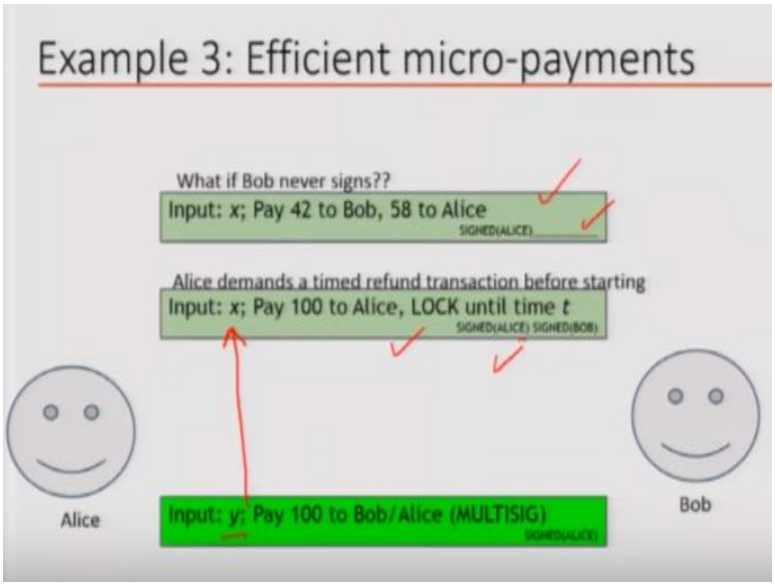
Introduction to Blockchain Technology & Applications
Prof. Sandeep Shukla
Department of Computer Science and Engineering
Indian Institute of Technology-Kanpur

Lecture - 11

Welcome back. So we were discussing in the previous session about micropayments, and we explained to you how the micropayment works. Now we at the end we said that in case the other party does not sign the last transaction, which would have completed this micropayment into a single payment, then Alice is going to get locked out with some of her changes that she should get back.

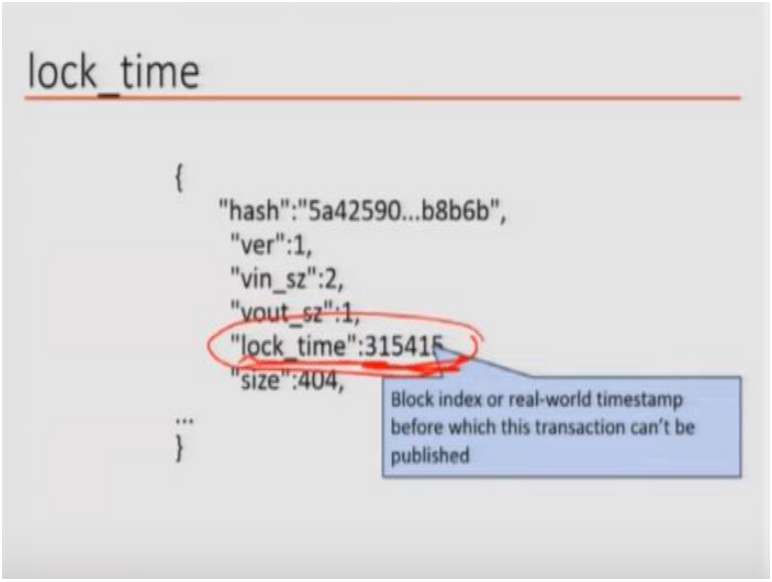
(Refer Slide Time: 00:46)



So therefore, in the beginning itself Alice takes in a demand from Bob a transaction that basically is a locked time transaction, which is signed by Bob and in which Alice is supposed to get back all the money. But it will be only be effective after time t. Now remember that this transaction is also using this as input. If this is ever accepted then this cannot be accepted, if this is accepted this cannot be accepted.

So once this transaction goes through it will not be effective. And if this transaction goes through this cannot be effective right. So that is the first thing. Second thing is that what lock until time t refers to is what we skipped earlier when we were discussing the metadata part of the transactions in bitcoin. So let us look at that again.

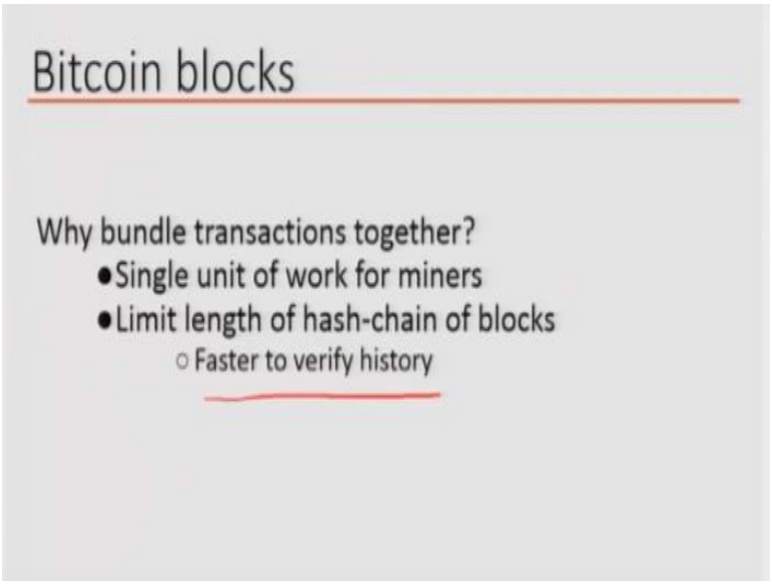
(Refer Slide Time: 01:37)



So this is the lock time feature and we earlier we saw it was zero which means that it is immediate transaction and here it says lock time as a time or the block number, right. So it could be the timestamp of the blocks. This transaction will not validate until that particular timestamp is arrived or crossed or if it is a block number then if that particular block number has been already is being mined.

So that is the idea of the lock time and this is being used in this micropayment transactions. So it could be a block index and or the timestamp and this transaction cannot be published in the sense that it cannot be put in a block until that block number or the timestamp has arrived. So next so we have seen the transactions now we look at the bitcoin blocks.

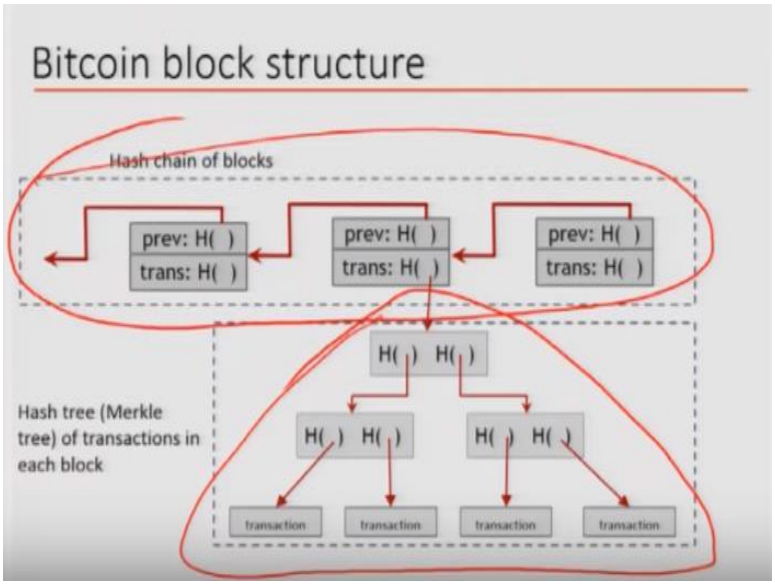
(Refer Slide Time: 02:36)



So blocks are when we bundle transactions together, we already know and then this is what the miners work on. So this is a single unit of work for miners and the length of the hash-chain. So if we had every transaction in one as a block, then you know you will have huge length blockchain. So therefore, when you put a lot of transactions together in a block, the number of blocks is much less.

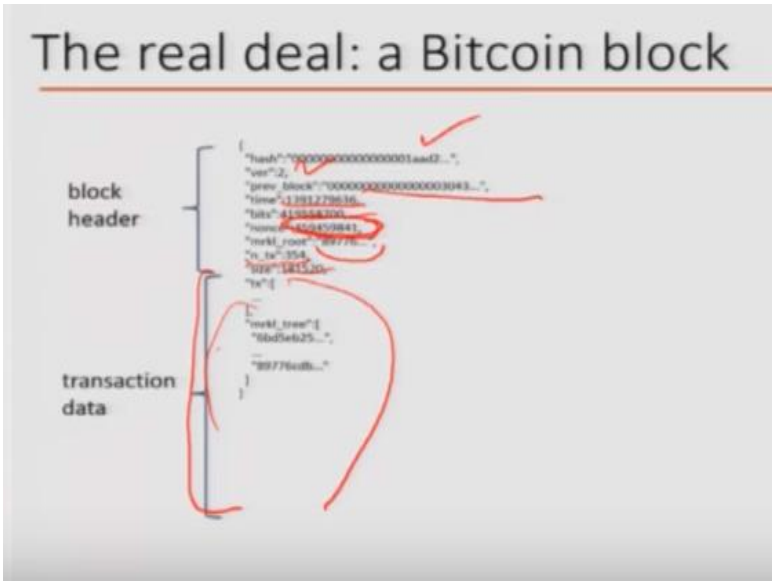
And therefore the length of the hash-chain is less. So that is the, and so if you want to verify history you do it block by block rather than transaction by transaction.

(Refer Slide Time: 03:12)



So the in the block structure, this is the actually the blockchain. So you have the hash of the previous block and the transactions and the transactions are put in a Merkle data structure. So this is how the transactions are stored so that you can quickly search through the transactions when you want to check, verify some history. So this is the block and this is the Merkle tree. So the Merkle tree root is stored here in the block.

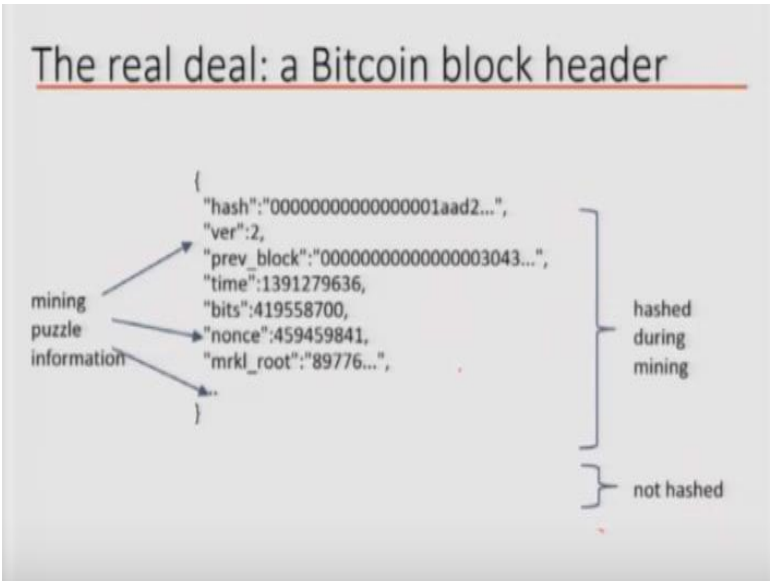
(Refer Slide Time: 03:39)



So the block header looks like this and this is where the transaction data is put. So this is the kind of like block header has a metadata. So this is the hash of the block. This is the version number. This is the previous block hash. And this is the timestamp. This is the number of bits. This is the nonce. This is what the when you solve the hash puzzle, remember there was a nonce that you have to discover in order to solve the hash puzzle.

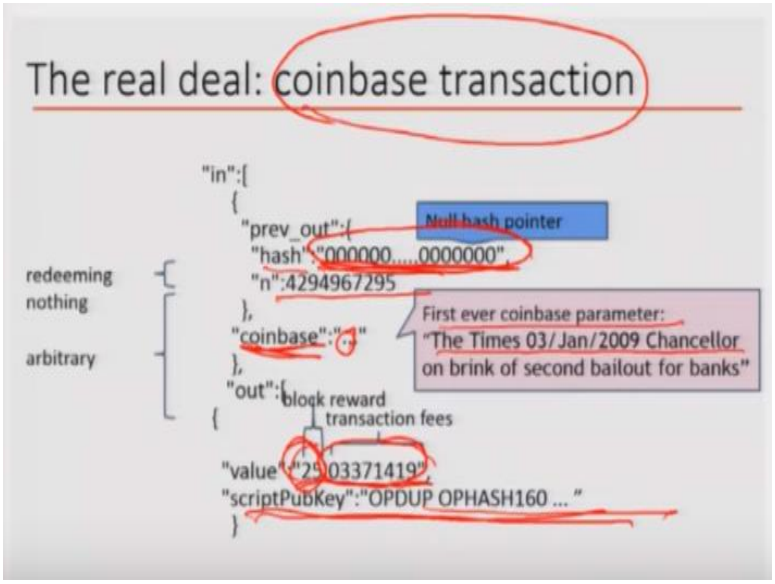
And that nonce can be checked to be the solution to the hash puzzle easily. So that nonce has to be stored here. And then the Merkle root, Merkle root is the root of this Merkle tree in which you have stood the transactions. And then you put the number of transactions, the size and then the transaction. And here is a Merkle tree for the transaction. So that is what the block structure looks like.

(Refer Slide Time: 04:36)



So this is what we talked about just now. These are all mining puzzle information, the version, the nonce, and some other maybe some other information. And then this is this part is hashed during mining. And the entire Merkle tree is not hashed. So this is the part of the block that is hashed for mining purposes.

(Refer Slide Time: 05:00)



Now there is when you make a block, you have to pay yourself whatever 25 coins or 12.5 coins whatever the current reward is. So anybody who mines the block, maybe his block will not eventually make it to the blockchain. But if I am mining a block and I have solved the hash puzzle, then what I do is I will create an extra transaction and put it into the block. And this is called a coinbase transaction.

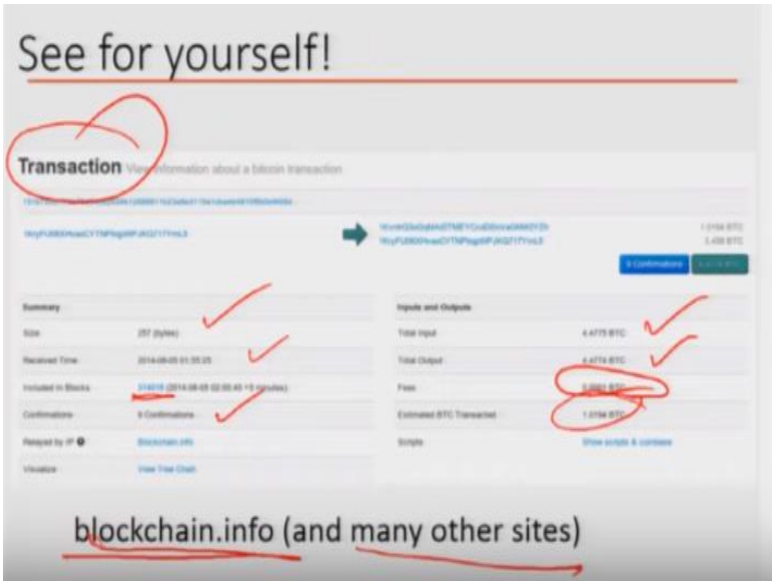
We mentioned this name before. So what the coinbase transaction does is that it creates a value of 25 or right now it is 12.5 and then add some transaction fee to it. And then you have this scriptPubKey that tells you that is one of your addresses, right because you are going to get the reward. So whoever is putting this transaction, see that the input is has is nothing. So it is a null pointer, right.

The hash of the previous output is all zeros, which means that there is no real input, right. And the output on the other hand is 25 point something. And right now it will be different, but we and then it is the hash of the public key of your address in which you want to get the reward. So this is what is called a coinbase transaction. In a coinbase transaction, there is also something here called coinbase.

And there you can put any information. So coinbase field can be an arbitrary information. And so this arbitrary information can also be used to store some information for the future. So what this is that in the first ever coinbase parameter in the very first block that is the genesis block the creator of bitcoin blockchain put this information. The Times 3rd January 2009 Chancellor on brink of second bailout for banks.

So we discussed this earlier that the bitcoin blockchain actually came as a rebellion against the 2008, 2009 slow down economic crisis, which was precipitated by the banking sector. So that seems to be this use of this particular arbitrary string in the first coinbase parameter kind of you know justifies that suspicion that that was the reason why the bitcoin blockchain was created.

(Refer Slide Time: 07:29)

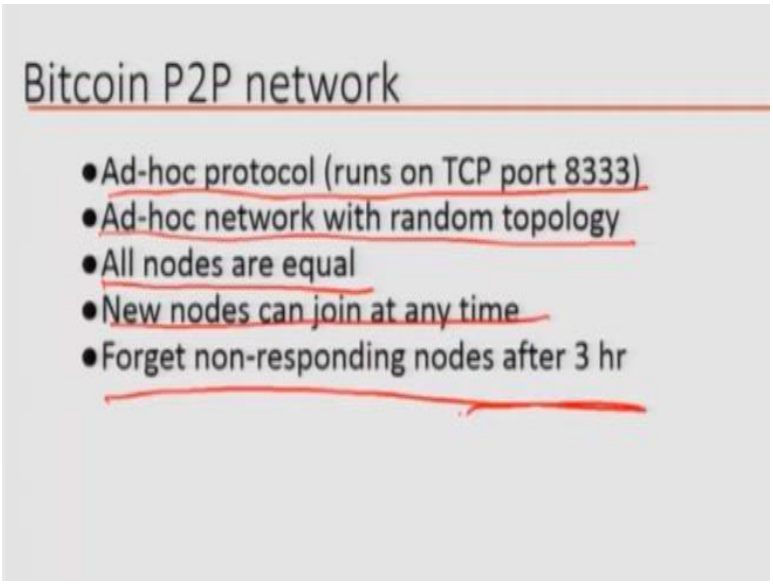


You can actually go to blockchain.info and there are many other sites. And you can actually look at browse through all transactions because the bitcoin blockchain is actually all public. So all the information, all the transactions all the blocks, they are all visible to anybody who wants to see and there are websites which get the data from the blockchain and make it viewable in nice format. So this is what you can see.

So this is about one particular transaction and it shows the size, the receive time, which block number it was added to, how many confirmations it has. And then you can actually see the tree chart. And also you can see inputs, amounts, output amounts. So there is a point 0001 transaction fee given in this transaction and how many transactions, how many bitcoins were transacted, this kind of information.

So you can actually browse this kind of information by going to the blockchain.info to get yourself some familiarity with this. So now let us talk about the bitcoin network. So we said earlier, that bitcoin actually runs on a P2P network. The P2P network is basically an overlay network on top of the TCP IP network.

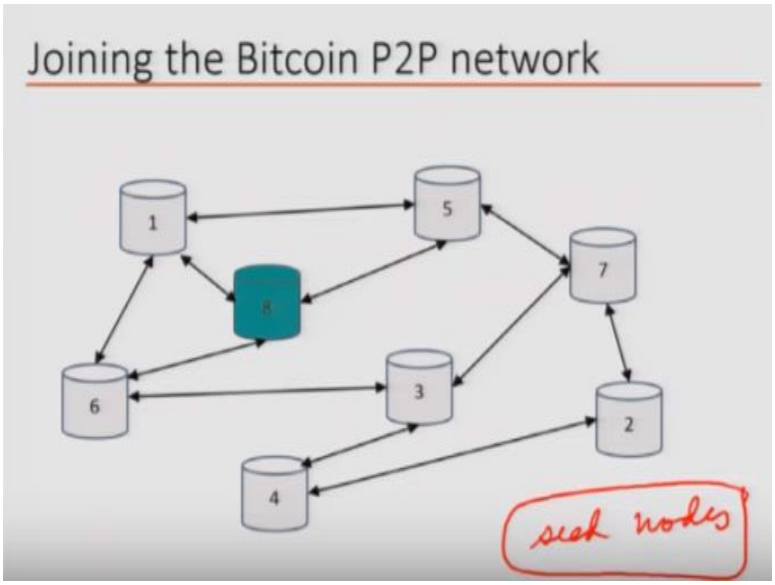
(Refer Slide Time: 08:51)



And on top of this P2P network is running the bitcoin application level protocol. So it is an ad-hoc protocol and it runs on TCP port number 8333. So an ad-hoc network with random topology gets generated as the overlay network on the standard internet. All nodes are equal. So there is no like a client server kind of architecture. New nodes can join any time, the barrier to entry is very small.

And also you can forget non-responding nodes after three hours, but the node can actually again wake up and join. So let us see how the joining works.

(Refer Slide Time: 09:32)



So the joining the bitcoin P2P network can be described with this figure, very nicely done by Arvind Narayanan. Here is let us say, the existing P2P network and suppose you want to join. So you say I want to join so you send out a message and there are

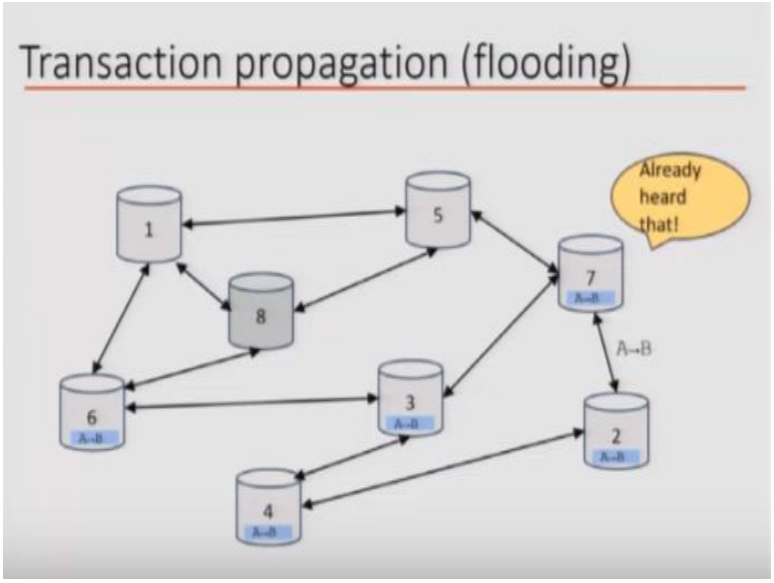
something called seed nodes. So seed nodes you can figure out most of the bitcoin software when you download them. It usually has the seed nodes listed in them.

So you can actually send message to the seed nodes first to start contacting the P2P network participants. So you let us say 5 is a seed node. So you send a message to 5 saying that tell me who else you know. Now 5 will tell that I know 1 and 7, I am connected to 1 and 7. So eight will also know 1 and 7. So it will send message to 1 and 7 saying that who else do you know.

So 1 and 7 will then give him some more information, and let us say 1 will say I know 6. And then 7 will say I know 3 and 2, but 8 may choose since this is does not have to be a fully connected network, 8 may choose that I will only be directly be connected to 1, 5 and 8, which means they are my neighbors. So whenever I want to forward a transaction or forward a block, I will only send to these three.

They can then forward it to the others they know. Or 8 may decide that I want to be more connected. So you might actually then get connected to 3, 7 and others. That is the ad-hoc nature of this P2P network.

(Refer Slide Time: 11:08)



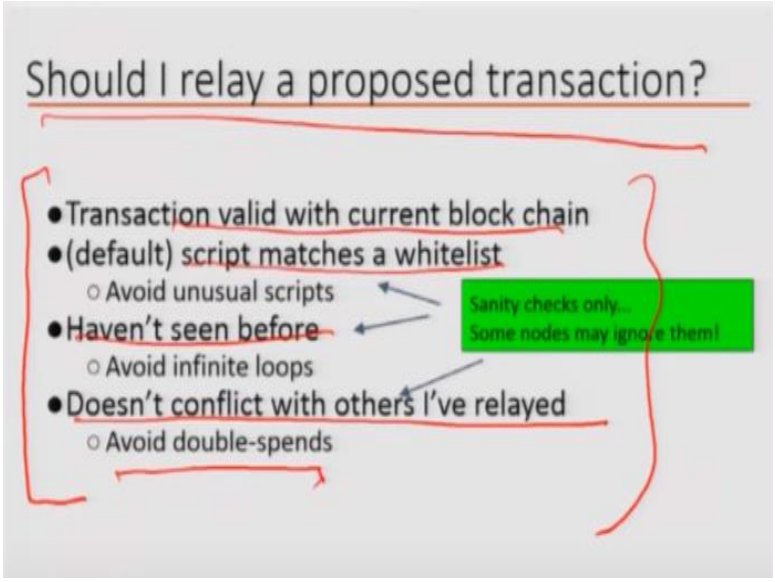
Now how do they do transaction propagation? So it is often called a gossip protocol. And also it is a flooding mechanism. So let us say there is a new transaction that 4 learns about or somebody who is actually connected to 4 puts that transaction on. So

the 4 will then send it to all its peer neighbors. So in this case, 4 is directly connected to only 3 and 2.

So it sends it to 3 and 2. 3 and 2 will check whether they have seen this transaction before. If they have not seen this transaction before, they will validate it. And if the validation goes through, then they will keep it in there you know pending transactions. The transactions that they are going to put in a block in case they are going to make a block. But they will also forward it to if they are satisfied that this is a valid transaction and this is forwardable, then they will forward to their neighbors.

So then 3 sends it to let us say 6 and 7. Now 6 and 7 now will learn about this. Now 2 will also send to 7. And then 7 will say I have seen this, so 7 will ignore it. So this is how this thing works.

(Refer Slide Time: 12:19)



So the question is, when do the nodes relay a proposed transaction? First of all, they check the validity of the transaction. Second thing they checks that the script matches a whitelist of scripts. So if the script in the transaction is unusual, they suspect that somebody is doing something malicious and they may not store it and they may not even forward it.

They must make sure that this is not an already seen transaction because if it is an already seen transaction, and if they still forward it, then there will be an infinite loop created. So anything that you have seen before you are not going to forward because

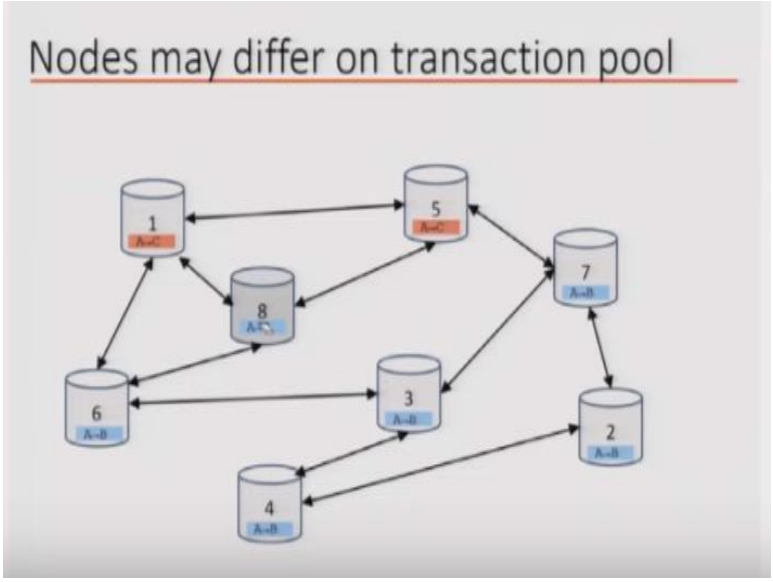
you have already forwarded it or you got it from somebody your neighbors. And it does not conflict with others that they have relayed. So this avoids the double-spend.

So these are the things that a good participant will check. Now if the participant is malicious, they may not check the validity and forward it. They might send something that they have seen before. They might actually not create whitelist and send any transactions out. So all that kind of stuff is also possible.

And therefore, the entire system should be designed in such a way so that even if some of the nodes does that kind of behavior, rogue behavior, not the prescribed behavior, then also the system works and it does not get into you know some kind of an infinite loop and other kinds of stuff. So we assume that most nodes will be sane and not rogue and therefore, they will stop.

Like even if you forward a malicious note forward something but the others have seen it before they will not forward it. So at some point the forwarding will stop.

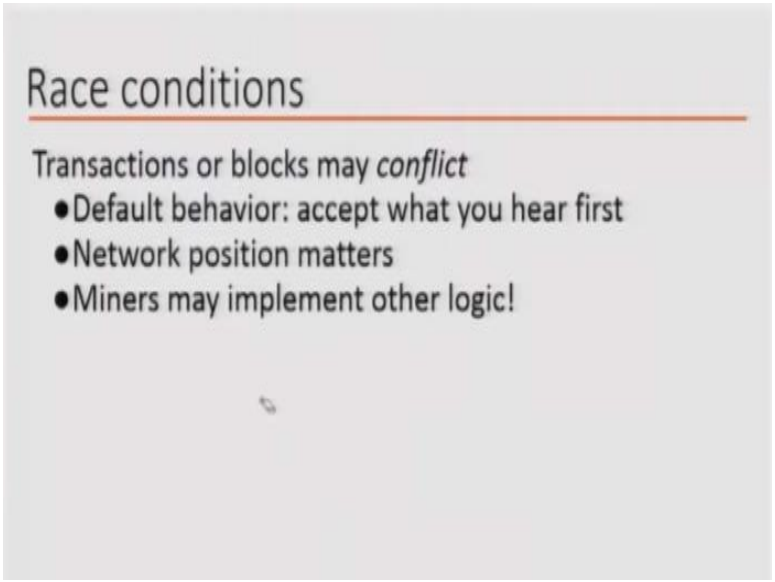
(Refer Slide Time: 14:03)



Now as we have seen in the previous sessions that nodes may differ on their pending transaction pool. So for example, you see that right now let us say these guys have these A to B transaction, and then one learns a new transaction A to C and starts flooding. So it sends A to C to its neighbors. 5 stores it but 8 may ignore it and 6 may ignore it. Now what that means is that there may be many reasons to ignore it.

Maybe 1 and 5 were malicious. So they are storing an invalid transaction, or maybe they are not malicious but they are the right ones and then 8 and 6 are maliciously ignoring this transaction. Anything can happen. And your system should be designed such that even if those things happen, because of the redundancy of mechanism in such a distributed system, eventually everything will work. That is the idea.

(Refer Slide Time: 15:04)

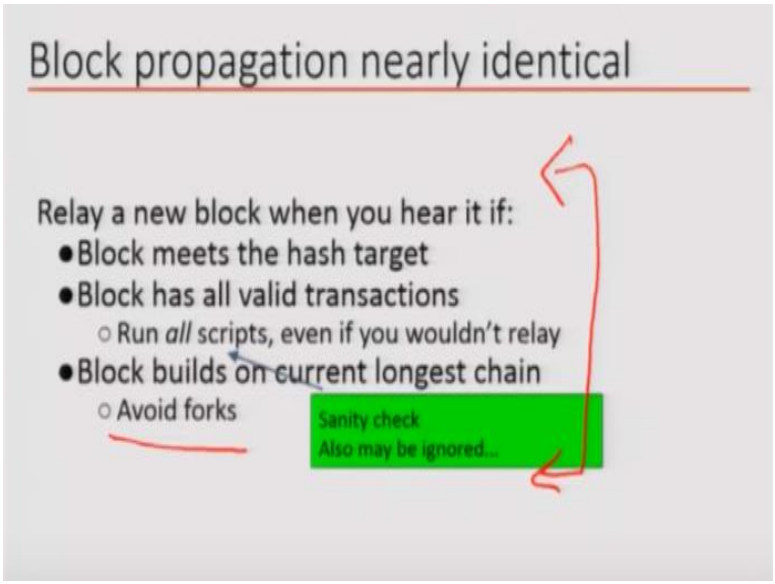


So transaction and blocks may also conflict, right. So as we discussed before that the nodes are geographically distributed very far, there is network delay and various kinds of things. So if I am close to the originator of the transaction or the originator of a block that is being broadcast, I might hear it first. But very far away node may hear it much later.

And therefore the blocks that I create would be different from each other. So each of the participant may have a different set of pending transactions and different set of blocks. So what they add in the blockchain or what they add in a block will differ from each other. So there would be some risk conditions that has to be automatically resolved at least probabilistically.

So with high probability, eventually the consensus branch will be unique in all the different nodes. So that is the goal.

(Refer Slide Time: 16:07)

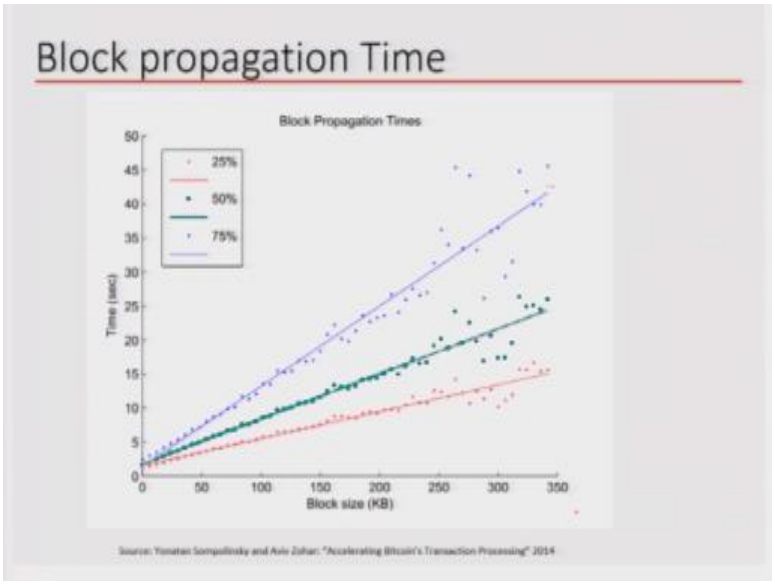


So the block propagation logic is also almost the same. You relay a new block when you hear it. First of all you have to check if the hash puzzle has been solved by this block by checking the nonce against the rest of the block. All the transactions in the block should be valid. So you check the validity of each transaction before you actually forward the blocks.

And the block is the, the current block in its previous hash is pointing to the last block in the longest chain. That because there may be some sub chains that are being created temporarily and then they get orphaned. But right now whatever is the longest chain, you should be building on top of that. So and this is to avoid forks. Again, this is what they say nodes or good nodes will do.

Now a malicious node can differ and do different things. And the protocol should be eventually robust to some of the nodes behaving differently than this.

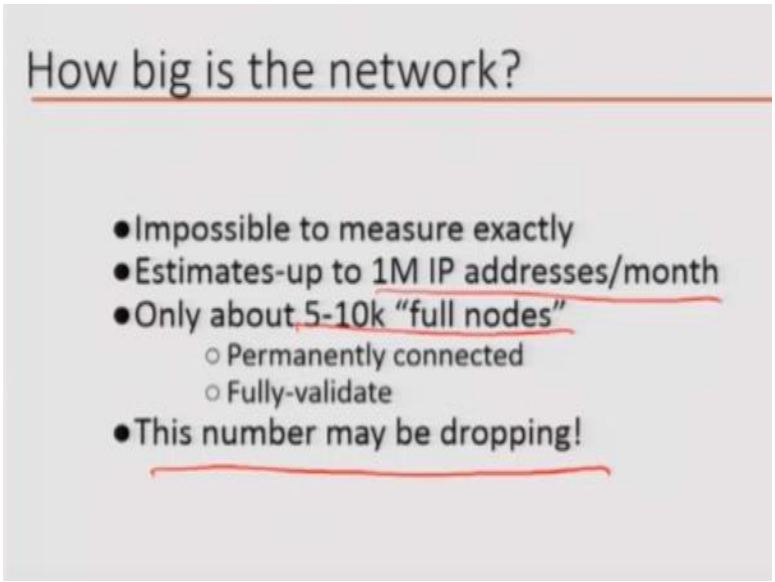
(Refer Slide Time: 17:11)



Now if we look at the real block propagation time, so this is the quartiles being plotted. This is the 25% quartile, and this is the 50% quartile and this is the 75% quartile. You see that 25% of them reaches all the nodes in like 15 seconds. The 50% of them reaches in about 25 seconds. And then there are some which actually reach, takes about 40, 45 seconds. So that is the overall propagation time.

And as the block size increases, the propagation time also increases. That is what we see here.

(Refer Slide Time: 17:52)

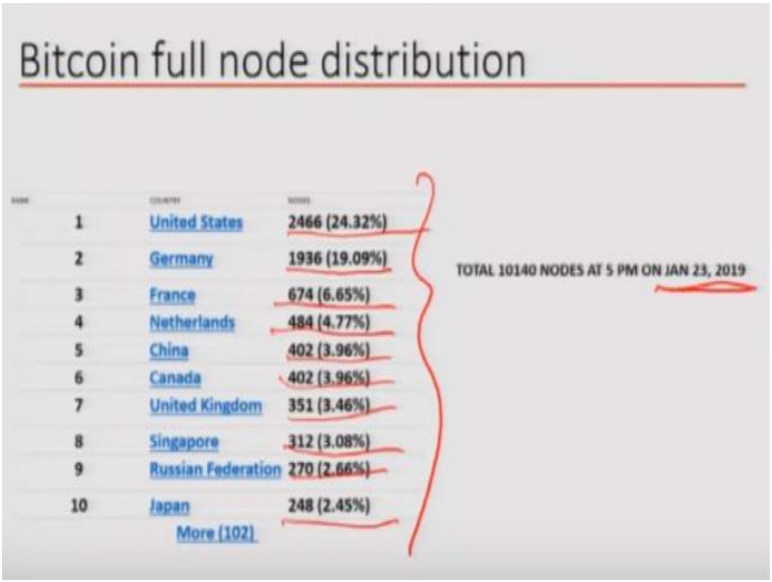


Now if you ask how big is this network, it is impossible to measure exactly. But it is estimated that up to 1 million IP addresses are involved in the blockchain per month, but only about 5 to 10,000 are full nodes. Full nodes are those which are permanently

connected, which have the complete blockchain copy. And they actually fully validate all the blocks.

They validate the hash puzzle has been solved, they validate that it is extending the longest chain and they also validate all the individual transactions in a block. And as the bitcoin is becoming less popular or less you know rewarding, this number may drop.

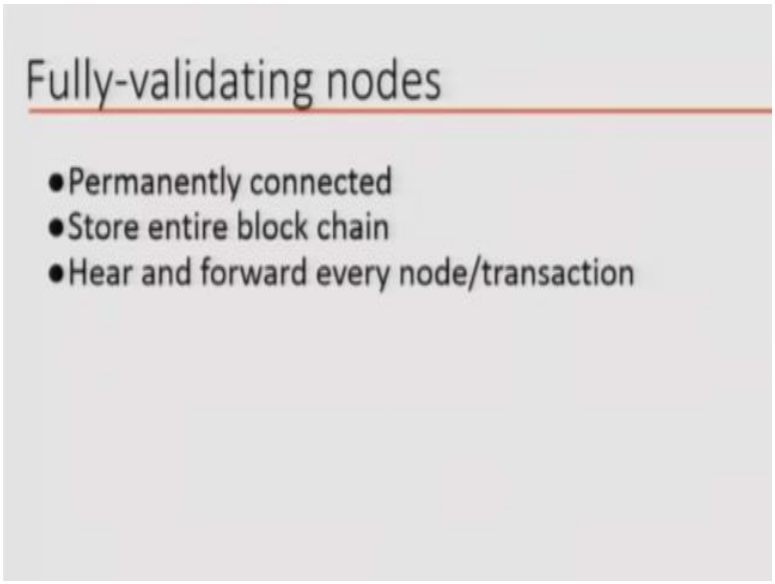
(Refer Slide Time: 18:34)



So let us look at the numbers as of January of 2019. So it is about 10,000 indeed at that time. And the United States had the highest number, almost 2500. Germany had almost 2000. France had about 674. Netherlands had about 500. China about 400. Canada about 400. UK about 350. Singapore about 300. Russia has about 270. Japan had 248 nodes. So these are the node distribution.

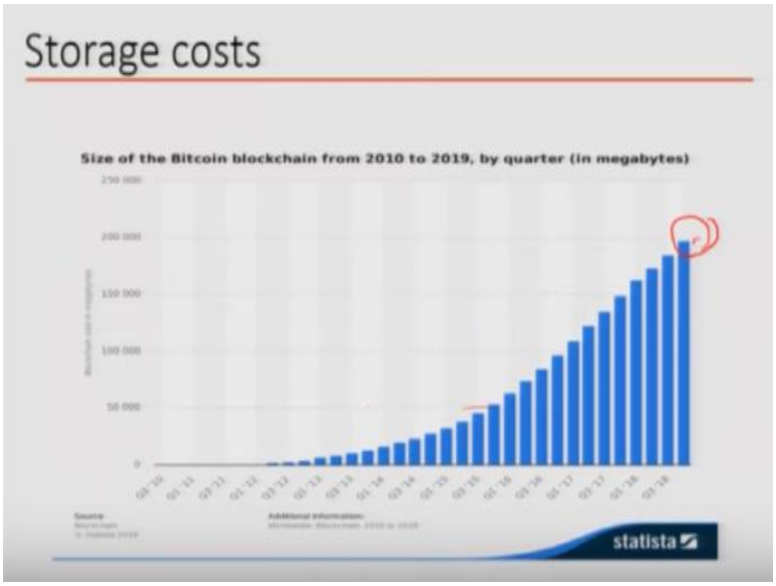
Full nodes require a lot of, although they may not be miners, but they are still going to be doing a lot of computation. And they have to be always connected. So that is a big deal to actually run full nodes.

(Refer Slide Time: 19:27)



So these are called fully validating nodes and they store the entire blockchain and they hear and forward every node and transaction.

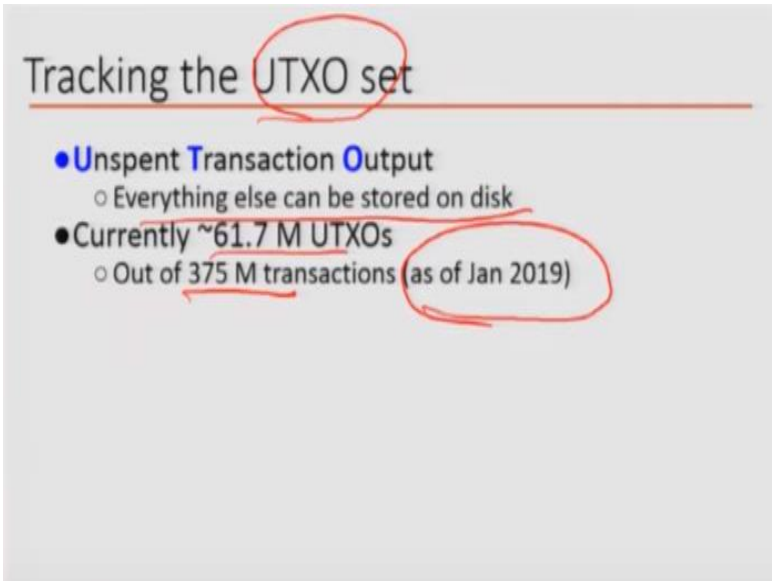
(Refer Slide Time: 19:37)



And then the storage cost. If you look at the numbers, right now it may be more but until 2019 it was about you know 200 gigabytes to store the entire blockchain. It may be about between 200 and 250 gigabytes. So you also have to if you have to store the entire blockchain, you need to have a lot of space also, right. 250 gigabytes is a lot of space. And it increased over time. It was in 2016, it was about 50 gigabytes.

And then by 2018, it was 200 gigabytes. So you can see how fast it was growing at that time.

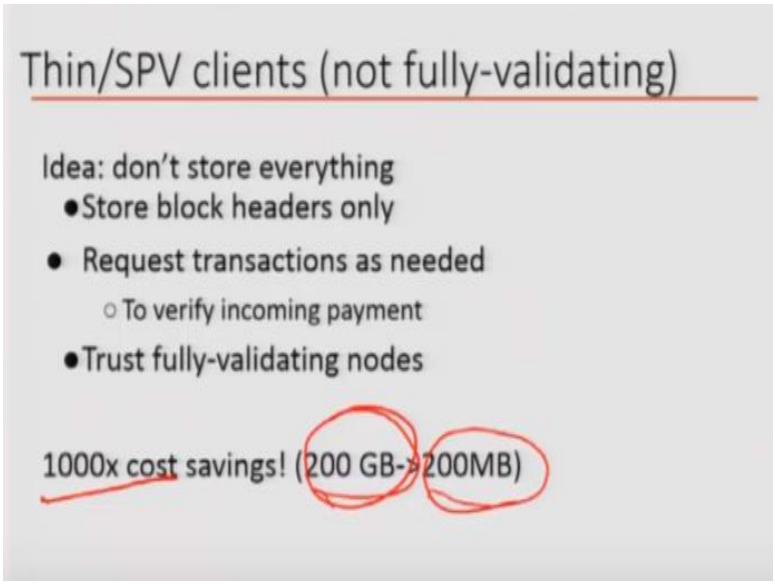
(Refer Slide Time: 20:15)



And also they have to track the unspent transaction output or UTXO set, because if a transaction is being referred to as input to a transaction, it must be one of the unspent transactions, everything else can be stored in the disk. So you can actually cache this information from the blockchain and rest you do not really need to use that often to validate transactions.

So at this point, as of January 19, there were about 62 million unspent transactions out of the 375 million transaction. So it was about one-sixth of the transactions. That is about 15% of the transactions were actually unspent. So you see that you still have to keep a lot of information around.

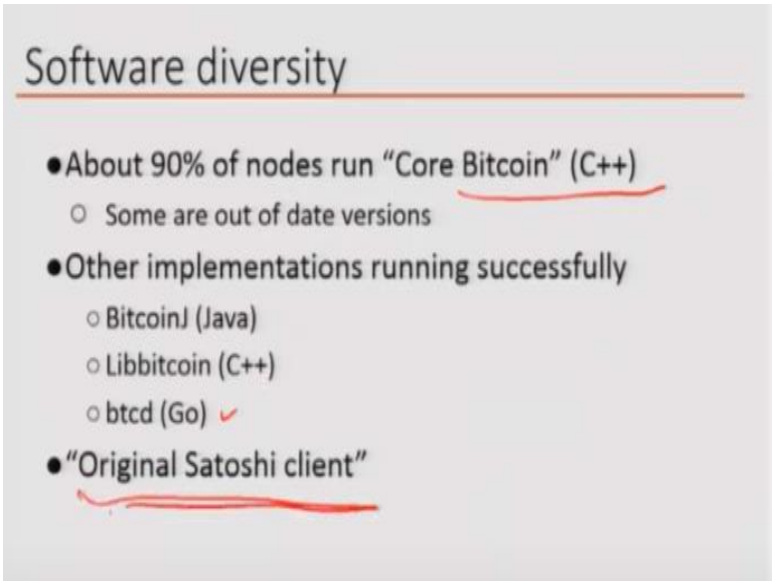
(Refer Slide Time: 21:05)



So thin clients which are not fully validating nodes, they do not store everything. They store the block headers only. They request transactions as needed. And they trust the fully validating nodes for doing the actual validation. Now these, just the headers the block headers will cost you at this point in January was about 200 megabytes. That is an actual full blockchain was 200 gigabytes.

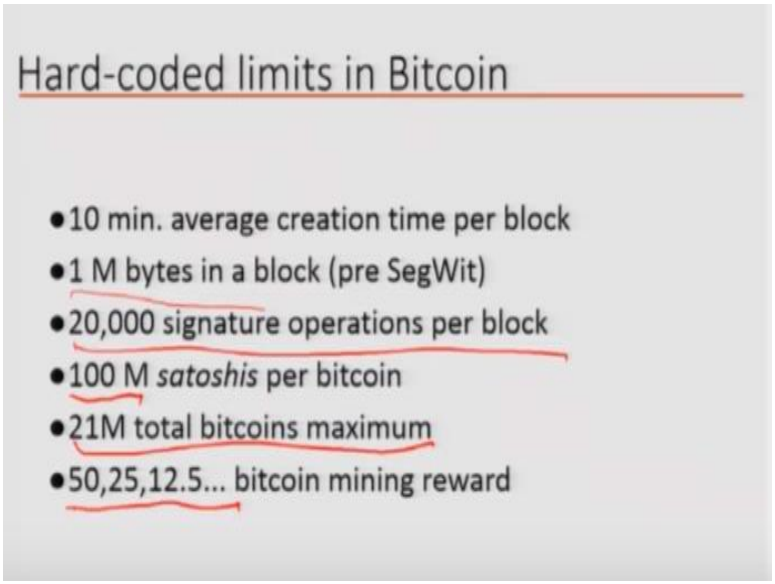
So it is about thousand times cost saving in terms of space. So wallets for example, if you have a blockchain wallet, that usually has an SPV built into it. It is a simple client. So they are not fully validating. They only worry about transactions that are relevant to them. So if somebody is paying your wallet money, then it actually keeps track of those and validate those.

(Refer Slide Time: 22:04)



Now there is a diversity of the software that it being used here. So for example 90% of the nodes run the core bitcoin that is C++. And then some there are some Java clients and there are some other C++ clients and there are Go clients. And then there is the original client that was designed by Satoshi Nakamoto. So we will now finish by discussing very quickly the limitations and improvements. So first of all, there are some hard coded limits in bitcoin.

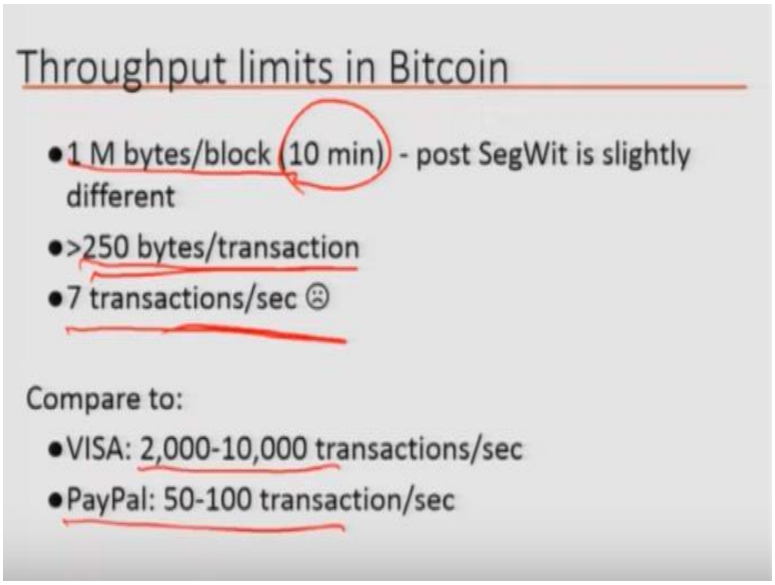
(Refer Slide Time: 22:35)



10 minute average creation time per block. 1 megabyte size for the block. There was a discussion on actually having a forking of the blockchain to actually increase it to 2 megabytes, but as far as I know it has not happened. 20,000 signature operations per block and there every bitcoin is divided into 10 to the 8 satoshis per bitcoin. So the resolution of paying bitcoin is about 10 to the 8.

And there is 21 million total bitcoins maximum, we discussed this before. And the only way to create bitcoin was through mining and the mining rewards have been decreasing and by 2040 it will reach 21 million when it will stop. So these are hard-coded limits of the bitcoin blockchain.

(Refer Slide Time: 23:29)

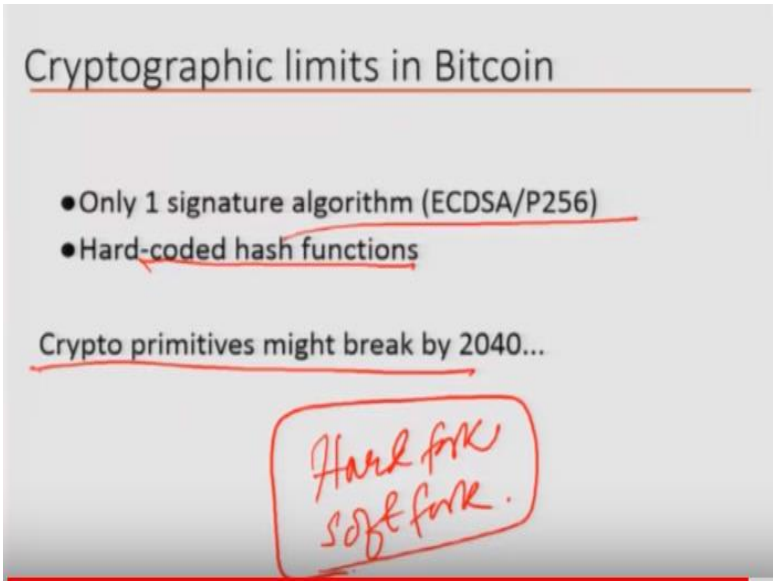


There is a throughput limit because if you have only 1 megabyte per block and each on an average each transaction is about 250 bytes, then you can have only so many about I think 40,000 or so transactions and they you know confirmed in 10 minutes because until the block is mined you have not seen it into permanence, although actually you have to wait for 60 minutes to see 6 confirmations.

But assuming that you do so many you are okay with zero confirmation even then you are actually having only a block full of transactions. And then that turns out to be about 7 transactions per second. And if you compare this to Visa, MasterCard, they can do 2000-10000 transactions per second and PayPal for example can do 50-100 transactions per second.

So compared to that, the throughput is very small, which basically is a problem in using bitcoin for real transactions.

(Refer Slide Time: 24:36)



So there is only 1 signature algorithm that is used. So and there is a hard-coded hash function that does it. And the other issue is that the crypto primitives that are implemented in the scripting language, they might break by 2040 because cryptoanalysis and cryptographers are always breaking the crypto's primitives. And then you have to redo the entire scripting language. The old scripts cannot be used again.

So there will be some problem and that will require something called a forking of the bitcoin blockchain. And so there are two types of forking, hard forking and the soft forking. And when we come back to the next session, we will talk about this two types of forking because this forking issue is true for other block chains also. So we have to understand what forking means.

So in this session, we will end here by just saying that there are certain limitations of the blockchain especially in terms of the longevity of the current version, because of the cryptographic primitives might be dated by the by 2040. And therefore will have some, we have to do some in the scripting language and that would require a change in the entire blockchain infrastructure.

And that would require some forking and we will discuss forking in the next session.