Distributed Trust and Blockchains		Date:	*29.08.2019*
Instructor: *Sujit Prakash Gujar*		Scribes:	Zubair Abid Akshay Kharbanda

Lecture 3: Distributed Consensus in Bitcoin

1 Recap

1.1 Quiz Review

1. (a) $\frac{1}{8}$ is the answer.
   (b) 216. Geometric RV reason should be given.
2. $Z_6^+ = 1, 2, 3, 4, 5$ is not a group. $5^{-1} = 5$
3. $log_2 7 = 7$

1.2 Last Class

Previously, we looked at some attempts at crypto-currency before Bitcoin, namely *MyCoin* and *Trust Me Coin.*

1. *MyCoin (GoofyCoin)*
   The rules of GoofyCoin are :-
   - A person can create new coins by simply signing a statement that he's making a new coin with a unique coin ID.
   - A person who owns a coin can send it to someone else by placing his signature on it.
   - The validity of this coin can be verified by following the chain of hash pointers back to it's creator, verifying all the signatures on the way.

   Challenges: Double spending attack is the major challenge with the GoofyCoin, as an owner of a coin can send the same coin to multiple people with his own signature, which can be verified, and thus appear to be valid-looking to the new owner of this coin.

2. *TrustMeCoin (ScroogeCoin)*
   This cryptocurrency solves the doublespending problem. Like the GoofyCoin, TrustMeCoin too has two types of transactions:
   (a) CreateCoins
   (b) PayCoins

   The key ideas of TrustMeCoin are:
   - A designated central entity publishes an *append-only ledger* containing the history of all the transactions. Doublespending can be prevented by requiring all transactions to be written the ledger before they are accepted. That way, it will be publicly visible if coins were previously sent to a different owner.
   - A person who owns a coin can send it to someone else by placing his signature on it.
   - It is sort of anonymous (*pseudo-anonymity*).

   Challenges: The main problem with TrustMeCoin is centralisation. The central entity has too much influence, as in he can deny some users services, making their coins unspendable. He could even refuse to publish transactions unless they transfer some mandated transaction fee to him. Thus, it is a major threat to TrustMeCoin.

2 Distributed Consensus

2.1 How does distributed consensus work for blockchain?

Distributed Consensus protocol has the following two properties :-

1. It must terminate with all honest nodes in agreement on the value
2. The value must have been generated by an honest node

2.2 Fischer-Lynch-Paterson(FLP) Impossibility Theorum

The FLP impossibility shows that in an asynchronous distributed system (under certain conditions), it is not possible for any distributed algorithm to solve the consensus problem even if a single node is being dishonest.

Paxos: Paxos is a protocol that largely works in practice, but is possible to come up with theoretical test cases where the algorithm gets stuck, calculating for infinite time and thus not getting any result.

What is Bitcoin doing differently?
The specific set of cases under which the FLP Impossibility holds are altered in this new setting. Namely, alongside all else there is also a reward (bitcoins) given for maintaining the database correctly, and a penalty for being dishonest.

2.3 How is Distributed Consensus achieved in Bitcoin?

In bitcoin, the consensus happens in a specific manner:

1. New transactions are broadcast to all nodes.
2. Each node collects new transactions into a block.
3. In each round a random node gets to broadcast its block
4. Other nodes accept the block only if all transactions in it are valid (unspent, valid signatures)
5. Nodes express their acceptance by including its hash in the next block they create.

2.4 Challenges of the process

1. Choosing the node should not take too long.
2. The random node should not be faulty.
3. "Randomising" the node in real life is not an easy task.

3 How to select a random node?

We are motivated by the idea that selecting a node at random is not an easy task. Random number generators that exist are pseudo-random, using a seed value. We could try to therefore implement a pseudo-random generator, with a completely random seed, or discard the approach entirely in favour of another one.
In general, there are two ways we consider of selecting a random node.

1. Noise
2. Solving a puzzle

3.1 Using noise to select the random node

Noise is used as the pseudo-random generator seed.
However, it needs to be accessible to all the nodes.
One idea suggested: At each round, the protocol generates a certain random string and we pick a node with identity having the same bit string in its identity.
Challenges to this idea

1. This is prone to Sybil attacks. One dishonest user can create multiple accounts, increasing their own probability for mining the next block, leading to centralisation.

   Say dishonest user creates 1 Million blocks, and 10,000 other users have 1 block each. This user writes the next block with

$$\textsc{Probability} = \frac{1,000,000}{1,010,000} \approx 1$$

2. We also know in which order the following people will be writing the next block. This opens up space for human harm.

3.2 How much damage can a malicious node do?

1. *Stealing* others' bitcoins is not particularly a possibility, as digital signatures cannot be forged. To steal someone's coin would require the attacker to create a valid transaction that spends that coin. This would require him to forge the owners' signatures which she cannot do if a secure digital signature scheme is used.

2. *Denial of Service* to certain people. Even this cannot be achieved by the malicious node. If someone's transaction doesn't make it into the next block that a malicious node proposes, he will just wait until an honest node gets the chance to propose a block and then his transaction will get into that block.

3. *Double Spending:* A malicious node that has to pay for a service to another node can send the same coin to different addresses too, like it's own address. The receiver of the coin will verify the signature of the sender and will accept it. But as it spend the same Bitcoins, only one of these transactions can be included in the block chain. The honest nodes follow the policy of extending the longest valid branch, and at this point, the two branches are the same length – they only differ in the last block and both of these blocks are valid. The node that chooses the next block then may decide to build upon either one of them, and this choice will largely determine whether or not the double-spend succeeds.

   Prevention of double spending : The receiver of the coins should wait until that transaction is included in the block chain and has several confirmations. If the block containing the payment to him has been orphaned, it means it was a double spending attack. He should abandon the transaction. Instead, if it happens that despite the double-spend attempt, the next several nodes build on the block with the valid transaction, then there is a high chance that this transaction will be on the long-term consensus chain.In general, the more confirmations a transaction gets, the higher the probability that it is going to end up on the long-term consensus chain. The most common heuristic that's used in the Bitcoin ecosystem is to wait for six confirmations, as it is a good trade-off between the amount of time to wait and the guarantee that the transaction ends up on the consensus block chain.

3.3 Puzzle approach

Another approach is to use a cryptographic puzzle (hash puzzle), and the one who solves it first will be appending to the ledger next.

An obvious *concern* is that machines with higher computational power will solve the puzzles faster, and can again control the system if that much computational power is available to them.
So the difficulty of the puzzle for each node is such that the probability of a node solving the puzzle is proportional to its computational power.

4 Cryptographic Puzzles

4.1 What is a Puzzle?

Generally, the cryptographic puzzle is the following :-

Find a *nonce* such that:
$H(nonce \ || \ prev\_hash \ || \ txn1 \ || \ txn2 \ || \ ...) \ < \ target$

This means in order to create a block, the node that proposes that block is required to find a number, or nonce such that when you concatenate the nonce, the previous hash, and the list of transactions that comprise that block and take the hash of this whole string, then that hash output should be a number that falls into a target space that is quite small in relation to the much larger output space of that hash function.

The *SHA-1* hash function gives a 256-bit hash. Hence, for *SHA-1*, the target will be a 256-bit number. Some observations about this puzzle are :-

- The puzzle is easy to verify, but not easy to solve. This is because the hash function is not invertible.
- If $target \ = \ 2^{256}$, then any nonce can solve the puzzle. Hence this is the easiest puzzle.
- If $target \ = \ 2^{255}$, this means that the nonce should have its most significant bit(MSB) as 0.
- In general, if $target = 2^{256-k}$, this means that the nonce should have its most significant k-bits as 0, for which the probability for the nonce= $\frac{1}{2^k}$.
- Since the number of trials for nonces required is a geomertic random variable, hence the expected number of trials, $E \ = 2^k$.
  Thus if $target = 2^{256-k}$, then a node has to perform 2^k trials on an average.
- Thus greater the k, lesser the *target*, and greater the difficulty.

4.2 Proof of Work and Incentive Engineering

The puzzle allows for choosing the next node that will append to the block. This gives us all the tools needed for a functioning system.

4.2.1 Agreement on the correct chain

So how is the public ledger maintained over a distributed system? As we see, multiple aspects of cryptography have been called in to help, but the final key is in Game Theory. To study in more detail, we take a look at the procedure for getting all peers to agree on a correct chain.

1. The longest chain is observed as correct

2. Appending to the longest chain is rewarded- this is incentive not accounted for in the FLP Impossibility theorem. This is where Game Theory comes into the picture.

3. Appending anywhere but to the longest chain is penalised (also Game Theory).

4.2.2 Reward for writing blocks

The current reward for writing a block to the bitcoin public blockchain is currently 12.5 BTC. The reward halves for every $210,000$ blocks, which works out to around 4 years in current day.

Limitation on number of coins

The reward for mining a block started off at 50 BTC. Since the reward halves every few years, the number of possible bitcoin is theoretically capped (as it is a converging progression):

$$\textsc{Max coins created} = 50(1 + \frac{1}{2} + \frac{1}{4} + ...)K$$

$$= \text{100K}$$
$$= 21 \text{ million BTC}$$

Note that it is not possible to actually split the reward till infinity, due to floating point precision concerns among others. The least denomination for BTC is 10^{-8}, also known as 1 *Satoshi*.