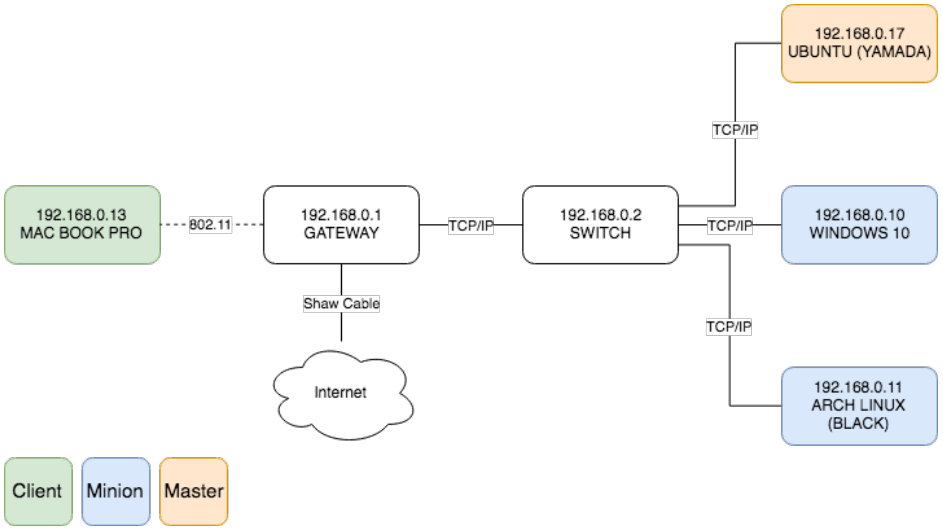


Figure 6 - typical network setup during development as shown in attached logs

2.5.3. Blockchain Layer

Unverified Transaction Pool:

The Blockchain will have a pool of client transaction requests awaiting verification. As chunks are uploaded and stored at nodes across the network, those nodes generate transactions which contain the information required for block creation. Any nodes designated as active miners will verify transactions from the pool and will create blocks for the chain.

Transaction Verification:

Part of the Blockchain transaction, the computational means by which client transactions are verified and new blocks are added the chain.

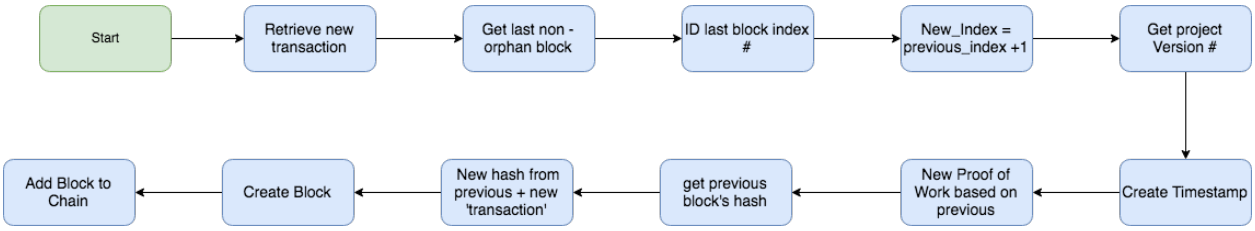


Figure 7 - block mining design flow

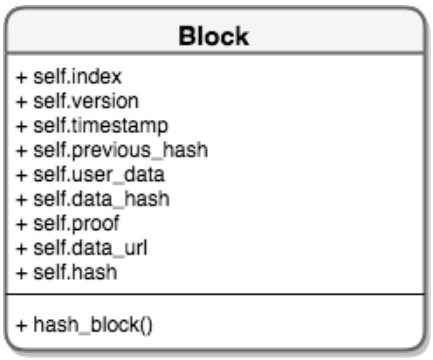


Figure 8 - block class diagram

Blockchain Network:

The physical devices and network connections between them. The JSON (Java-Script Object Notation) format is used for the local storage of blocks and for sharing and synchronizing the Blockchain between devices. The reason for choosing JSON is that is simple to read, by human and machine. It's text based, and requires very little storage and network capacity. A dedicated thread on each device listens for incoming Blockchain requests, packages its blocks to a byte format using Pickle and then transmits, thus sending its own 'copy' of the chain. Any machine that attempts to write to the Blockchain must first make such a request to all of its known peers. This sharing of JSON data forms part of the consensus process which is detailed later.

Blockchain Ledger:

The secure, consensually shared and synchronized database on which data storage location and access rights are stored. It is possible and does happen that 2 devices on the network will attempt to create a new block from the same previous block. Such blocks are not duplicated but would have the same index number and previous hash. In such a case, one block is chosen as the primary link and the other is not discarded but is identified as an orphan, from which no additional blocks will be created from.

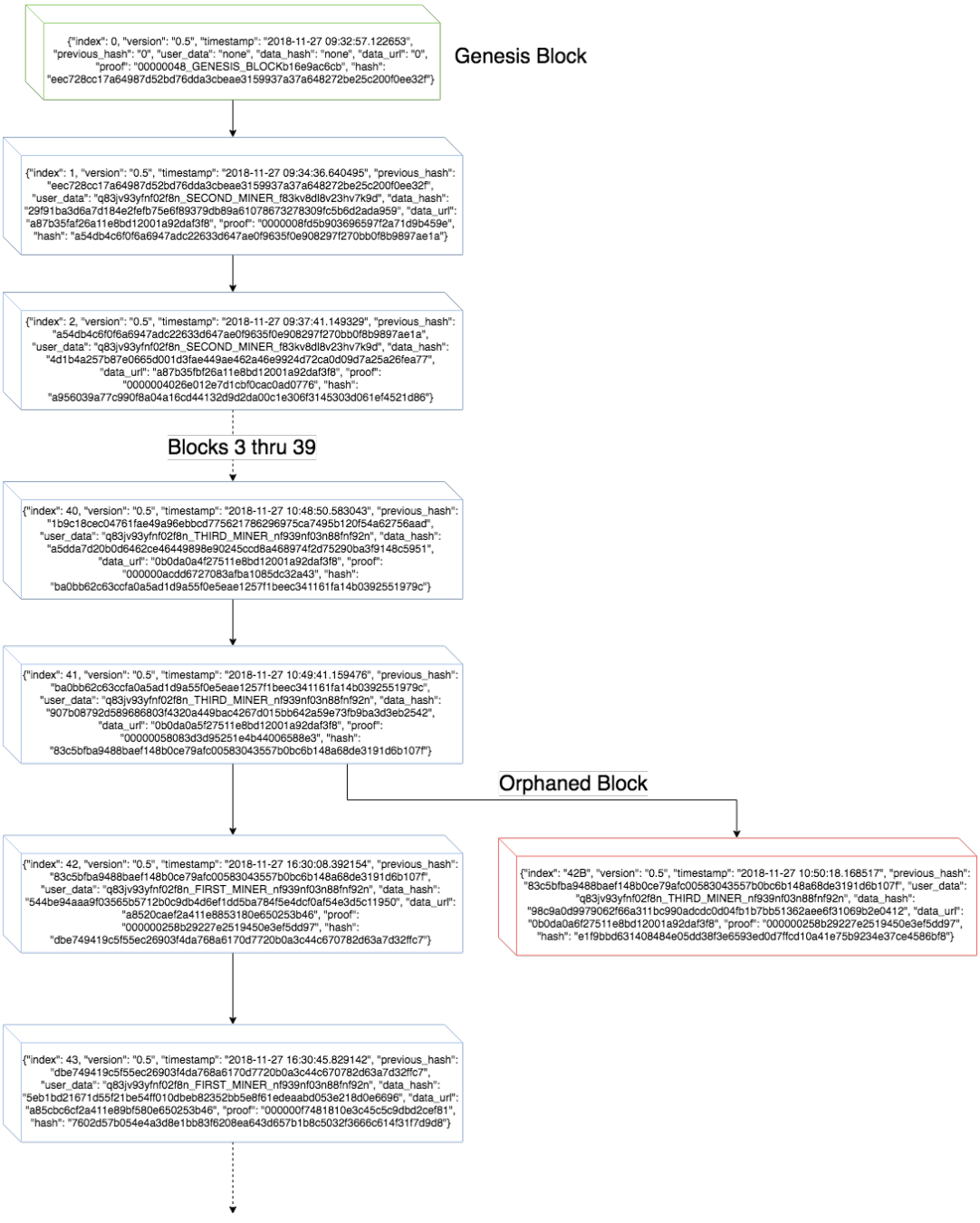


Figure 9 - visual graph of Blockchain with an orphan block

Consensus Node:

Any device connected to the Blockchain network is a node. A consensus node is one that writes the actual block to the ledger. Which node becomes the consensus may not be immediately apparent. Consensus, as in reality, is process that is made of several entities working together, repeating the process over and over until the chain is synchronized.

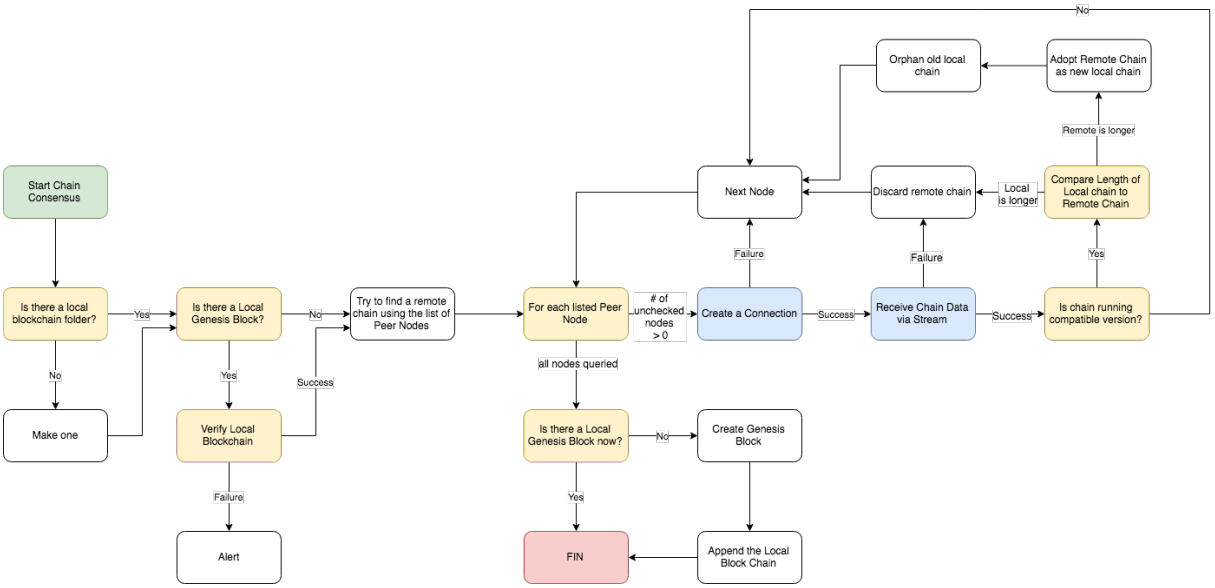


Figure 10 - Blockchain sharing and consensus design work flow

2.5.4. Storage Layer

Storage Master Node:

A master node is the point of first contact for the client . The master will determine the number of chunks required and will send the chunk allocation and distribution map to the client.

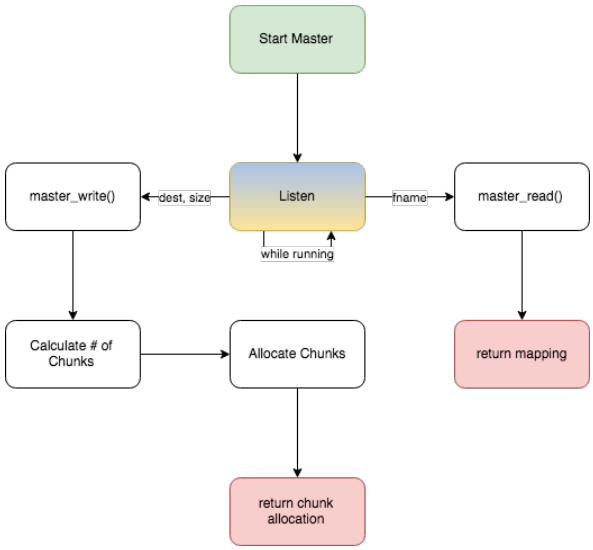


Figure 11 - storage master design work flow

Storage Node Assignment:

Part of the Blockchain transaction, the pseudo random means by which chunks are stored at nodes across the storage network. Each master node has a list of known minions. For each chunk that is required a URI is created. This creates what could be described as a pre-labeled empty container. A random storage minion node is selected as the primary destination for each container. With a number of other minions being redundant secondary destinations depending on the settings files listed replication factor. Each empty but labeled container is sent to the client, who will fill each container with encrypted data and send each to their assigned storage destinations. The shot below shows an incoming put request to a master node. 5 chunks are allocated, labeled with a URI and assigned to differing minion nodes [‘1’], [‘2’] and so forth.

```
2018-11-29 16:26:28.541617 -  - MASTER: incoming put request('192.168.0.13', 50578)
MASTER: 94366074f43611e88f1f80e650253b46 -> ['1']
MASTER: 94385c4ef43611e88f1f80e650253b46 -> ['2']
MASTER: 943a2fe2f43611e88f1f80e650253b46 -> ['1']
MASTER: 943f93ecf43611e88f1f80e650253b46 -> ['3']
MASTER: 94426f7cf43611e88f1f80e650253b46 -> ['2']
```

Figure 12 - storage node assignment performed by a master node

Storage Map:

The map which connects filenames and users with their chunk components and storage locations. This map is stored in memory while the program is running and is periodically written to a binary image file that is reloaded on program run. This way, the map is maintained even if the program is not running or fails.

Storage (Minion) Node:

A location on the storage network where encrypted chunks of data are stored. Each minion receives encrypted file chunks from clients and stores them. The minion then checks that chunks meta data for additional redundant storage locations and forwards those chunks to other minions as required. The minion also receives get requests from the client looking to retrieve specific chunks by their URI. No minion node has the means to decrypt any chunk. Nor can it derive a complete file from its disembodied chunk components.

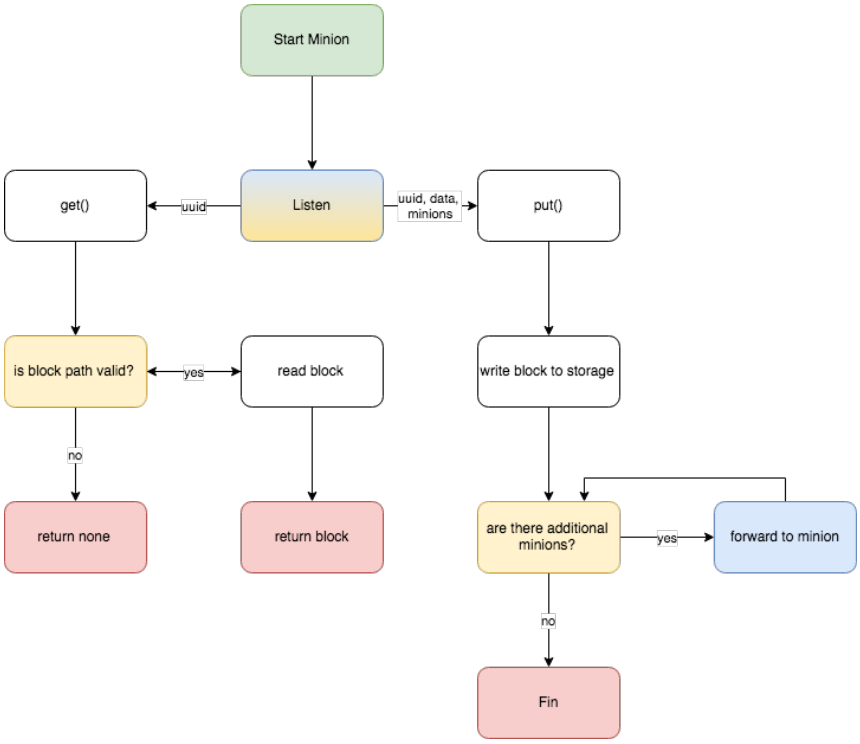


Figure 13 - storage node design work flow

2.6. User Guide / Setup & Installation

Requirements: The program is designed and intended for use on the Linux systems such as those used in the BCIT Datacomm lab only. However, the program has been run successfully on Microsoft Windows 10 and Apple MacOS but that is beyond the projects scope and therefore instructions for doing so are not included here.

Fedora 27 (as in the BCIT Datacomm lab):

Copy the program to a directory i.e. /prac or clone it from the available GitHub repository.

Run the following commands:

```
dnf install python3-devel

pip3 install pycrypto
```

Configure the settings.cfg file

```
[master]
active_master = yes       # is this machine a designated master node?
chunk_size = 100000       # desired chunk size in bytes for large files
replication_factor = 2    # number of times each chunk will have redundant copies across the network
minions = 1:192.168.0.14:8200,2:192.168.0.25:8200 # list of known storage network minions & ports

[miner]
active_miner = yes        # is this machine a designated miner? should be paired with active minions
miner_address = q83jv93yfnf02f8n_FIRST_MINER_nf939nf03n88fnf92n # the device and user name
peer_nodes = 192.168.0.25:8000 # list of known peers, whom we can obtain the Blockchain from.

[minion]
active_minion = yes       # Is this machine a designated storage minion?

[client]
active_client = yes       # is this machine a designated client device? allows for file management
master_address = 1:192.168.0.14:8100  # address & port of the known network master node
```

For use while in active_client mode:

Enable client mode in the settings.cfg file:

1. set active_client to yes
2. enter at least one master address like so:

```
[client]
active_client = yes
master_address = 1:192.168.0.14:8100
```

Run the program:

```
python3 main.py
```

Commands now available:

Upload a file:

```
CMD> put 'filename'
```

Download a file:

```
CMD> get 'filename'
```

Delete a file:

```
CMD> delete 'filename'
```

Force Mine Transactions

```
CMD> mine
```

Synchronize the Blockchain

```
CMD> sync
```

Run an integrity check on a specific block

```
CMD> block 'n'
```

Run an integrity check on a block / chunk relationship

```
CMD> chunk 'n'
```

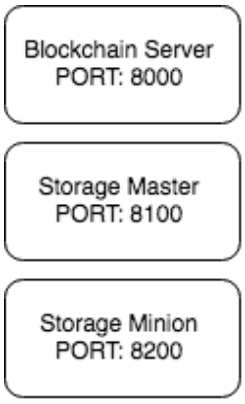


Figure 14 - default service ports for configuration file

2.7.Testing Details and Results

Testing was performed throughout the project to aid in a smooth delivery of a working, scalable application. Regular testing during development was done wherever possible by creating automated tests and other similar tests as the need arises. The overall aim is to minimize testing efforts while still ensuring all test cases can and will have been checked for and passed in the deliverable version.

Regression Testing: The focus here was on the validation of logic and the access between layers. This progressively involved the creation of automated scripts, unit tests, domain and range tests and the like as the application was developed.

Manual Testing: The focus is on those areas where automated testing is not possible or is impractical. Modules that are early in development, or areas concerning the user interface and the final functional requirements of the project.

2.7.1. Test Cases / Functional Requirements

Each test case is subsequently checked and results are detailed in the following section.

Case #	Description
1	Users shall be able to upload new files for storage. (Create)
2	Users shall be able to download files from storage when authorized. (Retrieve)
3	Users shall be able to update existing files when authorized. (Update)
4	Users shall be able to delete stored files when authorized. (Delete)
5	All files intended for upload shall first be chunked into blocks prior to encryption.
6	All user data blocks shall be encrypted using the Advanced Encryption Standard prior to any transmission.
7	Block Chain Transaction will involve the processing of data which consists of user info, data block storage location URLs, and block hash values.
8	The system shall tolerate a backlog of transactions to be processed.
9	Storage node assignment for data chunks will be pseudo randomized.
10	A file's integrity shall be verifiable. Including the source of a file and changes to those files.
11	Files will reassemble and be usable upon download as they were prior to upload.
12	No data block containing content shall be stored in the block chain.
13	Each user shall have a unique identifier.
14	The system shall tolerate file sizes sufficient enough for at least 4 chunks.
15	The SHA-256 algorithm shall be used as a minimum when generating secure hashes.
16	If a Blockchain does not exist, A user can elect to create the first block in a new chain. (Genesis)
17	All adjustable program settings will be controlled and loaded from a configuration file.
18	Storage locations will not be lost when the program (master node) ceases to run.
19	Blocks with duplicate indexes shall be orphaned to maintain a tidy Blockchain
20	Nodes can obtain a copy of the Blockchain from their peers

2.7.2. Test Results

Test Case 1 **- Users shall be able to upload new files for storage (Create)**

Setup: Program is running and the network has an active master storage node and at least 2 minion storage nodes.

Pass Condition: The user is able to upload a file and receives feedback indicating so.

Fail Condition: The user is unable to upload, and receives feedback detailing the error.

```
CMD> put testfile.jpg
2018-11-30 08:49:53.535363 -  - CLIENT: # of chunks:5
2018-11-30 08:49:53.536593 -  - CLIENT: CHUNK: f5b945c6f4bf11e885f4001a92daf3f8
2018-11-30 08:49:54.682068 -  - CLIENT: Sent to minion: 192.168.0.17
2018-11-30 08:49:54.682650 -  - CLIENT: CHUNK: f5b945c7f4bf11e885f4001a92daf3f8
2018-11-30 08:49:55.124042 -  - CLIENT: Sent to minion: 192.168.0.17
2018-11-30 08:49:55.124530 -  - CLIENT: CHUNK: f5b945c8f4bf11e885f4001a92daf3f8
2018-11-30 08:49:55.467961 -  - CLIENT: Sent to minion: 192.168.0.10
2018-11-30 08:49:55.468467 -  - CLIENT: CHUNK: f5b945c9f4bf11e885f4001a92daf3f8
2018-11-30 08:49:55.879902 -  - CLIENT: Sent to minion: 192.168.0.17
2018-11-30 08:49:55.880382 -  - CLIENT: CHUNK: f5b945caf4bf11e885f4001a92daf3f8
2018-11-30 08:49:56.123881 -  - CLIENT: Sent to minion: 192.168.0.10
2018-11-30 08:49:56.124373 -  - CLIENT: --->   UPLOADED FILE: testfile.jpg
CMD>
```

PASSED

Test Case 2 **- Users shall be able to download files from storage when authorized (Retrieve)**

Setup: Program is running and the network has an active master storage node and at least 2 minion storage nodes. The user has already uploaded a file into the network (Test case 1 is passed)

Pass Condition: The user is able to download a file and receives feedback indicating so.

Fail Condition: The user is unable to download the file, and receives feedback detailing the error.

```
CMD> get testfile.jpg
File Already Exists... Overwrite? y/n:y
2018-11-30 08:53:12.299864 -  - CLIENT: <---    DOWNLOADED FILE: testfile.jpg
CMD>
```

PASSED

Test Case 3 **- Users shall be able to update existing files when authorized (Update)**

Setup: Program is running and the network has an active master storage node and at least 2 minion storage nodes. The user has already uploaded a file into the network (Test case 1 is passed)

Pass Condition: The user is able to upload a file with the same filename as before and receives feedback indicating so.

Fail Condition: The user is unable to upload the file, and receives feedback detailing the error.

Note: The process of update is identical to that of create as far as the user is concerned.

```
CMD> put testfile.jpg
2018-11-30 08:55:58.600290 -  - CLIENT: # of chunks:5
2018-11-30 08:55:58.600779 -  - CLIENT: CHUNK: cf53c1daf4c011e885f4001a92daf3f8
2018-11-30 08:55:59.250290 -  - CLIENT: Sent to minion: 192.168.0.10
2018-11-30 08:55:59.250799 -  - CLIENT: CHUNK: cf53c1dbf4c011e885f4001a92daf3f8
2018-11-30 08:55:59.554343 -  - CLIENT: Sent to minion: 192.168.0.17
2018-11-30 08:55:59.555151 -  - CLIENT: CHUNK: cf53c1dcf4c011e885f4001a92daf3f8
2018-11-30 08:56:00.098588 -  - CLIENT: Sent to minion: 192.168.0.17
2018-11-30 08:56:00.099094 -  - CLIENT: CHUNK: cf53c1ddf4c011e885f4001a92daf3f8
2018-11-30 08:56:00.178369 -  - CLIENT: Sent to minion: 192.168.0.10
2018-11-30 08:56:00.178889 -  - CLIENT: CHUNK: cf53c1def4c011e885f4001a92daf3f8
2018-11-30 08:56:00.759318 -  - CLIENT: Sent to minion: 192.168.0.17
2018-11-30 08:56:00.759638 -  - CLIENT: --->   UPLOADED FILE: testfile.jpg
CMD>
```

PASSED

Test Case 4 **- Users shall be able to delete stored files when authorized (Delete)**

Setup: Program is running and the network has an active master storage node and at least 2 minion storage nodes. The user has already uploaded a file into the network (Test case 1 is completed and passed)

Pass Condition: The user is able to delete a file and receives feedback indicating so. The user is then unable to download that file from the network and receives feedback indicating file not found.

Fail Condition: The user is unable to delete a file and receives feedback detailing the error.

```
CMD> delete testfile.jpg
2018-11-30 08:58:20.180758 -  - CLIENT: NETWORK FILE DELETED
CMD> get testfile.jpg
2018-11-30 08:58:30.352744 -  - CLIENT: NETWORK FILE NOT FOUND
CMD>
```

PASSED

Test Case 5 **- All files intended for upload shall first be chunked into blocks prior to encryption.**

Setup: Program is running and the network has an active master storage node and at least 2 minion storage nodes.

Pass Condition: The user is able to upload a file and receives feedback indicating the number of chunks.

Fail Condition: The user is unable to upload the file, and receives feedback detailing the error.

Note: Testfile.txt has been broken into 5 chunks and sent to 2 different nodes. In this test as shown, the first chunk was sent to 192.168.0.17, the next 2 were sent to 192.168.10 and the final 2 were sent to 192.168.0.17.

```
CMD> put testfile.txt
2018-11-30 09:03:45.967956 -  - CLIENT: # of chunks:5
2018-11-30 09:03:45.968708 -  - CLIENT: CHUNK: e5e6d9fef4c111e885f4001a92daf3f8
2018-11-30 09:03:46.379166 -  - CLIENT: Sent to minion: 192.168.0.17
2018-11-30 09:03:46.379651 -  - CLIENT: CHUNK: e5e6d9fff4c111e885f4001a92daf3f8
2018-11-30 09:03:47.034297 -  - CLIENT: Sent to minion: 192.168.0.10
2018-11-30 09:03:47.034587 -  - CLIENT: CHUNK: e5e6da00f4c111e885f4001a92daf3f8
2018-11-30 09:03:47.539044 -  - CLIENT: Sent to minion: 192.168.0.10
2018-11-30 09:03:47.539524 -  - CLIENT: CHUNK: e5e6da01f4c111e885f4001a92daf3f8
2018-11-30 09:03:48.221044 -  - CLIENT: Sent to minion: 192.168.0.17
2018-11-30 09:03:48.221381 -  - CLIENT: CHUNK: e5e6da02f4c111e885f4001a92daf3f8
2018-11-30 09:03:48.732838 -  - CLIENT: Sent to minion: 192.168.0.17
2018-11-30 09:03:48.733168 -  - CLIENT: --->   UPLOADED FILE: testfile.txt
CMD>
```

PASSED

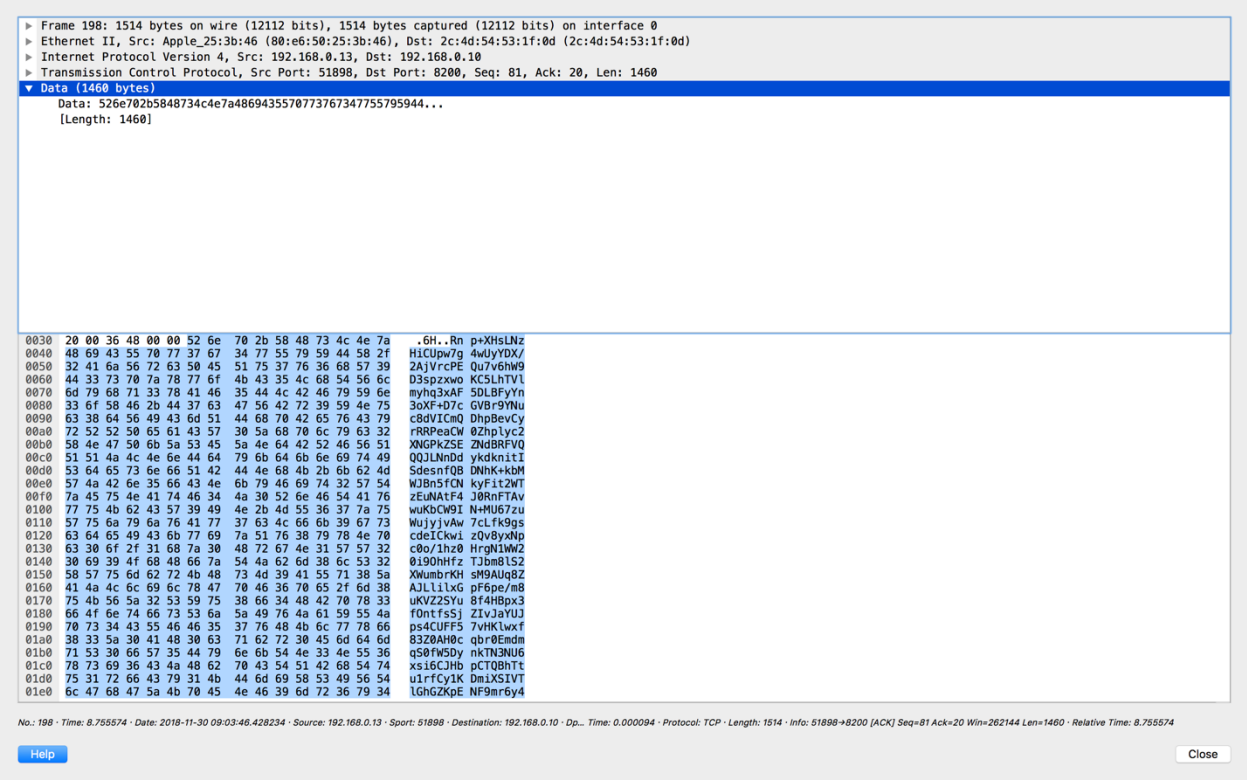
Test Case 6 **- All user data blocks shall be encrypted using the Advanced Encryption Standard prior to any transmission.**

Setup: A network packet capture using Wireshark can be examined for packet data details. A file upload of a plain text file such as in test case 5 is performed while a capture is made.

Pass Condition: The packets data payload is illegible (encrypted) .

Fail Condition: The packets data payload is legible in plain language.

This screenshot from Wireshark shows the chunk being sent from the client 192.168.0.13 to storage node 192.168.0.10 from the above test case 5. The original file was raw text. If not for encryption the text would be plainly readable in Wireshark.



PASSED

Test Case 7 **- Block Chain transactions will involve the processing of data which consists of user file info, data block storage location URLs and block hash values.**

Setup: Inspect the log read out from a storage node during and after a client upload.

Pass Condition: The transaction data of user info, url and block hash are shown in the log.

Fail Condition: The node will fail to mine blocks and no transaction info will be clear.

2018-12-05 14:16:56.326574 - - MINION: Port # 8200
2018-12-05 14:16:56.333055 - - PROGRAM COMPLETE, SERVING UNTIL MANUAL ABORT...
2018-12-05 14:17:36.490728 - - MINION: Received Chunk from:('192.168.0.17', 37590)
2018-12-05 14:17:36.914064 - - MINION: Received Chunk from:('192.168.0.17', 37598)
2018-12-05 14:17:37.329563 - - MINION: Forwarded to: 192.168.0.10
2018-12-05 14:17:37.380685 - - MINION: create transaction...
2018-12-05 14:17:37.425356 - - New Transaction: user data: q83jv93yfnf02f8n_FIRST_MINER_nf939nf03n88fnf92n data hash: 06694193c35383d90ae95b95b24d3dc7dbaecb65284ff513cf7080114b95b3ab url: 91af7c7ef8db11e8aeb8001a92daf3f8
The transaction is processed.

MINING... 91af7c7ef8db11e8aeb8001a92daf3f8
2018-12-05 14:20:00.390134 - - NEW BLOCK:: {'index': 153, 'version': '0.5', 'timestamp': '2018-12-05 14:20:00.389919', 'previous_hash': '0c45f1a7cee8e30afef93af486338212fddca995cdf48ec6f91973cbd7def208', 'user_data': 'q83jv93yfnf02f8n_FIRST_MINER_nf939nf03n88fnf92n', 'data_hash': '06694193c35383d90ae95b95b24d3dc7dbaecb65284ff513cf7080114b95b3ab', 'data_url': '91af7c7ef8db11e8aeb8001a92daf3f8', 'proof': '0000002892a525af39058a71486dc2d3', 'hash': 'e1efc78c23e44b6345d74f35d55af676b9b4c764690e148583737580527886af'}
MINING DONE... 91af7c7ef8db11e8aeb8001a92daf3f8
The block is mined and added to the chain.

```
[Richard@Richards-MBP:~/Documents/py/practicum/src/chunkdata$ ls
 0a566034f8db11e8aeb8001a92daf3f8  c46e98c3f8d911e8aeb8001a92daf3f8
 91af7c7ef8db11e8aeb8001a92daf3f8  c46e98c5f8d911e8aeb8001a92daf3f8
 91af7c81f8db11e8aeb8001a92daf3f8  c46e98c6f8d911e8aeb8001a92daf3f8
 95775778f8db11e8aeb8001a92daf3f8  cbbe4d02f8d911e8aeb8001a92daf3f8
 9577577af8db11e8aeb8001a92daf3f8  cbbe4d03f8d911e8aeb8001a92daf3f8
 9577577bf8db11e8aeb8001a92daf3f8  d2bdd334f8d911e8aeb8001a92daf3f8
 9577577cf8db11e8aeb8001a92daf3f8  d2bdd335f8d911e8aeb8001a92daf3f8
 9972ba99f8db11e8aeb8001a92daf3f8  d2bdd336f8d911e8aeb8001a92daf3f8
 9972ba9af8db11e8aeb8001a92daf3f8  d2bdd337f8d911e8aeb8001a92daf3f8
 9972ba9bf8db11e8aeb8001a92daf3f8  d2bdd338f8d911e8aeb8001a92daf3f8
 9972ba9cf8db11e8aeb8001a92daf3f8
 Richard@Richards-MBP:~/Documents/py/practicum/src/chunkdata$
```

The chunk data directory on the storage minion contains the related chunk, labeled with its url

PASSED

Test Case 8 **- The system shall tolerate a backlog of transactions to be processed.**

Setup: Inspect the log read out from a storage node during and after a client upload. The node will progress through transactions backlog, mining blocks one after another.

Pass Condition: New transactions are added to the backlog and removed from the backlog as they are mined into blocks.

Fail Condition: The backlog remains static and no blocks are mined.

```
2018-12-05 14:16:56.326574 -  - MINION: Port # 8200
2018-12-05 14:16:56.333055 -  - PROGRAM COMPLETE, SERVING UNTIL MANUAL ABORT...
2018-12-05 14:17:36.490728 -  - MINION: Received Chunk from:('192.168.0.17', 37590)
2018-12-05 14:17:36.914064 -  - MINION: Received Chunk from:('192.168.0.17', 37598)
2018-12-05 14:17:37.329563 -  - MINION: Forwarded to: 192.168.0.10
2018-12-05 14:17:37.380685 -  - MINION: create transaction...
2018-12-05 14:17:37.425356 -  - New Transaction: user data:
q83jv93yfnf02f8n_FIRST_MINER_nf939nf03n88fnf92n data hash:
06694193c35383d90ae95b95b24d3dc7dbaecb65284ff513cf7080114b95b3ab url: 91af7c7ef8db11e8aeb8001a92daf3f8
# of local transactions:1
2018-12-05 14:17:37.465519 -  - Synchronizing Blockchain...
2018-12-05 14:17:37.784249 -  - MINION: Forwarded to: 192.168.0.10
2018-12-05 14:17:37.784736 -  - MINION: create transaction...
2018-12-05 14:17:37.785209 -  - New Transaction: user data:
q83jv93yfnf02f8n_FIRST_MINER_nf939nf03n88fnf92n data hash:
c64cab015731e634e28a277db26847e8547cd2b0a4c5a19607b0bd5a7e2bf4d0 url: 91af7c81f8db11e8aeb8001a92daf3f8
2018-12-05 14:17:42.415190 -  - MINION: Received Chunk from:('192.168.0.17', 37610)
2018-12-05 14:17:42.415489 -  - MINION: create transaction...
2018-12-05 14:17:42.415964 -  - New Transaction: user data:
q83jv93yfnf02f8n_FIRST_MINER_nf939nf03n88fnf92n data hash:
06694193c35383d90ae95b95b24d3dc7dbaecb65284ff513cf7080114b95b3ab url: 95775778f8db11e8aeb8001a92daf3f8
2018-12-05 14:17:42.472787 -  - MINION: Received Chunk from:('192.168.0.17', 37618)
2018-12-05 14:17:42.500846 -  - MINION: Received Chunk from:('192.168.0.17', 37620)
2018-12-05 14:17:42.503224 -  - MINION: Forwarded to: 192.168.0.17
2018-12-05 14:17:42.503696 -  - MINION: create transaction...
2018-12-05 14:17:42.504174 -  - New Transaction: user data:
q83jv93yfnf02f8n_FIRST_MINER_nf939nf03n88fnf92n data hash:
c64cab015731e634e28a277db26847e8547cd2b0a4c5a19607b0bd5a7e2bf4d0 url: 9577577bf8db11e8aeb8001a92daf3f8
2018-12-05 14:17:42.560929 -  - MINION: Forwarded to: 192.168.0.11
2018-12-05 14:17:42.561530 -  - MINION: create transaction...
2018-12-05 14:17:42.562054 -  - New Transaction: user data:
q83jv93yfnf02f8n_FIRST_MINER_nf939nf03n88fnf92n data hash:
6332f315f0adbe4344d2d74ce0e08fe9f27ce165ce9129909b3a6b7809719bc6 url: 9577577cf8db11e8aeb8001a92daf3f8
2018-12-05 14:17:42.661278 -  - MINION: Received Chunk from:('192.168.0.17', 37616)
2018-12-05 14:17:42.661578 -  - MINION: create transaction...
2018-12-05 14:17:42.662037 -  - New Transaction: user data:
q83jv93yfnf02f8n_FIRST_MINER_nf939nf03n88fnf92n data hash:
db8419700d866e3d3fe65a811be50ce2028c983c630f41ce66368015a511bf50 url: 9577577af8db11e8aeb8001a92daf3f8
2018-12-05 14:17:49.122468 -  - MINION: Received Chunk from:('192.168.0.17', 37628)
2018-12-05 14:17:49.195181 -  - MINION: Forwarded to: 192.168.0.11
2018-12-05 14:17:49.195744 -  - MINION: create transaction...
2018-12-05 14:17:49.195932 -  - MINION: Received Chunk from:('192.168.0.17', 37632)
2018-12-05 14:17:49.196566 -  - New Transaction: user data:
q83jv93yfnf02f8n_FIRST_MINER_nf939nf03n88fnf92n data hash:
5eb1bd21671d55f21be54ff010dbeb82352bb5e8f61edeaabd053e218d0e6696 url: 9972ba99f8db11e8aeb8001a92daf3f8
2018-12-05 14:17:49.291072 -  - MINION: Received Chunk from:('192.168.0.11', 38038)
2018-12-05 14:17:49.291333 -  - MINION: create transaction...
2018-12-05 14:17:49.291918 -  - New Transaction: user data:
q83jv93yfnf02f8n_FIRST_MINER_nf939nf03n88fnf92n data hash:
242ea344a5d559e6b70b739c911a71faa535c68a1f4f667f7b1ab9e5cc1c2a3a url: 9972ba9cf8db11e8aeb8001a92daf3f8
2018-12-05 14:17:49.375189 -  - MINION: Received Chunk from:('192.168.0.11', 38036)
2018-12-05 14:17:49.375469 -  - MINION: create transaction...
2018-12-05 14:17:49.375952 -  - New Transaction: user data:
q83jv93yfnf02f8n_FIRST_MINER_nf939nf03n88fnf92n data hash:
2439e76509728dfd6f2c99d626cc704d3c284425665b2b385f4f237acdd7157e url: 9972ba9af8db11e8aeb8001a92daf3f8
2018-12-05 14:17:49.886079 -  - MINION: Forwarded to: 192.168.0.10
2018-12-05 14:17:49.886555 -  - MINION: create transaction...
```

```
2018-12-05 14:17:49.887180 -  - New Transaction: user data:
q83jv93yfnf02f8n_FIRST_MINER_nf939nf03n88fnf92n data hash:
09469f79daaa67fc6cf55b1f1d468da9cb903c76d84a0e92027a65182ffe8cf6 url: 9972ba9bf8db11e8aeb8001a92daf3f8
MINING... 91af7c7ef8db11e8aeb8001a92daf3f8
2018-12-05 14:20:00.390134 -  - NEW BLOCK:: {'index': 153, 'version': '0.5', 'timestamp': '2018-12-05
14:20:00.389919', 'previous_hash': '0c45f1a7cee8e30afef93af486338212fddca995cdf48ec6f91973cbd7def208',
'user_data': 'q83jv93yfnf02f8n_FIRST_MINER_nf939nf03n88fnf92n', 'data_hash':
'06694193c35383d90ae95b95b24d3dc7dbaecb65284ff513cf7080114b95b3ab', 'data_url':
'91af7c7ef8db11e8aeb8001a92daf3f8', 'proof': '0000002892a525af39058a71486dc2d3', 'hash':
'e1efc78c23e44b6345d74f35d55af676b9b4c764690e14858737580527886af'}
MINING DONE... 91af7c7ef8db11e8aeb8001a92daf3f8
# of local transactions:9
2018-12-05 14:20:00.391029 -  - Synchronizing Blockchain...
2018-12-05 14:20:33.214311 -  - ABOPTING BLOCK:: {'index': 154, 'version': '0.5', 'timestamp': '2018-
12-05 14:19:56.896651', 'previous_hash':
'9441f99cac2ee9c8caab555c911685cb93eb12d35ed8e6952cadddbce9a3dd8c', 'user_data':
'q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n', 'data_hash':
'09469f79daaa67fc6cf55b1f1d468da9cb903c76d84a0e92027a65182ffe8cf6', 'data_url':
'00eb1da9f8db11e8aeb8001a92daf3f8', 'proof': '000000141d6cdf5822f1bb1f12ba8fe7', 'hash':
'0bd06ef876a6864a5efd15d1947f8a47a26eca715bfcf2a75ec75434ca84403f'}
MINING... 91af7c81f8db11e8aeb8001a92daf3f8
2018-12-05 14:20:59.183202 -  - NEW BLOCK:: {'index': 155, 'version': '0.5', 'timestamp': '2018-12-05
14:20:59.182992', 'previous_hash': '0bd06ef876a6864a5efd15d1947f8a47a26eca715bfcf2a75ec75434ca84403f',
'user_data': 'q83jv93yfnf02f8n_FIRST_MINER_nf939nf03n88fnf92n', 'data_hash':
'c64cab015731e634e28a277db26847e8547cd2b0a4c5a19607b0bd5a7e2bf4d0', 'data_url':
'91af7c81f8db11e8aeb8001a92daf3f8', 'proof': '0000004f90d25028b2b125d1718ed408', 'hash':
'e1faf248ef22d7e22b38374f34e31007465981156aa429768b1d35a6a998b1aa'}
MINING DONE... 91af7c81f8db11e8aeb8001a92daf3f8
# of local transactions:8
2018-12-05 14:20:59.184082 -  - Synchronizing Blockchain...
MINING... 95775778f8db11e8aeb8001a92daf3f8
2018-12-05 14:22:19.050169 -  - NEW BLOCK:: {'index': 156, 'version': '0.5', 'timestamp': '2018-12-05
14:22:19.049951', 'previous_hash': 'e1faf248ef22d7e22b38374f34e31007465981156aa429768b1d35a6a998b1aa',
'user_data': 'q83jv93yfnf02f8n_FIRST_MINER_nf939nf03n88fnf92n', 'data_hash':
'06694193c35383d90ae95b95b24d3dc7dbaecb65284ff513cf7080114b95b3ab', 'data_url':
'95775778f8db11e8aeb8001a92daf3f8', 'proof': '000000c16a28b0bf08a46fd95c865813', 'hash':
'31fe362c8e840549bd5a28ea5c522af0c489efa4d1c16fd0c6606f5e4a713ebd'}
MINING DONE... 95775778f8db11e8aeb8001a92daf3f8
# of local transactions:7
2018-12-05 14:22:19.051074 -  - Synchronizing Blockchain...
MINING... 9577577bf8db11e8aeb8001a92daf3f8
2018-12-05 14:22:53.550746 -  - NEW BLOCK:: {'index': 157, 'version': '0.5', 'timestamp': '2018-12-05
14:22:53.550492', 'previous_hash': '31fe362c8e840549bd5a28ea5c522af0c489efa4d1c16fd0c6606f5e4a713ebd',
'user_data': 'q83jv93yfnf02f8n_FIRST_MINER_nf939nf03n88fnf92n', 'data_hash':
'c64cab015731e634e28a277db26847e8547cd2b0a4c5a19607b0bd5a7e2bf4d0', 'data_url':
'9577577bf8db11e8aeb8001a92daf3f8', 'proof': '0000006744d5daee2a370498528446f9', 'hash':
'988ba841ed12da097dec12c3890cb6c3693973f69cf4fb6b0142aed152a8e7e8'}
MINING DONE... 9577577bf8db11e8aeb8001a92daf3f8
# of local transactions:6
2018-12-05 14:22:53.551712 -  - Synchronizing Blockchain...
MINING... 9577577cf8db11e8aeb8001a92daf3f8
2018-12-05 14:23:57.858631 -  - NEW BLOCK:: {'index': 158, 'version': '0.5', 'timestamp': '2018-12-05
14:23:57.857142', 'previous_hash': '988ba841ed12da097dec12c3890cb6c3693973f69cf4fb6b0142aed152a8e7e8',
'user_data': 'q83jv93yfnf02f8n_FIRST_MINER_nf939nf03n88fnf92n', 'data_hash':
'6332f315f0adbe4344d2d74ce0e08fe9f27ce165ce9129909b3a6b7809719bc6', 'data_url':
'9577577cf8db11e8aeb8001a92daf3f8', 'proof': '000000c28b1fd8c90a3c99807bdc297d', 'hash':
'86ba447729823971e02a89255b8eb46cea5ed52d6d9e07511b6078617e48483a'}
MINING DONE... 9577577cf8db11e8aeb8001a92daf3f8
# of local transactions:5
```

As shown in the log, new transactions are added, bringing the backlog up to 9. As blocks are mined, the number of transactions in the backlog reduced to 5 and continued to reduce until no further transaction remained.

PASSED

Test Case 9 **- Storage node assignment for data chunks will be pseudo randomized.**

Setup: The client will attempt to upload files into the network where at least 3 minions are active.

Pass Condition: Read out from the client after several upload requests indicates no apparent reasonable pattern in the assignment of storage minion nodes.

Fail Condition: Read out from the client after several upload requests indicates a too obvious pattern in assignment of storage minion nodes.

```
CMD> put testfile.jpg
2018-12-02 14:27:14.780130 -  - CLIENT: # of chunks:5
2018-12-02 14:27:14.831218 -  - CLIENT: CHUNK: 6b481624f68111e885f4001a92daf3f8
2018-12-02 14:27:15.545700 -  - CLIENT: Sent to minion: 192.168.0.17
2018-12-02 14:27:15.592727 -  - CLIENT: CHUNK: 6b481625f68111e885f4001a92daf3f8
2018-12-02 14:27:15.778114 -  - CLIENT: Sent to minion: 192.168.0.11
2018-12-02 14:27:15.816149 -  - CLIENT: CHUNK: 6b481626f68111e885f4001a92daf3f8
2018-12-02 14:27:16.070107 -  - CLIENT: Sent to minion: 192.168.0.10
2018-12-02 14:27:16.113863 -  - CLIENT: CHUNK: 6b481627f68111e885f4001a92daf3f8
2018-12-02 14:27:16.268676 -  - CLIENT: Sent to minion: 192.168.0.17
2018-12-02 14:27:16.318362 -  - CLIENT: CHUNK: 6b481628f68111e885f4001a92daf3f8
2018-12-02 14:27:16.537429 -  - CLIENT: Sent to minion: 192.168.0.10
2018-12-02 14:27:16.587852 -  - CLIENT: --->   UPLOADED FILE: testfile.jpg
CMD> put testfile.jpg
2018-12-02 14:27:25.272049 -  - CLIENT: # of chunks:5
2018-12-02 14:27:25.309906 -  - CLIENT: CHUNK: 719fb612f68111e885f4001a92daf3f8
2018-12-02 14:27:25.429264 -  - CLIENT: Sent to minion: 192.168.0.17
2018-12-02 14:27:25.453811 -  - CLIENT: CHUNK: 719fb613f68111e885f4001a92daf3f8
2018-12-02 14:27:25.602058 -  - CLIENT: Sent to minion: 192.168.0.17
2018-12-02 14:27:25.639356 -  - CLIENT: CHUNK: 719fb614f68111e885f4001a92daf3f8
2018-12-02 14:27:25.791373 -  - CLIENT: Sent to minion: 192.168.0.10
2018-12-02 14:27:25.816527 -  - CLIENT: CHUNK: 719fb615f68111e885f4001a92daf3f8
2018-12-02 14:27:25.987348 -  - CLIENT: Sent to minion: 192.168.0.10
2018-12-02 14:27:26.037615 -  - CLIENT: CHUNK: 719fb616f68111e885f4001a92daf3f8
2018-12-02 14:27:26.239413 -  - CLIENT: Sent to minion: 192.168.0.10
2018-12-02 14:27:26.306772 -  - CLIENT: --->   UPLOADED FILE: testfile.jpg
CMD> put testfile.txt
2018-12-02 14:27:33.143828 -  - CLIENT: # of chunks:5
2018-12-02 14:27:33.167586 -  - CLIENT: CHUNK: 7639de82f68111e885f4001a92daf3f8
2018-12-02 14:27:33.372621 -  - CLIENT: Sent to minion: 192.168.0.11
2018-12-02 14:27:33.407983 -  - CLIENT: CHUNK: 7639de83f68111e885f4001a92daf3f8
2018-12-02 14:27:33.503843 -  - CLIENT: Sent to minion: 192.168.0.11
2018-12-02 14:27:33.535564 -  - CLIENT: CHUNK: 7639de84f68111e885f4001a92daf3f8
2018-12-02 14:27:34.069392 -  - CLIENT: Sent to minion: 192.168.0.17
2018-12-02 14:27:34.118560 -  - CLIENT: CHUNK: 7639de85f68111e885f4001a92daf3f8
2018-12-02 14:27:35.146718 -  - CLIENT: Sent to minion: 192.168.0.17
2018-12-02 14:27:35.202515 -  - CLIENT: CHUNK: 7639de86f68111e885f4001a92daf3f8
2018-12-02 14:27:35.349880 -  - CLIENT: Sent to minion: 192.168.0.10
2018-12-02 14:27:35.393488 -  - CLIENT: --->   UPLOADED FILE: testfile.txt
CMD> put earth.jpeg
2018-12-02 14:28:30.437868 -  - CLIENT: # of chunks:5
2018-12-02 14:28:30.489269 -  - CLIENT: CHUNK: 985fb00ef68111e885f4001a92daf3f8
2018-12-02 14:28:30.707976 -  - CLIENT: Sent to minion: 192.168.0.10
2018-12-02 14:28:30.745069 -  - CLIENT: CHUNK: 985fb00ff68111e885f4001a92daf3f8
2018-12-02 14:28:30.950885 -  - CLIENT: Sent to minion: 192.168.0.17
2018-12-02 14:28:30.985523 -  - CLIENT: CHUNK: 985fb010f68111e885f4001a92daf3f8
2018-12-02 14:28:31.227982 -  - CLIENT: Sent to minion: 192.168.0.17
2018-12-02 14:28:31.276304 -  - CLIENT: CHUNK: 985fb011f68111e885f4001a92daf3f8
2018-12-02 14:28:31.536859 -  - CLIENT: Sent to minion: 192.168.0.10
```

```
2018-12-02 14:28:31.586233 -  - CLIENT: CHUNK: 985fb012f68111e885f4001a92daf3f8
2018-12-02 14:28:31.782252 -  - CLIENT: Sent to minion: 192.168.0.17
2018-12-02 14:28:31.818932 -  - CLIENT: --->   UPLOADED FILE: earth.jpeg
CMD> put zelda.mp3
2018-12-02 14:28:42.355053 -  - CLIENT: # of chunks:5
2018-12-02 14:28:42.396033 -  - CLIENT: CHUNK: 9f74fdccf68111e885f4001a92daf3f8
2018-12-02 14:28:43.096497 -  - CLIENT: Sent to minion: 192.168.0.10
2018-12-02 14:28:43.146158 -  - CLIENT: CHUNK: 9f74fdcdf68111e885f4001a92daf3f8
2018-12-02 14:28:44.891629 -  - CLIENT: Sent to minion: 192.168.0.11
2018-12-02 14:28:44.928761 -  - CLIENT: CHUNK: 9f74fdcef68111e885f4001a92daf3f8
2018-12-02 14:28:46.264583 -  - CLIENT: Sent to minion: 192.168.0.11
2018-12-02 14:28:46.295762 -  - CLIENT: CHUNK: 9f74fdcff68111e885f4001a92daf3f8
2018-12-02 14:28:49.547480 -  - CLIENT: Sent to minion: 192.168.0.17
2018-12-02 14:28:49.590829 -  - CLIENT: CHUNK: 9f74fdd0f68111e885f4001a92daf3f8
2018-12-02 14:28:51.694130 -  - CLIENT: Sent to minion: 192.168.0.17
2018-12-02 14:28:51.738572 -  - CLIENT: --->   UPLOADED FILE: zelda.mp3
CMD> put logfile.txt
2018-12-02 14:29:07.820899 -  - CLIENT: # of chunks:5
2018-12-02 14:29:07.889963 -  - CLIENT: CHUNK: aea85500f68111e885f4001a92daf3f8
2018-12-02 14:29:08.019979 -  - CLIENT: Sent to minion: 192.168.0.10
2018-12-02 14:29:08.062606 -  - CLIENT: CHUNK: aea85501f68111e885f4001a92daf3f8
2018-12-02 14:29:08.192811 -  - CLIENT: Sent to minion: 192.168.0.11
2018-12-02 14:29:08.242488 -  - CLIENT: CHUNK: aea85502f68111e885f4001a92daf3f8
2018-12-02 14:29:08.368148 -  - CLIENT: Sent to minion: 192.168.0.11
2018-12-02 14:29:08.399890 -  - CLIENT: CHUNK: aea85503f68111e885f4001a92daf3f8
2018-12-02 14:29:08.788245 -  - CLIENT: Sent to minion: 192.168.0.17
2018-12-02 14:29:08.838223 -  - CLIENT: CHUNK: aea85504f68111e885f4001a92daf3f8
2018-12-02 14:29:09.130085 -  - CLIENT: Sent to minion: 192.168.0.17
2018-12-02 14:29:09.193120 -  - CLIENT: --->   UPLOADED FILE: logfile.txt
CMD> put testfile.jpg
2018-12-02 14:30:11.805244 -  - CLIENT: # of chunks:5
2018-12-02 14:30:11.848386 -  - CLIENT: CHUNK: d4cf669cf68111e885f4001a92daf3f8
2018-12-02 14:30:12.031155 -  - CLIENT: Sent to minion: 192.168.0.10
2018-12-02 14:30:12.056254 -  - CLIENT: CHUNK: d4cf669df68111e885f4001a92daf3f8
2018-12-02 14:30:12.194691 -  - CLIENT: Sent to minion: 192.168.0.11
2018-12-02 14:30:12.223216 -  - CLIENT: CHUNK: d4cf669ef68111e885f4001a92daf3f8
2018-12-02 14:30:12.780165 -  - CLIENT: Sent to minion: 192.168.0.17
2018-12-02 14:30:12.814913 -  - CLIENT: CHUNK: d4cf669ff68111e885f4001a92daf3f8
2018-12-02 14:30:12.942400 -  - CLIENT: Sent to minion: 192.168.0.11
2018-12-02 14:30:12.998093 -  - CLIENT: CHUNK: d4cf66a0f68111e885f4001a92daf3f8
2018-12-02 14:30:13.160441 -  - CLIENT: Sent to minion: 192.168.0.11
2018-12-02 14:30:13.192336 -  - CLIENT: --->   UPLOADED FILE: testfile.jpg
CMD>
```

There is no apparent pattern in the assignment of minion nodes. The distribution appears random amongst the 3 active minions.

PASSED

Test Case 10 **- A file's integrity shall be verifiable. Including the source of a file and changes.**

Setup: After blocks have been mined and network is stored on node. That node can double check the integrity of both blocks in the chain and their relevant chunks. Both the block and relevant chunk must be available locally. The user first runs the command 'block n' with n being a block index number. The block hash is re-written from available data to ensure that the block itself has not been tampered with. Second, the user runs the command 'chunk n' with n being the same block index number. This will compare the blocks data with the available raw chunk to ensure that either chunk or block have not been tampered with.

Pass Condition: When known to be intact, both block check and chunk check will report OK. When known to be tampered with the test will fail

Fail Condition: When known to be intact, both checks may report error. When known to be tampered with the test may pass in error or the program will fail.

```
CMD> block 33
BLOCK OK: b32521349176eb700dab4aaba6158d9449e2a3218752cd7f70b91c59e66b58bf
CMD> chunk 33
CHUNK AND BLOCK OK: b32521349176eb700dab4aaba6158d9449e2a3218752cd7f70b91c59e66b58bf == b32521349176eb700dab4aaba6158d9449e2a3218752cd7f7
0b91c59e66b58bf
CMD>
```

Above: Both intact block and then intact chunk pass a rehash check.

```
W4boiOmX2ln4ohRd4XBojemao0ZwRR9VIXeufp/jZiyH3qI8Qjzc6YeNqgZ7HKfmazwSMGizmmuneIDY
```

```
X4boiOmX2ln4ohRd4XBojemao0ZwRR9VIXeufp/jZiyH3qI8Qjzc6YeNqgZ7HKfmazwSMGizmmuneIDY
```

```
CMD> block 33
BLOCK OK: b32521349176eb700dab4aaba6158d9449e2a3218752cd7f70b91c59e66b58bf
CMD> chunk 33
CHECK FAIL: 7e3bfc4e67e7fe02cf44b0fd2b27c1c525c8b2875b422ed9cbaeca0740797833 != b32521349176eb700dab4aaba6158d9449e2a3218752cd7f70b91c59e
66b58bf
CMD>
```

Above: The raw chunk has been manually edited on the storage node. While the block itself remains intact, an integrity comparison shows a change in the chunk.

```
, "data_hash": "544be94aaa9f03565b5712b0c9db4d6ef1dd
```

```
 "data_hash": "644be94aaa9f03565b5712b0c9db4d6ef1dd
```

```
CMD> block 33
CHECK FAIL: 9201185849f2d10d089ce3bc843d46db0599c2a442479694355ff9dd32e1e129 != b32521349176eb700dab4aaba6158d9449e2a3218752cd7f70b91c59e
66b58bf
```

Above: The block has been manually edited and fails its hash check as expected.

PASSED

Test Case 11 **- Files will reassemble and be usable upon download as they were prior to upload.**

Setup: A large jpg file will be visually examined and then uploaded. The local copy will then be deleted. Once deleted, the file will be downloaded and then reexamined.

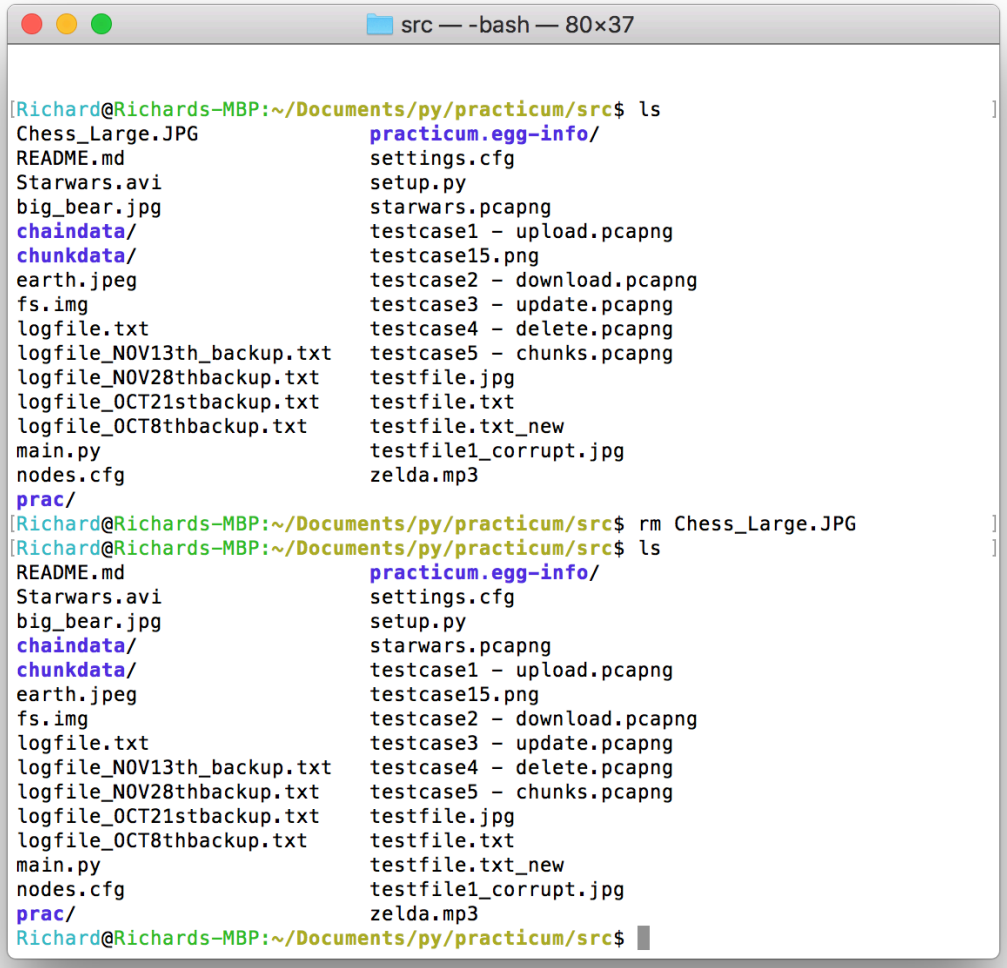
Pass Condition: The file opens after download and is intact.

Fail Condition: The file fails to download or has become corrupt.

file prior to upload

```
CMD> put Chess_Large.JPG
2018-12-05 15:56:54.784075 -  - CLIENT: # of chunks:5
2018-12-05 15:56:54.788833 -  - CLIENT: CHUNK: 714524eef8e911e8aeb8001a92daf3f8
2018-12-05 15:56:55.963487 -  - CLIENT: Sent to minion: 192.168.0.10
2018-12-05 15:56:55.964128 -  - CLIENT: CHUNK: 714524eff8e911e8aeb8001a92daf3f8
2018-12-05 15:56:57.719224 -  - CLIENT: Sent to minion: 192.168.0.10
2018-12-05 15:56:57.719643 -  - CLIENT: CHUNK: 714524f0f8e911e8aeb8001a92daf3f8
2018-12-05 15:57:07.399327 -  - CLIENT: Sent to minion: 192.168.0.11
2018-12-05 15:57:07.399967 -  - CLIENT: CHUNK: 714524f1f8e911e8aeb8001a92daf3f8
2018-12-05 15:57:08.028147 -  - CLIENT: Sent to minion: 192.168.0.10
2018-12-05 15:57:08.028818 -  - CLIENT: CHUNK: 714524f2f8e911e8aeb8001a92daf3f8
2018-12-05 15:57:08.676448 -  - CLIENT: Sent to minion: 192.168.0.17
2018-12-05 15:57:08.676923 -  - CLIENT: --->   UPLOADED FILE: Chess_Large.JPG
```

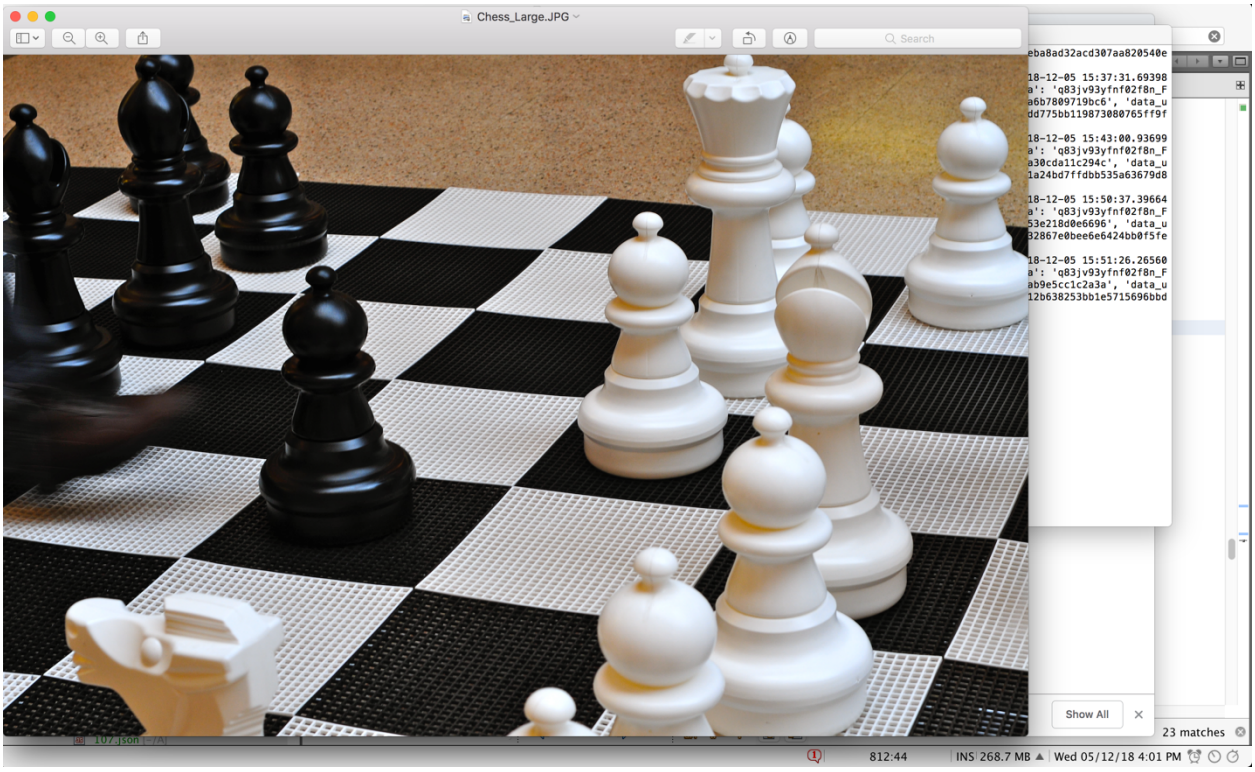
file uploaded



local copy deleted

```
CMD> get Chess_Large.JPG
2018-12-05 16:00:07.266921 -  - CLIENT: <---    DOWNLOADED FILE: Chess_Large.JPG
CMD>
```

file downloaded



file opens and is intact. note: timestamp bottom right of screenshot for reference

PASSED

Test Case 12 **- No data block containing content shall be stored in the block chain.**

Setup: The network and device has been operating for at least a short time. Files have been uploaded, and those transactions have been mined into blocks. The JSON file for those blocks can be examined.

Pass Condition: No file content in the block is present upon visual inspection.

Fail Condition: File content is discovered in the block.

{"index": 19, "version": "0.5", "timestamp": "2018-11-27 09:59:46.401907", "previous_hash": "14e77368c393a814add8708a9ece5a839c1505bd39ce19ae3a198570b7aee2df", "user_data": "q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n", "data_hash": "9be75ff613a52a356fe9a860dd0e11512b36e947e4be61ff6f13c3244a7d4cbc", "data_url": "1437a9c7f26e11e8bd12001a92daf3f8", "proof": "000000f976a0a0f7feec12aa3d187b11", "hash": "94009ef8d9624a376a619db4ed1dc7e6791a33b0f2c05a4c84564f1796bf4c0e"}

This block (index # 19) is typical of all blocks in the chain. The block contains no content, only hashes of, and proofs. No user data other than when and which device mined the block can be derived. As intended the block exists only as a verified record of the original storage network transaction.

PASSED

Test Case 13 **- Each user shall have a unique identifier.**

Setup: Each node will have had to set a user name in their configuration file. Master nodes will not accept null or duplicate user names.

Pass Condition: An active node is able to connect with the master node and thus upload and then download a file.

Fail Condition: An active node is unable to connect with the master node.

```
CMD> put Chess_Large.JPG
2018-12-05 15:56:54.784075 -  - CLIENT: # of chunks:5
2018-12-05 15:56:54.788833 -  - CLIENT: CHUNK: 714524eef8e911e8aeb8001a92daf3f8
2018-12-05 15:56:55.963487 -  - CLIENT: Sent to minion: 192.168.0.10
2018-12-05 15:56:55.964128 -  - CLIENT: CHUNK: 714524eff8e911e8aeb8001a92daf3f8
2018-12-05 15:56:57.719224 -  - CLIENT: Sent to minion: 192.168.0.10
2018-12-05 15:56:57.719643 -  - CLIENT: CHUNK: 714524f0f8e911e8aeb8001a92daf3f8
2018-12-05 15:57:07.399327 -  - CLIENT: Sent to minion: 192.168.0.11
2018-12-05 15:57:07.399967 -  - CLIENT: CHUNK: 714524f1f8e911e8aeb8001a92daf3f8
2018-12-05 15:57:08.028147 -  - CLIENT: Sent to minion: 192.168.0.10
2018-12-05 15:57:08.028818 -  - CLIENT: CHUNK: 714524f2f8e911e8aeb8001a92daf3f8
2018-12-05 15:57:08.676448 -  - CLIENT: Sent to minion: 192.168.0.17
2018-12-05 15:57:08.676923 -  - CLIENT: --->   UPLOADED FILE: Chess_Large.JPG
CMD> get Chess_Large.JPG
2018-12-05 16:00:07.266921 -  - CLIENT: <---    DOWNLOADED FILE: Chess_Large.JPG
```

PASSED

Test Case 14 **- The system shall tolerate files sizes sufficient enough for at least 4 chunks.**

Setup: The client shall upload then download 2 files. 1 Large (> 4 Mb) 1 Small (<1 Mb)

Pass Condition: Both files shall be successfully broken into at least 4 chunks prior to upload and reassembled after download.

Fail Condition: Either file fails to chunk and or upload/download.

```
Hello World
CMD> put Chess_Large.JPG
2018-12-05 19:50:44.949860 -  - CLIENT: # of chunks:10
2018-12-05 19:50:44.954364 -  - CLIENT: CHUNK: 1b91716cf90a11e8aeb8001a92daf3f8
2018-12-05 19:50:48.650965 -  - CLIENT: Sent to minion: 192.168.0.17
2018-12-05 19:50:48.651541 -  - CLIENT: CHUNK: 1b91716df90a11e8aeb8001a92daf3f8
2018-12-05 19:50:50.971129 -  - CLIENT: Sent to minion: 192.168.0.17
2018-12-05 19:50:50.971672 -  - CLIENT: CHUNK: 1b91716ef90a11e8aeb8001a92daf3f8
2018-12-05 19:50:51.846154 -  - CLIENT: Sent to minion: 192.168.0.11
2018-12-05 19:50:51.846688 -  - CLIENT: CHUNK: 1b91716ff90a11e8aeb8001a92daf3f8
2018-12-05 19:50:52.519306 -  - CLIENT: Sent to minion: 192.168.0.10
2018-12-05 19:50:52.519854 -  - CLIENT: CHUNK: 1b917170f90a11e8aeb8001a92daf3f8
2018-12-05 19:50:53.194331 -  - CLIENT: Sent to minion: 192.168.0.11
2018-12-05 19:50:53.194909 -  - CLIENT: CHUNK: 1b917171f90a11e8aeb8001a92daf3f8
2018-12-05 19:50:56.407267 -  - CLIENT: Sent to minion: 192.168.0.17
2018-12-05 19:50:56.407824 -  - CLIENT: CHUNK: 1b917172f90a11e8aeb8001a92daf3f8
2018-12-05 19:50:57.436143 -  - CLIENT: Sent to minion: 192.168.0.10
2018-12-05 19:50:57.438966 -  - CLIENT: CHUNK: 1b917173f90a11e8aeb8001a92daf3f8
2018-12-05 19:51:00.298365 -  - CLIENT: Sent to minion: 192.168.0.17
2018-12-05 19:51:00.298920 -  - CLIENT: CHUNK: 1b917174f90a11e8aeb8001a92daf3f8
2018-12-05 19:51:03.070184 -  - CLIENT: Sent to minion: 192.168.0.10
2018-12-05 19:51:03.070782 -  - CLIENT: CHUNK: 1b917175f90a11e8aeb8001a92daf3f8
2018-12-05 19:51:06.067685 -  - CLIENT: Sent to minion: 192.168.0.17
2018-12-05 19:51:06.068811 -  - CLIENT: --->   UPLOADED FILE: Chess_Large.JPG
CMD> put testfile.txt
2018-12-05 19:54:35.479065 -  - CLIENT: # of chunks:5
2018-12-05 19:54:35.480103 -  - CLIENT: CHUNK: a550bac0f90a11e8aeb8001a92daf3f8
2018-12-05 19:54:35.875499 -  - CLIENT: Sent to minion: 192.168.0.10
2018-12-05 19:54:35.875982 -  - CLIENT: CHUNK: a550bac1f90a11e8aeb8001a92daf3f8
2018-12-05 19:54:36.571745 -  - CLIENT: Sent to minion: 192.168.0.17
2018-12-05 19:54:36.572224 -  - CLIENT: CHUNK: a550bac2f90a11e8aeb8001a92daf3f8
2018-12-05 19:54:36.630356 -  - CLIENT: Sent to minion: 192.168.0.10
2018-12-05 19:54:36.631421 -  - CLIENT: CHUNK: a550bac3f90a11e8aeb8001a92daf3f8
2018-12-05 19:54:37.083924 -  - CLIENT: Sent to minion: 192.168.0.17
2018-12-05 19:54:37.084463 -  - CLIENT: CHUNK: a550bac4f90a11e8aeb8001a92daf3f8
2018-12-05 19:54:37.189820 -  - CLIENT: Sent to minion: 192.168.0.11
2018-12-05 19:54:37.190855 -  - CLIENT: --->   UPLOADED FILE: testfile.txt
CMD> get Chess_Large.JPG
File Already Exists... Overwrite? y/n:y
2018-12-05 19:59:28.608902 -  - CLIENT: <---    DOWNLOADED FILE: Chess_Large.JPG
CMD> get testfile.txt
File Already Exists... Overwrite? y/n:y
2018-12-05 19:59:43.390579 -  - CLIENT: <---    DOWNLOADED FILE: testfile.txt
CMD>
```

PASSED

Test Case 15 **- The SHA-256 algorithm shall be used as a minimum when generating secure hashes.**

Setup: The program generates hashes during the creation of a new block.

Pass Condition: By definition, A hash generated with the SHA-256 algorithm is 256 bits long. In a hexadecimal representation that is 64 characters. In any given block (other than genesis) created from the program, the 'previous_hash', 'data_hash', and 'hash' fields should contain such a 64 character representation.

Fail Condition: The 'previous_hash', 'data_hash', and 'hash' fields within any given block (other than genesis) do not contain such a 64 character representation of a SHA-256 hash.

Example block index # 42. All highlighted fields contain 64 characters indicating they are hexadecimal representations of SHA-256 Hashes.

{"index": 42, "version": "0.5", "timestamp": "2018-11-27 16:30:08.392154"
"previous_hash": "83c5bfba9488baef148b0ce79afc00583043557b0bc6b148a68de3191d6b107f",
"user_data": "q83jv93yfnf02f8n_FIRST_MINER_nf939nf03n88fnf92n",
"data_hash": "544be94aaa9f03565b5712b0c9db4d6ef1dd5ba784f5e4dcf0af54e3d5c11950",
"data_url": "a8520caef2a411e8853180e650253b46",
"proof": "000000258b29227e2519450e3ef5dd97",
"hash": "dbe749419c5f55ec26903f4da768a6170d7720b0a3c44c670782d63a7d32ffc7"}

PASSED

Test Case 16 **- If a Blockchain does not exist, A user can elect to create the first block in a new chain (Genesis)**

Setup: The local device has no blocks or Blockchain. Any peers devices that can be connected to likewise have no blocks or Blockchain.

Pass Condition: A new chain will be created with a genesis block with placeholders indicating so.

Fail Condition: The program will exit with an error.

```
[Richard@Richards-MBP:~/Documents/py/practicum/src$ python3 main.py                                                    ]
2018-11-30 17:49:53.731868 -  - Syncronizing Blockchain...
2018-11-30 17:49:55.735757 -  - ------NEW CHAIN-----
2018-11-30 17:49:55.736253 -  - NEW BLOCK:: {'index': 0, 'version': '0.5', 'timestamp': '2018-11-30 17:49:55.735677', 'previou
s_hash': '0', 'user_data': 'none', 'data_hash': 'none', 'data_url': '0', 'proof': '00000048_GENESIS_BLOCKb16e9ac6cb', 'hash':
'f71c23d6a86fcbfcd1ad9f073f8e8778f0ae904e68be1d52a026b7774fbf1fdc'}
Hello World
2018-11-30 17:49:58.251123 -  - MASTER: Port # 8100
2018-11-30 17:49:58.251970 -  - MINION: Port # 8200
CMD>
```

PASSED

Test Case 17 **- All adjustable program settings will be controlled and loaded from a configuration file.**

Setup:

Pass Condition: The program loads settings from the properly configured file and starts without error. Else if the configuration file is not configured properly the program will exit gracefully.

Fail Condition: The program will not load settings and will fail with an raised exception.

The program will not run if the configuration file is not present or if it contains errors.

```
[Richard@Richards-MBP:~/Documents/py/practicum/src$ python3 main.py
 2018-11-30 09:47:43.035679 -  - No Configuration File Present, Exiting...
[Richard@Richards-MBP:~/Documents/py/practicum/src$ python3 main.py
 2018-11-30 09:48:32.452398 -  - Error In Configuration File, Exiting...
 Richard@Richards-MBP:~/Documents/py/practicum/src$
```

Once the file is setup properly as per the user guide, the program will function.

```
[Richard@Richards-MBP:~/Documents/py/practicum/src$ python3 main.py
 2018-11-30 09:50:01.726558 -  - Syncronizing Blockchain...
 2018-11-30 09:50:22.846426 -  - NEW LONG CHAIN
```

PASSED

Test Case 18 **- Storage locations shall not be lost when the program (master node) ceases to run.**

Setup: The client uploads a file while the master is running. The master is then forced off and then on again.

Pass Condition: After the master is restart, the client is immediately able to download the uploaded file.

Fail Condition: After the master is restart, the master returns a file not found error.

```
CMD> put big_bear.jpg
2018-11-30 18:01:16.240716 -  - CLIENT: # of chunks:5
2018-11-30 18:01:16.241314 -  - CLIENT: CHUNK: fcaae9d2f50c11e893bc989096dcf54d
2018-11-30 18:01:18.503629 -  - CLIENT: Sent to minion: 192.168.0.03
2018-11-30 18:01:18.504640 -  - CLIENT: CHUNK: fcab0a20f50c11e893bc989096dcf54d
2018-11-30 18:01:20.751505 -  - CLIENT: Sent to minion: 192.168.0.02
2018-11-30 18:01:20.752338 -  - CLIENT: CHUNK: fcab5804f50c11e893bc989096dcf54d
2018-11-30 18:01:20.875207 -  - CLIENT: Sent to minion: 192.168.0.02
2018-11-30 18:01:20.875637 -  - CLIENT: CHUNK: fcab68daf50c11e893bc989096dcf54d
2018-11-30 18:01:20.887982 -  - CLIENT: Sent to minion: 192.168.0.03
2018-11-30 18:01:20.888388 -  - CLIENT: CHUNK: fcab77b2f50c11e893bc989096dcf54d
2018-11-30 18:01:20.894837 -  - CLIENT: Sent to minion: 192.168.0.03
2018-11-30 18:01:20.895162 -  - CLIENT: --->   UPLOADED FILE: big_bear.jpg
CMD>
```

client uploading big_bear.jpg

```
Hello World
2018-11-30 18:00:46.792871 -  - MASTER: Port # 8100
2018-11-30 18:00:46.793034 -  - PROGRAM COMPLETE, SERVING UNTIL MANUAL ABORT...
2018-11-30 18:01:16.126280 -  - MASTER: incomming put request('192.168.0.13', 52176)
MASTER: fcaae9d2f50c11e893bc989096dcf54d -> ['2']
MASTER: fcab0a20f50c11e893bc989096dcf54d -> ['1']
MASTER: fcab5804f50c11e893bc989096dcf54d -> ['1']
MASTER: fcab68daf50c11e893bc989096dcf54d -> ['2']
MASTER: fcab77b2f50c11e893bc989096dcf54d -> ['2']
```

master running and accepting put request from client

```
Hello World
2018-11-30 18:02:52.896387 -  - MASTER: Port # 8100
2018-11-30 18:02:52.896536 -  - PROGRAM COMPLETE, SERVING UNTIL MANUAL ABORT...
2018-11-30 18:03:14.000929 -  - MASTER: incomming get request('192.168.0.13', 52188)
```

master is restart (note: the hello world that always appears on program start) and receives the get request.

```
CMD> get big_bear.jpg
File Already Exists... Overwrite? y/n:y
2018-11-30 18:03:18.424109 -  - CLIENT: <---    DOWNLOADED FILE: big_bear.jpg
CMD>
```

Client has run the get request and retrieves the file successfully.

PASSED

Test Case 19 **- Blocks with duplicate indexes shall be orphaned to maintain a tidy Blockchain.**

Setup: The Program is running and the device attempts to synchronize with the Blockchain network. 2 devices on the network have near simultaneously created blocks with the same index number but from different network transactions.

Pass Condition: The late block is adopted into the chain but is flagged as an orphan to differentiate the block from the block which was completed just prior.

Fail Condition: The Blockchain is unintentionally forked, effectively resulting in multiple chains.

```
CMD> sync
2018-12-05 18:21:56.909976 -  - Syncronizing Blockchain...
2018-12-05 18:21:58.377907 -  - NEW LONG CHAIN
2018-12-05 18:21:58.379475 -  - ABOPTING BLOCK:: {'index': 0, 'version': '0.5', 'timestamp': '2018-12-05 18:16:16.690986', 'previous_hash
': '0', 'user_data': 'none', 'data_hash': 'none', 'data_url': '0', 'proof': '00000048_GENESIS_BLOCKb16e9ac6cb', 'hash': '35418378ccbb0095
f85c2d174293dbed8e56f392be99f3a2f5d5d981c1521de9'}
2018-12-05 18:21:58.380918 -  - ABOPTING BLOCK:: {'index': 1, 'version': '0.5', 'timestamp': '2018-12-05 18:17:36.131923', 'previous_hash
': '35418378ccbb0095f85c2d174293dbed8e56f392be99f3a2f5d5d981c1521de9', 'user_data': 'q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n', 'd
ata_hash': '73bfaac6ddcdc7aba50756413077af751949651303c1f714b482ad262e93d07e', 'data_url': '063d7d19f8fd11e8aeb8001a92daf3f8', 'proof': '
0000008fd5b903696597f2a71d9b459e', 'hash': '31ba53a6e7d97996f59bac113269fe5aa193af5e9e6ab94e75ae9401e0002d4f'}
2018-12-05 18:21:58.381996 -  - ABOPTING BLOCK:: {'index': 2, 'version': '0.5', 'timestamp': '2018-12-05 18:19:20.890444', 'previous_hash
': '31ba53a6e7d97996f59bac113269fe5aa193af5e9e6ab94e75ae9401e0002d4f', 'user_data': 'q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n', 'd
ata_hash': '6a42b19b0cdcbae8fe517dd3d8411ca0e6f3f7621e8fbf45d658d8957496499f', 'data_url': '063d7d1bf8fd11e8aeb8001a92daf3f8', 'proof': '
0000004026e012e7d1cbf0cac0ad0776', 'hash': 'f277e2603829cb428a50f00163d740287f0f8d36ac4e208dbb58e63b809970e8'}
2018-12-05 18:21:58.383054 -  - ABOPTING BLOCK:: {'index': 3, 'version': '0.5', 'timestamp': '2018-12-05 18:19:30.796486', 'previous_hash
': 'f277e2603829cb428a50f00163d740287f0f8d36ac4e208dbb58e63b809970e8', 'user_data': 'q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n', 'd
ata_hash': '544be94aaa9f03565b5712b0c9db4d6ef1dd5ba784f5e4dcf0af54e3d5c11950', 'data_url': '2e6117faf8fd11e8aeb8001a92daf3f8', 'proof': '
0000006f0d6d1c537fd252fe6da20dae', 'hash': '79b26f8789e325464c012bd06d963b1941e67629f3ebbed57ad53be46427605b'}
2018-12-05 18:21:58.384103 -  - ABOPTING BLOCK:: {'index': 4, 'version': '0.5', 'timestamp': '2018-12-05 18:19:50.653530', 'previous_hash
': '79b26f8789e325464c012bd06d963b1941e67629f3ebbed57ad53be46427605b', 'user_data': 'q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n', 'd
ata_hash': '09469f79daaa67fc6cf55b1f1d468da9cb903c76d84a0e92027a65182ffe8cf6', 'data_url': '2e6117fdf8fd11e8aeb8001a92daf3f8', 'proof': '
000000fc34c66bea8df8c9c18c70564e', 'hash': 'f776fc96826aa78620366093ae933b4a28d19cf19efe781bc138395491e151e9'}
2018-12-05 18:21:58.385075 -  - ABOPTING BLOCK:: {'index': 5, 'version': '0.5', 'timestamp': '2018-12-05 18:20:12.045418', 'previous_hash
': 'f776fc96826aa78620366093ae933b4a28d19cf19efe781bc138395491e151e9', 'user_data': 'q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n', 'd
ata_hash': '242ea344a5d559e6b70b739c911a71faa535c68a1f4f667f7b1ab9e5cc1c2a3a', 'data_url': '2e6117fef8fd11e8aeb8001a92daf3f8', 'proof': '
000000adb9d3903a059fff3888c21306', 'hash': '0c258d7fe0408dd7c1b6de15d9e41f127efd219cb8595b58371364f3aebe680a'}
CMD> sync
2018-12-05 18:26:56.970866 -  - Syncronizing Blockchain...
2018-12-05 18:26:58.757439 -  - NEW LONG CHAIN
2018-12-05 18:26:58.758978 -  - ABOPTING BLOCK:: {'index': 0, 'version': '0.5', 'timestamp': '2018-12-05 18:16:16.690986', 'previous_hash
': '0', 'user_data': 'none', 'data_hash': 'none', 'data_url': '0', 'proof': '00000048_GENESIS_BLOCKb16e9ac6cb', 'hash': '35418378ccbb0095
f85c2d174293dbed8e56f392be99f3a2f5d5d981c1521de9'}
2018-12-05 18:26:58.760437 -  - ABOPTING ORPHAN BLOCK:: {'index': '1B', 'version': '0.5', 'timestamp': '2018-12-05 18:17:46.309324', 'pre
vious_hash': '35418378ccbb0095f85c2d174293dbed8e56f392be99f3a2f5d5d981c1521de9', 'user_data': 'q83jv93yfnf02f8n_SECOND_MINER_nf939nf03n88
fnf92n', 'data_hash': 'cc0827bd85481caeed5cbec46fbbc8eb31a77feda2a5bb4d20bfb910b63b2eba', 'data_url': '063d7d18f8fd11e8aeb8001a92daf3f8',
 'proof': '0000008fd5b903696597f2a71d9b459e', 'hash': '8ec20d4e68edb636447176d9254d6f7fad216bb5aa441a8c2c88fd58cb1cafac'}
2018-12-05 18:26:58.763191 -  - ABOPTING BLOCK:: {'index': 6, 'version': '0.5', 'timestamp': '2018-12-05 18:24:06.366424', 'previous_hash
': '0c258d7fe0408dd7c1b6de15d9e41f127efd219cb8595b58371364f3aebe680a', 'user_data': 'q83jv93yfnf02f8n_SECOND_MINER_nf939nf03n88fnf92n', '
data_hash': '4c0fc970973412b9bbfd838c8501f7c0179c60e3f4c732d1ee0cfa13a32b3fd0', 'data_url': '063d7d1af8fd11e8aeb8001a92daf3f8', 'proof':
'000000980208c99eb510796eccb213f3', 'hash': '4dcd1c84037c6aa1cc1bc57392d863b6bf0d2401d9f5d31049dea65e68195841'}
2018-12-05 18:26:58.764261 -  - ABOPTING BLOCK:: {'index': 7, 'version': '0.5', 'timestamp': '2018-12-05 18:24:09.942777', 'previous_hash
': '4dcd1c84037c6aa1cc1bc57392d863b6bf0d2401d9f5d31049dea65e68195841', 'user_data': 'q83jv93yfnf02f8n_SECOND_MINER_nf939nf03n88fnf92n', '
data_hash': '5eb1bd21671d55f21be54ff010dbeb82352bb5e8f61edeaabd053e218d0e6696', 'data_url': '2e6117fbf8fd11e8aeb8001a92daf3f8', 'proof':
'000000425e20a96bb982ac6a3518d21c', 'hash': 'c8c9e1aacb9bc1b25feca1efc0320cce7e23364cb463f07b9efae9cac515853b'}
2018-12-05 18:26:58.765289 -  - ABOPTING BLOCK:: {'index': 8, 'version': '0.5', 'timestamp': '2018-12-05 18:24:15.183739', 'previous_hash
': 'c8c9e1aacb9bc1b25feca1efc0320cce7e23364cb463f07b9efae9cac515853b', 'user_data': 'q83jv93yfnf02f8n_SECOND_MINER_nf939nf03n88fnf92n', '
data_hash': '2439e76509728dfd6f2c99d626cc704d3c284425665b2b385f4f237acdd7157e', 'data_url': '2e6117fcf8fd11e8aeb8001a92daf3f8', 'proof':
'0000004c8ab9e0a067f302b7748ec375', 'hash': 'f83128d88ae949ea81fbe83d41618e264daefd4a81ef817746b69abc6649d7ce'}
2018-12-05 18:26:58.766212 -  - ABOPTING BLOCK:: {'index': 9, 'version': '0.5', 'timestamp': '2018-12-05 18:25:27.050223', 'previous_hash
': 'f83128d88ae949ea81fbe83d41618e264daefd4a81ef817746b69abc6649d7ce', 'user_data': 'q83jv93yfnf02f8n_SECOND_MINER_nf939nf03n88fnf92n', '
data_hash': '242ea344a5d559e6b70b739c911a71faa535c68a1f4f667f7b1ab9e5cc1c2a3a', 'data_url': '2e6117fef8fd11e8aeb8001a92daf3f8', 'proof':
'00000059a9e5fe8ae28573f9e1ba651d', 'hash': 'e669ea86fc7259fa79aa82a1a38a0b20adc7f63c97d5eb38db159aee91b31e69'}
CMD> sync
2018-12-05 18:32:03.845706 -  - Syncronizing Blockchain...
CMD>
```

The chain is synchronized and blocks 0 thru 5 are added to the local chain. After some time, the chain is synchronized again, a block with the same index number (1) is downloaded and is flagged as an orphan (1B).

PASSED

Test Case 20 **- Nodes can obtain the Blockchain from their peers.**

Setup: Client device attempts to synchronize their local block chain with an already functioning network with at least 2 peers. This is done either automatically or by manually entering the command ‘sync’.

Pass Condition: The client successfully downloads blocks and adopts them as needed.

Fail Condition: The client fails to download any blocks.

```
[Richard@Richards-MBP:~/Documents/py/practicum/src$ python3 main.py
2018-12-05 16:53:27.149584 -  - Syncronizing Blockchain...
2018-12-05 16:53:27.558498 -  - NEW LONG CHAIN
2018-12-05 16:53:27.560649 -  - ABOPTING BLOCK:: {'index': 0, 'version': '0.5', 'timestamp': '2018-12-05 16:50:01.036182', 'previous_hash
': '0', 'user_data': 'none', 'data_hash': 'none', 'data_url': '0', 'proof': '00000048_GENESIS_BLOCKb16e9ac6cb', 'hash': 'bb916d157808b8d8
88a5d888c6e96424a59801521ebe3aefef9618d5eed9e2c7'}
Hello World
CMD> put Chess_Large.JPG
2018-12-05 16:54:08.268808 -  - CLIENT: # of chunks:5
2018-12-05 16:54:08.272007 -  - CLIENT: CHUNK: 6f85b1def8f111e8aeb8001a92daf3f8
2018-12-05 16:54:14.292920 -  - CLIENT: Sent to minion: 192.168.0.17
2018-12-05 16:54:14.293458 -  - CLIENT: CHUNK: 6f85b1dff8f111e8aeb8001a92daf3f8
2018-12-05 16:54:21.872792 -  - CLIENT: Sent to minion: 192.168.0.11
2018-12-05 16:54:21.873429 -  - CLIENT: CHUNK: 6f85b1e0f8f111e8aeb8001a92daf3f8
2018-12-05 16:54:27.953938 -  - CLIENT: Sent to minion: 192.168.0.17
2018-12-05 16:54:27.959016 -  - CLIENT: CHUNK: 6f85b1e1f8f111e8aeb8001a92daf3f8
2018-12-05 16:54:34.161619 -  - CLIENT: Sent to minion: 192.168.0.17
2018-12-05 16:54:34.162283 -  - CLIENT: CHUNK: 6f85b1e2f8f111e8aeb8001a92daf3f8
2018-12-05 16:54:41.840927 -  - CLIENT: Sent to minion: 192.168.0.17
2018-12-05 16:54:41.841928 -  - CLIENT: --->   UPLOADED FILE: Chess_Large.JPG
CMD> sync
2018-12-05 16:59:38.414486 -  - Syncronizing Blockchain...
2018-12-05 16:59:39.543729 -  - NEW LONG CHAIN
2018-12-05 16:59:39.545836 -  - ABOPTING BLOCK:: {'index': 1, 'version': '0.5', 'timestamp': '2018-12-05 16:54:41.763265', 'previous_hash
': 'bb916d157808b8d888a5d888c6e96424a59801521ebe3aefef9618d5eed9e2c7', 'user_data': 'q83jv93yfnf02f8n_SECOND_MINER_nf939nf03n88fnf92n', '
data_hash': 'cc0827bd85481caeed5cbec46fbbc8eb31a77feda2a5bb4d20bfb910b63b2eba', 'data_url': '6f85b1def8f111e8aeb8001a92daf3f8', 'proof':
'0000008fd5b903696597f2a71d9b459e', 'hash': '92b63ac105551905fbae520fec15caeb3919a07bf66d7379d0adf4808c2826de'}
2018-12-05 16:59:39.547003 -  - ABOPTING BLOCK:: {'index': 2, 'version': '0.5', 'timestamp': '2018-12-05 16:58:41.896274', 'previous_hash
': '92b63ac105551905fbae520fec15caeb3919a07bf66d7379d0adf4808c2826de', 'user_data': 'q83jv93yfnf02f8n_SECOND_MINER_nf939nf03n88fnf92n', '
data_hash': '73bfaac6ddcdc7aba50756413077af751949651303c1f714b482ad262e93d07e', 'data_url': '6f85b1dff8f111e8aeb8001a92daf3f8', 'proof':
'0000004026e012e7d1cbf0cac0ad0776', 'hash': 'c0c5f4e5525170ad429c98e71a0d8b4dc0979035909207bf44c8c334aba08ab6'}
2018-12-05 16:59:39.547917 -  - ABOPTING BLOCK:: {'index': 3, 'version': '0.5', 'timestamp': '2018-12-05 16:56:24.764671', 'previous_hash
': '67e3860cb14017eb933dfb84baad9a05c6f603bfc360ba03bd7f9f767665e34f', 'user_data': 'q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n', 'd
ata_hash': '99427041938b531493f76c0bf5057d89356cfc7ea9dada750835ad1a0abed1cf', 'data_url': '6f85b1e2f8f111e8aeb8001a92daf3f8', 'proof': '
0000006f0d6d1c537fd252fe6da20dae', 'hash': '8a3a5b3acf866213cb4c07a56908bab16e711fc98fab94523778020bfe1753c6'}
2018-12-05 16:59:39.548798 -  - ABOPTING BLOCK:: {'index': 4, 'version': '0.5', 'timestamp': '2018-12-05 16:59:34.278207', 'previous_hash
': '8a3a5b3acf866213cb4c07a56908bab16e711fc98fab94523778020bfe1753c6', 'user_data': 'q83jv93yfnf02f8n_SECOND_MINER_nf939nf03n88fnf92n', '
data_hash': '4c0fc970973412b9bbfd838c8501f7c0179c60e3f4c732d1ee0cfa13a32b3fd0', 'data_url': '6f85b1e0f8f111e8aeb8001a92daf3f8', 'proof':
'000000fc34c66bea8df8c9c18c70564e', 'hash': '51d811b980d49a39f49ca05d54c8519d3ce4615742646cee6d3c695c9184183e'}
CMD>
```

PASSED

2.8.Implications of Implementation / Non-Functional Requirements

2.8.1. Extensibility, Coupling & Scope

The implementation of the application shall have in mind potential future growth. An appropriate balance of module cohesion and coupling will be maintained. Not only will this aid development and debugging for the duration of the project but will allow for later use and modification if desired.

Managing scope has had to be a priority. Particularly towards project end and sprints 4 and 5. The project has really started to come together at this point and not previously thought of ideas surface. Some of those ideas are discussed under future enhancements. While at this point in time classes are long finished, this project's start has been delayed by several months. Personal problems had repeatedly interrupted project proposal submissions. While those issues have been resolved, there is a reasonably limited time period in which this project could be completed. Prioritizing those features which are most important and taking steps to avoid deadline slippage has been important. The schedule and deadlines for sprint completion have been kept to rigidly.

2.8.2. Security

Security is of upmost importance and must be considered at all times and at all stages of development. Indeed, the very problem that this project seeks to combat is that of data security. Many existing cloud storage architectures have weak points in their own security such as a single, centralized storage location and little regard for data integrity. This projects aim is to not only provide for a more secure means of storage for users but also for the project to be considered secure in and of itself.

2.8.3. Reliability, Performance, Performance and Concurrency

Any applications made for this project have been designed with the intent of creating a stable and functional deliverable applications by project end. Testing will be as in depth as resources allow. Failures will be tracked during testing, fixed and tested for again to ensure that the application functions as it was intended. If an unexpected error were to occur, the application shall fail gracefully, allowing for further debugging effort. The final application shall be reliable as to pass test cases.

The application shall function in a timely manner as to not feel like it is taking 'too long' by users. Encryption and decryption functions can be computationally expensive. Upload and download rates can vary widely system to system. It is the designers intent to not waste system resources where it is not required.

It is a goal to have the application function in a fashion that does not feel slow to the user. To have a responsive networking application means multi-threading and other means of concurrency. Programming techniques and tools learned about in classes will be employed here to no doubt great effect.

2.9.Innovation

This project and the system created explored several modern computing concepts:

Block Chain technology is used in platforms such as Bitcoin and Etherium and provides a means for verifying electronic transactions between parties. Those transactions can involve cryptocurrency as in the aforementioned examples, but also other forms of processed data, records or other information. Once verified by a network of peers, those transactions are added as fresh blocks to the existing block chain creating a running ledger of all verified transactions. Block chain aims to solve the issue of trust between parties. For this project, the block chain is used as a method of trade. Only links, addresses and hash values are stored within block chain. Actual data is stored across random distributed nodes.

Distributed, Decentralized Storage removes the threat of single point of failure. A file is divided into chunks and encrypted individually. Data chunks are uploaded and are stored randomly to different nodes in the network. Their location addresses will be stored in the block chain. The capture of a single block will not result in any revealed data. The application will be in communication with other nodes in the network. Both for the purposes of creating and adding to the Blockchain but also for remote data storage. Network nodes may be unavailable and connections may be lost at unexpected times. The system should be able to handle and recover from such scenarios with a certain grace.

File Integrity Verification. In each block, the result of a pseudo-random hash (SHA-256) operation based on all transactions is stored. Through this process, it is possible to verify file integrity. The source of change to a file stored on the network can be pinpointed through this process. This will allow for an Administrator or security minded user to carry out investigation and/or carry out relevant action.

2.10. Complexity

This project involves several areas that should not be considered trivial.

2.10.1. Block chain technology

Blockchain tech has experienced a great deal of growth in interest and usage over the past 2 years. New developments in this area are made constantly and new use cases are being discovered. Increasingly, the principles are being applied to solve needs beyond cryptocurrencies. In the case of this project, the application of a block chain ledger will afford accountability and file integrity.

Although I've studied block chains at a very basic level, I have not yet fully implemented my own nor have I been involved with cryptocurrency in a technical capacity at all outside of school. I am naturally skeptical but also curious as to the possible genuine benefits for users concerning areas of privacy and data protection. I am currently and will continue to read additional papers, explore existing open-source designs, and experiment with working examples if possible to determine the most appropriate method of implementation. The preference is towards developing my own Blockchain from scratch with the aim of understanding the underlying technology better. The alternative would be to develop on an existing Blockchain platform such as Etherium although that may come with some unwanted financial cost.

2.10.2. Distributed storage

Writing functional software for distributed systems of any kind can be a very complex challenge. Failure of individual components, downed nodes and communication delays can prove difficult to overcome. The rigorous application of mathematics and finding the optimal algorithms are crucial to successful operation.

2.10.3. Data and Privacy Protection

The European Union has instituted new laws and regulations this year that are forcing many companies, even those operating outside the EU to rethink their polices of data capturing and storage. Companies wishing to move their mission critical data or processes to the 'cloud' will require a secure, verifiable supply chain. Currently, no major providers of cloud services provide for the power of a full audit or give a CIO or accountant sufficient peace of mind for the full acceptance of liability. This notion partly inspired this project. Although there may not obvious and deliberate steps to fulfil any GDPR requirements, The project will implement with those requirements in mind. This is because of the challenge that companies both large and small have to face in understand those requirements, and implementing changes in order to be compliant.

2.11. Future Enhancements

In retrospect, several areas of the project can be immediately further developed to enhance capability.

2.11.1. Blockchain Visualizer:

As the block chain files are stored and transmitted in JSON format, it might be a good exercise to design a simple graphical interpreter to map out the Blockchain relationships so they can be viewed, and clicked on for inspection. Although it might look kind of cool, this may prove a futile way for an administrator to inspect the chain however, the chain will no doubt become too long in practice for such an interface to be effective.

2.11.2. Transaction Batching:

One area that needs to be made a priority is the backlog of transactions. There is the reality that at present it takes a lot long for a block to be mined then it does for a data chunk to be stored and distributed. If the storage network is endlessly performing such transactions, the backlog would grow out of control and the block chain will not be able to keep up. A solution is to batch transactions together, making each block potentially more complex but there would be less blocks overall. A more radical solution is to alter the transaction itself, having it represent more work on the nodes behalf.

2.11.3. Storage Minion Heartbeat and Load Balancer:

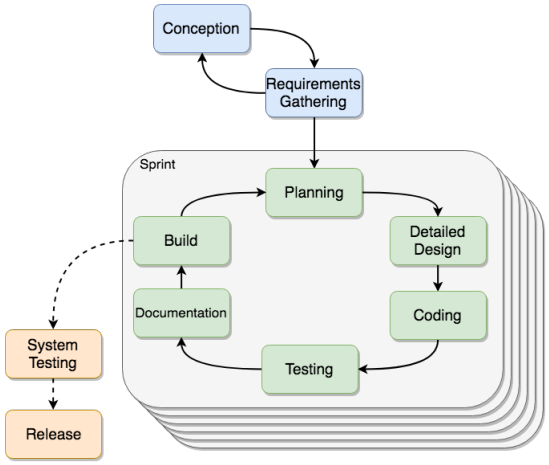
A lost opportunity in early design. Too much reliance was made on each node being manually configured. This was mainly due to each device having different or unknown capabilities and therefore different roles. For example, one machine used during development was running Windows 10 and could not perform the full duties of a client. This was because of the python cryptology library not reading end of file the same on Windows as it would on Linux. For Tests this was acceptable, as the machine was by far the fastest computer available, and could still perform storage minion and mining duties extremely well. Still though, as is present in many available distributed storage systems, each minion node regularly reports it status to the master node. The master node can maintain a list of each node and how much storage capacity it has available. For a minion to be ‘registered’ all it has to do is connect with the master the first time. No configuration per say of the master is needed, and the minion need only know the masters address.

2.11.4. Automatic List of Peers

Like above, too much reliance was made on manual configuration. At each node the settings files must contain a list of every other peer.

2.12. Timeline, Methodology and Milestones

The development methodology used for this project was a modified agile process consisting of two week long sprints each totaling 80 hours of work time. There were 5 Sprints. This flowchart , schedule and following details, outline the projects overall development and work flow.



2.12.1. Proposed Timeline

Timeline	Milestones	Breakdown	Time
October 1st to October 15th	Sprint 1 Blockchain Layer	Design, UML & Flow Diagrams	12
		Chain Genesis	12
		Blockchain Network	12
		Ledger	12
		Transaction Verification & Consensus	12
		Testing	12
		Documentation	8
			80 Hours
October 15th to October 29th	Sprint 2 Storage Layer	Design, UML & Flow Diagrams	12
		Data Storage Network	16
		Random Node Assignment	16
		Block Storage	16
		Testing	12
		Documentation	8
			80 Hours
October 29th to November 12th	Sprint 3 Client Layer	Design, UML & Flow Diagrams	12
		User Interface	8
		File & Block Management	10
		Encryption & Decryption	10
		Network Connections	10
		Transaction Requests	10
		Testing	12
		Documentation	8
			80 Hours
November 12th to November 26th	Sprint 4 Integration	Integration, Tests and Completion	50
		Documentation	30
			80 Hours
November 26th to December 10th	Sprint 5 Delivery	User Guide	30
		Prepare & Deliver Demonstration	15
		Final Report	30
			85 Hours
Estimated Total Hours			405 Hours

2.12.2. Actual Timeline

Timeline	Milestones	Breakdown	Time
October 1st to October 15th	Sprint 1 Blockchain Layer	Design, UML & Flow Diagrams	12
		Chain Genesis	12
		Blockchain Network	12
		Ledger	12
		Transaction Verification &Consensus	12
		Testing and Fixes	12
		Documentation	8
			80 Hours
October 15th to October 29th	Sprint 2 Client Layer	Design, UML & Flow Diagrams	12
		User Interface	8
		File & Block Management	10
		Encryption & Decryption	10
		Network Connections	10
		Transaction Requests	10
		Testing and Fixes	12
		Documentation	8
			80 Hours
October 29th to November 12th	Sprint 3 Storage Layer	Design, UML & Flow Diagrams	12
		Data Storage Network	16
		Random Node Assignment	16
		Block Storage	16
		Testing and Fixes	12
		Documentation	8
			80 Hours
November 12th to November 30th	Sprint 4 Integration	Integration, Tests and Completion	70
		Documentation	30
			100 Hours
November 30th to December 10th	Sprint 5 Delivery	User Guide	10
		Prepare Demonstration	15
		Final Report	30
			65 Hours
Total Hours			405 Hours

2.12.3. Timeline Progression and Adjustments

Sprint 1 - Blockchain Layer

This sprint went well even though I'd never had made a working Blockchain before. Still, it was this area of the project concerning Blockchain that I had been become most familiar with before the formal approval and start. I decided to use the JSON format to store blocks. This would allow local backup and ease transmission. An early proof of work method for mining blocks worked well early on but basically just became too slow when the blocks grew too high in quantity. A new proof of work based more on how the successful Blockchain examples operate was implemented. This should have been more obvious early on as the original proof of work could have easily been faked by a network ne'er do well. The system was then built to serve Blockchain requests to other peers and dummy data was used to create test transactions and then mine blocks with more dummy data fitting the anticipated data type structures. Although a hollow shell, the block chain ledger was functional and blocks could be verified against the original dummy generated data.

Sprint 2 - Client Layer

It was quickly realized that I needed to work on the client layer prior to the storage layer. The dummy data used in development of the Blockchain layer would not suffice moving forward and I wanted to have test machines interacting with one another as soon as possible.

Once machines were mining their own 'fake' transaction and blocks and sharing these with each other, the matter of simultaneous block mining became apparent and had be dealt with. Many tweaks were made to design and many areas recoded to try and smooth the process out but it's a reality that some machines are faster than others and will finish mining blocks sooner. More research was made into how block chains handle this issue and the orphaning of blocks seemed the best solution. A block could potentially be used as the 'previous block' for more than one block if two machine start work at the same time. The Blockchain ledger would start to resemble an awkward tree or net rather than a long thin chain as is intended. By orphaning the subsequent blocks, preventing them from being used by the chain to move forward, they are still part of the chain but are considered 'dead ends' and will not act as the 'previous block' to any future blocks in the ledger.

Although the interface itself is really simple, the client layer involved a great deal of code, delete and recode and triggered a great deal of thought into other project details. Many of these would trickle into the Storage layer.

Sprint 3 - Storage Layer

I got to this point and realized I didn't have a great idea of how my distributed storage network would operate. As with the Blockchain module, I'd never built such a system before. I had already constructed a network of interacting machines for the Blockchain and client layers so I decided to further utilize that. In their settings files, the roles of machines could be set. With the Blockchain sharing service running as a daemon thread as inspiration, I created additional optional daemons to support storage requests. A client would contact a master node, who would deliver meta data concerning the storage nodes back to the client. The client could then use that meta data to break up the file into the (encrypted) chunks prior to transmission as required by the project and then send (and subsequently retrieve) those chunks to (and from) the storage nodes. This began to mimic a simple Hadoop distributed file system which I was not very familiar with at the time. It's through further reading about this type of system that possible future enhancements were thought of, minion heartbeat for example. As the storage system grew I had to remain in scope to achieve sprint deadline. I only had wished I'd had taken more time to research distributed storage methods before settling on the master / minion nodes. I may have seen the parallels in other systems such as Hadoop. A redesign, although impractical at this stage would have probably resulted in a faster and more robust system. Lesson learned.

Sprint 4 - Integration

This sprint had the potential for disaster. Up to this point the Blockchain process and the Storage process were only linked in design and theory. As each storage node took in data chunks, they now generated transactions with actual data. Those transactions would now be mined into blocks, verified and added the Blockchain ledger. This proceeded well until Nov 13th when my cable connection to my internet service provider went down. It took a good part of 2 frustrating days to have the problem, which was caused in error by my service provider, to be corrected by said service provider. Some unfortunate timing but was overcome eventually. More extensive stand up tests were performed at home and in the Datacomm lab. A few complications involving version differences were solved by meticulous installation of the required python and cryptology libraries.

Prioritizing those features which are most important and taking steps to avoid deadline slippage has been important. The schedule and deadlines for sprint completion have been kept to rigidly.

Sprint 5 - Delivery

This sprint represents the final stages of the project. The only challenge here is to stop any fine tuning of the application, remain in scope, and complete all other deliverables by deadline. The application is functioning as intended and there is no time available for any further desired modification.

3. Conclusion and Lessons

The application developed for this project completes the goals set out during proposal. That was to create a working model for a decentralized, distributed block chain based secure 'cloud' file storage application. I had Initial difficulty in formulating a satisfactory and suitable project idea. It wasn't until I had begun reading articles about new laws in the European Union that were creating new challenges for online businesses that a theme came together. Many international companies and cloud providers were seeking to ensure that they were in legal compliance and wanted to convince clients that they truly did have their clients best interests in mind. Although I wasn't thinking in terms of how I could build a project to comply with such laws, after all such a project is well beyond my scope, I thought of what kind of potential solutions could I experiment with and thought of two. A distributed storage network would eliminate any single point of failure and would greatly aid security with file chunking and encryption. Second, a Blockchain ledger using the storage networks transactions as payload would add further security and accountability.

The project created an opportunity to complete an application where many different modern and possible future networking and security principles could be researched and explored. Having the application components all operating together in sprint 4, even in rough form was satisfying. All the more satisfying to complete as designed. Still, the application is begging for the enhancements I've thought of in hind-sight. I'd like an opportunity if given one by a future employer to work on something similar and more capable. I've added greatly to my education by completing this project. A good portion of the project involved practical use of hash algorithms, both in the Blockchain ledger and storage. There was network programming, data encryption, data management. I no doubt applied many other skills from both information technology diploma and Btech programs. I look forward to taking the lessons learned from this project and program and applying them to gain some much needed experience in the industry.

4. Appendix

4.1. List of Figures

Figure 1 - high level design 5
Figure 2 - Project Structure 10
Figure 3 - client node design work flow 11
Figure 4 - settings.cfg - configuration file 12
Figure 5 - genesis block data 13
Figure 6 - typical network setup during development as shown in attached logs 14
Figure 7 - block mining design flow 15
Figure 8 - block class diagram 15
Figure 9 - visual graph of Blockchain with an orphan block 16
Figure 10 - Blockchain sharing and consensus design work flow 17
Figure 11 - storage master design work flow 18
Figure 12 - storage node assignment performed by a master node 18
Figure 13 - storage node design work flow 19
Figure 14 - default service ports for configuration file 21

4.2.Typical Run Through

4.2.1. Client

```
Richard@Richards-MBP:~/Documents/py/practicum/src$ python3 main.py
2018-12-07 09:42:08.662206 -  - Synchronizing Blockchain...
2018-12-07 09:42:11.507753 -  - NEW LONG CHAIN
2018-12-07 09:42:11.509126 -  - ABOPTING BLOCK:: {'index': 5, 'version': '0.6', 'timestamp': '2018-12-
07 09:31:20.903075', 'previous_hash':
'10d7785336043dcb23fde757f748d3e16782d413a07df3b29db257381f25f494', 'user_data':
'q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n', 'data_hash':
'242ea344a5d559e6b70b739c911a71faa535c68a1f4f667f7b1ab9e5cc1c2a3a', 'data_url':
'8e49ab60fa4511e8b2ee001a92daf3f8', 'proof': '000000adb9d3903a059fff3888c21306', 'hash':
'6d5d6a1aa05a905510f6ca175f88f4fe2bef3cc198bb68f1e77a512766a5d625'}
2018-12-07 09:42:11.510814 -  - ABOPTING BLOCK:: {'index': 6, 'version': '0.6', 'timestamp': '2018-12-
07 09:32:20.573838', 'previous_hash':
'6d5d6a1aa05a905510f6ca175f88f4fe2bef3cc198bb68f1e77a512766a5d625', 'user_data':
'q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n', 'data_hash':
'8158b46a89cee8b92e7a030a489408761fe1e8b3f9de73d9d1c705461d810277', 'data_url':
'cb4d8406fa4511e8b2ee001a92daf3f8', 'proof': '000000980208c99eb510796eccb213f3', 'hash':
'76ed7cd802a12ab1fc5c6593e93d471cd04a292d17a6823021f07c160794be27'}
2018-12-07 09:42:11.511772 -  - ABOPTING BLOCK:: {'index': 7, 'version': '0.6', 'timestamp': '2018-12-
07 09:32:22.208233', 'previous_hash':
'76ed7cd802a12ab1fc5c6593e93d471cd04a292d17a6823021f07c160794be27', 'user_data':
'q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n', 'data_hash':
'210ac90d86661a3cf1e927f117c676c7a250982aa4ded654063ae8747da8e4c7', 'data_url':
'cb4d8407fa4511e8b2ee001a92daf3f8', 'proof': '000000425e20a96bb982ac6a3518d21c', 'hash':
'c64f1a5248be28a96197848e6c961cb2a4b08d37ef6f40110c948f8131bd92f0'}
2018-12-07 09:42:11.512615 -  - ABOPTING BLOCK:: {'index': 8, 'version': '0.6', 'timestamp': '2018-12-
07 09:32:24.368465', 'previous_hash':
'c64f1a5248be28a96197848e6c961cb2a4b08d37ef6f40110c948f8131bd92f0', 'user_data':
'q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n', 'data_hash':
'9f3e7d3d0158cc95d08c47a49cbae16e4c311c82247d6ac3b68a695cf628cadc', 'data_url':
'cb4d8408fa4511e8b2ee001a92daf3f8', 'proof': '0000004c8ab9e0a067f302b7748ec375', 'hash':
'070d253c01f69c723a2dee0e14fead65dc6bc0353972b7cc00718737ceebd743'}
2018-12-07 09:42:11.513436 -  - ABOPTING BLOCK:: {'index': 9, 'version': '0.6', 'timestamp': '2018-12-
07 09:32:52.404223', 'previous_hash':
'070d253c01f69c723a2dee0e14fead65dc6bc0353972b7cc00718737ceebd743', 'user_data':
'q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n', 'data_hash':
'2d2dd09223f986a1bdf433de91f2e8dc740dce6f3397b33e4d9477f1e65fdc2f', 'data_url':
'cb4d8409fa4511e8b2ee001a92daf3f8', 'proof': '00000059a9e5fe8ae28573f9e1ba651d', 'hash':
'576ff88e8253eca73e4db20b930183ce15b5aff835980da421783800e292d0ac'}
2018-12-07 09:42:11.514272 -  - ABOPTING BLOCK:: {'index': 10, 'version': '0.6', 'timestamp': '2018-12-
07 09:33:11.835289', 'previous_hash':
'576ff88e8253eca73e4db20b930183ce15b5aff835980da421783800e292d0ac', 'user_data':
'q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n', 'data_hash':
'1b4b993cb1ae756d9f657d9a9f3d6334e542bbd091e3237287cae8feeac34e39', 'data_url':
'cb4d840afa4511e8b2ee001a92daf3f8', 'proof': '0000008c16aaebf2d56ffe227dc05747', 'hash':
'1c77a54e39be8d3eb28e612a430ef8443f3a3d0b13586e861c469ef065269092'}
2018-12-07 09:42:11.515285 -  - ABOPTING BLOCK:: {'index': 11, 'version': '0.6', 'timestamp': '2018-12-
07 09:33:18.424166', 'previous_hash':
'1c77a54e39be8d3eb28e612a430ef8443f3a3d0b13586e861c469ef065269092', 'user_data':
'q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n', 'data_hash':
'0b79b8d727c8e1026ada60f3e416aad67cd50c6c03e47f1bd3267a52450be8d6', 'data_url':
'cb4d840bfa4511e8b2ee001a92daf3f8', 'proof': '000000d42631152973b23d5bc5194ca1', 'hash':
'e42e33120968ccd0495df08e77f7fe796a5b02c53e879a8568f9c029067390ce'}
2018-12-07 09:42:11.516170 -  - ABOPTING BLOCK:: {'index': 12, 'version': '0.6', 'timestamp': '2018-12-
07 09:33:19.875992', 'previous_hash':
'e42e33120968ccd0495df08e77f7fe796a5b02c53e879a8568f9c029067390ce', 'user_data':
'q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n', 'data_hash':
'3f65db0a5d62748f09623e0eac21e8db153b7c78b573617338d61c5f965974a8', 'data_url':
'cb4d840cfa4511e8b2ee001a92daf3f8', 'proof': '000000aa19680acc1f81cc080a2c431e', 'hash':
'4eb46cc9d93d9b21794ffe3271c6a0fbb7e32152ae215d50eb49ea9228e4ac79'}
2018-12-07 09:42:11.516933 -  - ABOPTING BLOCK:: {'index': 13, 'version': '0.6', 'timestamp': '2018-12-
07 09:33:31.792274', 'previous_hash':
'4eb46cc9d93d9b21794ffe3271c6a0fbb7e32152ae215d50eb49ea9228e4ac79', 'user_data':
'q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n', 'data_hash':
'37ae558c8b53b2206e87bc6e79a6fbe2aaceb24ed351e4afb4e6ca105ad5b683', 'data_url':
'cb4d840dfa4511e8b2ee001a92daf3f8', 'proof': '000000d8e59c1281ec46231ffd6d061e', 'hash':
'170b46584ab439790e91e1947c3005899b1d61a89168931b335e232a4ae7d0eb'}
```

```
2018-12-07 09:42:11.517604 -  - ABOPTING BLOCK:: {'index': 14, 'version': '0.6', 'timestamp': '2018-12-
07 09:34:11.428817', 'previous_hash':
'170b46584ab439790e91e1947c3005899b1d61a89168931b335e232a4ae7d0eb', 'user_data':
'q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n', 'data_hash':
'819e28111f67cf2a48b5f0907b0a862235aa81b057fe9698ba01942379116acb', 'data_url':
'cb4d840efa4511e8b2ee001a92daf3f8', 'proof': '000000d3acccddeeefde4fe0544c1750', 'hash':
'38add54ba651e6b3570baf249979a1dd58d73f2c6cd9bf5ea122d212000f1a2e'}
2018-12-07 09:42:11.518158 -  - ABOPTING BLOCK:: {'index': 15, 'version': '0.6', 'timestamp': '2018-12-
07 09:34:43.899177', 'previous_hash':
'38add54ba651e6b3570baf249979a1dd58d73f2c6cd9bf5ea122d212000f1a2e', 'user_data':
'q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n', 'data_hash':
'8591ec84faa0117327b0e7d39da85217e11910a10cc071333749fab372a644f8', 'data_url':
'cb4d840ffa4511e8b2ee001a92daf3f8', 'proof': '000000490e76915989ae367a6027bac1', 'hash':
'c68c450348ed6b5458cec92beb0354c0a255019d3b9482e2c73c21282da1424a'}
Hello World
CMD> put testfile.jpg
2018-12-07 09:43:49.998817 -  - CLIENT: # of chunks:5
2018-12-07 09:43:49.999750 -  - CLIENT: CHUNK: a7d0e002fa4711e8b2ee001a92daf3f8
2018-12-07 09:43:50.091104 -  - CLIENT: Sent to minion: 192.168.0.11
2018-12-07 09:43:50.091668 -  - CLIENT: CHUNK: a7d0e003fa4711e8b2ee001a92daf3f8
2018-12-07 09:43:50.162545 -  - CLIENT: Sent to minion: 192.168.0.11
2018-12-07 09:43:50.163161 -  - CLIENT: CHUNK: a7d0e004fa4711e8b2ee001a92daf3f8
2018-12-07 09:43:51.198904 -  - CLIENT: Sent to minion: 192.168.0.10
2018-12-07 09:43:51.199393 -  - CLIENT: CHUNK: a7d0e005fa4711e8b2ee001a92daf3f8
2018-12-07 09:43:51.282199 -  - CLIENT: Sent to minion: 192.168.0.10
2018-12-07 09:43:51.282618 -  - CLIENT: CHUNK: a7d0e006fa4711e8b2ee001a92daf3f8
2018-12-07 09:43:51.374849 -  - CLIENT: Sent to minion: 192.168.0.10
2018-12-07 09:43:51.375414 -  - CLIENT: --->   UPLOADED FILE: testfile.jpg
CMD> get testfile.jpg
File Already Exists... Overwrite? y/n:y
2018-12-07 09:43:58.973171 -  - CLIENT: <---    DOWNLOADED FILE: testfile.jpg
CMD> sync
2018-12-07 09:44:00.639254 -  - Synchronizing Blockchain...
CMD> delete testfile.jpg
2018-12-07 09:44:17.184763 -  - CLIENT: NETWORK FILE DELETED
CMD> get testfile.jpg
2018-12-07 09:44:34.146605 -  - CLIENT: NETWORK FILE NOT FOUND
CMD> put earth.jpg
2018-12-07 09:44:40.269283 -  - earth.jpg FILE NOT FOUND
CMD> put earth.JPEG
2018-12-07 09:44:48.462090 -  - CLIENT: # of chunks:5
2018-12-07 09:44:48.462530 -  - CLIENT: CHUNK: caa9b5f4fa4711e8b2ee001a92daf3f8
2018-12-07 09:44:48.594055 -  - CLIENT: Sent to minion: 192.168.0.10
2018-12-07 09:44:48.594511 -  - CLIENT: CHUNK: caa9b5f5fa4711e8b2ee001a92daf3f8
2018-12-07 09:44:48.653781 -  - CLIENT: Sent to minion: 192.168.0.11
2018-12-07 09:44:48.654293 -  - CLIENT: CHUNK: caa9b5f6fa4711e8b2ee001a92daf3f8
2018-12-07 09:44:48.877992 -  - CLIENT: Sent to minion: 192.168.0.10
2018-12-07 09:44:48.878467 -  - CLIENT: CHUNK: caa9b5f7fa4711e8b2ee001a92daf3f8
2018-12-07 09:44:49.065941 -  - CLIENT: Sent to minion: 192.168.0.10
2018-12-07 09:44:49.066553 -  - CLIENT: CHUNK: caa9b5f8fa4711e8b2ee001a92daf3f8
2018-12-07 09:44:49.160393 -  - CLIENT: Sent to minion: 192.168.0.11
2018-12-07 09:44:49.160900 -  - CLIENT: --->   UPLOADED FILE: earth.JPEG
CMD>
```

4.2.2. Master Node

```
rwgosse@Yamada-Star:~/Documents/prac$ python3 main.py
2018-12-07 09:28:25.410678 -  - Synchronizing Blockchain...
2018-12-07 09:28:25.548750 -  - NEW LONG CHAIN
2018-12-07 09:28:25.549200 -  - ABOPTING BLOCK:: {'index': 0, 'version': '0.6', 'timestamp': '2018-12-07
09:28:07.203393', 'previous_hash': '0', 'user_data': 'none', 'data_hash': 'none', 'data_url': '0', 'proof':
'00000048_GENESIS_BLOCKb16e9ac6cb', 'hash': 'dd8dab32200896b997500e0259eb572e4247bdde9e4c66c9a0f091450de439a0'}
Hello World
2018-12-07 09:28:26.207543 -  - MASTER: Port # 8100
2018-12-07 09:28:26.207666 -  - PROGRAM COMPLETE, SERVING UNTIL MANUAL ABORT...
2018-12-07 09:28:48.060870 -  - MASTER: incoming put request('192.168.0.13', 62622)
MASTER: 8e49ab5cfa4511e8b2ee001a92daf3f8 -> ['2', '1']
MASTER: 8e49ab5dfa4511e8b2ee001a92daf3f8 -> ['2', '1']
MASTER: 8e49ab5efa4511e8b2ee001a92daf3f8 -> ['1', '2']
MASTER: 8e49ab5ffa4511e8b2ee001a92daf3f8 -> ['2', '1']
MASTER: 8e49ab60fa4511e8b2ee001a92daf3f8 -> ['1', '2']
2018-12-07 09:30:30.431667 -  - MASTER: incoming put request('192.168.0.13', 62628)
MASTER: cb4d8406fa4511e8b2ee001a92daf3f8 -> ['1', '2']
MASTER: cb4d8407fa4511e8b2ee001a92daf3f8 -> ['1', '2']
MASTER: cb4d8408fa4511e8b2ee001a92daf3f8 -> ['1', '2']
MASTER: cb4d8409fa4511e8b2ee001a92daf3f8 -> ['1', '2']
MASTER: cb4d840afa4511e8b2ee001a92daf3f8 -> ['1', '2']
```

```
MASTER: cb4d840bfa4511e8b2ee001a92daf3f8 -> ['1', '2']
MASTER: cb4d840cfa4511e8b2ee001a92daf3f8 -> ['2', '1']
MASTER: cb4d840dfa4511e8b2ee001a92daf3f8 -> ['2', '1']
MASTER: cb4d840efa4511e8b2ee001a92daf3f8 -> ['1', '2']
MASTER: cb4d840ffa4511e8b2ee001a92daf3f8 -> ['2', '1']
2018-12-07 09:32:28.266361 -  - MASTER: incoming get request('192.168.0.13', 62644)
2018-12-07 09:32:41.438065 -  - MASTER: incoming delete request('192.168.0.13', 62650)
2018-12-07 09:32:41.438500 -  - MASTER: deleted file
2018-12-07 09:32:55.238110 -  - MASTER: incoming get request('192.168.0.13', 62651)
2018-12-07 09:32:55.238516 -  - MASTER: requested file not found
2018-12-07 09:33:42.388495 -  - MASTER: incoming get request('192.168.0.13', 62652)
2018-12-07 09:43:49.883909 -  - MASTER: incoming put request('192.168.0.13', 62676)
MASTER: a7d0e002fa4711e8b2ee001a92daf3f8 -> ['2', '1']
MASTER: a7d0e003fa4711e8b2ee001a92daf3f8 -> ['2', '1']
MASTER: a7d0e004fa4711e8b2ee001a92daf3f8 -> ['1', '2']
MASTER: a7d0e005fa4711e8b2ee001a92daf3f8 -> ['1', '2']
MASTER: a7d0e006fa4711e8b2ee001a92daf3f8 -> ['1', '2']
2018-12-07 09:43:56.254313 -  - MASTER: incoming get request('192.168.0.13', 62682)
2018-12-07 09:44:17.187717 -  - MASTER: incoming delete request('192.168.0.13', 62691)
2018-12-07 09:44:17.188154 -  - MASTER: deleted file
2018-12-07 09:44:33.147448 -  - MASTER: incoming get request('192.168.0.13', 62692)
2018-12-07 09:44:33.147887 -  - MASTER: requested file not found
2018-12-07 09:44:48.348659 -  - MASTER: incoming put request('192.168.0.13', 62694)
MASTER: caa9b5f4fa4711e8b2ee001a92daf3f8 -> ['1', '2']
MASTER: caa9b5f5fa4711e8b2ee001a92daf3f8 -> ['2', '1']
MASTER: caa9b5f6fa4711e8b2ee001a92daf3f8 -> ['1', '2']
MASTER: caa9b5f7fa4711e8b2ee001a92daf3f8 -> ['1', '2']
MASTER: caa9b5f8fa4711e8b2ee001a92daf3f8 -> ['2', '1']
```

4.2.3. Minion Node

```
C:\Users\user\Documents\GitHub\prac>py main.py
2018-12-07 09:28:03.970915 -  - Synchronizing Blockchain...
2018-12-07 09:28:07.203393 -  - ------NEW CHAIN-----
2018-12-07 09:28:07.219382 -  - NEW BLOCK:: {'index': 0, 'version': '0.6', 'timestamp': '2018-12-07 09:28:07.203393', 'previous_hash': '0', 'user_data': 'none', 'data_hash': 'none',
'data_url': '0', 'proof': '00000048_GENESIS_BLOCKb16e9ac6cb', 'hash': 'dd8dab32200896b997500e0259eb572e4247bdde9e4c66c9a0f091450de439a0'}
Hello World
2018-12-07 09:28:10.987323 -  - MINION: Port # 8200
2018-12-07 09:28:10.993302 -  - PROGRAM COMPLETE, SERVING UNTIL MANUAL ABORT...
2018-12-07 09:28:50.691416 -  - MINION: Received Chunk from:('192.168.0.11', 33816)
2018-12-07 09:28:50.748594 -  - MINION: create transaction...
2018-12-07 09:28:50.853311 -  - New Transaction: user data: q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n data hash:
544be94aaa9f03565b5712b0c9db4d6ef1dd5ba784f5e4dcf0af54e3d5c11950 url: 8e49ab5cfa4511e8b2ee001a92daf3f8
2018-12-07 09:28:50.925115 -  - Synchronizing Blockchain...
2018-12-07 09:28:50.932097 -  - MINION: Received Chunk from:('192.168.0.11', 33818)
2018-12-07 09:28:50.932097 -  - MINION: Received Chunk from:('192.168.0.13', 62625)
2018-12-07 09:28:50.936086 -  - MINION: create transaction...
2018-12-07 09:28:50.940049 -  - New Transaction: user data: q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n data hash:
5eb1bd21671d55f21be54ff010dbeb82352bb5e8f61edeaabd053e218d0e6696 url: 8e49ab5dfa4511e8b2ee001a92daf3f8
2018-12-07 09:28:50.961135 -  - MINION: Forwarded to: 192.168.0.11
2018-12-07 09:28:50.964129 -  - MINION: create transaction...
2018-12-07 09:28:50.968166 -  - New Transaction: user data: q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n data hash:
2439e76509728dfd6f2c99d626cc704d3c284425665b2b385f4f237acdd7157e url: 8e49ab5efa4511e8b2ee001a92daf3f8
last block index:0
MINING... 8e49ab5cfa4511e8b2ee001a92daf3f8
2018-12-07 09:28:51.161610 -  - MINION: Received Chunk from:('192.168.0.11', 33826)
2018-12-07 09:28:51.208452 -  - MINION: create transaction...
2018-12-07 09:28:51.242390 -  - New Transaction: user data: q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n data hash:
09469f79daaa67fc6cf55b1f1d468da9cb903c76d84a0e92027a65182ffe8cf6 url: 8e49ab5ffa4511e8b2ee001a92daf3f8
2018-12-07 09:28:51.440917 -  - MINION: Received Chunk from:('192.168.0.13', 62627)
2018-12-07 09:28:51.610467 -  - MINION: Forwarded to: 192.168.0.11
2018-12-07 09:28:51.684266 -  - MINION: create transaction...
2018-12-07 09:28:51.741115 -  - New Transaction: user data: q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n data hash:
242ea344a5d559e6b70b739c911a71faa535c68a1f4f667f7b1ab9e5cc1c2a3a url: 8e49ab60fa4511e8b2ee001a92daf3f8
2018-12-07 09:29:01.539907 -  - NEW BLOCK:: {'index': 1, 'version': '0.6', 'timestamp': '2018-12-07 09:29:01.538909', 'previous_hash':
'dd8dab32200896b997500e0259eb572e4247bdde9e4c66c9a0f091450de439a0', 'user_data': 'q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n', 'data_hash':
'544be94aaa9f03565b5712b0c9db4d6ef1dd5ba784f5e4dcf0af54e3d5c11950', 'data_url': '8e49ab5cfa4511e8b2ee001a92daf3f8', 'proof':
'0000008fd5b903696597f2a71d9b459e', 'hash': '735e9fb9b7b55c1fe26bd37b89a9654a8e0d91475f47bc79172298486c328f5c'}
MINING DONE... 8e49ab5cfa4511e8b2ee001a92daf3f8
2018-12-07 09:29:01.547866 -  - Synchronizing Blockchain...
last block index:1
MINING... 8e49ab5dfa4511e8b2ee001a92daf3f8
2018-12-07 09:30:31.260313 -  - NEW BLOCK:: {'index': 2, 'version': '0.6', 'timestamp': '2018-12-07 09:30:31.259315', 'previous_hash':
'735e9fb9b7b55c1fe26bd37b89a9654a8e0d91475f47bc79172298486c328f5c', 'user_data': 'q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n', 'data_hash':
'5eb1bd21671d55f21be54ff010dbeb82352bb5e8f61edeaabd053e218d0e6696', 'data_url': '8e49ab5dfa4511e8b2ee001a92daf3f8', 'proof':
'0000004026e012e7d1cbf0cac0ad0776', 'hash': '416ed614eb73400fc891be959892333cd18d6f8f288d87dd047131b259f85bc8'}
MINING DONE... 8e49ab5dfa4511e8b2ee001a92daf3f8
2018-12-07 09:30:31.267762 -  - Synchronizing Blockchain...
last block index:2
MINING... 8e49ab5efa4511e8b2ee001a92daf3f8
2018-12-07 09:30:36.373812 -  - MINION: Received Chunk from:('192.168.0.13', 62629)
```

2018-12-07 09:30:37.478997 - - MINION: Received Chunk from:('192.168.0.13', 62630)
2018-12-07 09:30:38.616052 - - MINION: Received Chunk from:('192.168.0.13', 62631)
2018-12-07 09:30:38.628018 - - MINION: Forwarded to: 192.168.0.11
2018-12-07 09:30:38.709815 - - MINION: create transaction...
2018-12-07 09:30:38.778731 - - New Transaction: user data: q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n data hash:
8158b46a89cee8b92e7a030a489408761fe1e8b3f9de73d9d1c705461d810277 url: cb4d8406fa4511e8b2ee001a92daf3f8
2018-12-07 09:30:39.482983 - - MINION: Forwarded to: 192.168.0.11
2018-12-07 09:30:39.575778 - - MINION: create transaction...
2018-12-07 09:30:39.645590 - - New Transaction: user data: q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n data hash:
210ac90d86661a3cf1e927f117c676c7a250982aa4ded654063ae8747da8e4c7 url: cb4d8407fa4511e8b2ee001a92daf3f8
2018-12-07 09:30:40.649199 - - MINION: Forwarded to: 192.168.0.11
2018-12-07 09:30:40.725056 - - MINION: create transaction...
2018-12-07 09:30:40.791881 - - New Transaction: user data: q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n data hash:
9f3e7d3d0158cc95d08c47a49cbae16e4c311c82247d6ac3b68a695cf628cadc url: cb4d8408fa4511e8b2ee001a92daf3f8
2018-12-07 09:30:41.081077 - - NEW BLOCK:: {'index': 3, 'version': '0.6', 'timestamp': '2018-12-07 09:30:41.079113', 'previous_hash':
'416ed614eb73400fc891be959892333cd18d6f8f288d87dd047131b259f85bc8', 'user_data': 'q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n', 'data_hash':
'2439e76509728dfd6f2c99d626cc704d3c284425665b2b385f4f237acdd7157e', 'data_url': '8e49ab5efa4511e8b2ee001a92daf3f8', 'proof':
'0000006f0d6d1c537fd252fe6da20dae', 'hash': 'a3c33e7989293f11d1cecf427876063f560ab5838172dae9eec962b0d627ac3a'}
MINING DONE... 8e49ab5efa4511e8b2ee001a92daf3f8
2018-12-07 09:30:41.096037 - - Synchronizing Blockchain...
2018-12-07 09:30:41.109030 - - MINION: Received Chunk from:('192.168.0.13', 62632)
last block index:3
MINING... 8e49ab5ffa4511e8b2ee001a92daf3f8
2018-12-07 09:30:41.458852 - - MINION: Received Chunk from:('192.168.0.13', 62633)
2018-12-07 09:30:43.068835 - - MINION: Forwarded to: 192.168.0.11
2018-12-07 09:30:43.128667 - - MINION: create transaction...
2018-12-07 09:30:43.200450 - - New Transaction: user data: q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n data hash:
2d2dd09223f986a1bdf433de91f2e8dc740dce6f3397b33e4d9477f1e65fdc2f url: cb4d8409fa4511e8b2ee001a92daf3f8
2018-12-07 09:30:43.585450 - - MINION: Forwarded to: 192.168.0.11
2018-12-07 09:30:43.693165 - - MINION: create transaction...
2018-12-07 09:30:43.799937 - - New Transaction: user data: q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n data hash:
1b4b993cb1ae756d9f657d9a9f3d6334e542bbd091e3237287cae8feeac34e39 url: cb4d840afa4511e8b2ee001a92daf3f8
2018-12-07 09:30:43.934605 - - MINION: Received Chunk from:('192.168.0.13', 62634)
2018-12-07 09:30:45.905337 - - MINION: Forwarded to: 192.168.0.11
2018-12-07 09:30:45.978147 - - MINION: create transaction...
2018-12-07 09:30:46.025016 - - New Transaction: user data: q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n data hash:
0b79b8d727c8e1026ada60f3e416aad67cd50c6c03e47f1bd3267a52450be8d6 url: cb4d840bfa4511e8b2ee001a92daf3f8
2018-12-07 09:30:47.890388 - - MINION: Received Chunk from:('192.168.0.11', 33828)
2018-12-07 09:30:47.921333 - - MINION: create transaction...
2018-12-07 09:30:47.968336 - - New Transaction: user data: q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n data hash:
3f65db0a5d62748f09623e0eac21e8db153b7c78b573617338d61c5f965974a8 url: cb4d840cfa4511e8b2ee001a92daf3f8
2018-12-07 09:30:50.965932 - - MINION: Received Chunk from:('192.168.0.11', 33830)
2018-12-07 09:30:51.042733 - - MINION: create transaction...
2018-12-07 09:30:51.156429 - - New Transaction: user data: q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n data hash:
37ae558c8b53b2206e87bc6e79a6fbe2aaceb24ed351e4afb4e6ca105ad5b683 url: cb4d840dfa4511e8b2ee001a92daf3f8
2018-12-07 09:30:52.202628 - - MINION: Received Chunk from:('192.168.0.13', 62637)
2018-12-07 09:30:54.317973 - - MINION: Forwarded to: 192.168.0.11
2018-12-07 09:30:54.368837 - - MINION: create transaction...
2018-12-07 09:30:54.409728 - - New Transaction: user data: q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n data hash:
819e28111f67cf2a48b5f0907b0a862235aa81b057fe9698ba01942379116acb url: cb4d840efa4511e8b2ee001a92daf3f8
2018-12-07 09:30:57.639513 - - MINION: Received Chunk from:('192.168.0.11', 33832)
2018-12-07 09:30:57.683392 - - MINION: create transaction...
2018-12-07 09:30:57.735254 - - New Transaction: user data: q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n data hash:
8591ec84faa0117327b0e7d39da85217e11910a10cc071333749fab372a644f8 url: cb4d840ffa4511e8b2ee001a92daf3f8
2018-12-07 09:31:00.384272 - - NEW BLOCK:: {'index': 4, 'version': '0.6', 'timestamp': '2018-12-07 09:31:00.383306', 'previous_hash':
'a3c33e7989293f11d1cecf427876063f560ab5838172dae9eec962b0d627ac3a', 'user_data': 'q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n', 'data_hash':
'09469f79daaa67fc6cf55b1f1d468da9cb903c76d84a0e92027a65182ffe8cf6', 'data_url': '8e49ab5ffa4511e8b2ee001a92daf3f8', 'proof': '000000fc34c66bea8df8c9c18c70564e',
'hash': '10d7785336043dcb23fde757f748d3e16782d413a07df3b29db257381f25f494'}
MINING DONE... 8e49ab5ffa4511e8b2ee001a92daf3f8
2018-12-07 09:31:00.391253 - - Synchronizing Blockchain...
last block index:4
MINING... 8e49ab60fa4511e8b2ee001a92daf3f8
2018-12-07 09:31:20.903075 - - NEW BLOCK:: {'index': 5, 'version': '0.6', 'timestamp': '2018-12-07 09:31:20.903075', 'previous_hash':
'10d7785336043dcb23fde757f748d3e16782d413a07df3b29db257381f25f494', 'user_data': 'q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n', 'data_hash':
'242ea344a5d559e6b70b739c911a71faa535c68a1f4f667f7b1ab9e5cc1c2a3a', 'data_url': '8e49ab60fa4511e8b2ee001a92daf3f8', 'proof':
'000000adb9d3903a059fff3888c21306', 'hash': '6d5d6a1aa05a905510f6ca175f88f4fe2bef3cc198bb68f1e77a512766a5d625'}
MINING DONE... 8e49ab60fa4511e8b2ee001a92daf3f8
2018-12-07 09:31:20.910878 - - Synchronizing Blockchain...
last block index:5
MINING... cb4d8406fa4511e8b2ee001a92daf3f8
2018-12-07 09:32:20.573838 - - NEW BLOCK:: {'index': 6, 'version': '0.6', 'timestamp': '2018-12-07 09:32:20.573838', 'previous_hash':
'6d5d6a1aa05a905510f6ca175f88f4fe2bef3cc198bb68f1e77a512766a5d625', 'user_data': 'q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n', 'data_hash':
'8158b46a89cee8b92e7a030a489408761fe1e8b3f9de73d9d1c705461d810277', 'data_url': 'cb4d8406fa4511e8b2ee001a92daf3f8', 'proof':
'000000980208c99eb510796eccb213f3', 'hash': '76ed7cd802a12ab1fc5c6593e93d471cd04a292d17a6823021f07c160794be27'}
MINING DONE... cb4d8406fa4511e8b2ee001a92daf3f8
2018-12-07 09:32:20.581578 - - Synchronizing Blockchain...
last block index:6
MINING... cb4d8407fa4511e8b2ee001a92daf3f8
2018-12-07 09:32:22.208233 - - NEW BLOCK:: {'index': 7, 'version': '0.6', 'timestamp': '2018-12-07 09:32:22.208233', 'previous_hash':
'76ed7cd802a12ab1fc5c6593e93d471cd04a292d17a6823021f07c160794be27', 'user_data': 'q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n', 'data_hash':

'210ac90d86661a3cf1e927f117c676c7a250982aa4ded654063ae8747da8e4c7', 'data_url': 'cb4d8407fa4511e8b2ee001a92daf3f8', 'proof':
'000000425e20a96bb982ac6a3518d21c', 'hash': 'c64f1a5248be28a96197848e6c961cb2a4b08d37ef6f40110c948f8131bd92f0'}
MINING DONE... cb4d8407fa4511e8b2ee001a92daf3f8
2018-12-07 09:32:22.216193 - - Synchronizing Blockchain...
last block index:7
MINING... cb4d8408fa4511e8b2ee001a92daf3f8
2018-12-07 09:32:24.368465 - - NEW BLOCK:: {'index': 8, 'version': '0.6', 'timestamp': '2018-12-07 09:32:24.368465', 'previous_hash':
'c64f1a5248be28a96197848e6c961cb2a4b08d37ef6f40110c948f8131bd92f0', 'user_data': 'q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n', 'data_hash':
'9f3e7d3d0158cc95d08c47a49cbae16e4c311c82247d6ac3b68a695cf628cadc', 'data_url': 'cb4d8408fa4511e8b2ee001a92daf3f8', 'proof':
'0000004c8ab9e0a067f302b7748ec375', 'hash': '070d253c01f69c723a2dee0e14fead65dc6bc0353972b7cc00718737ceebd743'}
MINING DONE... cb4d8408fa4511e8b2ee001a92daf3f8
2018-12-07 09:32:24.375446 - - Synchronizing Blockchain...
last block index:8
MINING... cb4d8409fa4511e8b2ee001a92daf3f8
2018-12-07 09:32:35.150367 - - MINION: CHUNK SENT TO CLIENT ('192.168.0.13', 62647)
2018-12-07 09:32:35.495447 - - MINION: CHUNK SENT TO CLIENT ('192.168.0.13', 62649)
2018-12-07 09:32:52.405221 - - NEW BLOCK:: {'index': 9, 'version': '0.6', 'timestamp': '2018-12-07 09:32:52.404223', 'previous_hash':
'070d253c01f69c723a2dee0e14fead65dc6bc0353972b7cc00718737ceebd743', 'user_data': 'q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n', 'data_hash':
'2d2dd09223f986a1bdf433de91f2e8dc740dce6f3397b33e4d9477f1e65fdc2f', 'data_url': 'cb4d8409fa4511e8b2ee001a92daf3f8', 'proof':
'00000059a9e5fe8ae28573f9e1ba651d', 'hash': '576ff88e8253eca73e4db20b930183ce15b5aff835980da421783800e292d0ac'}
MINING DONE... cb4d8409fa4511e8b2ee001a92daf3f8
2018-12-07 09:32:52.412182 - - Synchronizing Blockchain...
last block index:9
MINING... cb4d840afa4511e8b2ee001a92daf3f8
2018-12-07 09:33:11.835289 - - NEW BLOCK:: {'index': 10, 'version': '0.6', 'timestamp': '2018-12-07 09:33:11.835289', 'previous_hash':
'576ff88e8253eca73e4db20b930183ce15b5aff835980da421783800e292d0ac', 'user_data': 'q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n', 'data_hash':
'1b4b993cb1ae756d9f657d9a9f3d6334e542bbd091e3237287cae8feeac34e39', 'data_url': 'cb4d840afa4511e8b2ee001a92daf3f8', 'proof':
'0000008c16aaebf2d56ffe227dc05747', 'hash': '1c77a54e39be8d3eb28e612a430ef8443f3a3d0b13586e861c469ef065269092'}
MINING DONE... cb4d840afa4511e8b2ee001a92daf3f8
2018-12-07 09:33:11.842270 - - Synchronizing Blockchain...
last block index:10
MINING... cb4d840bfa4511e8b2ee001a92daf3f8
2018-12-07 09:33:18.425164 - - NEW BLOCK:: {'index': 11, 'version': '0.6', 'timestamp': '2018-12-07 09:33:18.424166', 'previous_hash':
'1c77a54e39be8d3eb28e612a430ef8443f3a3d0b13586e861c469ef065269092', 'user_data': 'q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n', 'data_hash':
'0b79b8d727c8e1026ada60f3e416aad67cd50c6c03e47f1bd3267a52450be8d6', 'data_url': 'cb4d840bfa4511e8b2ee001a92daf3f8', 'proof':
'000000d42631152973b23d5bc5194ca1', 'hash': 'e42e33120968ccd0495df08e77f7fe796a5b02c53e879a8568f9c029067390ce'}
MINING DONE... cb4d840bfa4511e8b2ee001a92daf3f8
2018-12-07 09:33:18.432144 - - Synchronizing Blockchain...
last block index:11
MINING... cb4d840cfa4511e8b2ee001a92daf3f8
2018-12-07 09:33:19.875992 - - NEW BLOCK:: {'index': 12, 'version': '0.6', 'timestamp': '2018-12-07 09:33:19.875992', 'previous_hash':
'e42e33120968ccd0495df08e77f7fe796a5b02c53e879a8568f9c029067390ce', 'user_data': 'q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n', 'data_hash':
'3f65db0a5d62748f09623e0eac21e8db153b7c78b573617338d61c5f965974a8', 'data_url': 'cb4d840cfa4511e8b2ee001a92daf3f8', 'proof':
'000000aa19680acc1f81cc080a2c431e', 'hash': '4eb46cc9d93d9b21794ffe3271c6a0fbb7e32152ae215d50eb49ea9228e4ac79'}
MINING DONE... cb4d840cfa4511e8b2ee001a92daf3f8
2018-12-07 09:33:19.883978 - - Synchronizing Blockchain...
last block index:12
MINING... cb4d840dfa4511e8b2ee001a92daf3f8
2018-12-07 09:33:31.793271 - - NEW BLOCK:: {'index': 13, 'version': '0.6', 'timestamp': '2018-12-07 09:33:31.792274', 'previous_hash':
'4eb46cc9d93d9b21794ffe3271c6a0fbb7e32152ae215d50eb49ea9228e4ac79', 'user_data': 'q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n', 'data_hash':
'37ae558c8b53b2206e87bc6e79a6fbe2aaceb24ed351e4afb4e6ca105ad5b683', 'data_url': 'cb4d840dfa4511e8b2ee001a92daf3f8', 'proof':
'000000d8e59c1281ec46231ffd6d061e', 'hash': '170b46584ab439790e91e1947c3005899b1d61a89168931b335e232a4ae7d0eb'}
MINING DONE... cb4d840dfa4511e8b2ee001a92daf3f8
2018-12-07 09:33:31.802179 - - Synchronizing Blockchain...
last block index:13
MINING... cb4d840efa4511e8b2ee001a92daf3f8
2018-12-07 09:33:57.064952 - - MINION: CHUNK SENT TO CLIENT ('192.168.0.13', 62653)
2018-12-07 09:33:57.314285 - - MINION: CHUNK SENT TO CLIENT ('192.168.0.13', 62654)
2018-12-07 09:33:57.610653 - - MINION: CHUNK SENT TO CLIENT ('192.168.0.13', 62655)
2018-12-07 09:33:57.924862 - - MINION: CHUNK SENT TO CLIENT ('192.168.0.13', 62656)
2018-12-07 09:33:58.193233 - - MINION: CHUNK SENT TO CLIENT ('192.168.0.13', 62657)
2018-12-07 09:33:58.486457 - - MINION: CHUNK SENT TO CLIENT ('192.168.0.13', 62658)
2018-12-07 09:33:59.278324 - - MINION: CHUNK SENT TO CLIENT ('192.168.0.13', 62661)
2018-12-07 09:34:11.428817 - - NEW BLOCK:: {'index': 14, 'version': '0.6', 'timestamp': '2018-12-07 09:34:11.428817', 'previous_hash':
'170b46584ab439790e91e1947c3005899b1d61a89168931b335e232a4ae7d0eb', 'user_data': 'q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n', 'data_hash':
'819e28111f67cf2a48b5f0907b0a862235aa81b057fe9698ba01942379116acb', 'data_url': 'cb4d840efa4511e8b2ee001a92daf3f8', 'proof':
'000000d3acccddeeefde4fe0544c1750', 'hash': '38add54ba651e6b3570baf249979a1dd58d73f2c6cd9bf5ea122d212000f1a2e'}
MINING DONE... cb4d840efa4511e8b2ee001a92daf3f8
2018-12-07 09:34:11.436494 - - Synchronizing Blockchain...
last block index:14
MINING... cb4d840ffa4511e8b2ee001a92daf3f8
2018-12-07 09:34:43.899177 - - NEW BLOCK:: {'index': 15, 'version': '0.6', 'timestamp': '2018-12-07 09:34:43.899177', 'previous_hash':
'38add54ba651e6b3570baf249979a1dd58d73f2c6cd9bf5ea122d212000f1a2e', 'user_data': 'q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n', 'data_hash':
'8591ec84faa0117327b0e7d39da85217e11910a10cc071333749fab372a644f8', 'data_url': 'cb4d840ffa4511e8b2ee001a92daf3f8', 'proof':
'000000490e76915989ae367a6027bac1', 'hash': 'c68c450348ed6b5458cec92beb0354c0a255019d3b9482e2c73c21282da1424a'}
MINING DONE... cb4d840ffa4511e8b2ee001a92daf3f8
2018-12-07 09:43:52.913090 - - MINION: Received Chunk from:('192.168.0.11', 33858)
2018-12-07 09:43:52.965952 - - MINION: Received Chunk from:('192.168.0.11', 33860)
2018-12-07 09:43:53.031770 - - MINION: create transaction...
2018-12-07 09:43:53.055703 - - MINION: create transaction...

2018-12-07 09:43:53.073654 - - New Transaction: user data: q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n data hash:
907b08792d589686803f4320a449bac4267d015bb642a59e73fb9ba3d3eb2542 url: a7d0e003fa4711e8b2ee001a92daf3f8
2018-12-07 09:43:53.114546 - - New Transaction: user data: q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n data hash:
a5dda7d20b0d6462ce46449898e90245ccd8a468974f2d75290ba3f9148c5951 url: a7d0e002fa4711e8b2ee001a92daf3f8
2018-12-07 09:43:53.127512 - - Synchronizing Blockchain...
2018-12-07 09:43:53.143441 - - MINION: Received Chunk from:('192.168.0.13', 62679)
2018-12-07 09:43:53.195596 - - MINION: Forwarded to: 192.168.0.11
2018-12-07 09:43:53.211564 - - MINION: create transaction...
2018-12-07 09:43:53.231508 - - New Transaction: user data: q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n data hash:
98c9a0d9979062f66a311bc990adcdc0d04fb1b7bb51362aee6f31069b2e0412 url: a7d0e004fa4711e8b2ee001a92daf3f8
2018-12-07 09:43:53.241482 - - MINION: Received Chunk from:('192.168.0.13', 62680)
2018-12-07 09:43:53.297352 - - MINION: Forwarded to: 192.168.0.11
2018-12-07 09:43:53.318302 - - MINION: create transaction...
2018-12-07 09:43:53.323250 - - MINION: Received Chunk from:('192.168.0.13', 62681)
2018-12-07 09:43:53.335250 - - New Transaction: user data: q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n data hash:
9be75ff613a52a356fe9a860dd0e11512b36e947e4be61ff6f13c3244a7d4cbc url: a7d0e005fa4711e8b2ee001a92daf3f8
2018-12-07 09:43:53.391138 - - MINION: Forwarded to: 192.168.0.11
2018-12-07 09:43:53.399152 - - MINION: create transaction...
2018-12-07 09:43:53.408097 - - New Transaction: user data: q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n data hash:
5c188ebf34e5584193d31e390d5bbaa6be0fbc990b43ebad80b21f517ec369c9 url: a7d0e006fa4711e8b2ee001a92daf3f8
last block index:15
MINING... a7d0e003fa4711e8b2ee001a92daf3f8
2018-12-07 09:44:00.496061 - - MINION: CHUNK SENT TO CLIENT ('192.168.0.13', 62685)
2018-12-07 09:44:00.688547 - - MINION: CHUNK SENT TO CLIENT ('192.168.0.13', 62686)
2018-12-07 09:44:00.943898 - - MINION: CHUNK SENT TO CLIENT ('192.168.0.13', 62687)
2018-12-07 09:44:04.340868 - - NEW BLOCK:: {'index': 16, 'version': '0.6', 'timestamp': '2018-12-07 09:44:04.339896', 'previous_hash':
'c68c450348ed6b5458cec92beb0354c0a255019d3b9482e2c73c21282da1424a', 'user_data': 'q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n', 'data_hash':
'907b08792d589686803f4320a449bac4267d015bb642a59e73fb9ba3d3eb2542', 'data_url': 'a7d0e003fa4711e8b2ee001a92daf3f8', 'proof':
'00000027f863f28919077b2379e72064', 'hash': '2adb7884a9717f9c57a2f6a7fc774f13c88eb9d195e77e7c2d76ad3b2792604a'}
MINING DONE... a7d0e003fa4711e8b2ee001a92daf3f8
2018-12-07 09:44:04.347556 - - Synchronizing Blockchain...
last block index:16
MINING... a7d0e002fa4711e8b2ee001a92daf3f8
2018-12-07 09:44:28.336558 - - NEW BLOCK:: {'index': 17, 'version': '0.6', 'timestamp': '2018-12-07 09:44:28.336558', 'previous_hash':
'2adb7884a9717f9c57a2f6a7fc774f13c88eb9d195e77e7c2d76ad3b2792604a', 'user_data': 'q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n', 'data_hash':
'a5dda7d20b0d6462ce46449898e90245ccd8a468974f2d75290ba3f9148c5951', 'data_url': 'a7d0e002fa4711e8b2ee001a92daf3f8', 'proof':
'00000056d98af3403aa981a5a3f9c870', 'hash': 'ed98d609df6d1f4aa1dadec3bcdda733e9e4ddcadc9633668b979b7f455d7a43'}
MINING DONE... a7d0e002fa4711e8b2ee001a92daf3f8
2018-12-07 09:44:28.343508 - - Synchronizing Blockchain...
last block index:17
MINING... a7d0e004fa4711e8b2ee001a92daf3f8
2018-12-07 09:44:50.545871 - - MINION: Received Chunk from:('192.168.0.13', 62695)
2018-12-07 09:44:50.727381 - - MINION: Forwarded to: 192.168.0.11
2018-12-07 09:44:50.808167 - - MINION: create transaction...
2018-12-07 09:44:50.838086 - - MINION: Received Chunk from:('192.168.0.13', 62697)
2018-12-07 09:44:50.838086 - - MINION: Received Chunk from:('192.168.0.11', 33868)
2018-12-07 09:44:50.844070 - - New Transaction: user data: q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n data hash:
29f91ba3d6a7d184e2fefb75e6f89379db89a61078673278309fc5b6d2ada959 url: caa9b5f4fa4711e8b2ee001a92daf3f8
2018-12-07 09:44:50.877983 - - MINION: create transaction...
2018-12-07 09:44:50.924858 - - MINION: Forwarded to: 192.168.0.11
2018-12-07 09:44:50.953781 - - New Transaction: user data: q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n data hash:
4d1b4a257b87e0665d001d3fae449ae462a46e9924d72ca0d09d7a25a26fea77 url: caa9b5f5fa4711e8b2ee001a92daf3f8
2018-12-07 09:44:50.995636 - - MINION: create transaction...
2018-12-07 09:44:51.013621 - - MINION: Received Chunk from:('192.168.0.13', 62698)
2018-12-07 09:44:51.058498 - - New Transaction: user data: q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n data hash:
1e5d08c022cd1aa45149962f0eff30032fb43ef65d82489e54c40bffedd8773c url: caa9b5f6fa4711e8b2ee001a92daf3f8
2018-12-07 09:44:51.129307 - - MINION: Forwarded to: 192.168.0.11
2018-12-07 09:44:51.197134 - - MINION: create transaction...
2018-12-07 09:44:51.279976 - - New Transaction: user data: q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n data hash:
477ffe00249c9ed6bef6ad4c660d2fb51454128f5998a5ee1563205a698b465d url: caa9b5f7fa4711e8b2ee001a92daf3f8
2018-12-07 09:44:51.309893 - - MINION: Received Chunk from:('192.168.0.11', 33870)
2018-12-07 09:44:51.370729 - - MINION: create transaction...
2018-12-07 09:44:51.438552 - - New Transaction: user data: q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n data hash:
a9ca06f2f3e804af29d782b6e2fc2e90305a559c5063fe152b7602bee7d460c1 url: caa9b5f8fa4711e8b2ee001a92daf3f8
2018-12-07 09:45:10.558451 - - NEW BLOCK:: {'index': 18, 'version': '0.6', 'timestamp': '2018-12-07 09:45:10.558451', 'previous_hash':
'ed98d609df6d1f4aa1dadec3bcdda733e9e4ddcadc9633668b979b7f455d7a43', 'user_data': 'q83jv93yfnf02f8n_THIRD_MINER_nf939nf03n88fnf92n', 'data_hash':
'98c9a0d9979062f66a311bc990adcdc0d04fb1b7bb51362aee6f31069b2e0412', 'data_url': 'a7d0e004fa4711e8b2ee001a92daf3f8', 'proof':
'0000002a6454faae6e39dfcfa297886b', 'hash': '0ba90f9a9b6a2d95090cc839d7626114ac5b7e1639f75a4f7d54f9f83aa7813b'}
MINING DONE... a7d0e004fa4711e8b2ee001a92daf3f8
2018-12-07 09:45:10.565433 - - Synchronizing Blockchain...
last block index:18
MINING... a7d0e005fa4711e8b2ee001a92daf3f8

5. References

Ago, C. A. • 6 M. (2018, January 29). #archisteem 1: Blockchain Technology in City Architecture and Planning. Retrieved August 11, 2018, from https://steemit.com/archisteem/@cklai/archisteem-1-blockchain-technology-in-city-architecture-and-planning

Apache Hadoop. (n.d.). Retrieved December 7, 2018, from https://hadoop.apache.org/

Baron, J., & Schneider, R. (2010). Storage Options in the AWS Cloud: Use Cases, 12.

BlockApps. (2017, December 13). How Blockchain Will Disrupt Data Storage. Retrieved December 7, 2018, from https://blockapps.net/blockchain-disrupt-data-storage/

Blockchain technology for cloud storage: This looks like the future. (2018a, February 8). Retrieved December 7, 2018, from http://techgenix.com/blockchain-technology-for-cloud-storage/

Blockchain technology for cloud storage: This looks like the future. (2018b, February 8). Retrieved July 17, 2018, from http://techgenix.com/blockchain-technology-for-cloud-storage/

BlockCloud: Re-inventing Cloud with Blockchains. (n.d.). Retrieved July 17, 2018, from https://guardtime.com/blog/blockcloud-re-inventing-cloud-with-blockchains

Chin, A. (2017, October 29). A simple and secure Blockchain Database API written in Python. Retrieved August 11, 2018, from https://hackernoon.com/how-finance-data-can-be-secured-by-blockchain-technology-a-fast-and-simple-adoption-with-200-5e762299b67

Deploying Large File Transfer on an HTTP Content Distribution Network. (n.d.). Retrieved August 14, 2018, from https://www.usenix.org/legacy/publications/library/proceedings/worlds04/tech/full_papers/park/park_html/codeploy.html

Develop a mini fully-functional blockchain-based project from scratch in Python. (2018, April 3). Retrieved July 17, 2018, from http://www.ibm.com/developerworks/cloud/library/cl-develop-blockchain-app-in-python/index.html

Developer Glossary - Bitcoin. (n.d.). Retrieved December 7, 2018, from https://bitcoin.org/en/developer-glossary#numbers

Es-Samaali, H., Outchakoucht, A., & Leroy, J. P. (2017). A Blockchain-based Access Control for Big Data, 11.

EU GDPR Information Portal. (n.d.). Retrieved August 21, 2018, from http://eugdpr.org/eugdpr.org-1.html

HDFS Architecture Guide. (n.d.). Retrieved December 7, 2018, from https://hadoop.apache.org/docs/r1.2.1/hdfs_design.html

How to Build Your Own Blockchain Part 1 — Creating, Storing, Syncing, Displaying, Mining, and Proving Work. (2017, October 17). Retrieved December 7, 2018, from https://bigishdata.com/2017/10/17/write-your-own-blockchain-part-1-creating-storing-syncing-displaying-mining-and-proving-work/

Information | Free Full-Text | BBDS: Blockchain-Based Data Sharing for Electronic Medical Records in Cloud Environments | HTML. (n.d.). Retrieved August 13, 2018, from http://www.mdpi.com/2078-2489/8/2/44/htm

Jul'18 2017-04-11T00:36:04+00:00, J. G.-L. U. 11. (2017, April 11). Best DIY Cloud Storage Tools: Build Your Own Personal Cloud Server. Retrieved July 17, 2018, from https://www.cloudwards.net/diy-cloud-storage-tools/

Leading the Blockchain in Healthcare Code-A-Thon for ONC. (2017, March 23). Retrieved August 11, 2018, from https://nucleushealth.io/leading-blockchain-in-healthcare-code-a-thon-onc/

Li, J., Liu, Z., Chen, L., Chen, P., & Wu, J. (2017). Blockchain-Based Security Architecture for Distributed Cloud Storage. In *2017 IEEE International Symposium on Parallel and Distributed Processing with Applications and 2017 IEEE International Conference on Ubiquitous Computing and Communications (ISPA/IUCC)* (pp. 408–411). https://doi.org/10.1109/ISPA/IUCC.2017.00065

Nakamoto, S. (n.d.). Bitcoin: A Peer-to-Peer Electronic Cash System, 9.

REGULATION (EU) 2016/ 679 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL - of 27 Apri l 2016 - on the protection of natural persons with regard to the processing of personal dat a and on the free movement of such data, and repealing Directive 95/ 46/ EC (General Data Protection Regulation). (n.d.), 88.

Risk Management – 6 Steps to GDPR Implementation. (n.d.-b). Retrieved September 26, 2018, from http://www.rmmagazine.com/2018/04/02/6-steps-to-gdpr-implementation/

Security — pysheeet. (n.d.). Retrieved December 7, 2018, from https://www.pythonsheets.com/notes/python-security.html

Sia. (n.d.). Retrieved July 17, 2018, from https://sia.tech/

Xia, Q., Sifah, E., Smahi, A., Amofa, S., & Zhang, X. (2017). BBDS: Blockchain-Based Data Sharing for Electronic Medical Records in Cloud Environments. *Information*, *8*(2), 44. https://doi.org/10.3390/info8020044

Zaninotto, F. (n.d.). The Blockchain Explained to Web Developers, Part 1: The Theory. Retrieved August 1, 2018, from https://marmelab.com/blog/2016/04/28/blockchain-for-web-developers-the-theory.html

6. Change Log

None as yet.