Cryptography (BITS F463)
Programming Assignment 2

March 31st, 2021

1 Introduction

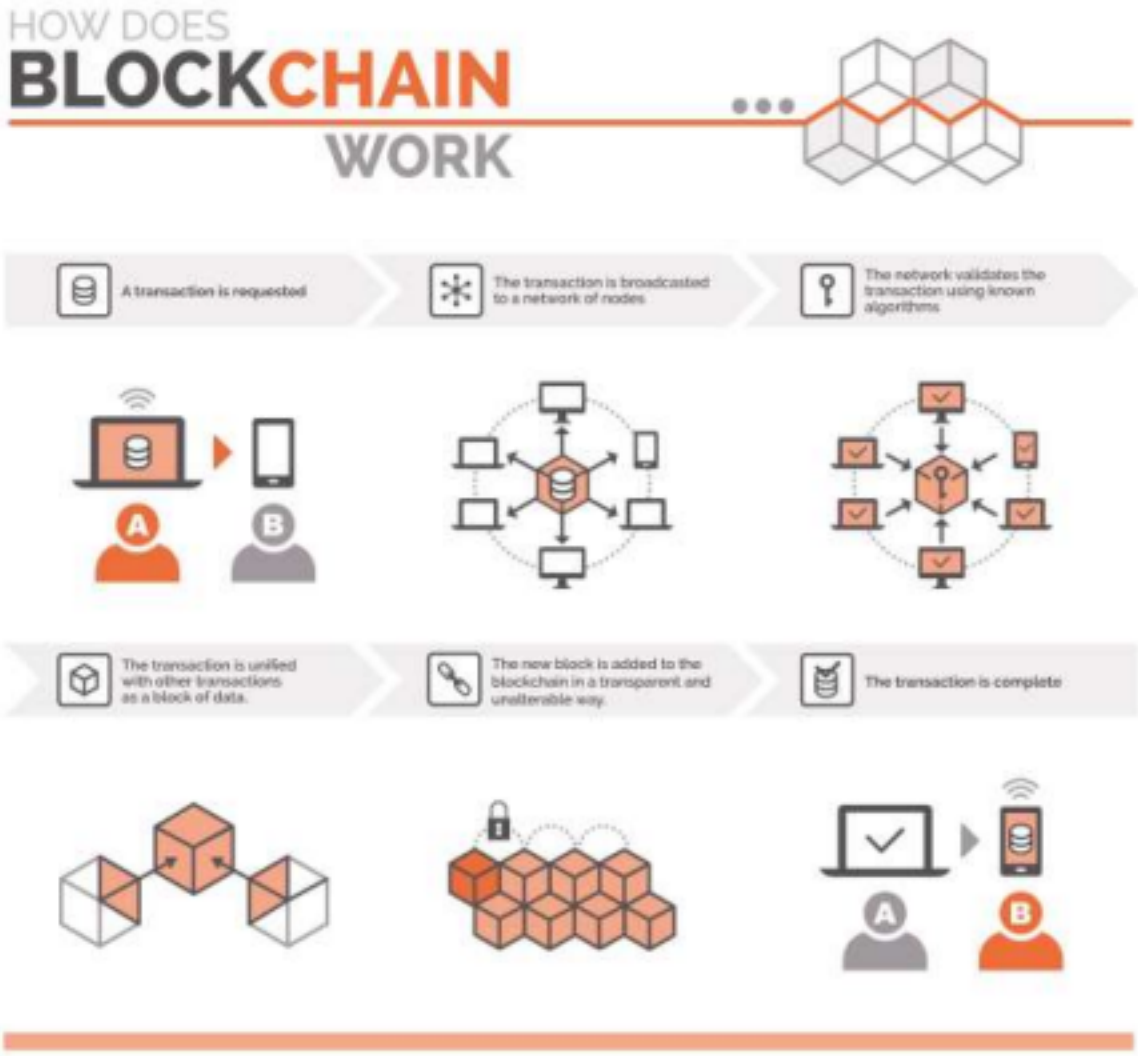
There has been a lot of buzz around blockchains in recent years, ever since Bitcoin became popular. Blockchain technology offers new tools for authentication and authorization in the digital world that preclude the need for many centralized administrators. The scope of this assignment is to help you get accustomed to blockchain development. You are required to make an Asset management technology using blockchain. You can work in teams of upto **4** members. You can register your teams here before 12 noon 2nd April, 2021.

2.1 Blockchain Fundamentals

A blockchain is a digital and distributed ledger of transactions, recorded and replicated in real time across a network of computers or nodes. Every transaction must be cryptographically validated via a consensus mechanism executed by the nodes before being permanently added as a new “block” at the end of the “chain.” There is no need for a central authority to approve the transaction, which is why blockchain is sometimes referred to as a peer-to-peer trustless mechanism.

Blockchain can be thought of as a linked list with each node containing multiple transactions. Each transaction has a hash which depends on the previous transaction's hash as well. So we can see that the order of transactions is important. If we were to change one transaction somewhere, it would have a ripple effect and change the hash of all subsequent transactions. This is one of the reasons why blockchain is a powerful medium for storing transactions.

The placing of a transaction in a block is called a successful conclusion to a proof of work challenge, and is carried out by special nodes called miners. Proof of Work is a system that requires some work from the service requester, usually meaning processing time by a computer. Producing a proof of work is a random process with low probability, so normally a lot of trial and error is required for a valid proof of work to be generated. When it comes to Bitcoins, hash is what serves as a proof of work. Miners on a Blockchain are nodes that produce blocks by solving proof of work problems. If a miner produces a block that is approved by an electronic consensus of nodes then the miner is rewarded with coins. This essentially is the crux of blockchain. Proof of Work is what is keeping all transactions on the blockchain secure and protecting it from malicious attempts to alter these transactions.



This is just a high level introduction. You are expected to dig deep and understand how the blockchain truly works. Different applications might implement it differently, but the core ideas remain the same.

2.2 Quick Introduction to Zero Knowledge Proofs

Zero-Knowledge Proofs (ZKP) refer to a proof construction that allow you to convince someone that you know something without revealing any information about what it is you know in the process. To explain ZKPs with the help of an example consider the following scenario:

Scenario:

Imagine two friends, Alice & Bob. Bob is color-blind. Alice has two identical balls that only differ in color: red and green. To Bob they seem completely identical and he is skeptical that they are actually distinguishable. Alice wants to prove to him they are in fact differently-colored, but nothing else, thus he does not reveal which one is red and which is green.

Alice gives the two balls to Bob and he hides it. Next, he takes out one ball and displays it. After that he hides the ball again and shows a ball. There is a 50% probability that he switched the balls. Alice is asked if the ball was switched. She could guess and answer correctly with a probability of 50% but if this exercise is repeated multiple times, we can see that this probability will eventually become negligible. So with 5 rounds, he will have a 1 in 32 chance of successfully faking. With 10 rounds, it is 1 in 1024, and with 20 rounds, it is about one in a million. This way one can reach any probabilistic level of proof that is desired, although an absolute certainty can never be achieved.

The above proof is zero-knowledge because Bob never learns which ball is green and which is red; but he can indeed verify that the balls differ in color.

As part of the assignment you will have to implement a similar proof when handling operations on data that you feel should remain secure (example your Aadhar Number when doing some sort of authentication). An algorithm for the same is outlined below.

Alice has sensitive data x for which she chooses two numbers p and g. p can be a large prime and g is a generator for p .The meaning of these terms will be covered as the course progresses. For now you can assume p=11 and g = 2. She calculates as y = (g^x) mod(p) she performs the following steps to create a zero knowledge proof for .

1. Alice chooses a random number 0<= r <p-1 and sends it to Bob as h = (g^r) mod(p)
2. Bob receives h and sends back a random bit b (could be 0/1).
3. Alice sends s = $(r + bx) \bmod (p-1)$ to Bob.
4. Bob computes (g^s) mod(p) which should equal h*(y^b) mod p

Here Bob acts as a verifier and checks if Alice knows the value of x without actually getting to know what x is. You will have to implement the same in your blockchain application. While implementing the proof it will be up to you to decide on the number of rounds this should go on for. Alternatively, you can also make use of the public-private key system as part of the zero knowledge proof.

3. Deliverables

1. You are free to code in any language but your solution must implement the following methods:

-- createBlock()
-- verifyTransaction()
-- mineBlock() or something equivalent for proof of work
-- viewUser()

2. All transactions must be verified before they can be added to a block. As part of the verification process, you are required to use zero knowledge proof (as described in section 2.2) to verify at least one attribute.

3. viewUser() should list all (successful) transactions against the user.

4. For obtaining the hash value of a block, you are required to use SHA-256 encoding. You can use any library function for this part. You also have to implement a DES (you cannot use library functions here) to encode the 256 bit output of the SHA-256 before returning the hash. Since DES takes a 64 bit input, you will take the first 64 bits of the SHA-256 output and encode using DES, then the next encode the next 64 bits of the SHA-256 output and so on. You will combine them in the same order. (ex: SHA-256 output = abcd, then you will output DES(a) + DES(b) + DES(c) + DES(d), here '+' represents concatenation). You can get the link to the keys of the DES used here.

5. You can submit your work here by 11:59PM 12th April, 2021. Any plagiarism found will be awarded zero marks for the assignment.

4. Further Reading

-- WTF is a Blockchain
-- What is a Blockchain
-- A guide to blockchain Implementation
-- Zero Knowledge Proofs
-- Asset Management using Blockchain

5. Marking Scheme

1. Successful implementation of Blockchain according to the deliverables: 20 Marks

2. Successful implementation of DES in the Blockchain: 10 Marks
3. Preserving Blockchain and users: 5 marks (The Blockchain and the users are preserved even after the application has been closed)

An alternative to the user to user transaction is to have a central bank with unlimited funds and preloaded assets. Any user can buy and sell to the bank.
You are free to add admin roles (to add assets to a user), but this is not necessary. Assets can also be added to the user manually.