# Image Integrity Analysis with BlockChain Technology

Bachelor Thesis

*Submitted By*

Anuj Kr. Pathak
&
Sayan Shankhari

*A thesis submitted to*

Indian Institute of Information Technology Kalyani

*for the partial fulfillment of the degree of*

**Bachelor of Engineering in Computer Science**
**in**
**Department of Computer Science and Information Technology**

May, 2019

*To my beloved parents and friends who have supported me and prayed for my success throughout my life.*

# Certificate

This is to certify that the thesis entitled **"Image Integrity Analysis with BlockChain Technology"** being submitted by undergraduate students **Anuj Kr. Pathak** (Reg. No.: 000000102) and **Sayan Shankhari** (Reg. No.: 00000121) in the Department of Computer Science and Information Technology, Indian Institute of Information Technology Kalyani, Nodia, 741235, India, for the award of **Bachelors of Technology** in **Computer Science & Engineering**, is an original research work carried by them under my supervision and guidance. The synopsis has fulfilled all the requirements as par the regulation of **IIIT Kalyani** and in my opinion, has reached the standards needed for submission. The works, techniques and the results presented have not been submitted to any other university or Institute for the award of any other degree or diploma.

——————————-

(**Dr. Imon Mukherjee**)

Assistant Professor

Department of Computer Science and Information Systems

Indian Institute of Information Technology Kalyani

IIIT Kalyani Campus, West Bengal 741235, India. May 2019

# Declaration

We hereby declare that the work being presented in this thesis entitled, **"Image Integrity Analysis with BlockChain Technology"**, submitted to Indian Institute of Information Technology Kalyani in partial fulfilment for the award of the degree of Bachelor of Technology in Computer Science and Engineering during the period from July, 2018 to May, 2019 under the supervision of Dr. Imon Mukherjee, Department of Computer Science and Engineering, Indian Institute of Information Technology Kalyani, West Bengal 741235, India, does not contain any classified information.

**Anuj Kumar Pathak**
39/CSE-15145 :: 00000102
Computer Science and Engineering,
Indian Institute of Information
Technology Kalyani,
WEBEL IT Park, West Bengal
741235 India

**Sayan Shankhari**
67/IT-15026 :: 00000121
Information Technology,
Indian Institute of Information
Technology Kalyani,
WEBEL IT Park, West Bengal
741235 India

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

(**Dr. Imon Mukherjee**)
Assistant Professor
Department of Computer Science and Information Systems
Indian Institute of Information Technology Kalyani
IIIT Kalyani Campus, West Bengal 741235, India
May 2019

# Acknowledgments

First of all, We would like to take this opportunity to thank my supervisor Dr. Imon Mukherjee without whose effort this thesis would not have been possible. We are so grateful to him for working tirelessly after us, clearing our doubts whenever and wherever possible. We are most grateful to Department of Computer Science and Information Technology, Indian Institute of Information Technology Kalyani, West Bengal, 741235, India, for providing us this wonderful opportunity to complete our bachelor thesis. We would like to thank our friends for providing us help as and when required. We would like to thank our team mates for being a great motivators and great friends.

And last but the biggest of all, We want to thank our parents, for always believing in us and letting us do what we wanted, but keeping a continuous check that we never wandered off the track from my goal.

**Anuj Kumar Pathak**
39/CSE-15145 :: 00000102
Computer Science and Engineering,
Indian Institute of Information
Technology Kalyani,
WEBEL IT Park, West Bengal
741235 India

**Sayan Shankhari**
67/IT-15026 :: 00000121
Information Technology,
Indian Institute of Information
Technology Kalyani,
Webel IT Park, West Bengal
741235 India

# Contents

# List of Figures

# Abstract

Images are electronic documents that plays an important role in the society to store and give some information about some event as a historical document. With the digitalization in developing country like India there comes image scams with the image editing tools. We can not use a normal database to maintain the context of the image because a server's security can be compromised by the hackers or the masters of the internet and electronic devices with the flaws of technology. An emerging technology *i.e.* BlockChain can be used to store the data (here image) in such a way that before storing it is verified and no one can change the data as it is a peer-to-peer decentralized network of a group of people where everybody is having a copy of public ledger containing information of a portion having the image in it. We created a system that serves users a social media platform to store images and verify an image by checking in our system. It is not easy to create a blockchain and it is about impossible to change the data of blockchain because it works collaboratively.

**Keywords:** Peer-to-Peer network, Proof-of-work, Proof-of-Stake, Distributed Ledger, Consensus Mechanism, Hash Function, Image Formatting, Image File formats, Digital Signature, Watermarking, WebServer.

# Chapter 1

# Introduction

This chapter presents the introduction of the thesis that includes the brief description of the purpose of the project, the problem being investigated, the background of the problem, previous works, our thesis and general approach and the criteria of success.

We are at a point of time where electronic devices like mobile phones and computers with internet connection become an important part of our daily life. The social media platforms are the main spreading source of the media files (image, audio or video). The amount of media files in the world is continously increasing. The social media platforms gets and shows these media contents but does not verify the originality of the content. This makes it very important to be able to verify online image content specially when and by whom it was captured, what was the geo location, is the image edited, if yes by whom, when and other information.

A news channel or a news feed in a social media platform, is a place where different informative media are shown to the world. People come to the social media platform to see what is other's status in the news feed.

In the context of unaccredited news, some videos have been manipulated or the image taken out of its context. This thesis investigates methods to verify image content based on a specific criteria: integrity (i.e., non-modification) of the recording. This would not only flag as unverified a lot of fake news images but might also be used to give increased credibility to images which can be verified.

This chapter gives a brief introduction to this thesis. Section 1.1 gives a general background to the reader. Section 1.2 further discusses the problems that this thesis will investigate. Section 1.3 gives a brief overview of our ideas regarding how the problem will be solved and why. Section 1.4 succinctly breaks down and quantizes the project into measurable goals. Section 1.5 presents a general framework and approach to the complete project. Section 1.6 explains the initial idea regarding what the final proof of concept prototype will look like. In Section 1.7 the delimitations are discussed and why certain areas have been excluded. Finally, Section 1.8 outlines the entirety of this thesis.

## 1.1 Background

Image integrity is the task of checking if the image file's bits are changed or not at any point. A digital file might change by platforms. This would be fully online system. Nowadays with increasing people in digital world the editing softwares are getting smart. Most of the files that might be non-editable in some operating systems or file systems, but the so called hackers or masters of electronic devices have file systems that can easily access and change any file data. So in that way the media files can not be checked for integrity. And also while the images are shared in different platforms, the files' data might be changed a little bit according to their protocols for data compression or security. Online platforms or social media just use public data, but they neither care about the originality of the data, nor there exists a fixed verification process for it.

The topic of this thesis is, as previously mentioned, highly relevant and widely discussed but usually not in the context of blockchains. Blockchain technology has the potential to provide data security and enable validation of the integrity of the data

The proposed system in this thesis will help us to detect those shared files and compare them not bit by bit, but context wise and signature wise. That means if one shares an image both our system as well as one of those online platforms and download them as separate file and check in our system, it should return truth values if data portion are same or atleast 95% same. The system we are introducing the social media like platform that have the capability to store images and show them in news feed. There is a option for every photo to download in user's computer or mobile devices and after sharing and getting back the person can check if the file is intact or not in our proposed system by the digital signature.

## 1.2 Problem

More and more electronic devices like mobile phones and computers are equipped with cameras, internet connections, and verious sensors. The increasing capabilities of current devices enable the end user to produce and distribute media contents more easily than before. One issue that has arisen from this is the inability to be able to distinguish between a modified image and the original image. Additionally, images are taken out of context or claimed to be from a place and/or time different from what might actually be the truth. It is getting increasingly difficult for the viewer to know what to believe, who to believe, and tools to help counter these issues are rare and limited.

A service as proposed by this thesis will need to be trusted by its users. It is therefore of utmost importance to be able to ensure the integrity of the service and its software. The user needs to be able to trust the files they are downloading, and using on their devices. No additional software is required for the user to download except a reliable web-browser that very often comes with smartphone or computer.

## 1.3 Purpose

The purpose of this thesis is to set up and evaluate the usage of a blockchain in order to verify and store signature hashes from live-captured image content as well as uploaded images from any device. The creation of the signature hashes should be done as soon as the upload request comes, leaving as little time as possible for data manipulation by anyone and enabling the establishment of the precedence of this signed hash (*i.e.*, it can be shown that it was signed at a particular period in time (based upon the entries before and after it in the blockchain), therefore the hash had to be calculated before this, and hence the media over which the hash is computed has to be even earlier than this). In this thesis project, the signature hash is stored in a blockchain. Verification methods for these hashes are implemented and accessible to users via a web-based interface. This proof of concept prototype showcases the potential of combining blockchains with hashes over chunks of media content and meets the requirements of the evolving world of streaming media content.

About all compression schemes says all the information of image is not useful we have to identify the useful regions of image and then try to integrate these regions of images. The objective why we write this thesis is If an image in system then it is original but if it not in system System cannot confirm about it.

## 1.4 Goals of the Thesis

The measurable goals for this Thesis can be enumerate as three goals through which we aim to develop and evaluate a proof of concept prototype. The prototype when complete should be able to:

1. Determine whether an image has been taken at a specified time.
2. Determine which parts of the image whose integrity can/cannot be verified
3. The process of verifying the image through the entire blockchain should require less than 3 seconds.

## 1.5 Research Methodology

The qualitative hypothetico-deductive method is used for this thesis. We have selected a deductive method since it uses previous theory and the main goal of the thesis is to prove a theory;

specifically, to prove that blockchain technology is a suitable tool to ensure image integrity for any image. This is the hypothesis we wish to either confirm or dismiss.

We have chosen a qualitative method as this project is primary exploratory research within the combined areas of image integrity and blockchains. We seek to provide insights into the problem being investigated. The data collection method in this thesis will be unstructured and dependent upon the actual progress toward the project's goals.

The system that we are trying to build is capable of processing, storing and showing image files. Not only that, it also allows viewers to veryfy his own copy of the image. It is not easy to store and veryfy in a moment and it is about impossible to change the whole blockchain, because the miners have their own copy of the public ledger and their processed computation powers in their computers. The automated server should back up its data from both the Virtual miners as well as remote miners' computers. The studied and inspired technologies are discussed in the thesis.

The images collected during the investigation will be highly reliable since it will be collected by software without any human interaction. The image data is replicable as long as the same hardware and software are used. This is due to the fact that different hardware has different characteristics (such as processing power and read/write memory speed).

## 1.6 Setup

The system service (LAMPP web server) runs on a Dell Inspiron with Ubuntu Bionic Beaver (18.04.15). The client system is also in the same computer but the system we are building for a website that codes will be uploaded on a rermote dedicated linux web server.

The system can be seen in next figure. The decentralized web-application is built using PHP-7 (Personal Home Page or Hypertext PreProcessesor), CSS-3 (Cascading Style Sheets) and JS-5 (JavaScipt) language. PHP runs in server computer and JS runs in clients' browsers.

There are two phases of processing user image, i.e. whether the user wants to store image or verify image. So there are two separate page for these two. Moreover there are two different ways an user can request to store image, either by image uploadation or by web-camera. So we have separate page for that too.

## 1.7 Delimitations

This thesis will investigate and develop a proof of concept prototype showcasing the merger of an app running on an Android device with a backend security system using blockchain verification through smart contracts. However, the thesis will also examine what type of manipulation or tampering has been done to the recorded data, and it will simply verify if the hash of the image content is the same as that of the signed hash as recorded in the blockchain. There already

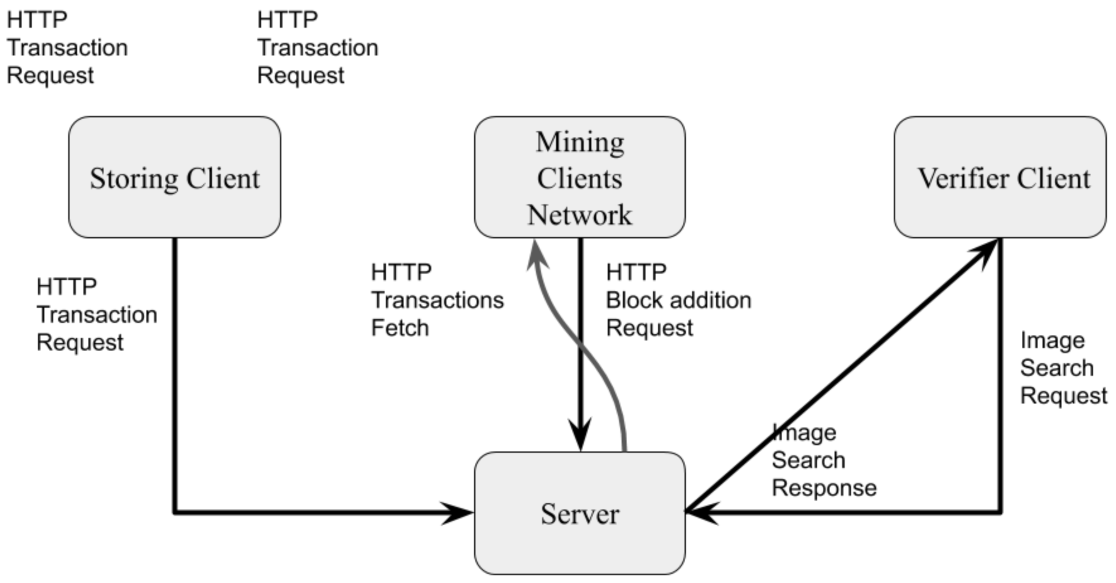


Figure 1.1: Overview of Proposed System Model

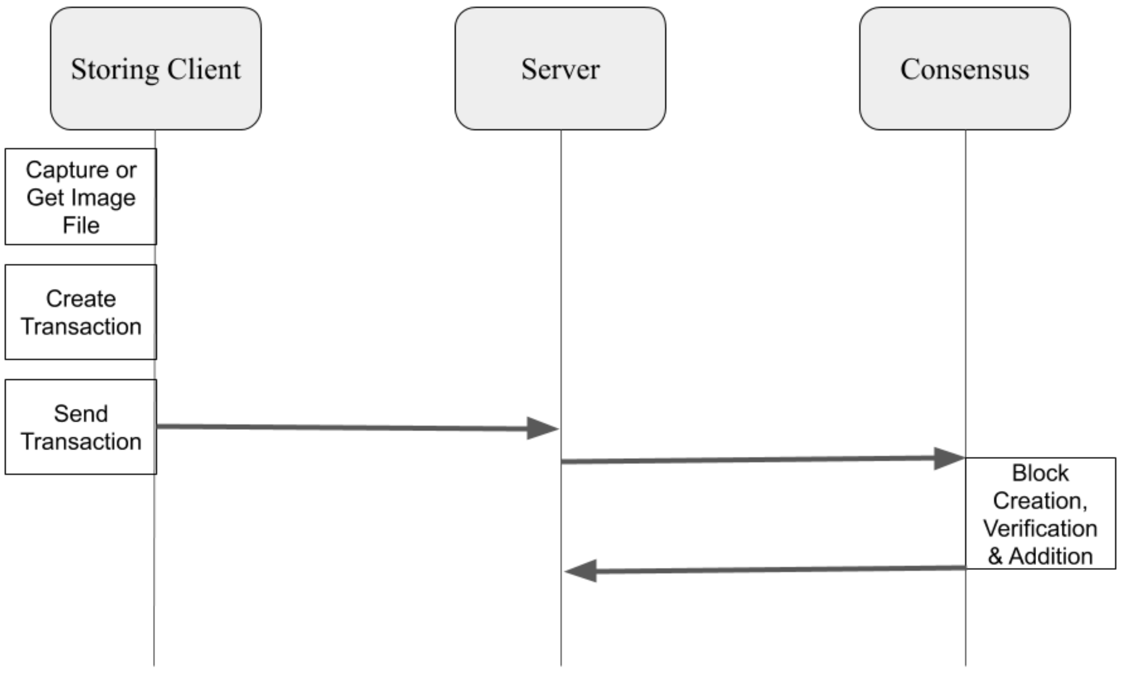

Storing Client
Server
Consensus
Capture or Get Image File
Create Transaction
Send Transaction
Block Creation, Verification & Addition

Figure 1.2: A process diagram for Image Insertion

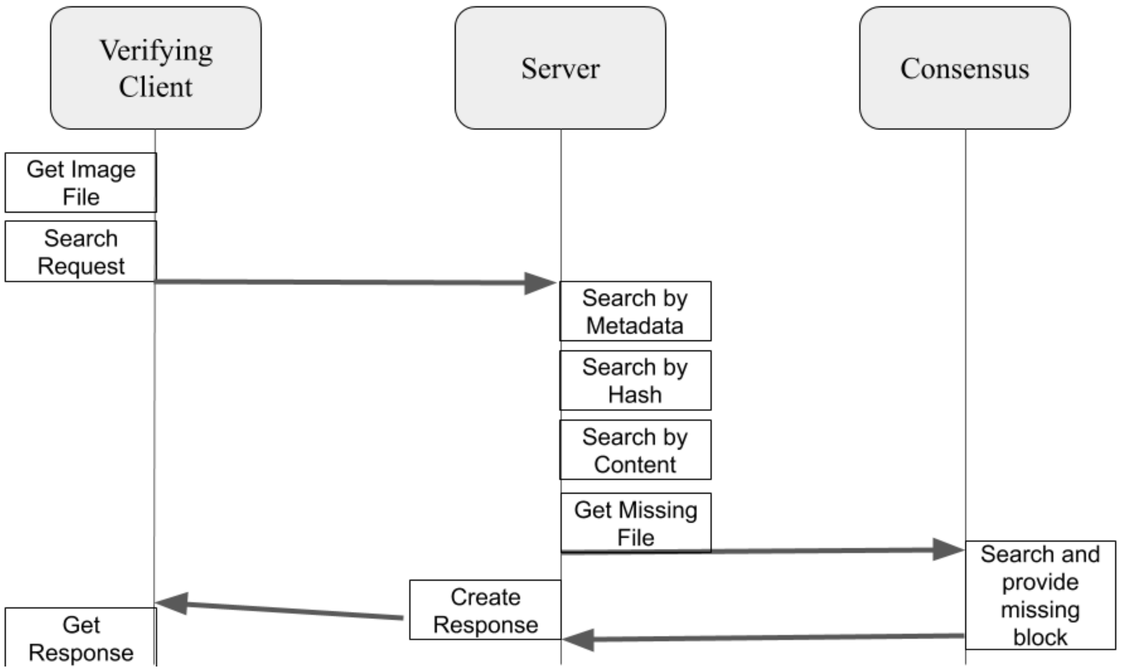


Figure 1.3: A process diagram for Image Verification

exists a huge amount of methods for detecting data tampering and detecting what has actually been manipulated. The planned proof of concept prototype will simply determine whether or not the hash of the image subjected to verification matches the original image's hash saved by the client device in the blockchain. Additionally, the use of a blockchain is not limited here to setting up a working blockchain which can handle and store unique hashes from connected Android devices.

Other delimitations are the assumption of a perfect connection between the client device and the server, *i.e.*, without any data loss. This loss free communication is required for both the upload of the image to be compared and the upload of hashes (with the meta data) to the blockchain. The issue of handling packet-loss in these scenarios are outside of the scope of this thesis.

## 1.8 Roadmap of the Thesis

The structure of the thesis is as follows:

1. The Chapter 1 is an introductory part which discusses the scope of the thesis, about the contribution of this thesis and the motivation for writing it.

2. The Chapter 2 provides the background of Image integrity security aspects of it and previous works.

3. The detailed descrption of the proposed mechanism is discussed in Chapter 3.

4. The implementation of Chapter 4, where an informal implementation of the proposed protocol has been discussed.

5. The Chapter 5 comprises of the conclusion and further work of the Project in future.

6. The Final Chapter is the Appendix *i.e.* the detaied information of the technologies used as a combination model in this project.

## 1.9 Scope of Discussion

This thesis focuses on building a hybrid architechture inspired by one of the major implementations of BlockChain i.e. BitCoin [19]. By the advancement of PHP (Hypertext PreProcessor) and JS (JavaScript) which are the basic web building languages or platforms that can be run in any modern devices, it might be quite easy to make a peer-to-peer web-api (Application Programming Interface) running like bitcoin decentraized network as well as social media platform that might be a real Truth Machine. While comparing image files data we will first compress the data using our own protocol and compare by bit matching Euclid, or Deep CNN (Convolutional Neural Network).

## 1.10 Thesis Contribution

The main contributions of this thesis includes

1. Proposes an user anonymity-preserving algorithm to be a part of Image Integrity Program.

2. Formally analyzes the security of the newly designed protocol as well as its performance.

3. The scheme, as compared to the existing schemes, not only authenticates the users but, also establishes a session key between the user and the System after successful mutual authentication.

4. The scheme provides many security and robustness features of user authentication and Block Processing scheme for BCTs.

5. No installation required to be a part of the system, except you want to be the miner.

# Chapter 2

# Background

As we all know that in India the politics and the social media is a big thing for people to consider as an important part of life. But the problem is that some social media users does think of exploiting with the content either by downloading the image or video or recording on screen and uploading it to the social media platforms. Those new images might be very sensitive and controversial and that becomes viral and put bad impact on the society.

The integrity of data has become more and more important. Security breaches that have leaked passwords and user emails are commonplace and regularly appear in the news. When a system is hacked, the first issue that usually comes to mind is the compromise of confidentiality, specifically what confidential information is now in the hands of the adversary. Another aspect, which is getting increasingly important, is to know the integrity of data, has anything been changed contextually in order to address the higher level question: “Can we trust that this data is the actual data?”. The need to verify the integrity of the data varies between different organizations. For example, in health care, incorrect information could be life threatening, while in industries such as finance in authentic data could be associated with great cost. Social media platforms are presenting images on their platform with only a very low (almost nothing) level of verification, often next to paid content and advertising. Companies paying for distribution of content and advertising have an inherent interest to know that their content is being displayed reliably and in the case of advertisements that the content displayed next to their content is reliable, *i.e.*, that it has not been compromised. This thesis will investigate the possibility of verifying the integrity of image content using blockchains. The resulting system should be capable of being implemented on various platforms, such as social media platforms.

This chapter introduces the reader to all the relevant background knowledge needed to grasp all of this thesis. Section 2.1 details what an image is generated and how it can be decoded. In Section 2.2 describes the intricacies of the main underlying concepts that comprises a blockchain. The Section 2.3 is a little introductory part of the new emarging technology Machine Learning where computers have programmed in such a way that they seems they are learning and adapting with the changes of the environment. We introduced it because we are gonna use this to compare images using their contexts. Section 2.4 talks about the security of our system with regards to device to device communication. Section 2.5 details the client framework and what components & Application Programming Interfaces (APIs) will be used to create the prototype mobile or computer client. Section 2.6 explains the front end development done for the web server, focusing mainly on the necessary JavaScript development. Section 2.7 discusses related projects that have been done both in academia and in the industry. Lastly, Section 2.8 summarizes the background information and lays a theoretical foundation for this thesis.

## 2.1 Image

Image can be described what we see at a particular instance of time, as we can sense that we are in a 4 dimentional video space 3 D's for space (height, width, depth) and 1 D for time. Images that are captured in the digital camera (like an eye) by burning a integrated circuit chip portion called memory can be stored in different file formats for future use.

Image file formats are standardized means of organizing and storing digital images. Image files are composed of digital data in one of these formats that can be rasterized for use on a computer display or printer. An image file format may store data in uncompressed, compressed, or vector formats. Once rasterized (making a matrix of intensity vaues of pixel positions), an image becomes a grid of pixels, each of which has a number of bits to designate its color equal to the color depth of the device displaying it.

The size of raster image files is positively correlated with the number of pixels in the image and the color depth (bits per pixel). Images can be compressed in various ways, however. A compression algorithm stores either an exact representation or an approximation of the original image in a smaller number of bytes that can be expanded back to its uncompressed form with a corresponding decompression algorithm. Images with the same number of pixels and color depth can have very different compressed file size. Considering exactly the same compression, number of pixels, and color depth for two images, different graphical complexity of the original images may also result in very different file sizes after compression due to the nature of compression algorithms. With some compression formats, images that are less complex may result in smaller compressed file sizes. This characteristic sometimes results in a smaller file size for some lossless formats than lossy formats. For example, graphically simple images (*i.e.* images with large continuous regions like line art or animation sequences) may be losslessly compressed into a GIF (Graphics Interchange Format) or PNG (Portable Network Graphics) format and result in a smaller file size than a lossy JPEG (Joint Photographic Experts Group) format. With vector images the file size increases only with the addition of more vectors.

There are two types of image file compression algorithms: lossless and lossy. **Lossless**

**compression** algorithms reduce file size while preserving a perfect copy of the original uncompressed image. Lossless compression generally, but not always, results in larger files than lossy compression. Lossless compression should be used to avoid accumulating stages of re-compression when editing images. **Lossy compression** algorithms preserve a representation of the original uncompressed image that may appear to be a perfect copy, but it is not a perfect copy. Often lossy compression is able to achieve smaller file sizes than lossless compression. Most lossy compression algorithms allow for variable compression that trades image quality for file size.

## 2.2 BlockChain

A BlockChain or **"The Truth Machine"** [7] can be broadly described as a peer-to-peer network of nodes that makes a collaborative effort in sensing certain specified chain of blocks of data around its periphery and thereby controls the surrounding environment. Accrrding to Wikipedia [25] a blockchain, originally block-chain, is a growing list of records, called blocks, which are linked using cryptography. Each block contains a cryptographic hash of the previous block, a timestamp, and transaction data (generally represented as a Merkle tree). In BlockChains, each node consists of processing capability, it may contain multiple types of memory like program, data and memories, having a Web-Service transceiver, Client-side processors, and a power source. The nodes communicate with each other using web-services and self-organized. There are certain nodes called miners that veryfies each transactions or entry of data in the chain and are the most reliable personnel in the network who always have the updated copy of blocks of data.

### 2.2.1 What is Blockchain

By definition blockchain is a decentralized computation and information sharing platform which enables multiple authoritative domains who do not trust each other to cooperate coordinate and collaborate in a rational decision making process.

The technology is particularly useful when multiple parties or individual they want to cooperate it with each other, and they want to come to a common platform to share the information among themselves.

Blockchain is that whenever we are talking about multiple authoritative domains.

This multiple authoritative domains they do not trust each other. So, this is an important aspect of blockchain that you can combine multiple authoritative domains who do not trust with each other and they can come to a common platform where they can cooperate, coordinate and collaborate in application development process at the business intelligence process.

Let's suppose in a blockchain platform Alice has her own copy of a document and Bob has

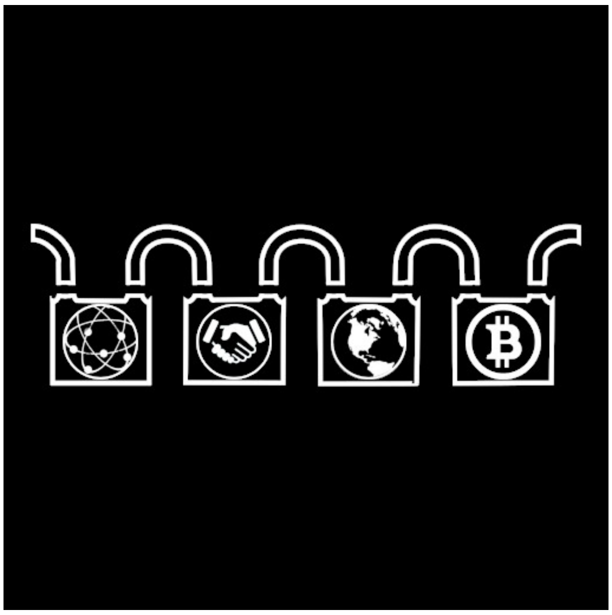


Figure 2.1: Overview of Block-Chain Technology (BCTs).

his own copy of another document. So, the seperate copy belongs to Alice and Bob, and they can simultaneously write to their own document and here I have the network in between and the network has the task to ensure that the information consistencies maintained between the documents which Bob and Alice hold individually.

The advantage is that that they do not need to rely on the internet or if a server crashes they do not depend on the server files.

So, by definition we can say that a blockchain is decentralized data based platform with strong consistency support.

### 2.2.2 Blockchain Arcitecture

So, the protocol for commitment means whenever someone is making a new transaction, during that time you need to ensure that this particular transaction if it is valid, it will get committed to the existing public ledger or the existing blockchain, otherwise that entry will not be there in the blockchain. So, there should be a mechanism for validity checking of every upcoming transactions from the clients and then based on that validity checking you will be able to either accept the transaction and include it in the existing blockchain or you can delete the transaction or discard the transaction.

The second requirements is a consensus. Consensus is an important aspect in the in the context of blockchain. So, in case of blockchain as we have discussed that everyone has a local copy of the information available to every individual parties and there is no such central platform like a bank which will maintain the consistency of the transactions or the consistency of the information. So, that is why the consensus mechanism ensures that whatever local copy every individual party has they are consistent with each other that everyone has the most updated copy, and a copy, that the individuals have, are identical or similar to each other.

The third important aspect is the security. That means, the data that one is inserting in a public ledger or inside the blockchain, now because this blockchain is distributed to individual parties everyone is maintaining their local copy of the blockchain. So, that person may change something in that local copy and broadcast that saying that "see this is the updated information".

Fourth aspects is the privacy and authenticity. The data or the transactions which is their inside the blockchain it belong to various clients. So, it is coming from various clients and you are putting that information inside the blockchain and a copy of the blockchain is available to every parties and that is why the privacy and authenticity of the information needs to be ensured.

Some of the underlying concepts of the BlockChain especially BitCoin (the most popular implementation of BlockChain) and some other important technology are briefed.

### 2.2.3 Types

There are two types of blockchain,

#### 2.2.3.1 Permissionless BlockChain

Permissionless or Public blockchain networks power up most of the market's digital currencies. They allow every user to create a personal address and begin interacting with the network, by submitting transactions, and hence adding entries to the ledger. Additionally, all parties have the choice of running a node on the system, or employing the mining protocols to help verify transactions.

Additionally, for digital currencies such as Ethereum, the blockchain network also supports smart contracts, which are automated transactions that self-execute when certain criteria are met. As Ethereum also employs a permisionless blockchain, anyone can develop and add smart contracts onto the network, with no limitation imposed by the developers. Apart from allowing anyone to get involved on the network, there are few more characteristics associated with the permisionless model. These are:

- **Decentralization:** No central entity has the authority to edit the ledger, shut down the network, or change its protocols.
- **Digital assets:** The presence of a financial system on the network.
- **Anonymity:** It do not require users to submit personal information prior to being able to create an address, or submit transactions. However, in certain cases, personal information is required for legal purposes

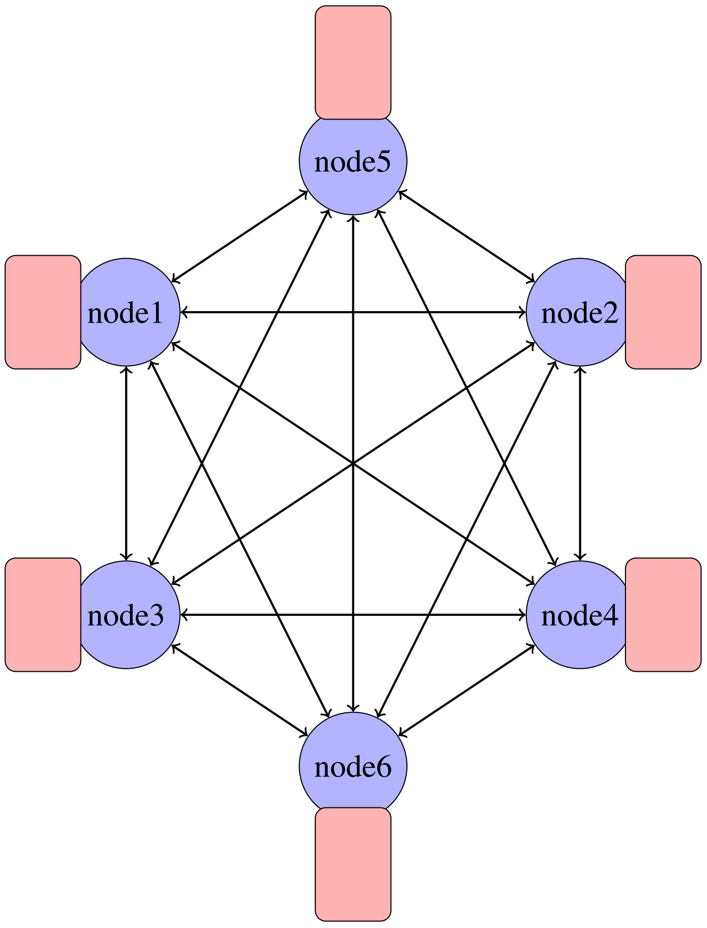
- **Transparency:** It needs to freely give users access to all information apart from the private keys from addresses, to how transactions are processed into blocks, and the freedom to see all transactions processed by the network.

#### 2.2.3.2 Permissioned BlockChain

Permissioned or Private blockchain closed ecosystems, where users are not freely able to join the network, see the recorded history, or issue transactions of their own. Permissioned blockchains are preferred by centralized organizations, which leverage the power of the network for their own, internal business operations. Company consortiums are also likely to employ private blockchains to securely record transactions, and exchange information between one another.

### 2.2.4 Peer-to-Peer Network

This is the internet protocol that connects different logged in users as a node which is having some computation power. The users who are only uploading and verifying may have computer or smartphone, but the verifiers or the miners must have to work on computer with sufficient amount of computation power.



### 2.2.5 Transactions

Everything in crypto-currency comes under transactions, i.e. someone is sending some amount of money to someone else at some time. So a basic or overall transaction data can be structured as,

```
Transaction ::  {
     <Transaction_id>,
     <TimeStamp>,
     <Sender_Id>,
     <Receiver_Id>,
     <Amount_Unit>
}
```

This transaction details is send to every peer to verify. If they heard already about it, it is true, or it is false (same as women's un-manipulated gossip in village). If more than 50% population declare it true, then it is allowed to be in the Public Ledger.

### 2.2.6 Public Ledger

This is the publicly shared record of transactions kept as a list of blocks. At a particular point of time everybody (every node), who are connected to the network, should have a same copy of ledger in their own devices of what server has. Whenever a new person logs into the network aotomatically the server forces to update the ledger to the person's device. So basically the ledger is the chain file.

### 2.2.7 Chain of Blocks

It means list of blocks of data having some common part with previous and next block. This is like Single-way linked-list(every node consists of data and program memory address to the next node). Every block contains a number of transactions and many more things.

```
Block ::  {
     <Block_Id>,
     <TimeStamp>,
     <Merkle_Root>,
     <Verifier_Id>,
     <Nonce_Value>,
     <Previous_Hash>,
     <Current_Hash>,
     <Data ::  ANumberOfTransactions>
}
```

So it is kind of backword linked-list which is propagating by having the previous block's kind of identity (because there is a mild chance of collision i.e. multiple values' hashes are same) hash.

### 2.2.8 Timestamp

The time means absolute global date-time in the format:

```
TimeStamp :: String ( <day_of_week_code :: ddd>,
<month_code :: mmm>, <day_of_month :: dd>, <year ::
yyyy>, <time :: hh:mm:ss>, <Distance from Mean-TimeLine
:: GMT+hhmm>, <Time Zone :: Country_Name Standard Time>)
```

Example: **"Sat May 25 2019 20:45:04 GMT+0530 (India Standard Time)"**. The timestamp is one of the most needed to prove it later for verification of the record. It is used to create the current block's hash.

### 2.2.9 Hash

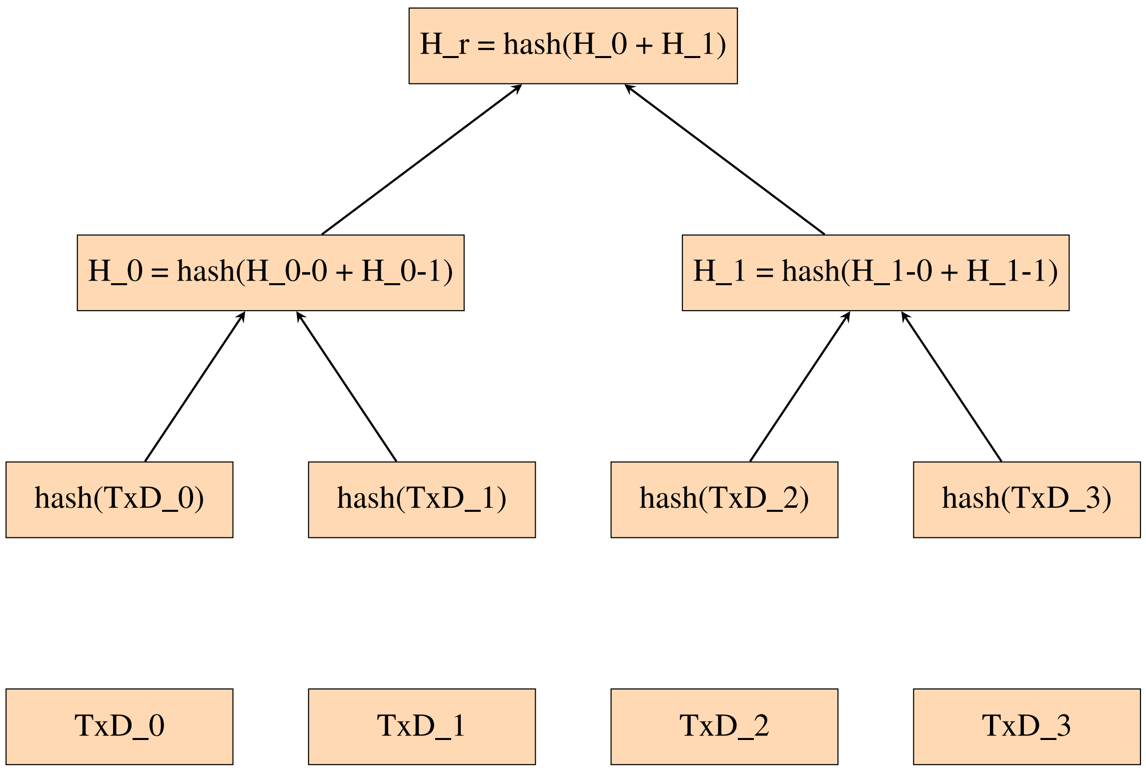
The job of a hash algorithm is to map any size of domain to a particular size of range. SHA-256 is one of the most popular hash algorithms which takes any length input and returns 256 bit output. All input/output operations can be transferred into strings.

### 2.2.10 Merkle Root Hash

Merkle tree is a complete binary tree which has the hash values of transaction data as leaf nodes. The tree propagation occures from leaves to root as tournament tree form. For any point,

parent hash = hash(child-1 hash + child-2 hash);

A little change in any data of any transaction will change the merkle root, and thus the block's hash and the complete chain. Because it is used to create the current block's hash.



### 2.2.11 Previous Hash

The previous block's hash. It is required to maintain the chain system because we can not create the next address and we don't know when it would be created, so it is better to store what we already have. It is used to create the current block's hash.

### 2.2.12 Nonce

It is the quantity of computation power used to solve a mathematical problem which is not so hard but not so easy either. Too easy solution will be easy to break and too hard solution will take so long time to create a block that the adversary with huge computation power have a chance to alter the data before creating and verifying a block. Rather a medium hard problem will be better. It is used to create the current block's hash.

This is to show the verifiers that the block creator have spent sufficient amount of computation power before creating the block and also to delay the process a little bit and this is called POW (Proof of Work). In bitcoin the problem is to find the first hash of given values which is having 'd' number of leading 'zero's where the 'd' represents the difficulty of the problem i.e. the more 'd' gets it will take longer to calculate. Typical value of 'd' is 32 bits and average delay for the whole block addition (create, verify then add) is about 10 minutes.

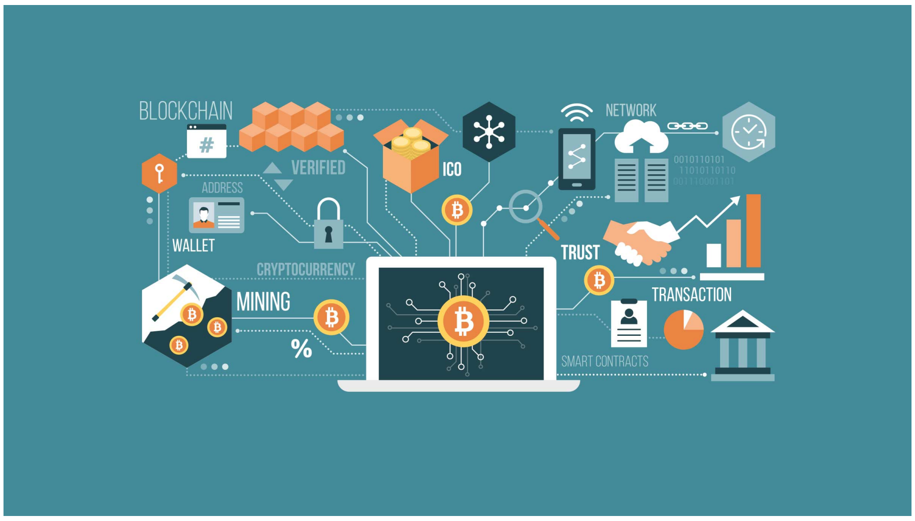


Figure 2.2: Overview of BitCoin Technology (BCTs).

### 2.2.13 Consensus Mechanism

It is the contract or the protocol by which the blocks are verified and the winning blocks are added to everyone's ledger as well as the central server. The actual consensus algorithm is not published for security reasons. But by possible ways or reverse eengineering people have created different models. Some of those models are:

1. Probable bitcoin consensus mechanism
2. Paxos (Part-Time Parliament) consensus mechanism [16, 17]
3. Raft consensus mechanism [21]

Bitcoins one is probably the simplest one, but having bugs. The Paxos allows different types of sources and faults to come, it learns and fixes it. Raft does not allow any fault to happen.

The basic overall way in which consensus happen in bitcoin might be the following,

1. The transaction comes to server from client nodes;
2. Server stores that in file system database as unverified transaction;
3. At the point of interval of 10 minutes server broadcasts it to the miners network i.e. to every miner;
4. Each miner individually verifies and adds correct transaction in a block. Each node holds its block creation and validates the new block as soon as it gets new block from network. Who's block is valid and introduced first to network will be added to everyone's chain and they will start making new block on top of it. Sometimes forks might be created for the

networking distance between distant nodes, at that moment conflict will come. If a new introduced block's previous hash does not match with the last block's current hash the node requests to the network to get the missing blocks one by one until the blocks match, it validates it, delete the wrong blocks (make them orphan) and add missing blocks to own chain to resolve fork and maintain longest updated chain. So it is a race between miner nodes.

5. Server gets a copy from miners group and updates own copy and delete the added tansactions from file system.

### 2.2.14 Applications of BCTs

Block-Chain Technology provides one major advantage over conventional centralized database system: immunity from unexpected data changes or Hacks, which gives rise to numerous applications. Some of them include

- Crypto-currency: Creating and transferring digital money, Data Mining.
- Military applications: Secure and verified records of Every Military Events and documents.
- Structural health Monitoring
- e-Biding Systems
- Election System or e-Voting Systems
- Selling Records and other Commercial Applications
- Music Copyright Verification System
- Integrity Analysis of Media Files

### 2.2.15 Application

Some of the commercial applications built on blockchain technology are as follows,

#### 2.2.15.1 BitCoin

One year after creating protocols of BlockChian for cryptocurrency, Satoshi Nakamoto in 2009 released the first source code for BitCoin (BTC) application. Bitcoin has since grown rapidly to a market footprint of around 39 billion U.S. dollars (as of 12 Jul 2017). BTC is a decentralized cryptocurrency using a permissionless blockchain that every user has the ability to download and thereby check. The users of the network use their personal computational power to verify transactions through hashes, i.e., mining. The total computational power of the Bitcoin network is estimated to be more than 256 times the power of the world's 500 top supercomputers. Site: BitCoin.

#### 2.2.15.2 HyperLedger

Hyperledger is a project under the Linux Foundation which seeks to provide a unified blockchain architecture for industrial applications. Hyperledger is a collaboration of more than 100 organizations focused on banking, supply chain, and other transaction networks. It uses BFT system to reach consensus by fewer number of nodes. HyperLedger

#### 2.2.15.3 Ethereum

It is a permissionless blockchain made as a smart contracts platform run by Ether (ETH). The verification is done through the peer to peer network which constitutes Ethereum together with ETH which fuels the network, the consensus algorithm to share the state of the network, and the Turing complete scripting language which enables users to write complex scripts (smart contracts). It is the consensus engine which sets the speed of the Ethereum network by updating the state of the network at specific time intervals (shorter than for the Bitcoin network). The Ethereum network is fueled by gas which is a unit connected to ETH which is used to pay for the storage and computations an application (smart contract) needs. Ethereum

Solidity is a programing language build specifically to target the Ethereum virtual Machine (EVM) which in turn is the protocol to access the blockchain. Solidity has a syntax similar to JavaScript and is the language in which many smart contracts are written. The smart contracts are then compiled into bytecode and fed into to EVM and thus they can operate on the blockchain. In Ethereum are every transaction and smart contract is saved on the blockchain. The blockchain is in turn distributed and the states are handled by the consensus algorithm. Solidity

#### 2.2.15.4 Tendermint

Tendermint is not a blockchain in itself but rather a general purpose blockchain consensus engine with a consensus layer for blockchaining. Tendermint can be used as a replacement for the consensus engines of other blockchains. It runs BFT algorithm. Tendermint

#### 2.2.15.5 Eris:db

Eris:db is a controllable (permissible), smart contract-enabled, PoS based blockchain design with the PoS based on Tendermint. Eris:db is an application layer for blockchain applications. Eris is the backbone for deploying and interacting with the application logic. Eris:db

#### 2.2.15.6 Cumulus

Cumulus is a blockchain solution developed by Ericsson. Cumulus builds on and works with the Ethereum Virtual Machine (EVM) but separates the consensus algorithm and the blockchain from the EVM. Cumulus works similar to Ethereum; however, it is a private blockchain which means it does not run on gas. Cumulus

### 2.2.16 Comparison

There are a number of blockchain solutions to choose from, some of the most important ones are described in the previous subsections. For this thesis we will use a hybrid one, that supplies our requirements or logical proposal.

| Name | Distribution | Transactions/second | Consensus Algorithm |
|---|---|---|---|
| **Hyperledger** | Private | 10k | BFT |
| **Eris:db** | Private | 10k | Tendermint(PoS) |
| **Ethereum** | Public | 20 | PoS |
| **Tendermint** | Private | 10k | Tendermint BFT |
| **Bitcoin** | Public | 7 | PoW |
| **Cumulus** | Private | (untested) | (unknown) |

### 2.2.17 Security and Integrity in BCTs

As the data is not sitting on a single data server so there is no security issue for Server Hacking. And also the hash-Chain with Cryptography makes it near to impossible to figure out or change previous data block in the blockchain. Before adding any data block with the help of consensus mechanism the blocks are verified with the digital signature of the node and some solution of nonce (the number of times the cllient application required to calculate the mathematical problem) and various other meta data. As in some interval the system is refreshing itself, if some error seems to be occured, it backs up itself by contacting the miners (the trusted nodes), compare their files and back up with the valid one. So no integrity problem is there, unless and until the internet works fine.

## 2.3 Machine Learning

A machine is said to be learning from past Experiences(data feed in) with respect to some class of Tasks, if it's Performance in a given Task improves with the Experience.

According to a famous person, Machine Learning is a system that can learn from example through self-improvement and without being explicitly coded by programmer. The breakthrough comes with the idea that a machine can singularly learn from the data (*i.e.* media database of real life objects, numbers dataset from algorithms, face images, job dtabase) to produce accurate results.

Machine learning combines data with statistical tools to predict an output. This output is then used by corporate to makes actionable insights. Machine learning is closely related to data mining and Bayesian predictive modeling. The machine receives data as input, use an algorithm to formulate answers.

A typical machine learning tasks are to provide a recommendation. For those who have a Netflix account, all recommendations of movies or series are based on the user's historical data. Tech companies are using unsupervised learning to improve the user experience with personalizing recommendation.

### 2.3.1 Machine Learning vs. Traditional Programming

Traditional programming differs significantly from machine learning. In traditional programming, a programmer code all the rules in consultation with an expert in the industry for which software is being developed. Each rule is based on a logical foundation; the machine will execute an output following the logical statement. When the system grows complex, more rules need to be written. It can quickly become unsustainable to maintain.

Machine learning is supposed to overcome this issue. The machine learns how the input and output data are correlated and it writes a rule. The programmers do not need to write new rules each time there is new data. The algorithms adapt in response to new data and experiences to improve efficacy over time.

### 2.3.2 How does Machine learning work

Machine learning is the virtual brain where all the learning takes place. The way the machine learns is similar to the human being. Humans learn from experience. The more we know, the more easily we can predict. By analogy, when we face an unknown situation, the likelihood of success is lower than the known situation. Machines are trained the same. To make an accurate prediction, the machine sees an example. When we give the machine a similar example, it can figure out the outcome. However, like a human, if its feed a previously unseen example, the machine has difficulties to predict.

The core objective of machine learning is the learning and inference. First of all, the machine learns through the discovery of patterns. This discovery is made thanks to the data. One crucial part of the data scientist is to choose carefully which data to provide to the machine. The list of attributes used to solve a problem is called a feature vector. You can think of a feature vector as a subset of data that is used to tackle a problem.

The machine uses some fancy algorithms to simplify the reality and transform this discovery into a model. Therefore, the learning stage is used to describe the data and summarize it into a model.

For instance, the machine is trying to understand the relationship between the wage of an individual and the likelihood to go to a fancy restaurant. It turns out the machine finds a positive relationship between wage and going to a high-end restaurant: This is the model

#### 2.3.2.1 Inferring

When the model is built, it is possible to test how powerful it is on never-seen-before data. The new data are transformed into a features vector, go through the model and give a prediction. This is all the beautiful part of machine learning. There is no need to update the rules or train again the model. You can use the model previously trained to make inference on new data.

The life of Machine Learning programs is straightforward and can be summarized in the following points:

1. Define a question
2. Collect data
3. Visualize data
4. Train algorithm
5. Test the Algorithm
6. Collect feedback
7. Refine the algorithm
8. Loop 4-7 until the results are satisfying
9. Use the model to make a prediction

There are two main types of Learning,

#### 2.3.2.2 Supervised learning

An algorithm uses training data and feedback from humans to learn the relationship of given inputs to a given output. For instance, a practitioner can use marketing expense and weather forecast as input data to predict the sales of cans.

You can use supervised learning when the output data is known. The algorithm will predict new data.

There are two categories of supervised learning:

**Classification** Imagine you want to predict the gender of a customer for a commercial. You will start gathering data on the height, weight, job, salary, purchasing basket, etc. from your customer database. You know the gender of each of your customer, it can only be male or female. The objective of the classifier will be to assign a probability of being a male or a female (*i.e.*, the label) based on the information (*i.e.*, features you have collected). When the model learned how to recognize male or female, you can use new data to make a prediction. For instance, you just got new information from an unknown customer, and you want to know if it is a male or female. If the classifier predicts male = 70%, it means the algorithm is sure at 70% that this customer is a male, and 30% it is a female.

The label can be of two or more classes. The above example has only two classes, but if a classifier needs to predict object, it has dozens of classes (*e.g.* glass, table, shoes, etc. each object represents a class).

**Regression** When the output is a continuous value, the task is a regression. For instance, a financial analyst may need to forecast the value of a stock based on a range of feature like equity, previous stock performances, macroeconomics index. The system will be trained to estimate the price of the stocks with the lowest possible error.

#### 2.3.2.3 Unsupervised learning

In unsupervised learning, an algorithm explores input data without being given an explicit output variable (*e.g.* explores customer demographic data to identify patterns)

You can use it when you do not know how to classify the data, and you want the algorithm to find patterns and classify the data for you

Challenges and Limitations of Machine learning The primary challenge of machine learning is the lack of data or the diversity in the dataset. A machine cannot learn if there is no data available. Besides, a dataset with a lack of diversity gives the machine a hard time. A machine needs to have heterogeneity to learn meaningful insight. It is rare that an algorithm can extract information when there are no or few variations. It is recommended to have at least 20 observations per group to help the machine learn. This constraint leads to poor evaluation and prediction.

### 2.3.3 Application

There are plenty of application fields where ML is being used for better performance,

#### 2.3.3.1 Augmentation

Machine learning, which assists humans with their day-to-day tasks, personally or commercially without having complete control of the output. Such machine learning is used in different ways such as Virtual Assistant, Data analysis, software solutions. The primary user is to reduce errors due to human bias.

#### 2.3.3.2 Automation

Machine learning, which works entirely autonomously in any field without the need for any human intervention. For example, robots performing the essential process steps in manufacturing plants.

#### 2.3.3.3 Finance Industry

Machine learning is growing in popularity in the finance industry. Banks are mainly using ML to find patterns inside the data but also to prevent fraud.

#### 2.3.3.4 Government organization

The government makes use of ML to manage public safety and utilities. Take the example of China with the massive face recognition. The government uses Artificial intelligence to prevent jaywalker.

#### 2.3.3.5 Healthcare industry

Healthcare was one of the first industry to use machine learning with image detection.

#### 2.3.3.6 Marketing

Broad use of AI is done in marketing thanks to abundant access to data. Before the age of mass data, researchers develop advanced mathematical tools like Bayesian analysis to estimate the value of a customer. With the boom of data, marketing department relies on AI to optimize the customer relationship and marketing campaign.

#### 2.3.3.7 Supply Chain

Machine learning gives terrific results for visual pattern recognition, opening up many potential applications in physical inspection and maintenance across the entire supply chain network.

Unsupervised learning can quickly search for comparable patterns in the diverse dataset. In turn, the machine can perform quality inspection throughout the logistics hub, shipment with damage and wear.

For instance, IBM's Watson platform can determine shipping container damage. Watson combines visual and systems-based data to track, report and make recommendations in real-time.

In past year stock manager relies extensively on the primary method to evaluate and forecast the inventory. When combining big data and machine learning, better forecasting techniques have been implemented (an improvement of 20 to 30 % over traditional forecasting tools). In term of sales, it means an increase of 2 to 3 % due to the potential reduction in inventory costs.

**Google Car** For example, everybody knows the Google car **Waymo**. The car is full of lasers on the roof which are telling it where it is regarding the surrounding area. It has radar in the front, which is informing the car of the speed and motion of all the cars around it. It uses all of that data to figure out not only how to drive the car but also to figure out and predict what potential drivers around the car are going to do. What's impressive is that the car is processing almost a gigabyte a second of data.

For our case we use logistic regression (discussed in Appendix app:4) and the algorithm is discussed in Proposal chapter 3.

## 2.4 Security

In this thesis project data integrity is of paramount importance. In order to be able to trust the data stored within the blockchain, a reliable secure connection is created between the application running on cleent's device and the computer that is computing the blockchain. User side application collects the data and creates the consecutive transaction which is to be sent between the device and the server where the blockchain computations are done, therefore it is impossible for a third party to know what the plaintext representation of the image is. As a result encryption of this transaction might be of less importance. However, it is crucial for the system to know that the integrity of the hash is intact. The image captured and stored at a device can later be verified as to what is received by another device by using the blockchain. An HTTP connection is used between the device and the server. The HTTP connection will in future implementation be replaced with a HTTPS connection to secure the communication.

## 2.5 Client Side

A client may have a smartphone (of the brand Android, iPhone) or a computer (laptop or desktop). All of these devices comes with a default web-browsing tool like Google Chrome, Microsoft Edge, Mozilla Firefox or Mac Safari etc. All else a client needs an active reliable internet connection to communicate with sever. Moreover the miners have to have computers with better computation power. With the advancement of web programming languages with lots of APIs now it is very easy to capture an image or handle a local file and pass the data to the peer-to-peer network.

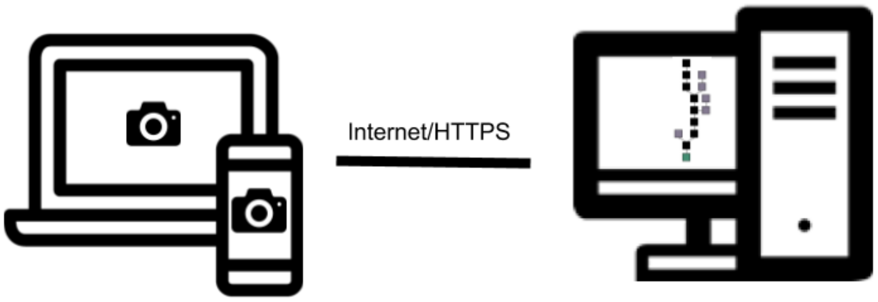


Figure 2.3: A secure connection between Client and Server

### 2.5.1 JavaScript

In this thesis, basic js is used as the run-time environment together with the inbuilt JS library installed on the browser. We are trying to use Node.js (also along with our code) that runs there servers, i.e., the consensus server. Note that all three servers run within the node.js environment on the computer hosting the web server and blockchain. JavaScript is the front end development language which realizes the functionality of the web page. JavaScript works together with PHP (HTML in it) and CSS in order to deliver a nice looking functional web page.

## 2.6 Server Side

##### 2.6.0.1 PHP Server

The Server side is made of PHP (HTML-5). In order to easy communicate with the blockchain from both the client device and the server running the verification client PHP APIs used over the HTTP protocol, usually to the server's TCP port 80. An apache server is basically a web server architecture which enables clients to make a request of servers. The HTTP protocol builds upon a request/response relationship between a client and a server and in this case used within the LAMPP architecture. In HTTP a string is sent to a specific port on the server with a specific method predefined by the HTTP standard. The most common methods are: GET, HEAD, POST, and PUT. The POST method is used by both the Android device and the web client of this thesis project to give both the ability to add and via the reply to this POST request to get information from the server. The communication between JS and PHP is done with XML-HTTP request with JSON byte stream. There is every type of handwriten PHP code to handle any type of request to Server.

#### 2.6.0.2 POST Method

The POST method is a standard method in the HTTP protocol and gives the user the ability to add and receive data from the server. A HTTP transaction session is established through a TCP connection to a specific port on the server. After the TCP connection is established, of plain text HTTP message is sent to the other who either confirms the transaction or responds with an error message. It is in the body of this message that the response to the sender will be sent. Within the scope of this thesis the response will either be a boolean variable showing a successful transmission or a string of hashes requested from the blockchain.

#### 2.6.0.3 HTTPS

Hypertext Transfer Protocol Secure (HTTPS) is an encrypted version of the HTTP protocol. HTTPS is usually used for sensitive information such as banking, private information, or classified information. HTTPS usually uses port 443 rather than HTTP port 80. In order for a user to be able to trust that a server is who it says that it issigned, third party certificates are used, usually in conjuction with the use of the Secure Socket Layer (SSL) as a secure transport protocol. A third party provides the server with a signed certificate which the user of the service verifies. Now that the user has verified the identity of the client and the server has verified the identity of the client - and encryption between the two parties may be set up. The security depends on that the certification chain work, that the user knows which server to use and that the user (client) notices when the certificate is incorrect or missing. Furthermore, in order to provide a high level of security the keys and the algorithms used need to be strong.

## 2.7 Related Works

### 2.7.1 Old possible works

The process of image integrity checking is old job and many algorithms have been made. The following is the overview of them.

1. Store the real images in some database. Match the testing image with the existing one bit by bit.

2. Like in previous case, store it and check the new one's hash with the existing one's hash.

3. Searching the image in different databases using object detection and context similarity.

4. Every digital image is created in an electronic device which is having some unique property or signature in the world. The softwares that has created the image are designed in such a way that they put the digital signature in the metadata field of the image, let's say EXIF data for JPEG images. Any software that knows the byte signatures can detect the random image is original or not by checking the signature and named context that lies inside of the image file.

5. By checking the image context with reality or possiblity, some images can be checked.

6. An expert or hacker not only seeks for the context or visual similarity, but an expert knows that secret figures (WaterMarking) secret messages (Steganography) can be embedded in the image file, because the main image is a 2d matrix of intensity values (list of integers for BnW, RGBA, CMYK, or other) at different pixel positions.

7. The modern approach is to use deep neural network that learns and tries to detect image integrity in about 90% efficiency and accuracy.

8. Image Forensics uses about all of the previous ones.

### 2.7.2 Related Work

This section presents related work done both in the academic world and the industry.

#### 2.7.2.1 Academic

Three different academic papers have been identified as relevant to this research. The following subsections highlight their main points and relevance to this thesis project.

**Timestamping video footage in traffic incidents** In [13] Gipp, Krosti, and, Breitinger investigated the use of the Bitcoin network to timestamp a video feed from a smartphone camera placed in a car in the event of an accident.

**Trusted Timestamping** With regards to trusted timestamping two reports were identified: "Trusted Timestamping" [14] and "Commitcoin" [8]. Both solutions leverage the time stamp made using the Bitcoin protocol when creating a transaction together with the carbon dating nature of the blockchain (i.e., one can tell the rough date of an entry by looking at the sequence of time stamps). These two solutions are slightly different in terms of their execution, but the basic theory of using the existing Bitcoin blockchain is the same.

**Forensics Investigations of Multimedia Data** In [22], R. Poisel and S. Tjoa review the latest trends in forensic investigations of multimedia data, i.e. images, videos, and audio files. They describe different methods for determining what has been done to images and expose fabrications down to which details within a picture have been tampered with.

**Digital Watermarking** I. Echizen, et al. [12] insert digital watermarks into video files to detect data tampering. They begin by breaking a video file into its composonents: the video, the audio, the timecodes, and the eader. The header is used together with the timecodes to separately watermark the audio and video.

#### 2.7.2.2 Industrial

Three different industry solutions have been identified as being relevant. The following subsections highlight their main points and relevance to this thesis project.

**Nexan - Assureon Archive Storage** In Assureon Archive Storage [20] fingerprints are created from the data in order to prove the integrity of files within a data archive system. The original files are stored on at least two different disks or at two different geographical locations.

**Enigio - time:beat** The product series "time:beat" [3] includes: time:shot, time:stamp, time:grab, and time:mail: by Enigio. It uses to archive time stamped fingerprints of integrity sensitive materials: email, pictures, documents, and websites in a permissioned blockchain controlled and owned by Enigio.

**Ascribe** Ascribe [15] helps artists to create a digital copy of their work and time-stamp it in the Bitcoin blockchain. When a file is uploaded, Ascribe creates a digital certificate which can be traded, tracked, or loaned via the blockchain using SPOOL [11].

### 2.7.3 Curent Works

:

1. There are plenty of image storing and searching by image in the websites. In the list Google Image, Yandex Images they use the image searching by name, context.
2. Some of them uses object detection Google Images, Wolfram Images.
3. Some website programs use reverse image search like Google Images, Reverse Image Search.
4. There are plenty of softwares like the photo eding softwares itself like Photoshop, GNU Image Manipulation Program, and other editors.

5. In [5] they used CNN for image comparing and many other that can be mentioned for well known face recognition problem.
6. In [6] they used a mechanism for video integrity analysis.

## 2.8 Summary

The different parts and aspects of the proof of concept prototype were broken down and analyzed. From the smallest components of the blockchain consensus algorithms, to which existing blockchain platform should be utilized. It also talked about the ways of communication between server and client.

# Chapter 3

# Proposal

Using the concepts of Block-Chain we propose our algoorithms for Image Integrity Analysis D-APP system.

We are trying to build a web-based social media application like WhatsApp and that will be connected with the online server running blockchain in it. The process of verification of the media (text, image or video) for checking fakeness or scam in a nutshell will be as follows,

As mentioned earlier, the main jobs of our project to build a small social media platform that will store images, show them in news feed and not only that, someone can bring an image and perform search operation in the system to check if we know about the image or not, i.e. if the image is valid or not.

The system we are proposing is a hybrid of previous strict systems in terms of Permission (Permissioned/Non-permisionalized) and Centralization (Centralized/Decentralized) along with some extra mechanisms. And also we preserve the scalability of the implementation that it is not necessary to have a dedicated huge computation power for all.

## 3.1 Workflow

The following is the control flow of the overall processes,

1. An user will have to log into the system first to be a part of the system;

2. There are two types of users, some are normal users and rest are miners who have some extra powers as well as responsibilities.

3. Normal users are those who will come to share their images by upload or taken from their camera interface and also search images for checking. In their home page they can scroll and click an image for viewing, search signatures, download images to own devices.

Figure 3.1: Consensus mechanism

4. The miners are the creators of the blocks i.e. they will have computation power to generate the blocks, store the blocks in their own computers and broadcast them in the miners network, gossip among each others. They verify transactions before putting into a block.

## 3.2 System Design

In one side of the system, the normal user can request the system either to store an image or to verify. On storing request the server will help the client's device to resize the image in a particular dimention (image preprocessing) and generate the correct request format in terms of transaction (Tx). Then the server will create a copy of it making a symmetric key encryption over assymetric key encryption of the user and store in own file system database in unverified or new transaction file.

On a certain time interval (not so long, not so small, bitcoin takes 10 minutes, we will take it 5 minutes) the miners will be verifying these pool of unverified transactions and adding them in new blocks in parallel through consensus mechanism. Our consensus mechanism works as follows,

### 3.2.1 Consensus mechanism

1. At a time interval the server will broadcast the transactions or the miners will get them on request from the server database through XML-HTTP or SFTP protocol.

2. The miners either individually or by gossiping will verify the transaction data. Verifying by own will create the uniqueness, by gossiping maintains the correctness that prevent

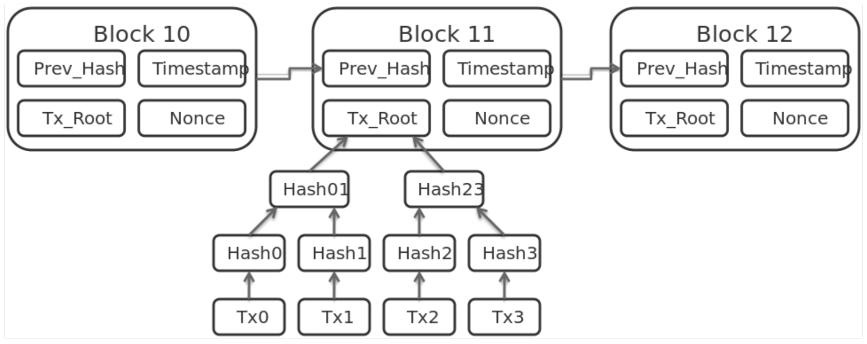


Figure 3.2: Block Creation

the fault from any side (server, other miners, itself).

3. After they delete the invalid transaction set they will start creating new block out of it.

4. After a node discovers or completes creating a new block, it broadcasts to the miner network and waits for confirmation and others blocks. After a node gets a new block, it holds it's own computation and validate the new block. If the new block is valid and the node have all previous blocks that should be linked in the chain by previous hash the node keeps it into own temporary space, and wait for others for confirmation, and if not valid then resume own computation. After all blocks say we have a valid block (either own or someone else's) to each other, they compare with what other nodes are having. The winner is one of the blocks which is having most of the criterias below,

   (a) Biggest in size $\Longrightarrow$ It is having maximum number of transactions

   (b) Bigger nonce $\Longrightarrow$ Most computation power is spent for it.

   (c) Lowest rewarded miner $\Longrightarrow$ To remove partiality and create balance and good understanding between miners.

   After 5 minutes the server requests the miners to get the decided block to be added in the chain. Server validates the block and adds it in the chain and remove the transactions that are either added or marked as invalid from the transaction pool.

5. On request of an image verification, first the image metadata is being checked (1st phase). If the metadata contains the our system generated signature, it will decrypt it and check in that particular block or the range of the blocks for the transaction that contains the image as well as the user's profile for it. Becacuse while creating downloadable file the system encrypts and adds the metadata in the image file. If found, it generates the confirmation message and the links regarding it. And if not then the system requests for time and search the entire blockchain database for hashes (2nd phase). If found it does the same. And still if it not found, it runs a deep CNN for searching for contexts or by object for maximum possible proof (3rd phase). The result it gives now is the final result.

6. The process continues again and again recursively.

## 3.3 Image Comparison Scheme

The following diagram shows how all the 3 steps the system should follow to chech if an image is already in the system or not *i.e.* Signature Match, Hash Match, Context Match.

The image comparison technique we are trying to build at 3rd level of verification is as follows. Comparing the two images byte by byte, if the difference between two images is less then the threshold value then image will be same, otherwise we consider that comparison as -(ve) result. To decide the value of threshold we will use machine learning, in which we train our system to decide the threshold value.

1. We will "Get Uploaded Bytes" (GUB) from the file using file uploader api of HTML-5;

2. Then we will try to "Extract Metadata" and Search for digital signatures that is expexted to be generated and included by our system;

3. Checking if "Signature Found" is true go to 4, else go to 9;

4. Process Signature to get Blocks and Transaction metadata, *i.e.* **who**, **where**, **when** created **what**.

5. "Match Current Metadata with Transactions" (MCMwT) in that Block.

6. If "Is Match Found" gives No do nothing, else go for 2nd level matching (next step).

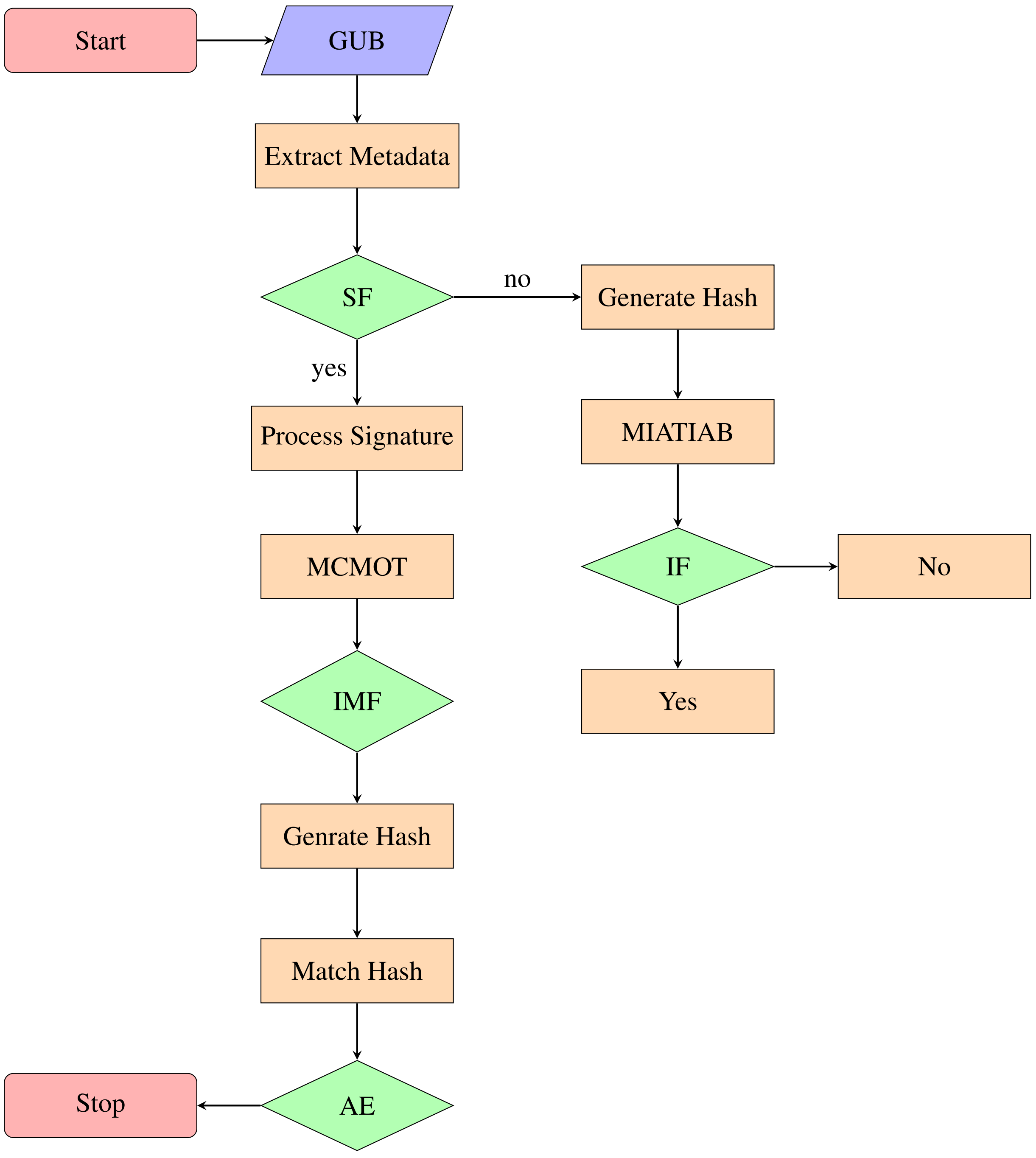
7. "Generate Hash" from the image and "Match Hash" between the narrowed down two images.

8. If "Are Equal" gives true then the Exact match has found, else go for level-3 verification for the two images (discussed in previous paragraph).

9. "Generate Hash" for 2nd level verification.

10. “Match in All Transactions in All Blocks” for the hash.

11. If “Is Found” gives Yes that means an exact match found, else go for level-3 verification for context “Match in All Transactions in All Blocks” and it will take a huge amount of time.



# Chapter 4

# Implementation

## 4.1 Setup

- Computer: Dell, Ubuntu 16.04 LTS
- Runtime Platform: 1and1 web-server in website TheScienceUniverse
- Server: Localhost, Apache, MySQL, PHP, PHP-MyAdmin
- Client: Browser (Chrome, FireFox) with HTML-5, CSS-3, JavaScript-5 support, Active Internet Connection

## 4.2 Tools

The list of tools and important functions used:

- Image API (JS-5 inbuilt)
- File API (JS-5 inbuilt)
- SHA-256 (Discussed in A.1)
- Base-64 (6 bits encoding scheme replacement of 8 bit executable scheme)
- Image API with Canvas (JS-5 inbuilt)
- AES (Discussed in A.3)
- RSA (Discussed in A.2)
- LR (Discussed in B.1)

We have made basic blockchain platform for a single node processing. The screenshots of the processes are the following steps.

## 4.3 Directory Structure

The file system we built are as follows,

```
blockcain/ – the root folder
|       index.php – Home Page (ToDo, News Feed, Others)
|       auth/ – files for LAMPP authentication system (registration & login)
|       client/ – files to be run on clients’ machine
|       |       b_crt.php – home page for creating blocks by upload or webcam
|       |       b_upl.php – upload block directly from client software (NOT BUILT YET)
|       |       b_vrf.php – verify page by uploading image
|       |       profile.php – profile for user account
|       |       mine.php – mining home for miner account
|       |       webcam.php – use webcam to capture and create transaction to send
|       |       upload.php – use file upload to create and create transaction to send
|       |       css/ – cascading style sheets for home and other pages
|       |       js/ – JavaScript files to be run by browsers
|       |       |       script.js – for index page scripting
|       |       |       base.js – basic function collection
|       |       |       sha256.js – sha256 implementation (Hex String In – Hex String Out)
|       |       |       base64.js – base64 implementation (codec base 64 String – base 256 String)
|       |       |       upload.js -}
|       |       |       webcam.js -} get, modify image by canvas, create transaction, send to server
|       |       |       verify.js – scripting for image verification
|       |       |       crt_blk.js – get data, send to php for creating files
|       |       |       mine_comm.js – scripting for miners
|       server/ – server side computation
|       |       gfc.php – get from chain (give data from existing chain)
|       |       atc.php – add to chain (update chain, add file to filesystem)
|       |       rtc.php – real time connect (creating peer-to-peer network with JS-5)
|       |       cron.php – periodic cron task script
|       |       vrf.php – verify chain for security and integrity
|       res/ – the resources
|       |       chain.json – list of blocks
|       |       uvf_txd/ – unverified pool of json transactions
|       |       tmp_blk/ – json blocks held for review
|       |       blk/ – verified and added json blocks
```

## 4.4 Process Flow

The process flow in the users' (normal user or miner) perspective is as follows,

1. User have to be authosized (Figure: 4.1) means register (sign-up) and log in (sign-in) to the system

2. User becomes a node and gets its home page (Figure: 4.2)

3. There are links to reach Profile (Figure: 4.6), Block Creation (Figure: 4.3), Block Uploadation, Image Verification (Figure: 4.8) Pages

4. Profile page consists of users personal details only the user can view and change. If the user is a miner, then the list of blocks created is mentioned in the page and link to mine blocks further.

5. In the home page after clicking on the Block Creation link the user gets on to block creation page (Figure: 4.3). Actually the name of the page should be Image Insertion. There are two options (image links for two pages) there, one is WebCam (Figure: 4.4), another is Upload (Figure: 4.5) page.
   WebCam page uses User's webcam after the confirmations from user. It takes real time photo and resizes it in Canvas and gets base64 string for the image pixel values, encrypts it with RSA with server's public key, then creates transaction json structure by following the given rules.
   Upload page takes image using file uploadation in HTML-5 & JS-5 file api, gets metadata and stores it, and resizes it in Canvas and gets base64 string for the image pixel and do the same as the webcam script does.
   Then it sends to server php page using XML-HTTP for further processing. After getting the request server decrypts it using RSA private key, encrypts it with AES and stores it as the transaction json format in unverified pool *i.e.* uvf_txd/ directory.

6. After each 5 minutes the server collects the unverified transactions and publishes to the virtual miners if any of the human miners asked for mining from the mine link included in their profile (Figure: 4.6) page. A human miner gets all the transaction json files downloaded in own computer by PHP and Secure File Transfer Protocol (SFTP). After getting the file it starts verifying the transactions by gossiping with others. After validating the them the miner starts creating the block (Figure: 4.7) until it finishes it or gets another block from the network for verification. As soon as it gets another block for verification it holds own process, and validates it. If it is valid it discards it's own calculation and stores the block in temporary location. If not then it completes creating own block and publish it

to the network for the same validation. Up to 5 minutes they gossip in the fully connected graph of peer-to-peer network and reach to a decision which block to keep according to some criteria. After 5 minutes they reach to a final decision and Then it shares the block to server to store as a temporary block.

7. After 5 minutes the server knows that every miner is having mostly choosen copy of the new block after each miner node reaches to a final decision. It validates all and re-check the criteria and gets to a final decision and publishes it to final one to be added in the chain. It then deletes the transactions which are included in the new block from the transaction pool.

8. The server decryps the blocks to get image data and adds in form of canvases in the news feed in Home Page with downloadable link each. After clicking download it creates digital signature and adds in the image metadata for downloading with it, that marks that the image is created in our system.

9. Whenever a peprson goes for image verification (Figure: 4.8) it checks it in 3 steps that is discussed in the image verification portion in Proposal Chapter.

10. The machine learning step comparison *i.e.* step-3, the contextual analysis is interrupted because of some technical limitations that we are going to cover up very soon.

11. Every server file system related works are done in PHP and computational works are done in JS and communicated between them always whenever required.

## 4.5 JSON Files

### 4.5.1 JSON File Structure

The JSON (JavaScript Object Notation) files are semi-structured alternate database files inspired by JS object structure. The database structure for the character rules in the file are,
File starts and ends with { and }
data attributes and values are set in the way, List of key-pair:
"key" = "value", . . . the value can be Number (0-9*.0-9*), Array ([value, . . . ]), or String ("characters")

```
{ "officer" =
      "name" = "ABC XYZ",
      "salary" = 1000.00,
      "friends" = ["A X", "B Y", "C Z"]
}
```

### 4.5.2 JSON Files Used

The JSON files we are gonna use are,

#### 4.5.2.1 txd_u.json

– This file (filename: Unverified Transactions) consists of a list of un-added transactions that are gonna be added in the new block. The JSON file consists of an array named txd, and the attribute fields are the list of following bunch of information for each transaction
**t_id** Transaction Identifier is created by appending user_id + "_" + photo_id, where user_id is the username, and photo_id is the Algebra(number of photos the user inserted + 1),
**t_ts** Transaction Timestamp is the JS generated date-time-location string in an standard format,
**t_md** Transaction Metadata is the important portion of the metadata in the uploaded digital image file read in hex string of image bytes,
**t_dh** Transaction Data-Hash is the SHA-256 calculated hex hash string,
**t_rd** Transaction Raw-Data is the RSA encrypted hex string of Canvas conversion of image bytes.

#### 4.5.2.2 blk_i.json

– This file (i in filename: i-th Block is the block number) includes the exact following fields,
**b_id** Block Identifier is created by appending "blk_" + blk_id, blk_id is the Algebra(number of blocks in the chain + 1),
**b_ts** Block's Timestamp is the JS generated date-time-location string in an standard format,
**b_ms** Block Miner's Signature, we are currently using creator's user_name here
**b_mr** Block's Merkle Root hash is stored in hex string format here
**b_ph** Block's Previous Hash is the previous block's hash identifier hex string
**b_ch** Block's Current Hash is the calculated hash of this block, the calculation goes as follows, SHA-256 of (concatinated hex(b_id, b_ts, b_ms), b_mr, b_ph),
**b_dt** Block Data is the list of all the validated transactions' fields that are included in the block;

#### 4.5.2.3 chain.json

– This file is the pointer or the book keeper for easy access of the collection of the blocks created. The fields are the list of blocks' information and for each in each entry,
**b_mn** Block Miner's Signature (discussed earlier)
**b_fn** Block's File-Name is created by appending "blk_" + i from b_id + ".json" which is stored in the same resource directory
**c_ch** Chain's Current Hash is the last block's current hash (b_ch) to keep an eye on the last block

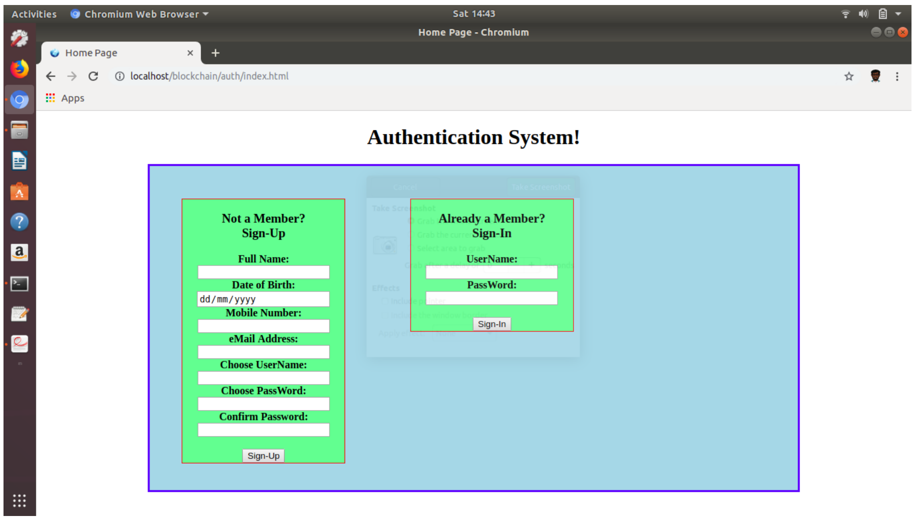
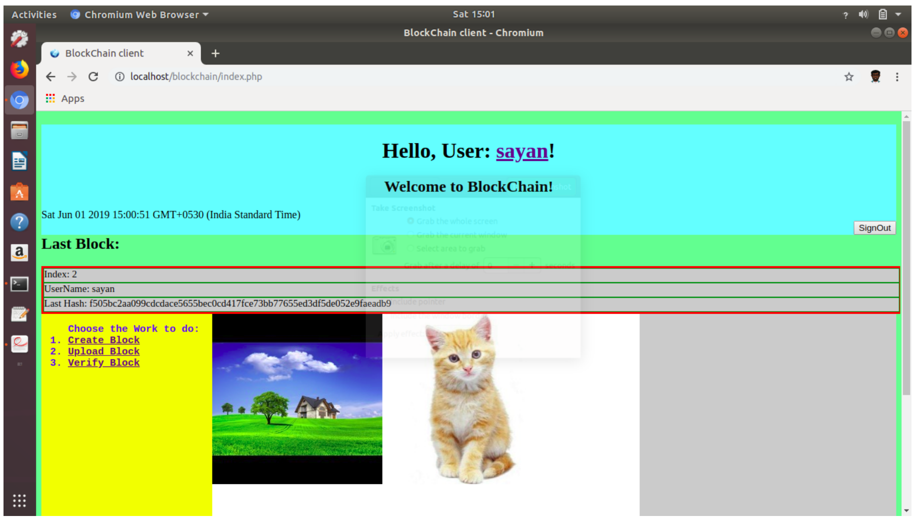


Figure 4.1: Authentication Page



Hello, User: sayan!

Welcome to BlockChain!

Sat Jun 01 2019 15:00:51 GMT+0530 (India Standard Time)

Last Block:

Index: 2
UserName: sayan
Last Hash: f505bc2aa099cdcdace5655bec0cd417fce73bb77655ed3df5de052e9faeadb9

Choose the Work to do:
1. Create Block
2. Upload Block
3. Verify Block

Figure 4.2: Home Page

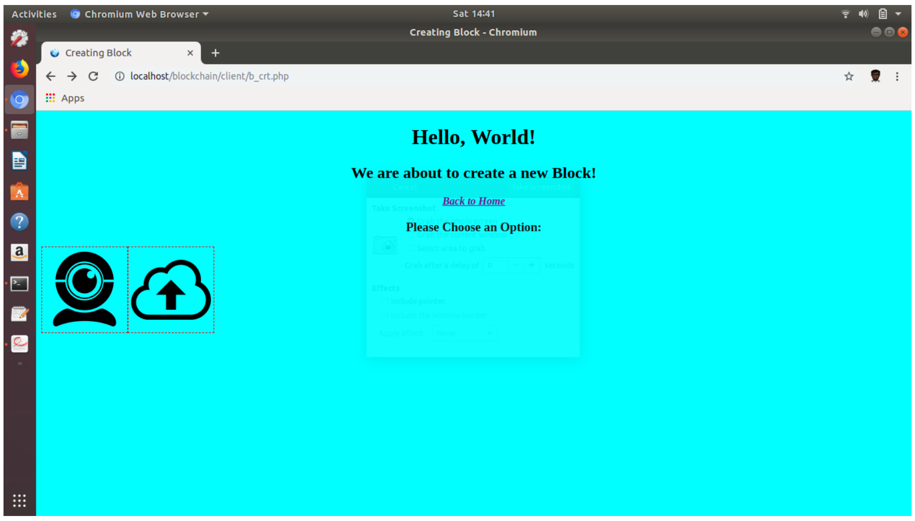


Figure 4.3: Choose How to Insert Image

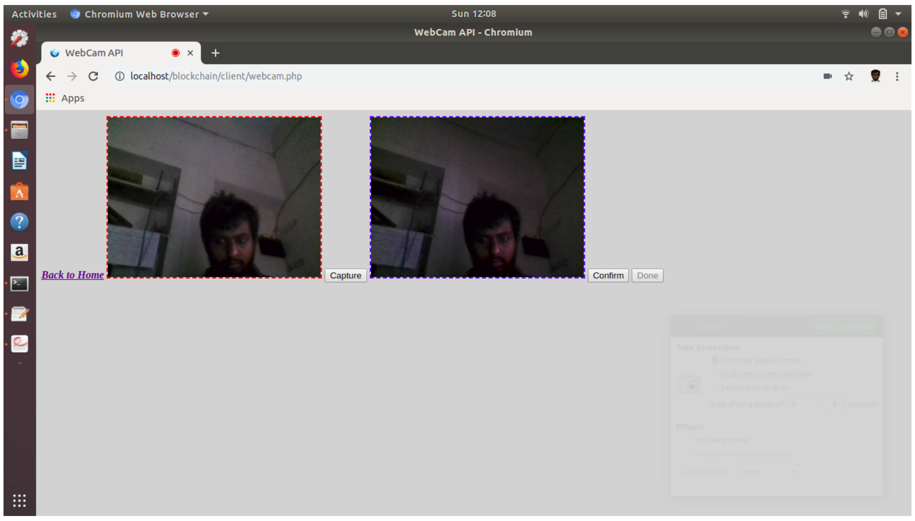


Figure 4.4: WebCam

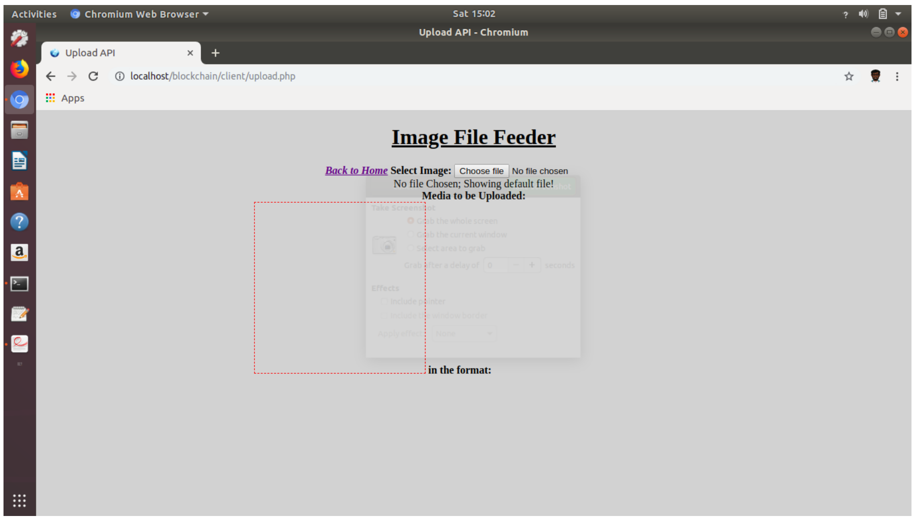


**Image File Feeder**

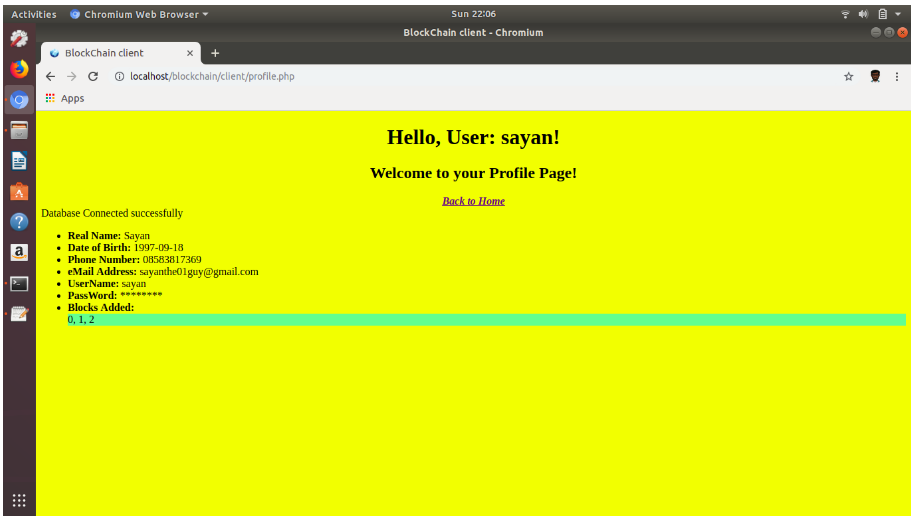
*Back to Home* **Select Image:** Choose file No file chosen

No file Chosen; Showing default file!

**Media to be Uploaded:**

**in the format:**

Figure 4.5: Upload



**Hello, User: sayan!**

**Welcome to your Profile Page!**

*Back to Home*

Database Connected successfully

- **Real Name:** Sayan
- **Date of Birth:** 1997-09-18
- **Phone Number:** 08583817369
- **eMail Address:** sayanthe01guy@gmail.com
- **UserName:** sayan
- **PassWord:** ********
- **Blocks Added:**
  0, 1, 2

Figure 4.6: Profile

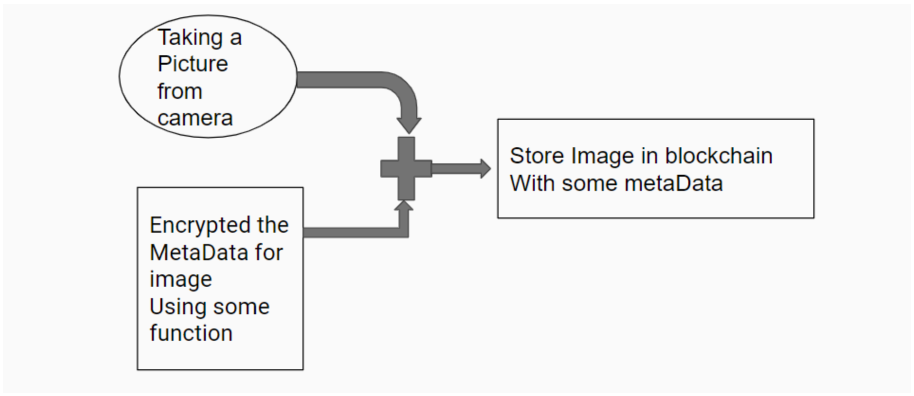

Taking a Picture from camera
Encrypted the MetaData for image Using some function
Store Image in blockchain With some metaData

Figure 4.7: Image Insertion

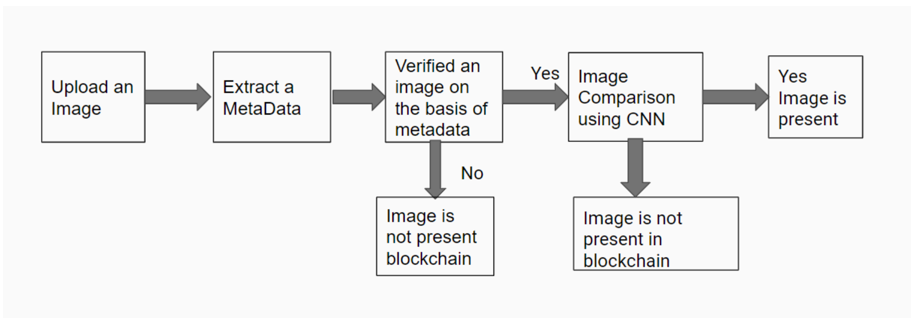

Upload an Image
Extract a MetaData
Verified an image on the basis of metadata
Yes
Image Comparison using CNN
Yes Image is present
No
Image is not present blockchain
Image is not present in blockchain

Figure 4.8: Image Verification

# Chapter 5

# Conclusion and Further Work

## 5.1 Conclusion

This synopsis provides a detailed description of an Practical implementation of Online Biding system which provides Secure Key Exchange and agreement. We have implemented system for

- Capturing or uploading image;
- Show status in Home and Profile Page;
- Processing the image;
- Signing the image
- Create transactions
- Create block in server
- Adding to chain

## 5.2 Further Work

The next targets are to create a system that will standalone perform the following tasks

- making a peer-to-peer network for android that will perform the following tasks
- connecting to Online Verifier service
- sending and Receive messages containing data between peers
- improving Verification Service
- connecting in mobile application
- estimating risks and reconfigure concensus protocols

# Appendices

# Appendix A

# Cryptographic Technologies

## A.1 SHA-256

### A.1.1 Introduction

SHA-256 (Secure Hash Algorithm) comes under 6 member group of SHA-2. SHA-2 (Secure Hash Algorithm 2) is a set of cryptographic hash functions designed by the United States National Security Agency (NSA). They are built using the Merkle-Damgård structure, from a one-way compression function itself built using the Davies-Meyer structure from a (classified) specialized block cipher.

SHA-2 includes significant changes from its predecessor, SHA-1. The SHA-2 family consists of six hash functions with digests (hash values) that are 224, 256, 384 or 512 bits: SHA-224, SHA-256, SHA-384, SHA-512, SHA-512/224, SHA-512/256.

SHA-256 and SHA-512 are novel hash functions computed with 32-bit and 64-bit words, respectively. They use different shift amounts and additive constants, but their structures are otherwise virtually identical, differing only in the number of rounds. SHA-224 and SHA-384 are truncated versions of SHA-256 and SHA-512 respectively, computed with different initial values. SHA-512/224 and SHA-512/256 are also truncated versions of SHA-512, but the initial values are generated using the method described in Federal Information Processing Standards (FIPS) PUB 180-4. SHA-2 was published in 2001 by the National Institute of Standards and Technology (NIST) a U.S. federal standard (FIPS). The SHA-2 family of algorithms are patented in US patent 6829355. The United States has released the patent under a royalty-free license.

### A.1.2 Property

SHA-256 is one of the successor hash functions to SHA-1 (collectively referred to as SHA-2), and is one of the strongest hash functions available. SHA-256 is not much more complex to code than SHA-1, and has not yet been compromised in any way. The 256-bit key makes it a good partner-function for AES. It is defined in the NIST (National Institute of Standards and Technology) standard 'FIPS 180-4'. NIST also provide a number of test vectors to verify correctness of implementation.

SHA-256 takes message sige of bit-length a multiple of 512 bits and returns a hash of 256 bits. The message have to be pre-processed before sending it to main hash function. The hash function's job is to take 512 bit input and return 256 bit output.

The message is divided into n blocks of 512 bits. No matter what message length is a 1 and enough 0's are appended to it along with 64 bit message length in the last block. Then one by one the n or n+1 blocks are sent for hashing or changing the default 8 digest values (each words are of 32 bits).

### A.1.3 Algorithm

The pseudocode for the implementation in JS is as follows,

Note 1: All variables are 32 bit unsigned integers and addition is calculated modulo 232

Note 2: For each round, there is one round constant k[i] and one entry in the message schedule array w[i], $0 \le i \le 63$

Note 3: The compression function uses 8 working variables, a through h

Note 4: Big-endian convention is used when expressing the constants in this pseudocode, and when parsing message block data from bytes to words, for example, the first word of the input message "abc" after padding is 0x61626380

#### A.1.3.1 Initialize hash values

(first 32 bits of the fractional parts of the square roots of the first 8 primes 2..19):
*h0 := 0x6a09e667*
*h1 := 0xbb67ae85*
*h2 := 0x3c6ef372*
*h3 := 0xa54ff53a*
*h4 := 0x510e527f*
*h5 := 0x9b05688c*
*h6 := 0x1f83d9ab*
*h7 := 0x5be0cd19*

#### A.1.3.2 Initialize array of round constants

(first 32 bits of the fractional parts of the cube roots of the first 64 primes 2..311): *k[0..63] := {*
*0x428a2f98, 0x71374491, 0xb5c0fbcf, 0xe9b5dba5, 0x3956c25b, 0x59f111f1, 0x923f82a4, 0xab1c5ed5,*
*0xd807aa98, 0x12835b01, 0x243185be, 0x550c7dc3, 0x72be5d74, 0x80deb1fe, 0x9bdc06a7, 0xc19bf174,*
*0xe49b69c1, 0xefbe4786, 0x0fc19dc6, 0x240ca1cc, 0x2de92c6f, 0x4a7484aa, 0x5cb0a9dc, 0x76f988da,*
*0x983e5152, 0xa831c66d, 0xb00327c8, 0xbf597fc7, 0xc6e00bf3, 0xd5a79147, 0x06ca6351, 0x14292967,*
*0x27b70a85, 0x2e1b2138, 0x4d2c6dfc, 0x53380d13, 0x650a7354, 0x766a0abb, 0x81c2c92e, 0x92722c85,*
*0xa2bfe8a1, 0xa81a664b, 0xc24b8b70, 0xc76c51a3, 0xd192e819, 0xd6990624, 0xf40e3585, 0x106aa070,*
*0x19a4c116, 0x1e376c08, 0x2748774c, 0x34b0bcb5, 0x391c0cb3, 0x4ed8aa4a, 0x5b9cca4f, 0x682e6ff3,*
*0x748f82ee, 0x78a5636f, 0x84c87814, 0x8cc70208, 0x90befffa, 0xa4506ceb, 0xbef9a3f7, 0xc67178f2*
*}*

#### A.1.3.3 Pre-processing (Padding)

begin with the original message of length L bits
append a single '1' bit

append K '0' bits, where K is the minimum number ≥ 0 such that L + 1 + K + 64 is a multiple of 512

$$(L+1+K+64) = 512\times$$

append L as a 64-bit big-endian integer, making the total post-processed length a multiple of 512 bits

#### A.1.3.4 Process the message in successive 512-bit chunks

```
break message into 512-bit chunks
for each chunk
    create a 64-entry message schedule array w[0..63] of 32-bit words
```

(The initial values in w[0..63] don’t matter, so many implementations zero them here)
copy chunk into first 16 words w[0..15] of the message schedule array
Extend the first 16 words into the remaining 48 words w[16..63] of the message schedule array:

```
for i from 16 to 63
    s0 := (w[i-15] rightrotate 7) xor (w[i-15] rightrotate 18) xor (w[i-15] rightshift 3)
    s1 := (w[i- 2] rightrotate 17) xor (w[i- 2] rightrotate 19) xor (w[i- 2] rightshift 10)
    w[i] := w[i-16] + s0 + w[i-7] + s1
Initialize working variables to current hash value:
a := h0
b := h1
c := h2
d := h3
e := h4
f := h5
g := h6
h := h7
```

### A.1.3.5 Compression function main loop:

```
for i from 0 to 63
    S1 (Sigmoid-1) := (e rightrotate 6) xor (e rightrotate 11) xor (e rightrotate 25)
    ch (Choice) := (e and f) xor ((not e) and g)
    temp1 (Sum) := h + S1 + ch + k[i] + w[i]
    S0 (Sigmoid-0) := (a rightrotate 2) xor (a rightrotate 13) xor (a rightrotate 22)
    maj (Major) := (a and b) xor (a and c) xor (b and c)
    temp2 (Sum) := S0 + maj
    Update the middle-man values
    h := g
    g := f
    f := e
    e := d + temp1
    d := c
    c := b
    b := a
    a := temp1 + temp2
```

#### A.1.3.6 Add the compressed chunk to the current hash value

```
    h0 := h0 + a
    h1 := h1 + b
    h2 := h2 + c
    h3 := h3 + d
    h4 := h4 + e
    h5 := h5 + f
    h6 := h6 + g
    h7 := h7 + h
Get another chunk and calculate again
```

#### A.1.3.7 Produce the final hash value (big-endian)

*digest := hash := h0 append h1 append h2 append h3 append h4 append h5 append h6 append h7*

The computation of the ch and maj values can be optimized the same way as described for SHA-1.

### A.1.4 Application

The SHA-2 hash function is implemented in some widely used security applications and protocols, including TLS and SSL, PGP, SSH, S/MIME, and IPsec.

SHA-256 partakes in the process of authenticating Debian software packagesand in the DKIM message signing standard; SHA-512 is part of a system to authenticate archival video from the International Criminal Tribunal of the Rwandan genocide. SHA-256 and SHA-512 are proposed for use in DNSSEC. Unix and Linux vendors are moving to using 256- and 512-bit SHA-2 for secure password hashing.

Several cryptocurrencies like Bitcoin use SHA-256 for verifying transactions and calculating proof of work or proof of stake. The rise of ASIC SHA-2 accelerator chips has led to the use of scrypt-based proof-of-work schemes.

### A.1.5 Citing

A good discussion is given here [2]. And a good implementation is found here [24]

## A.2 RSA

### A.2.1 Introduction

RSA (inventors: Rivest-Shamir-Adleman) belongs to PKCS (public key cryptosystem) where a pair of keys are required, one to encrypt another to decrypt the encrypted one. Encryption means changing a message to another using such a popular rule using some secret value (that no one but the person who is encrypting knows) no one can understand. In case of PKCS it is asymmetric cryptographic algorithm means one key is kept secret (private key) and another one is sent to everyone (public key) for decryption. The keys are long enough to make the adversary break both the keys.

### A.2.2 Property

The security of RSA depends on the strengths of two separate functions. The RSA cryptosystem is most popular public-key cryptosystem strength of which is based on the practical difficulty of factoring the very large numbers.

- **Encryption Function:** It is considered as a one-way function of converting plaintext into ciphertext and it can be reversed only with the knowledge of private key d.
- **Key Generation:** The difficulty of determining a private key from an RSA public key is equivalent to factoring the modulus n. An attacker thus cannot use knowledge of an RSA public key to determine an RSA private key unless he can factor n. It is also a one way function, going from p & q values to modulus n is easy but reverse is not possible.

If either of these two functions are proved non one-way, then RSA will be broken. In fact, if a technique for factoring efficiently is developed then RSA will no longer be safe.

The strength of RSA encryption drastically goes down against attacks if the number p and q are not large primes and (or) chosen public key e is a small number.

### A.2.3 Algorithm

1. Choose two different large random prime numbers $p$ and $q$
2. Calculate Maximum range of considered numbers:

$$n = p \times q$$

3. Calculate Euler's totient or Carmichael's totient (number of $+(ve)$ coprimes that are less than the given numbers):

$$\phi(n) = (p-1) \times (q-1)$$

4. Choose a random number in the range $[2, \phi(n))$ that is co-prime with both $n$ and $\phi(n)$; two numbers $a$ and $b$ are are co-prime means they have no common factor *i.e.*

$$gcd(a,b) = 1$$

5. Compute $d$ to satisfy the congruence relation

$$d \times e \equiv 1 \mod \phi(n)$$

or

$$d \times e = 1 + k \times \phi(n)$$

or

$$d = \frac{1 + k \times \phi(n)}{e}$$

6. Private Key $\{d, n\}$ is kept secret to decrypt the encrypted message after receiving from others and Public Key $\{e, n\}$ is sent to everyone in the network to encrypt the encrypted message sent to the person who shared the public key.

7. Sender side encryption:

$$c = m^e \mod n$$

8. Reciever side decryption:

$$m = c^d \mod n$$

9. Afterwards they changed $\phi(n)$ to $\lambda(n)$ where,

$$\lambda(n) = lcm(p-1, q-1)$$

### A.2.4 Application

Suppose Alice uses Bob's public key to send him an encrypted message. In the message, she can claim to be Alice but Bob has no way of verifying that the message was actually from Alice since anyone can use Bob's public key to send him encrypted messages. So, in order to verify the origin of a message, RSA can also be used to sign a message.

Suppose Alice wishes to send a signed message to Bob. She produces a hash value of the message, raises it to the power of d mod n (just like when decrypting a message), and attaches it as a "("signature) to the message. When Bob receives the signed message, he raises the signature to the power of e mod n (just like encrypting a message), and compares the resulting hash value with the message's actual hash value. If the two agree, he knows that the author of the message was in possession of Alice's secret key, and that the message has not been tampered with since.

Note that secure padding schemes such as RSA-PSS are as essential for the security of message signing as they are for message encryption, and that the same key should never be used for both encryption and signing purposes.

### A.2.5 Citing

There is a nice description in wikipedia [9] and [1]. The real paper [23] and the paper [18] helps making the code ourselves.

## A.3 AES

### A.3.1 Introduction

The AES (Advanced Encryption Standard), also known by its original name Rijndael is a specification for the encryption of electronic data established by the U.S. National Institute of Standards and Technology (NIST) in 2001. AES is a subset of the Rijndael block cipher developed by two Belgian cryptographers, Vincent Rijmen and Joan Daemen, who submitted a proposal to NIST during the AES selection process. Rijndael is a family of ciphers with different key and block sizes.

### A.3.2 Properties

It is one of the symmetric key block cipher technique *i.e.* same key is used to encrypt an decrypt. It takes 128 bit message block at a time and 128 bit (or 192 bit or 256 bit) key block.

AES is based on a design principle known as a substitution-permutation network, and is efficient in both software and hardware. Unlike its predecessor DES, AES does not use a Feistel network. AES operates on a 4 × 4 column-major order array of bytes (1 byte = 8 bits), termed the state. Most AES calculations are done in a particular finite field.

For instance, if there are 16 bytes these bytes are represented as this two-dimensional array (state matrix)

$$\begin{bmatrix} b_0 & b_4 & b_8 & b_C \\ b_1 & b_5 & b_9 & b_D \\ b_2 & b_6 & b_A & b_E \\ b_3 & b_7 & b_B & b_F \end{bmatrix}$$

The key size used for an AES cipher specifies the number of transformation rounds that convert the input, called the plaintext, into the final output, called the ciphertext. The number of rounds are as follows:

- 10 rounds for 128-bit keys.
- 12 rounds for 192-bit keys.
- 14 rounds for 256-bit keys.

Each round consists of several processing steps, including one that depends on the encryption key itself. A set of reverse rounds are applied to transform ciphertext back into the original plaintext using the same encryption key.

All the calculation of AES can be calculated using either Galois Field $GF(2^8)$ and Euclidean algebra (takes more time but less space) or pre-calculated table driven way (takes less time but more space).

### A.3.3 Algorithm

#### A.3.3.1 Message Division

Divide the message by 128 bits chunks of state matrices

#### A.3.3.2 Key Expansion

Round keys are derived from the cipher key using Rijndael's key schedule. AES requires a separate 128-bit round key block for each round plus one more.

#### A.3.3.3 Initial Step

For each message block:

```
    $K_{ini} \leftarrow genRoundKey(0, K_{main})$
    $M_0 \leftarrow addRoundKey(M_{main}, K_{ini})$
```

#### A.3.3.4 Rounds

```
    for each round $0 \leq i < (totalRounds - 1)$ :
        $M_i \leftarrow subBytes(M_i)$
        $M_i \leftarrow shiftRows(M_i)$
        $M_i \leftarrow mixColumns(M_i)$
        $K_i \leftarrow genRoundKey(i, K_{main})$
        $M_i \leftarrow addRoundKey(M_i, K_i)$
```

#### A.3.3.5 Final Step

It is $i = (totalRounds - 1)^{th}$ round;
No column mixing here or it will get back close to original message block.

$$M_i \leftarrow subBytes(M_i)$$
$$M_i \leftarrow shiftRows(M_i)$$
$$K_{fin} \leftarrow genRoundKey(i, K_{main})$$
$$S_{fin} \leftarrow addRoundKey(M_i, K_{fin})$$

#### A.3.3.6 Used Functions

**subBytes(state) :: state**
In the SubBytes step, each byte in the state array is replaced with a SubByte $S(a_{i,j})$ using an 8-bit substitution box. This operation provides the *non-linearity* in the cipher. The S-box used is derived from the multiplicative inverse over $GF(2^8)$, known to have good non-linearity properties. To avoid attacks based on simple algebraic properties, the S-box is constructed by combining the inverse function with an invertible affine transformation. The S-box is also chosen to avoid any fixed points (and so is a derangement), i.e. $S(b_{i,j}) \neq b_{i,j}$ and also any opposite fixed points, i.e. $S(b_{i,j}) \oplus b_{i,j} \neq FF_{16}$. While performing the decryption, the InvSubBytes step (the inverse of SubBytes) is used, which requires first taking the inverse of the affine transformation and then finding the multiplicative inverse. Returns:

$$\begin{bmatrix} S(b_0) & S(b_4) & S(b_8) & S(b_C) \\ S(b_1) & S(b_5) & S(b_9) & S(b_D) \\ S(b_2) & S(b_6) & S(b_A) & S(b_E) \\ S(b_3) & S(b_7) & S(b_B) & S(b_F) \end{bmatrix}$$

**shiftRows(state) :: state**
The ShiftRows step operates on the rows of the state; it cyclically left shifts the bytes in each row by a certain offset which is equal to Zero initiated row number *i.e.* in the row number set {0, 1, 2, 3}. It provides *shuffling* to avoid the columns being encrypted independently. Returns:

$$\begin{bmatrix} b_0 & b_4 & b_8 & b_C \\ b_5 & b_9 & b_D & b_1 \\ b_A & b_E & b_2 & b_6 \\ b_F & b_3 & b_7 & b_B \end{bmatrix}$$

**mixColumns(state) :: state**
In the MixColumns step, the four bytes of each column of the state are combined using an invertible linear transformation. The MixColumns function takes four bytes as input and outputs

four bytes, where each input byte affects all four output bytes. Together with ShiftRows, MixColumns provides diffusion in the cipher. During this operation, each column is transformed using a fixed matrix (matrix left-multiplied by column gives new value of column in the state): Returns:

$$\begin{bmatrix} FF_{16} & FF_{16} & FF_{16} & FF_{16} \\ FF_{16} & FF_{16} & FF_{16} & FF_{16} \\ FF_{16} & FF_{16} & FF_{16} & FF_{16} \\ FF_{16} & FF_{16} & FF_{16} & FF_{16} \end{bmatrix} \oplus \begin{bmatrix} b_0 & b_4 & b_8 & b_C \\ b_5 & b_9 & b_D & b_1 \\ b_A & b_E & b_2 & b_6 \\ b_F & b_3 & b_7 & b_B \end{bmatrix} \times \begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix}$$

**genRoundKey(round, state) :: state**
AES (Rijndael) uses a key schedule to expand a short key into a number of separate round keys. This is known as the Rijndael key schedule. The three AES variants have a different number of rounds. Each variant requires a separate 128-bit round key for each round plus one more. The key schedule produces the needed round keys from the initial key.

Each round has a round constant $rcon_i$ which is calculated by,

$$rcon_i = \begin{bmatrix} rc_i & 00_{16} & 00_{16} & 00_{16} \end{bmatrix}$$

where,

$$rc_i = \begin{cases} 1, & \text{if}(i = 1) \\ 2 \times rc_{i-1}, & \text{if}(i > 1)\text{and}(rc_{i-1} < 80_{16}) \\ (2 \times rc_{i-1}) \oplus 1B_{16}, & \text{if}(i > 1)\text{and}(rc_{i-1} > 80_{16}) \end{cases}$$

Two main functions used are,

$$RotWord(\begin{bmatrix} b_0 & b_1 & b_2 & b_3 \end{bmatrix}) = \begin{bmatrix} b_1 & b_2 & b_3 & b_0 \end{bmatrix}$$

and

$$SubWord(\begin{bmatrix} b_0 & b_1 & b_2 & b_3 \end{bmatrix}) = \begin{bmatrix} S(b_1) & S(b_2) & S(b_3) & S(b_0) \end{bmatrix}$$

For each of $4 \times r + 1$ rounds,

$$W_i = \begin{cases} K_i, & \text{if}(i < n) \\ W_{i-n} \oplus SubWord(RotWord(W_{i-1})) \oplus rcon_{i/n}, & \text{if}(i \geq n) and \text{and}(i \equiv 0 \mod n) \\ W_{i-n} \oplus SubWord(W_{i-1}), & \text{if}(i \geq n), (n > 6) and \text{and}(i \equiv 4 \mod n) \\ W_{i-n} \oplus W_{i-1}, & \text{otherwise} \end{cases}$$

**addRoundKey(state, state) :: state**
In the AddRoundKey step, the subkey is combined with the state. For each round, a subkey is

derived from the main key using Rijndael's key schedule; each subkey is the same size as the state. The subkey is added by combining each byte of the state with the corresponding byte of the subkey using bitwise XOR. Returns:

$$\begin{bmatrix} m_0 & m_4 & m_8 & m_C \\ m_5 & m_9 & m_D & m_1 \\ m_A & m_E & m_2 & m_6 \\ m_F & m_3 & m_7 & m_B \end{bmatrix} \oplus \begin{bmatrix} k_0 & k_4 & k_8 & k_C \\ k_5 & k_9 & k_D & k_1 \\ k_A & k_E & k_2 & k_6 \\ k_F & k_3 & k_7 & k_B \end{bmatrix}$$

#### A.3.3.7 Message Generation

After calculating the cipher-text the bytes are converted to consecutive readable strings of characters.

#### A.3.3.8 Decryption

The decryption is the reverse processing with the ciphertext state blocks with the same key. The reverse operations for each functions are,

- $rev(subBytes(state)) \implies invSubBytes(state)$
- $rev(shiftRows(state)) \implies invShiftRows(state) or shiftRows(state, right)$
- $rev(mixColumns(state))$'s mixing matrix changes to it's inverse matrix
- $rev(genRoundKey(round, state)) \implies genRoundKey(totalRound - round, state)$
- $rev(addRoundKey(state, state))$remains same as if$\{m \oplus k = c\}$then$\{c \oplus k = m\}$

### A.3.4 Application

AES is implemented in secure file transfer protocols like FTPS, HTTPS, SFTP, AS2, WebDAVS, and OFTP.

### A.3.5 Citing

Well known real paper [10] describes a lot along with we took help from wikipedia [4].

# Appendix B

# Machine Learning Technologies

## B.1 Linear Regression

### B.1.1 Introduction

Linear regression is a statistical approach for modelling relationship between a dependent variable with a given set of independent variables.

In order to provide a basic understanding of linear regression, we start with the most basic version of linear regression, i.e. Simple linear regression.

#### B.1.1.1 Simple Linear Regression

Simple linear regression is an approach for predicting a **response** using a **single feature**.

It is assumed that the two variables are linearly related. Hence, we try to find a linear function that predicts the response value(y) as accurately as possible as a function of the feature or independent variable(x).

Now, the task is to find a line which fits best in above scatter plot so that we can predict the response for any new feature values. (i.e a value of x not present in dataset)

This line is called **Regression line**. This equation of regression line is represented as:

$$h(x_i) = \beta_0 + \beta_1 x_i; \tag{B.1}$$

Here, • represents the predicted response value for i-th observation.

• b_0 and b_1 are regression coefficients and represent y-intercept and slope of regression line respectively.

To create our model, we must "learn" or estimate the values of regression coefficients b_0 and b_1. And once we've estimated these coefficients, we can use the model to predict responses! For this we use Least Squared Technique.

### B.1.1.2 Least Square Technique

.

$$y_i = \beta_0 + \beta_1 \times x_i + \epsilon_i = h(x_{i+}) + \epsilon_i \implies \epsilon_i = y_i - h(x_i); \tag{B.2}$$

Here, e_i is residual error in ith observation. We try to minimize the total residual error. We define the squared error or cost function, J as:

$$J(\beta_0, \beta_1) = \frac{1}{2n}\sum_{i=1}^{n} \epsilon_i^2; \tag{B.3}$$

and our task is to find the value of b_0 and b_1 for which J(b_0, b_1) is minimum! Without going into the mathematical details, we present the result here:

$$\beta_1 = \frac{SS_{xy}}{SS_{xx}}; \tag{B.4}$$

$$\beta_0 = \bar{y} - \beta_1\bar{x}; \tag{B.5}$$

where SS_xy is the sum of cross-deviations of y and x:

$$SS_{xy} = \sum_{i=1}^{n}(x - \bar{x})(y - \bar{y}) = \sum_{i=1}^{n} y_i x_i - n\bar{x}\bar{y}; \tag{B.6}$$

and SS_xx is the sum of squared deviations of x:

$$SS_{xx} = \sum_{i=0}^{n}(x_i - \bar{x})^2 = \sum_{i=0}^{n} x_i^2 - n(\bar{x})^2; \tag{B.7}$$

### B.1.1.3 Multiple Linaer Regression

Multiple linear regression attempts to model the relationship between two or more features and a response by fitting a linear equation to observed data. Clearly, it is nothing but an extension of Simple linear regression. Consider a dataset with p features(or independent variables) and one response(or dependent variable). Also, the dataset contains n rows/observations.

X(features matrix) = a matrix of size nXp where x_i denotes the values of jth feautres for ith observation.

So,

$$\begin{pmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,n} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m,1} & x_{m,2} & \cdots & x_{m,n} \end{pmatrix}$$

and

y = $\begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots y_n \end{pmatrix}$

The regression line for p is represented as:

$$h(x_i) = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + ... + \beta_p x_{ip}; \tag{B.8}$$

where h(x_i) is predicted response value for ith observation and b_0,b_1, ...,b_p are the regression coefficients. Also, we can write:

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + ... + \beta_p x_{ip} + \epsilon_i; \tag{B.9}$$

or

$$y_i = h(x_i) + \epsilon_i \implies \epsilon_i = y_i - h(x_i); \tag{B.10}$$

We can generalize our linear model a little bit more by representing feature matrix X as:

$\begin{pmatrix} 1 & x_{1,1} & x_{1,2} & \cdots & x_{1,p} \\ 1 & x_{2,1} & x_{2,2} & \cdots & x_{2,p} \\ \vdots & \vdots & \ddots & \vdots & \\ 1 & x_{n,1} & x_{n,2} & \cdots x a_{n,p} & \end{pmatrix}$

So now, the linear model can be expressed in terms of matrices as:

$$y = X\beta + \epsilon$$

where,

$$\beta = \begin{bmatrix} \beta_0 \\ \beta_1 \\ . \\ . \\ \beta_p \end{bmatrix}$$

and

$$\epsilon = \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ . \\ . \\ \varepsilon_n \end{bmatrix}$$

Now, we determine estimate of b, i.e. b' using Least Squares method.

As already explained, Least Squares method tends to determine b for which total residual error is minimized.

We present the result directly here:

$$\hat{\beta} = (XX')^{-1}X'y$$

where 'represents the transpose of the matrix while -1 represents the matrix inverse.

Knowing the least square estimates, b', the multiple linear regression model can now be estimated as:

$$\hat{y} = X\hat{\beta}$$

where $\hat{y}$ is estimated response vector.

#### B.1.1.4 Assumptions

the basic assumptions that a linear regression model makes regarding a dataset on which it is applied:

**Linear relationship:** Relationship between response and feature variables should be linear. The linearity assumption can be tested using scatter plots. As shown below, 1st figure represents linearly related variables where as variables in 2nd and 3rd figure are most likely non-linear. So, 1st figure will give better predictions using linear regression.

**Little or no multi-collinearity:** It is assumed that there is little or no multicollinearity in the data. Multicollinearity occurs when the features (or independent variables) are not independent from each other.

**Little or no auto-correlation:** Another assumption is that there is little or no autocorrelation in the data. Autocorrelation occurs when the residual errors are not independent from each other. You can refer here for more insight into this topic.

**Homoscedasticity:** Homoscedasticity describes a situation in which the error term (that is, the "noise" or random disturbance in the relationship between the independent variables and the dependent variable) is the same across all values of the independent variables. As shown below, figure 1 has homoscedasticity while figure 2 has heteroscedasticity.

### B.1.2 Application

Some of the application of linear regression model are :

1. Trend lines: A trend line represents the variation in some quantitative data with passage of time (like GDP, oil prices, etc.). These trends usually follow a linear relationship. Hence, linear regression can be applied to predict future values. However, this method suffers from a lack of scientific validity in cases where other potential changes can affect the data.

2. Economics: Linear regression is the predominant empirical tool in economics. For example, it is used to predict consumption spending, fixed investment spending, inventory investment, purchases of a country's exports, spending on imports, the demand to hold liquid assets, labor demand, and labor supply.

3. Finance: Capital price asset model uses linear regression to analyze and quantify the systematic risks of an investment.

4. Biology: Linear regression is used to model causal relationships between parameters in biological systems.

### B.1.3 Citing

A well known implementation is given in Geeks-for-Geeks and a well discussion is done at Linear regression Wiki.

For our project we will use simple linear regression (with features the difference in the number of bits).

# Bibliography

[1] Rsa algorithm in cryptography. https://www.geeksforgeeks.org/rsa-algorithm-cryptography/. Accessed on 2019-03-15.

[2] 62.210.124.242. Sha-2. https://en.wikipedia.org/wiki/SHA-2, Mar 2005. Accessed on 2019-02-01.

[3] Goran Almgren, Mats Stengard, and Hans Almgren. Enigio ab. https://enigio.com/.

[4] The Anome. Advanced encryption standard. https://en.wikipedia.org/wiki/Advanced_Encryption_Standard, Nov 2001. Accessed on 2019-03-15.

[5] Srikar Appalaraju and Vineet Chaoji. Image similarity using deep cnn and curriculum learning.

[6] Adam Hemlin Billström and Fabian Huss. Video integrity through blockchain technology. 2017.

[7] Michaeel J. Casey and Paul Vigna. *The Truth Machine: The Blockchain and the Future of Everything*. HarperCollins, 2018.

[8] J. Clark and A. Essex. Carbon dating commitments with bitcoin. *Springer Berlin Heidelberg*, 2012.

[9] Odhan Cohen. Rsa algorithm. https://simple.wikipedia.org/wiki/RSA_algorithm, Jun 2006. Accessed on 2019-03-15.

[10] Joan Daemen and Vincent Rijmen. Aes proposal: Rijndael. 2001.

[11] Dimitri de Jonghe and Trent McConaghy. Spool. https://github.com/ascribe/spool.

[12] I. Echizen, S. Singh, T. Yamada, K. Tanimoto, S. Tezuka, and B. Huet. Integrity verification system for video content by using digital watermarking. *International Conference on Service Systems and Service Management*, 2006.

[13] B. Gipp, J. Kosti, and C. Breitinger. Securing video integrity using decentralized trusted timestamping on the bitcoin blockchain. *MCIS 2016 Proceedings.*, 2016.

[14] B. Gipp, N. Meuschke, and A. Gernandt. Decentralized trusted timestamping using the crypto currency bitcoin. *Clinical Orthopaedics and Related Research*, 2015.

[15] S. Higgins. Blockchain startup raises $2 million for intellectual property solution. .

[16] Leslie Lamport. Paxos made simple. 2001.

[17] David Mazieres. Paxos made practical.

[18] Evgeny Milanov. The rsa algorithm. 2009.

[19] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. 2008.

[20] Nexsan. Assureon archive storage data sheet. https://www.nexsan.com/wp-content/uploads/datasheets/Assureon-DS-v2.pdf, Nov 2016. Accessed: 2019-03-14.

[21] Diego Ongaro and John Ousterhout. In search of an understandable consensus algorithm (extended version). 2014.

[22] R. Poisel and S. Tjoa. Forensics investigations of multimedia data: A review of the state-of-the-art. *Sixth International Conference on IT Security Incident Management and IT Forensics*, 2011.

[23] R.L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 1978.

[24] Chris Veness. Sha-256 cryptographic hash algorithm. https://www.movable-type.co.uk/scripts/sha256.html, Jan 2018. Accessed on 2019-05-31.

[25] Wikipedia. Blockchain. https://en.wikipedia.org/wiki/Blockchain, Oct 2014. Accessed on 2018-12-01.