Blockchains Architecture, Designs and Use Cases

Prof. Sandip Chakraborty
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture – 51
Research Aspects – IV (Byzcoin)

Welcome back to the course on Blockchain. So, we are discussing about this is Byzcoin Consensus Mechanism. In the last class, we have looked into a little bit of background of nice cryptographic signing mechanism called CoSi or the Collective signing which Byzcoin uses for signing individual blocks.

(Refer Slide Time: 00:41)

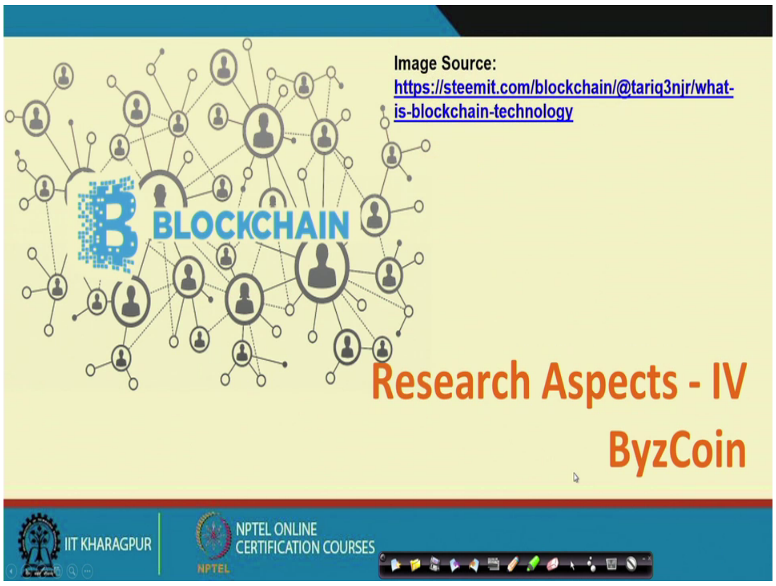


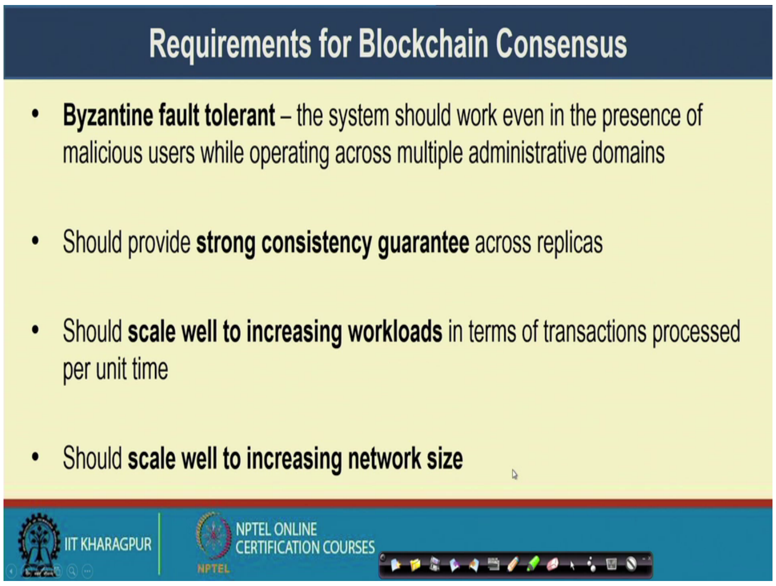
Image Source:
https://steemit.com/blockchain/@tariq3njr/what-is-blockchain-technology

Research Aspects - IV
ByzCoin

BLOCKCHAIN
IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES

So, today we will look into the details of this Byzcoin mechanism; about how Byzcoin and how Byzcoin solves the problem which was there in case of Bitcoin and Bitcoin-NG protocol. And Byzcoin is a nice architecture in the sense that it the combines the concept from proof of work and PBFT together and it takes the good of both Bitcoin as well as Bitcoin-NG. And by combining all this concepts all together, it designs a nice protocol and nice mechanism for ensuring consensus in a Blockchain base system. So, let us look into the Byzcoin in details.

(Refer Slide Time: 01:22)



Requirements for Blockchain Consensus

- **Byzantine fault tolerant** – the system should work even in the presence of malicious users while operating across multiple administrative domains
- Should provide **strong consistency guarantee** across replicas
- Should **scale well to increasing workloads** in terms of transactions processed per unit time
- Should **scale well to increasing network size**

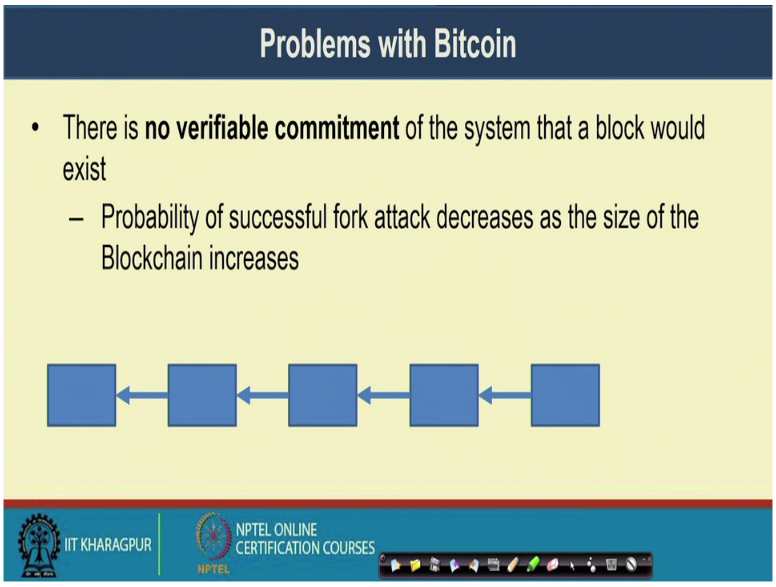
IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, if you remember the requirement for a Blockchain Consensus Protocol that I have mentioned in the last class, the first one is the Byzantine fault tolerance. The protocol need to be Byzantine fault tolerance. The second one is it should provide strong consistency guarantee. The third one is it should scale well to increasing work load and the fourth one was it should scale well to increasing network size.

Now, if you look into the Byzcoin protocol the Byzcoin protocol will satisfy completely the first two requirements. That means, the protocol is Byzantine fault tolerant and it provides strong consistency guarantee by avoiding the faults; the second one, the third and fourth one that two requirements, it makes the nice trade off between the performances. In terms of the transaction that can be supported; transaction throughput that can be supported and the scalability in terms of network size.

So, this is the Byzcoin protocol as I have mentioned it takes the good for from Bitcoin as well as from Bitcoin-NG and combines the concept of proof of works scalability and PBFT scalability; PBFT scalability altogether to increase the scalability in terms of number of nodes as well as in terms of number of transactions that can be supported. So, first let us look into the some problems in Bitcoin which motivates behind the design of this Byzcoin protocol.

(Refer Slide Time: 02:55)



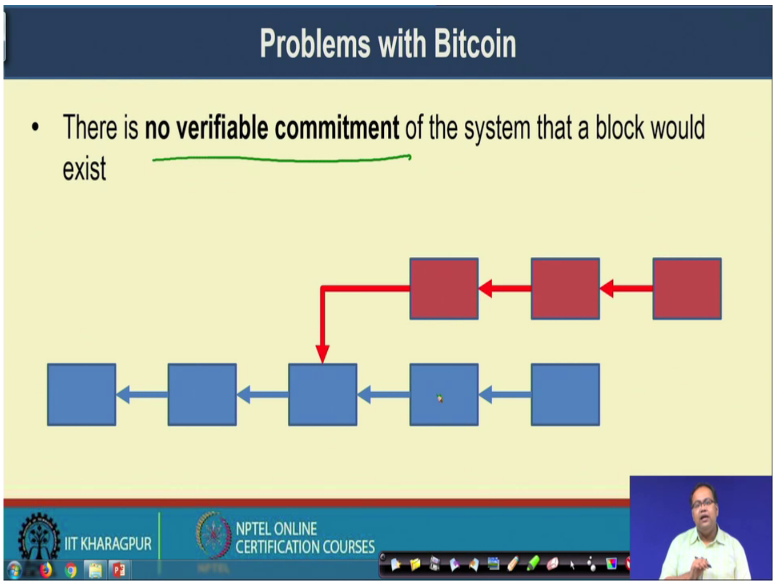
Problems with Bitcoin

- There is **no verifiable commitment** of the system that a block would exist
  - Probability of successful fork attack decreases as the size of the Blockchain increases

IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES

So, in case of Bitcoin, there is no verifiable commitment of the system that a block would exist. So, for example, the Probability of successful fork attack, what we have learned earlier that it decreases as the size of the block chain increases. So, whenever you have a large Blockchain with a longer during that time, you have the lower probability of having a fork attack. But whenever the Blockchain is growing at that time, you always have a possibility of the fork attack and fork attack actually preference some of the valid block to exist in Blockchain.

(Refer Slide Time: 03:35)



Problems with Bitcoin

- There is **no verifiable commitment** of the system that a block would exist

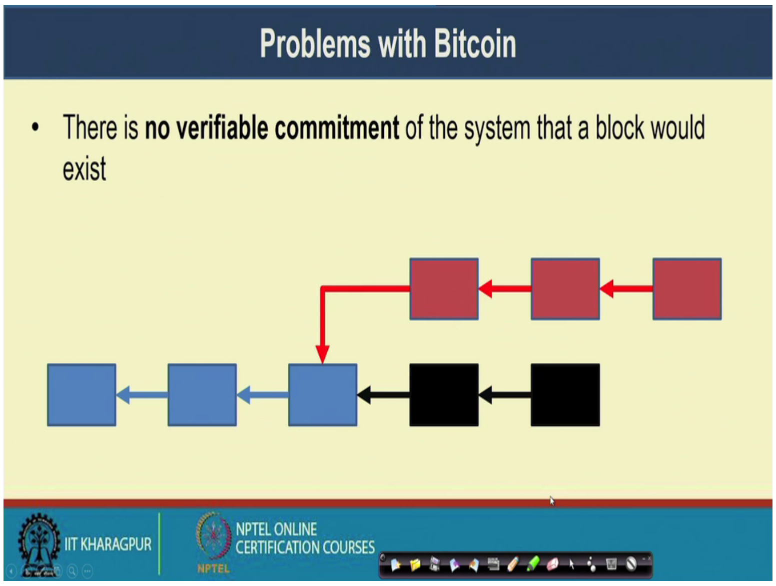
IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES

So, what happens that sometime it may happen that well, earlier say this was your longest chain and after having a fork; it may happen that this rate chain becomes the longest chain.

So, in that case this two nodes, this two blue nodes which were there earlier that they will be considered fork chain and they will not be used in further. So, that is why we say that in Bitcoin, we do not have any verifiable commitment of the system. So, even if you have committed this block and this block seems to be accorded block. But there is no guarantee that this block will be a part of your final Blockchain and that way it will be a kind of waste of resources. So, because you know that Bitcoin anyway utilizes the significant amount of mining power and the depending on the system power and your computation efficiency, you are able to generate the mining block.

Now, that tree is that you have utilised so much of resource in the system, but ultimately the block that you have generated that block is useless, because it has become a part of the fork. So, that is one major problem in Bitcoin.

(Refer Slide Time: 04:56)

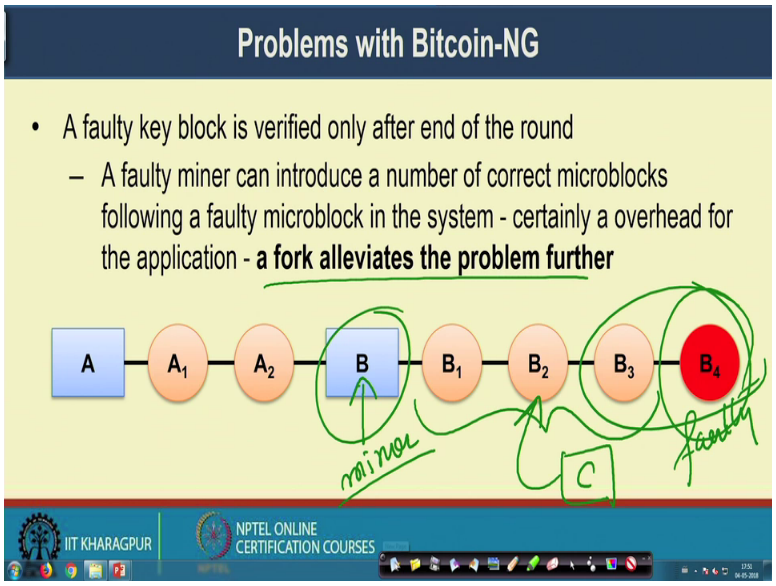


Problems with Bitcoin

- There is **no verifiable commitment** of the system that a block would exist

And in this case as I have mentioned that this two block becomes for and they are not utilised any further.

(Refer Slide Time: 05:03)



Problems with Bitcoin-NG

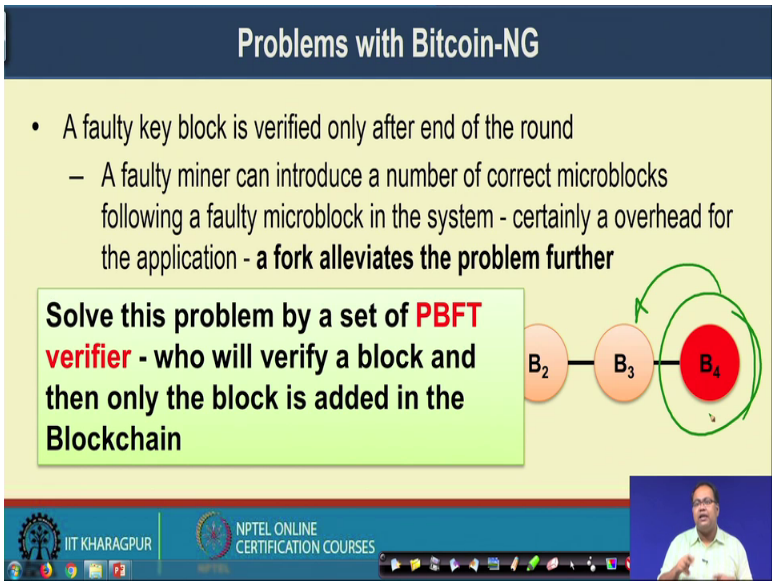
- A faulty key block is verified only after end of the round
  - A faulty miner can introduce a number of correct microblocks following a faulty microblock in the system - certainly a overhead for the application - **a fork alleviates the problem further**

So, that was the problem with Bitcoin and the problem is Bitcoin-NG as I was mentioning in the last class that a faulty key block is verified only after the end of the round. So, a faulty miner, it can introduce a number of correct micro blocks following a faulty microblock in the system. For example, in this case this particular key block, it has been say introduced by a miner. So, this miner has introduced a number of correct micro blocks follow, followed by a faulty microblock. So, this before is a faulty microblock.

So, in this particular case, you can detect that this miner which was here which has generated this particular key block, it is the byzantine miner or it is the malicious miner or a faulty miner only after it has generated this particular block. And the interesting thing is that by the time it has already added this block to this blockchain, because you have asked or you have voted or better to say that the system has given the power to his miner to introduce new block in the system. So, that way the miner has already added the block and after that you are verifying that that particular block is incorrect.

So, that is the case now a fork alleviate this problem further because even if there is a fork. So, it may happen certain forks are invalidated. So, at this stage, if it happens that another node creates the key block here that gets point at here. So, all these valid block becomes invalid.

(Refer Slide Time: 06:48)



Problems with Bitcoin-NG

- A faulty key block is verified only after end of the round
  - A faulty miner can introduce a number of correct microblocks following a faulty microblock in the system - certainly a overhead for the application - **a fork alleviates the problem further**

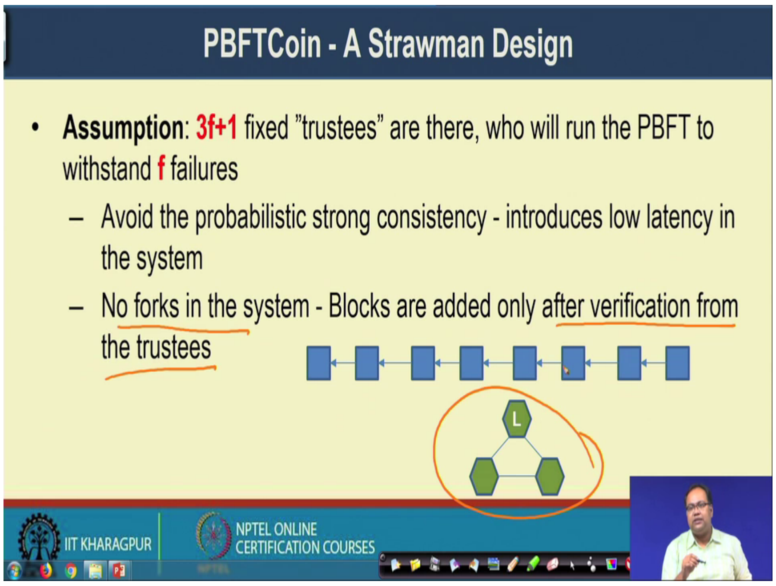
Solve this problem by a set of PBFT verifier - who will verify a block and then only the block is added in the Blockchain

So, we solved this problem by a set of PBFT verifier. So now, that idea that we talked earlier that before putting the block in the blockchain rather than a single block or a single node or a single signer signing the block you make a set of signer will collectively signing the block. So, that is the solution we are introducing here that you solve this problem by a set of PBFT verifier, who will verify your block and then only the block is added in the block chain.

That means, we are not giving the power to a single miner to introduce multiple micro blocks one after another. Although, that miner will be the leader of the system, but a set of witness which will apply this collective signing protocol that I have discussed in the last class. They will apply the collective signing protocol and after applying the collective signing protocol if that particular microblock is verified by multiple such co signers, then only the will get added in the blockchain.

So, in this case if you look into the earlier problem, this block before, it will never get added in this blockchain and at this stage because other signers they will detect that well, see this miner is faulty. So, remove this miner and adopt a new miner by introducing the proof of work mechanism to introduce another new key block in the system. So, that way you can resolve this particular problem in case of blockchain consensus. Well.

(Refer Slide Time: 08:28)



PBFTCoin - A Strawman Design

- **Assumption**: 3f+1 fixed "trustees" are there, who will run the PBFT to withstand f failures
  - Avoid the probabilistic strong consistency - introduces low latency in the system
  - No forks in the system - Blocks are added only after verification from the trustees

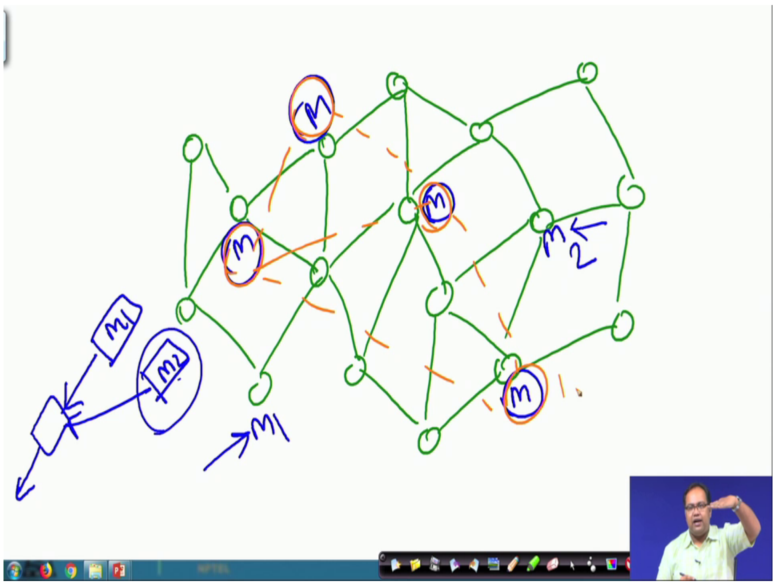
L
IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES

So, let us look into the solution of this in details that how you can use PBFT in this case. So, in that Byzcoin paper initially they have discussed about the straw man design which they call as the PBFT coin. In PBFT coin their assumption is standard to the normal PBFT system that you have a 3f plus 1 fixed “trustees” who are there, who will run the PBFT to withstand f number of failures. So, to sustain f number of failures as we have looked into the concept of PBFT that you would require 3f plus 1 number of nodes in the system.

So, the same thing is use here. So, this avoids the probabilistic strong consistency, it in by introducing low latency in the system. So why: because say whenever you have multiple signers, who are sign in a block and then, only you are adding a block in the system. So, when multiple signer is signing a new block during that time. Well, if there is another block which is being created by another side of the node; it is more likely that well this signing will take some time and by that time one block will propagate in the entire network.

So, that way it avoids this probabilistic strong consistency that well. Now among the set of miners who are there in the world, whenever a new block is being proposed then those miners in the world. They are collectively voting for only one block and that particular block is getting added in your system. So, that way you can avoid this forking in the system.

(Refer Slide Time: 10:33)



So, this entire concepts; so, let me try to explain it again with the help of one example. So, the idea is there, you have now multiple nodes in the network who are connected with each other say these are multiple nodes in the network. They are assumed that they are connected with each other through some means and now among them some of the nodes are working as miners. So, we have these nodes and then some of the nodes were designated as miners node, where I am writing as M. They are the miners in the system.

Now, in case of the earliest case, it may happen that well. This miner is trying to validate the new block or create the new block by using the proof of work mechanism. At the same time this miner is create a new block. So that way, if both of them are able to successfully miner new block at the same time. Then, you have two different block which are pointing to the same pair in block in case of blockchain.

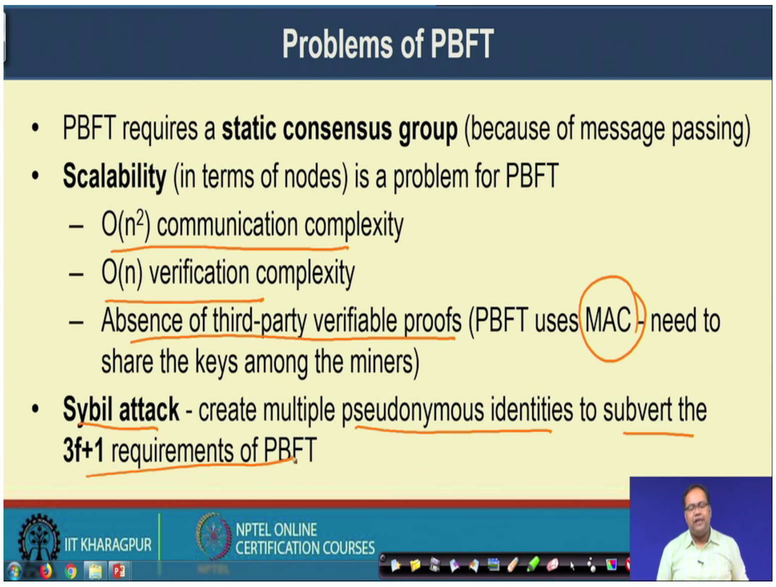
So, now, one miner is pointing to this block. So, this was the previous block, another is pointing the block. So, I am just naming them as M 1 and 2. So, M 1 has created this block and M 2 has created this block and both are valid block pointing to the same pair of block. So, that is why we have the problem of the strong consistency here that well, we are generating certain blocks which can result in a fork in the system.

Now, whenever in case of this particular case what you are doing, you are running a PBFT among this miner. So, out of these miners, you are taking certain miners who are? So, these are the miners which I have marked as circle. So, these are the miner. So, let us use a different colour. So, these are the miners who are authorised to sign the block. So, we are applying this kind of collective signing concept. Now these miners now they will run a PBFT protocol among themselves. So, this miner create another level of hierarchy. So, they will run the PBFT among themselves to add a new block in the system.

So, that way every time, you are making a consensus among the miners that which block is going to add in the system and that way you are preventing multiply block to get added to block chain at the same time. At that way you are preventing the fork to be happening and which in turn gives you a strong consistency support in the system well. So, that is the broad idea here that we are avoiding the forks in the system. So, these blocks we are added only after verification from the trustees.

So, here these are the set of nodes who are working as the miners and this miners are arranged as a set of which we call as a set of trustees, who collectively decides that which particular block need to be added to the system and that particular block is only added to the existing block chain ok. So, that was the solution using PBFT. Now let us see the problem here.

(Refer Slide Time: 13:57)



Problems of PBFT

- PBFT requires a **static consensus group** (because of message passing)
- **Scalability** (in terms of nodes) is a problem for PBFT
  - O(n^2) communication complexity
  - O(n) verification complexity
  - Absence of third-party verifiable proofs (PBFT uses MAC - need to share the keys among the miners)
- **Sybil attack** - create multiple pseudonymous identities to subvert the **3f+1** requirements of PBFT

So, we have certain problems of PBFT which are there which we need to tackle in this particular scenario.

So, the first problem is that PBFT requires a static consensus group. Now, in case of a proof of work based system, we do not want a static consensus group. Why? Because it is open environment and anyone can join as a miner. Now say today I am joining as a miner, if there is a static consensus group; then, what is the guarantee or when I will get the chance to join to that particular consensus group. So, our basic idea that the system will behave like a open environment in case of proof of work what we called as a Permission List Blockchain that concept of permission list blockchain gets wide, if you are thinking about the about the static consensus group. So, that is one of the major limitation for this PBFT type of protocol.

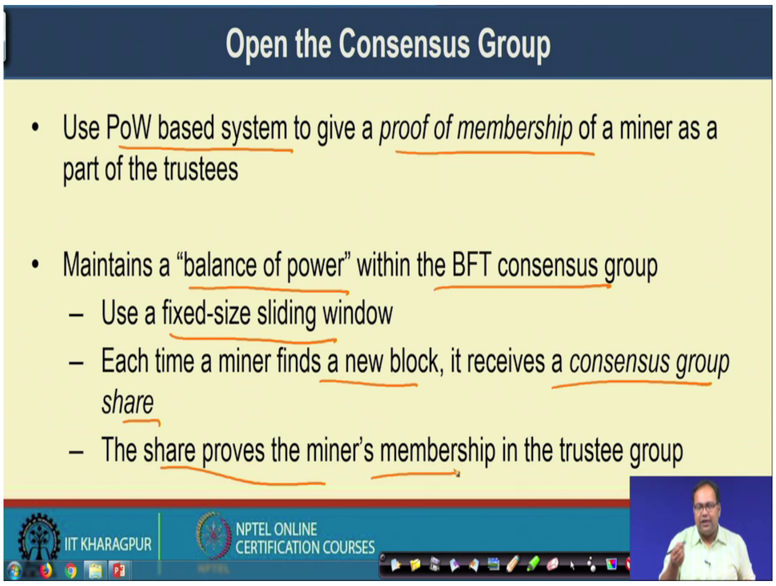
Now, this PBFT type of protocol because of it works on this message passing architecture, it uses a static consensus scope and scalability in terms of number of nodes it is also a problem. Because PBFT it uses O of n square communication complexity and O of n verification complexity. So, if you remember the PBFT mechanism that every node in the prepare and the comic phase every node need to sends the messages to all other nodes in that closed consensus group and they will get that message and only then, they will be able to verify it. So, that way your communication complexities now O of n square; under verification complexity of this O of n because that particular statement or the particular block, need to be verified by all the nodes in the system.

So, that is the two major things and the third problem in PBFT is that you do not have at a third party verifiable proof; that means, if you remember PBFT, PBFT uses Message Authentication Code. So, in case of Message Authentication Code, you have share secret between any two pair of nodes, if you have share secret between any two pair of nodes; that means, if you have some miner in the system done. Then, every pair of nodes every pair of miner, need to have a share secret among themselves which is a overhead obviously, for the entire system and which prevents the system to behave like open system.

And another problem which can be there in case of PBFT is the Sybil attack. Now in case of Sybil attack, a node it can create multiple pseudonymous identities. So, the node can create multiple such identities to subvert that the 3f plus 1 requirement of PBFT. So that means I assume that a malicious node, there is malicious node and that malicious node creates some 100 different identities which apparently looks different.

So, if you are getting the message from those nodes and that Sybil attack becomes prominent even if you are using an open environment, where any malicious node can join in the system and can then subvert the entire system by applying this kind of Sybil attack by faults; the identities multiple faults identities simultaneously. So, to the other node it just looks like that it is receiving the message from 1000 other nodes, but actually 1000 other nodes are pointing to a single malicious node. So, that is the kind of Sybil attack which can happen in case of this kind of PBFT based consensus groups.

(Refer Slide Time: 17:40)



Open the Consensus Group

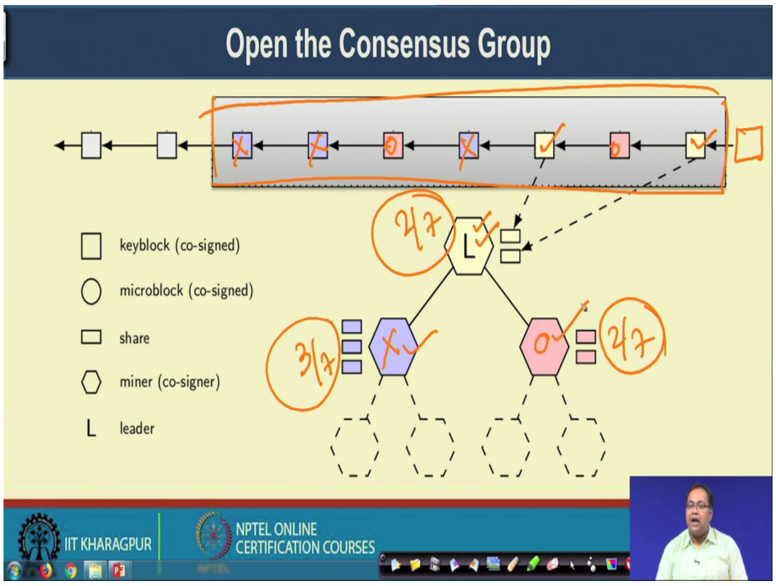
- Use PoW based system to give a *proof of membership* of a miner as a part of the trustees
- Maintains a "balance of power" within the BFT consensus group
  - Use a fixed-size sliding window
  - Each time a miner finds a new block, it receives a *consensus group share*
  - The share proves the miner's membership in the trustee group

IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES

Now, to solve this, our first target would be to open the consensus group. So, how will you open the consensus group? To open the consensus group, we use the proof of work based system. So, we now use the proof of work based system to give a proof of membership of a miner as a part of the trustees; that means we utilise proof of work to determine who will be getting the membership in the trustee board or in the group of trustees.

So, I will explain that with the help of a diagram, but before going to that this particular architecture, it maintain a balance of power within the BFT consensus group. So, how it maintains this balance of power? So, it uses of fixed-size sliding window. Each time a miner finds a new block, it receives a consensus group share and this share provides the miner's membership in the trustee group.

(Refer Slide Time: 18:36)

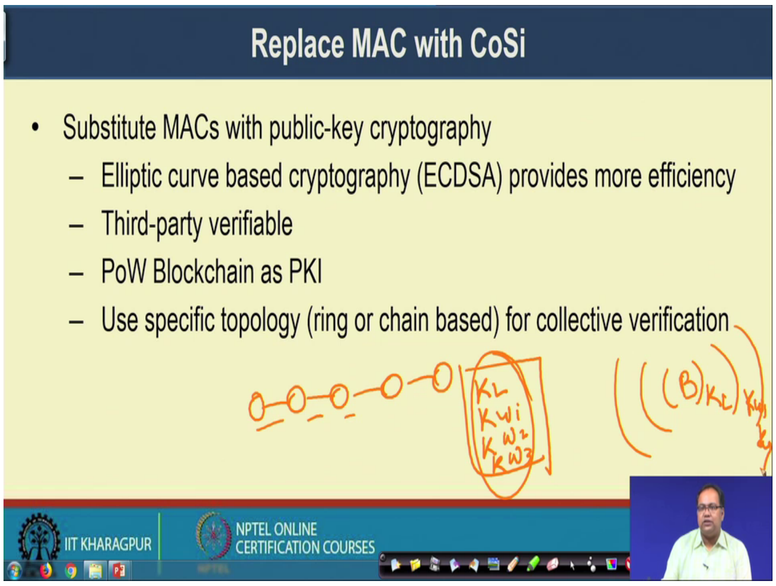


So, let us look into an example to make that him much clear. So, here this is the sliding window, this is my current window where I am considering that, which are the miners who will be there in the trustee group. Now here we see that well this particular miner, it has mined these 2 blocks; then this particular miner, it has mined 3 blocks and this particular miner; it has mined 2 blocks. So, within this window; so, here the window size is 1 2 3 4 5 6 7. With this window size of 7, so it gets the share of 2. This gets the share of 3. This node gets the share of 2. So, this is their voting power or the power to participate in the set of or the group of trustees.

So, in this particular window, who are the in the last window who has mined the maximum number of blocks they become the part of the trustee group to vote further or to sign further the about of validity of a particular block. Now, here whenever you are generating a block say after this whenever a new block will be generated, this new block obviously, will be generated by this leader. Here written as L, but this block need to be collectively signed by this leader along with all these two signers who are there and who are the part of the miner. So that to way, we are distributing this entire power among a set of trustees and at the same time, we are preventing inactive miner to be there in the part of the trustee group.

So, if you put in a inactive miner in that set of trustee group, you have a question about the validity of that miner because that miner has not proven enough or given a proof that it can sufficiently or it has the sufficient resource or sufficient motivation to generate a new block in the system well. So, that is the idea of opening the consensus group. Now your consensus group is not limited to a set of miners rather dynamically based on the miners to a generating the blocks and their share in the current window, you are deciding that which miners will part of this witness group will be signing a new black.

(Refer Slide Time: 21:19)



Replace MAC with CoSi

- Substitute MACs with public-key cryptography
  - Elliptic curve based cryptography (ECDSA) provides more efficiency
  - Third-party verifiable
  - PoW Blockchain as PKI
  - Use specific topology (ring or chain based) for collective verification

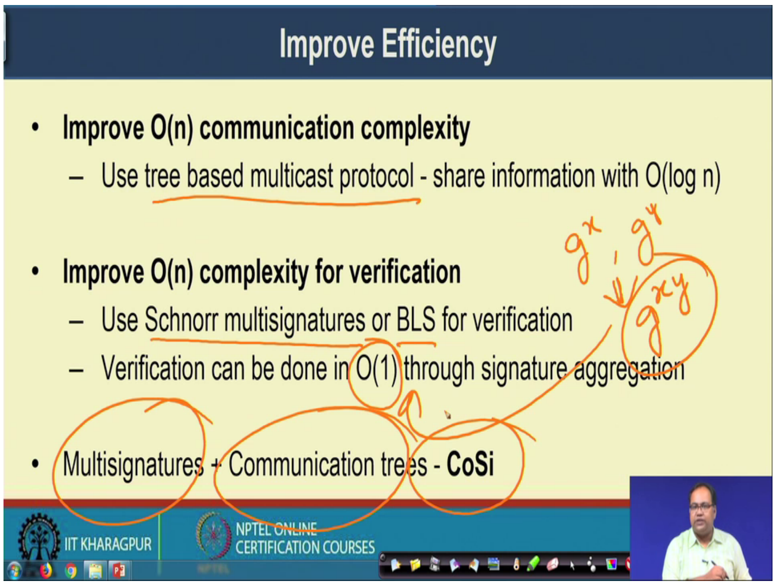
Well, then the second is our problem was with the Message Authentication Code because the Message Authentication Code used private key cryptography, you need a shared secret between any 2 pair of the miners. Now here we are utilising the core size protocol.

So, you substitute them at with public key cryptography, you can utilise this Elliptic curve based cryptography or ECDSA mechanism to provide more efficiency say to use the BLS, you can use normal signature or to have a more scalability in the Cosi-architecture in the tree-based architecture; you can use the BLS based encryption. So, it is start part verifiable, you have a public key. So, once every node has the public key, now in the key block rather than put rather than putting the public key of just a leader, along with that you also put the public key of all the witnesses who has signed those block.

So now, if you remember in the Bitcoin-NG key block structure; so, in the Bitcoin-NG key block, you had the key of the leader. Now apart from the key of the leader you also add a the key of all the witnesses say W 1 W 2 W 3 who has collectively signed a block and here, you can use a specific topology like ring, chain this kind of topology for the collective verification. Say if these miners are arranged in a chain kind of topology, then during the verification, you can pass that particular block to this miner. So, say they have signed.

So, you had this blog B in the collective signing say if it is first encrypted by or signed by say K L. Then it has been signed by K W 1, then it has been signed by K W 2 if that is the case; then, in a reverse order you can just keep on checking the verification. But we can again apply the concept of CoSi here to have a scalable approach of verification.

(Refer Slide Time: 23:34)



Improve Efficiency

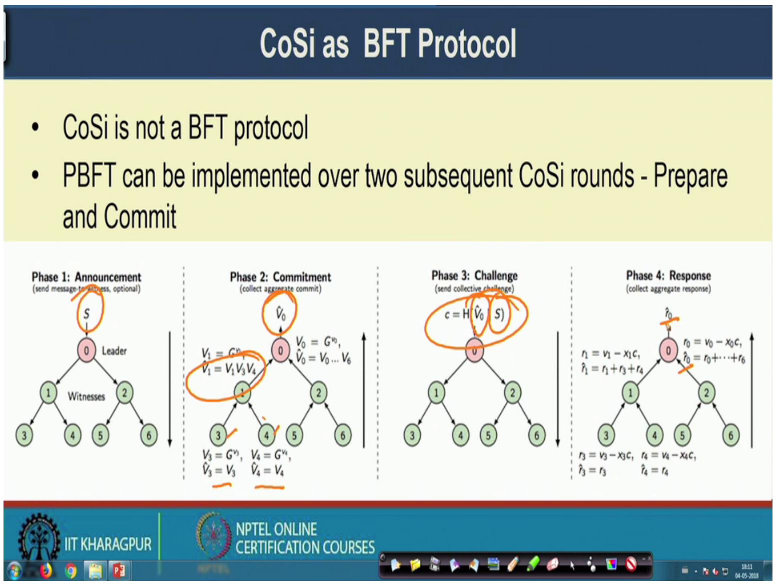
- Improve O(n) communication complexity
  - Use tree based multicast protocol - share information with O(log n)
- Improve O(n) complexity for verification
  - Use Schnorr multisignatures or BLS for verification
  - Verification can be done in O(1) through signature aggregation
- Multisignatures + Communication trees - CoSi

So, let us see how? So, how we are improving efficiency using Cosi? First of all for the communication complexity, your standard PBFT protocol was O of n square communication complexity. Now, we are thinking of well even if you are going a linear complexity by utilizing this chain base system, can I improve the complexity further and the answer is yes, you can do it using the tree based multicast protocol which has been utilised in CoSi. So, you can share the information in the tree with O of n log n message complexity.

And then, for the verification, you can use this Schnorr signature out the BLS for verification, where you have a way of combining all the keys together, all the public keys together and get a final public keys. So, if you remember the BLS encryption if g to the power x and g to the power y are the public key, then from there you can generate the combined public key as g to the power xy. So, with the help of this combine public key, you can directly verify the message without going for the chaining procedure that I have explain just a couple of minutes before and with this combined verification with the message complexity of O of n, O of 1 with a constant message complexity or constant complexity you can verify the signature.

Now, this multisignature and the community tree, together it uses this CoSi protocol and this CoSi gives you a scalable way of doing the signing performing the signing, with O of login complexity and verify the Si with O of 1 complexity. Now if you if you just remember the key block. So, let me go this Bitcoin-NG architecture.

(Refer Slide Time: 25:25)



CoSi as BFT Protocol

- CoSi is not a BFT protocol
- PBFT can be implemented over two subsequent CoSi rounds - Prepare and Commit

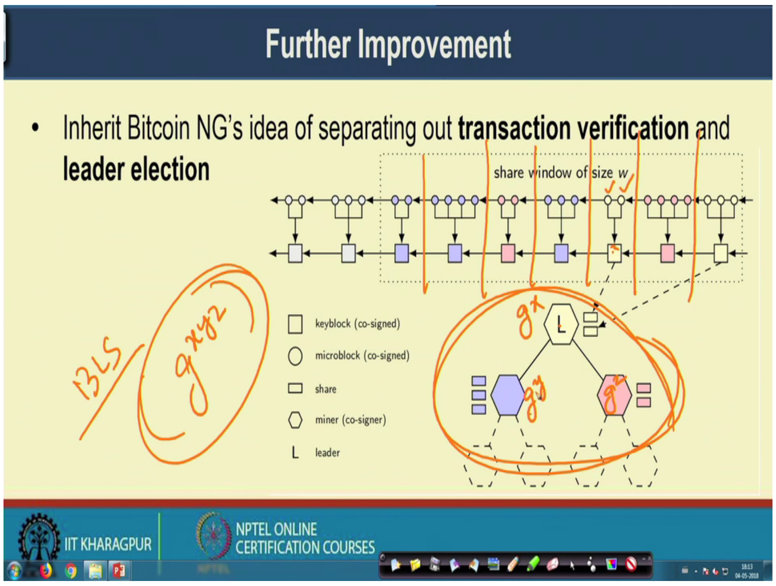
Phase 1: Announcement
(send message-to-witness, optional)
S
Leader
Witnesses
Phase 2: Commitment
(collect aggregate commit)
$\hat{V}_0$
$V_0 = G^{v_0}$,
$\hat{V}_0 = V_0 \dots V_6$
$V_1 = G^{v_1}$,
$\hat{V}_1 = V_1 V_3 V_4$
$V_3 = G^{v_3}$, $\hat{V}_3 = V_3$
$V_4 = G^{v_4}$, $\hat{V}_4 = V_4$
Phase 3: Challenge
(send collective challenge)
$c = H(\hat{V}_0 \| S)$
Phase 4: Response
(collect aggregate response)
$\hat{r}_0$
$r_0 = v_0 - x_0 c$,
$\hat{r}_0 = r_0 + \cdots + r_6$
$r_1 = v_1 - x_1 c$,
$\hat{r}_1 = r_1 + r_3 + r_4$
$r_3 = v_3 - x_3 c$, $\hat{r}_3 = r_3$
$r_4 = v_4 - x_4 c$, $\hat{r}_4 = r_4$
IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES

Well, before going to that as we have mentioned earlier that this CoSi; in CoSi protocol you can implement PBFT with 2 subsequent CoSi rounds with prepare and commit.

So, if you remember the CoSi description that we discussed in the last class that in the first round. So, every round has 2 phases; one downward phase and one upward phase. In the downward phase of first round, the leader it shares the statement to all the nodes. That means, my S and in the upward phase of the first round everyone generates these values, individual values and the combine keys set every round. So, here this nodes generator their own keys, share the public key with one; one computes the combine key.

So, that way these combined keys are created. In the third in the downward phase of the second round; that means, in the phase 3, the leader sense a challenge by using this combine value and the corresponding block statement that you want to validate that gets propagated. And in the final phase; that means, upward phase of the second round, you are generating these individual signatures with the help of R and C's. So, that way you are ovulating this behaviour of PBFT using the 2 downs of CoSi protocol.

(Refer Slide Time: 26:58)



Further Improvement

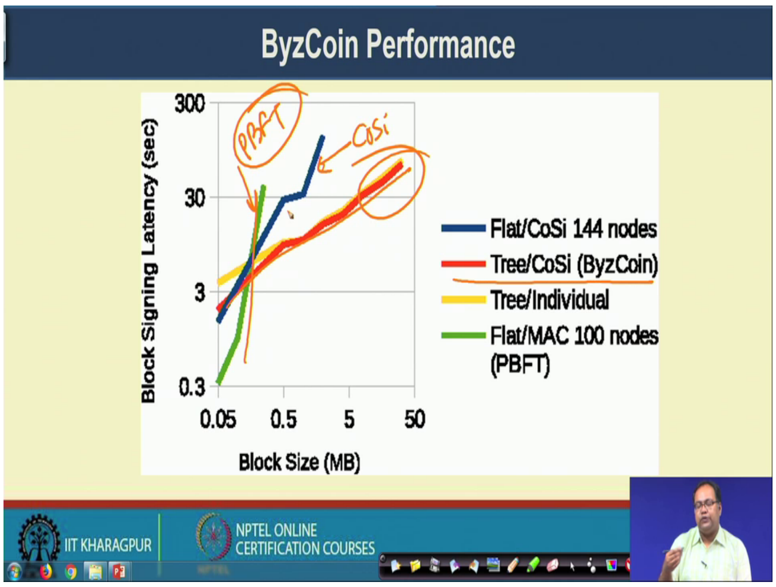
- Inherit Bitcoin NG's idea of separating out **transaction verification** and **leader election**

Now, as I have mentioned, it also takes the good thing from Bitcoin-NG. So, Byzcoin also uses the Bitcoin in this idea of separating out the transaction verification and leader election. So, here again you have the key blocks and a micro blocks and individual key block, now the difference is here. Earlier these individual key blocks, they were containing the key of only 1 node or only 1 miner. Now rather than having the key of only 1 miner, you have the combined key of all these nodes which are collectively signing all the microblocks which are there.

Now, say in different rounds different Bitcoin-NG round or here, I will say the Byzcoin Byzcoin round at every round you say this are the individual rounds. Now at every round among this set of miners, 1 node becomes the leader. So, if I consider this round, in this round this node has became the leader L. This yellow node as became the leader L and it has generated this 2 microblocks and both of these microblocks are now collectively signed by this 3 miners who are the part of this witness committee. Now in the key block rather than putting the key of this node you are. So, if you are using a BLS signature in the earlier case, you have put just the key of say the key of this node is g to the power x, the key of this node is g to the power y and the key of this node is g to the power z.

Now, in this key block, you are putting the combined public if you are using BLS; you are putting the combined public key which is g to the power xyz. Now using this key everyone can verify that and whenever a micro block is generated; by utilising this tree base structure you are verifying you are getting this microblock verified by this witnesses who are there among the miner. And then, only adding the micro block in the blockchain. So, that way it solves the entire problem of Bitcoin-NG as well as the standard Bitcoin ok.

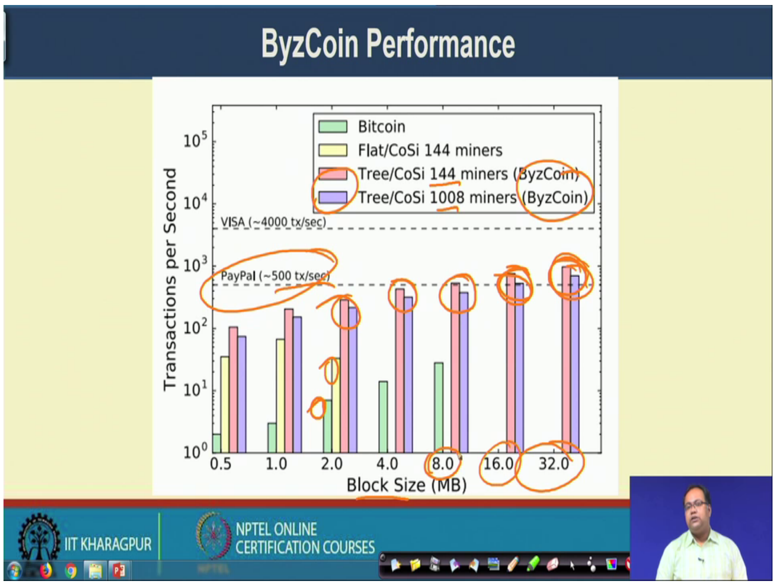
(Refer Slide Time: 29:19)



ByzCoin Performance

So, let us look into the performance aspect. So, in the performance aspect this shows the block signing latency. So, here if you look into this line; so, this is the line corresponds to Byzcoin. So, if you look into this line; this line is as you increase the block size, this line is less than the other protocol. So, this is the standard CoSi or the flat CoSi; that means, it is not utilising the key base structure that are the flat structure in O of n it is performing and a verification and O of n square it is performing the signing. Here, it is done by the tree-based protocol and this particular line corresponds to the standard PBFT protocol. So, you can see that it is significantly reducing the block commitment latency as you are increasing the block sizes.

(Refer Slide Time: 30:15)

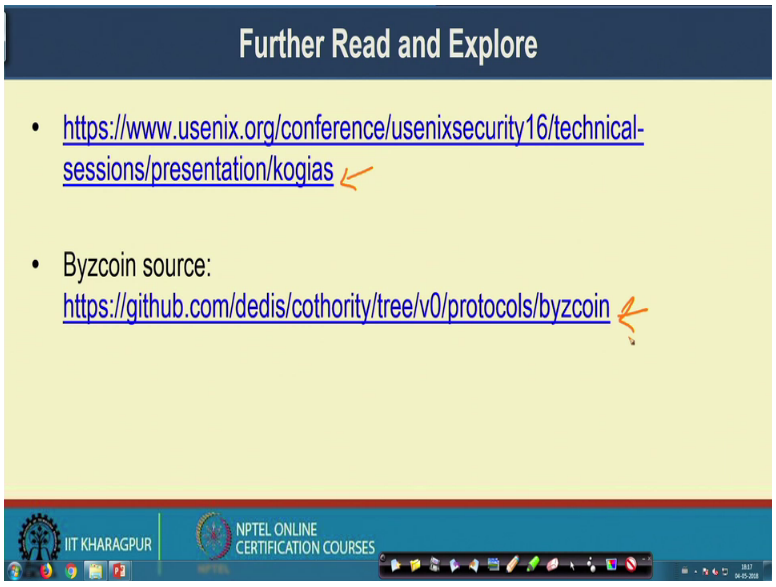


The second thing if you look into the transaction per second which was our major concern. So, if you look into this Byzcoin performance of transaction per second; so, so, this particular lines, this 2 line this lines, this lines they shows the transaction per second for this coin with different numbers of miners and this is the standard CoSi protocol and this is the Bitcoin protocol the standard proof of work.

So, you can see that well this Byzcoin, it is significantly able to improve the transaction per second which is supported by the consensus protocol. So, for Bitcoin the standard block size that they have considered up to 8 MB which is now standard for proof of work based protocol. You can go up to 8 MB of block size. Now if you increase the block size further, you can get around say if you think about the people which the standard transaction platform. So, people supports around 500 transactions per second. So, with Byzcoin, you can reach up to that scale with block size of 16 MB or 32 MB.

So, that way we see that well Byzcoin was able to significantly improve the performance in terms of transaction per second that can be supported in the system and because it is a proof of work based protocol, again you can support large number of nodes or you can work on top of large number of nodes ok.

(Refer Slide Time: 32:10)


Further Read and Explore

- https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/kogias
- Byzcoin source: https://github.com/dedis/cothority/tree/v0/protocols/byzcoin

IIT KHARAGPUR
NPTEL ONLINE CERTIFICATION COURSES

So, that is about the Byzcoin protocol; 2 interesting things that I want to share with you. So, this particular the first preference the first preference, it provides the original people of Byzcoin. So, the original talk from the authors of the papers are also available there. So, I suggest all of you to go through that and you also have the Byzcoin source shows publicly available. So, you can run of Byzcoin network and look into the different performance aspect of Byzcoin by exploring that particular architecture.

That is all about the Byzcoin. In the next class, we will discuss about some of the other protocols which recently people or the researchers have developed to tackle this scalability problem in Blockchain based network.

Thank you all for attending this course.