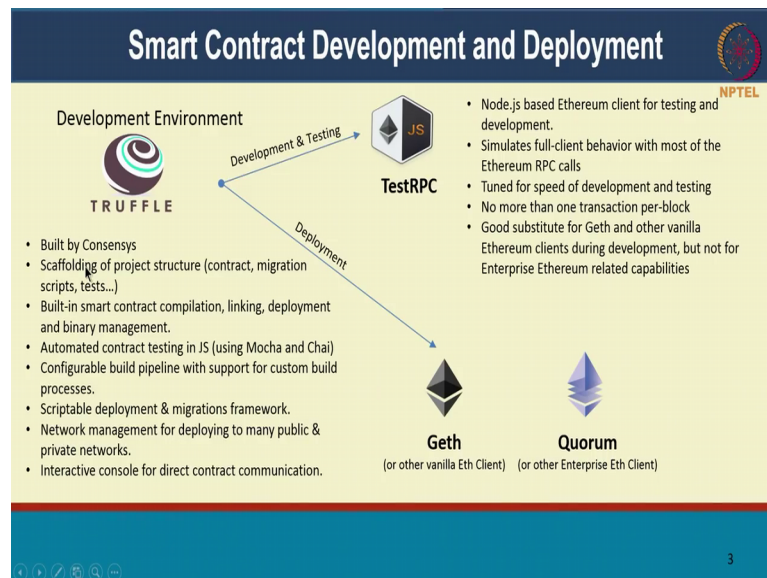


Blockchains Architecture, Design and Use Cases
Prof. Sandip Chakraborty
Prof. Praveen Jayachandran
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture – 57
Comparing Ecosystems – Ethereum development tools and Quorum

Hello everyone, welcome back to our next lecture, we are looking at different blockchain platforms with specifically last lecture, we looked at Ethereum, now we will go into some of the Ethereum developer tools and also a permissioned version of Ethereum Quorum specifically in this lecture. So, Ethereum has over the last maybe 2 3 years, it is really developed in terms of the ecosystem itself, there are plenty of developer tools available. We just going talk about maybe a few of them, there are a lot more that are available, I will have a pointer towards the end of the lecture, in our front reading section on some of the large number of tools that have come up to help developers code on Ethereum.

(Refer Slide Time: 01:06)



So, Truffle is really a development environment hence, become popular over a period of time it is built by consensus and it helps give you a scaffolding of project structure. So, you can have contracts, scripts, tests all as a project almost then, you can have it has support for built in smart contract compilation linking in deployment, you can manage

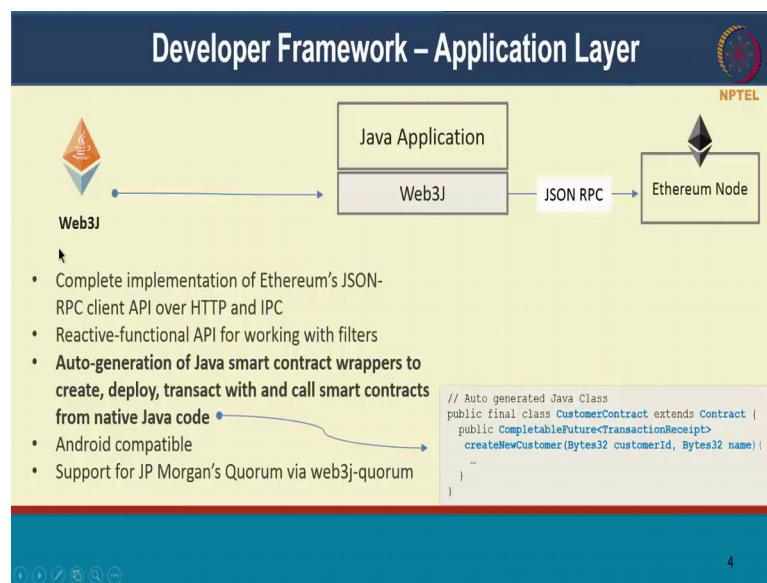
the binaries for your smart contract, it gives you automatic test framework in Java script. It gives you a configurable built in test pipeline, it gives you a scriptable deployment framework and it also gives you the ability to manage your network deploy into many public and private networks.

So, gives you a host of capabilities all around managing your smart contracts, your development at providing an environment for that giving you the build and test pipelines needed for you to deploy into different networks. And the deployment could go into for instance, you can deployment can connect to go Ethereum client, you just Geth, it is very popular vanilla Ethereum client or you could also connect to other Ethereum variants such as Quorum right.

Test RPC is node JS based Ethereum client for testing and development, it sub basically simulates a full client behavior as if you were running on the actual public Ethereum network, it actually simulates that behavior and it helps really improve the speed, if testing in development and testing because, you have a single node almost with yourself for development and testing. But as a so some limitations, it can only create one transaction per block.

So, you can you cannot have the full, let us say full blown on proof of work style multiple transactions, created in a block and people are actually mining all that is not there. This is just, this not contract execution in an EVM, box right and it is services a good subsist substitute for Geth or other Ethereum clients during development phase, but it does not have any enterprise Ethereum related capabilities.

(Refer Slide Time: 03:15)



So, web 3J is again a developer framework and it gives you a complete implementation of Ethereum's JS, JSON, RPC client, but it provides this over HTTP and inter process communication, it has a reactive functional API, if you are familiar with reactive and what did a cool thing, you can do is it can auto generate Java smart contract wrappers for your contract, for your create deploy transactions to call your smart contracts on blockchain.

So, it gives you the Java application wrappers to invoke, your create deploy transactions on your smart contracts. It is also android compatible and is also again supports Quorum right.

(Refer Slide Time: 04:07)



Enterprise Ethereum Alliance

Mission: "Learn from and build upon the only smart contract supporting blockchain currently running in real-world production and to define enterprise-grade software capable of handling the most complex, highly demanding applications at the speed of business."

Focus: To support enterprise use cases

- Enterprise security and data privacy
- Permissioning
- High throughput
- Pluggable architecture

Launch Members

<http://entethalliance.org/>

The slide displays a grid of logos for various member organizations, including Accenture, IBM, ANULI, BBVA, NPTEL, Brainbot, BNY Mellon, CME Group, ConsenSys, Chronicle, Credit Suisse, CRYPTAPE, ING, The Institutes, Intel, J.P. Morgan, Monax, Microsoft, Nuco, Thomson Reuters, Santander, String, Telindus, Tendermint, UBS, VidRo, and Wipro.

So, the enterprise Ethereum alliance, I mentioned last lecture was formed in March of 2017, the main objective was that there was a group of entities that realized that Ethereum, while it had the cool properties, it had a public network and there was an ether inherent cryptocurrency.

It was raw really not well suited for enterprise applications for in terms of scale, in terms of security and privacy, in terms of throughput, in terms of plug ability, in terms of what different enterprise applications need. There were, there were a lot of shortcomings with the public Ethereum itself.

So, they want it to use some of the main constructs from Ethereum, but they wanted to change it in certain ways actually fork Ethereum, but then modify it for specifically use in enterprise applications, but they wanted to do this in such a way that, they can still leverage a lot of the innovation capabilities. For instance, they retained the EVM itself that was the intent. So, all the contracts that developed for the public Ethereum could also run on the private networks and you can also leverage many of the developer tools and the ecosystem the developer community that was creating all of these tools, those can also be used with the your enterprise variants right.

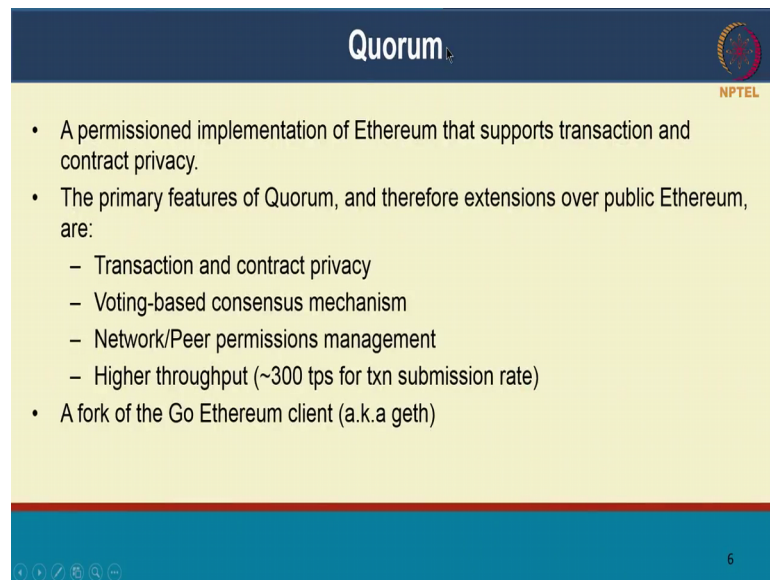
So, what are some of the main critical properties, they were looking for? They were looking for security and data privacy, we have looked at how Hyperledger Fabric provides many of these capabilities, but they are trying to bring these capabilities into an

enterprise version of Ethereum. They want to bring in permissioning so, the public Ethereum is permissionless, which means that anybody absolutely, anybody can come and join the network can see the transactions happening, but in enterprise applications we wanted to be restricted to an authorized set of permissioned entities, only those that are performing our part of a transaction.

So, permissioning is being brought in, they also want to improve throughput. So for instance, one of the things they have done is, they have gotten rid of proof of what they are leveraging, other notions of consensus, we will talk about it when we talk about Quorum and also trying to bring in a pluggable architecture right. So, I want to be able to plug in different consensus algorithms, I want to be able to plug in different data bases, there are different notions, that are different plug ability notions that are being brought in right.

So, they are very trying to modify Ethereum, in some way the public Ethereum for enterprise requirements, such as security privacy and scale and if as you can see there are many well known large corporations both technology companies as well as financial companies, that are coming together in this Ethereum alliance. As of today, at least I have not seen an actual codebase put out by Ethereum alliance, they are trying to put out us a policy statement or a statement of purpose, in some sense of what are some of the things that they want to do? And in and it and Quorum appears to be the main platform, that seems to be adhering to these specifications and there is a community building around Quorum.

(Refer Slide Time: 07:28)

A presentation slide for Quorum, an NPTEL course. The slide has a dark blue header with the 'Quorum' logo and the NPTEL logo. The main content area is yellow and contains a bulleted list of features. The footer is blue with navigation icons and the number '6'.

- A permissioned implementation of Ethereum that supports transaction and contract privacy.
- The primary features of Quorum, and therefore extensions over public Ethereum, are:
 - Transaction and contract privacy
 - Voting-based consensus mechanism
 - Network/Peer permissions management
 - Higher throughput (~300 tps for txn submission rate)
- A fork of the Go Ethereum client (a.k.a geth)

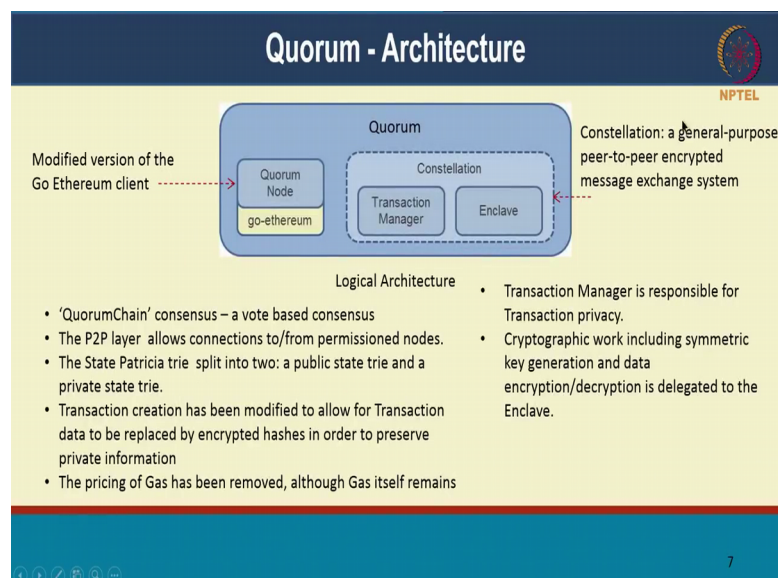
So, Quorum itself is coming to a Quorum as a blocking platform, it is a variant of Ethereum, it is a permission implementation that is geared towards transaction and contract privacy right. So, the primary features of Quorum it is an extension of the public key theorem, it actually leverages the Ethereum virtual machine asses. So, you can code in solidity or some of the other languages that is available, the same bytecode is maintained. It gives you notions of transaction and contract privacy, it has a pluggable consensus mechanism involved.

So, they have gotten rid of proof of work completely, they brought in a voting based consensus algorithm and they also have support for raft consensus and they brought in permissions management. So, all the peers and users of Quorum will be permissioned so they will have an identity on the block chain finally, they are also looking for higher throughput of the order of about 300 transactions a second and this is the transaction submission rate right.

This is really a fork of the Go Ethereum client. So, they want to be compatible with that going forward. So, that as the get client as the get client goes through versions new capabilities are added, we talked about some of the capabilities with secure messaging with charting and so on, as they get added to Geth, Quorum wants to be able to leverage that. So, they are keeping it as a 4 for their key they making sure at least portions of the code are not too different from what they are currently, they are not making too many

changes, they are only changing the consensus mechanism and they are there are a few other changes, we will talk about those.

(Refer Slide Time: 09:13)



Let us look at the Quorum architecture itself. So, there is a Quorum node, which is the go eth client. So, this is a slightly modified version of the go eth client, but right they are trying to make sure, this is not too looking too different so, that future innovations into the go eth client can be brought in quickly. And they have a notion of constellation, which is really a general purpose peer to peer encrypted message exchange system. So, this allows you; allows spheres to exchange messages in a privacy preserving manner and this constellation is also responsible for transaction execution.

And the different Quorum nodes adopt our notion of consensus and the consensus itself as, I said is pluggable there are this Quorum chain consensus, we will talk about that and there is also raft consensus that is supported and there is a peer to peer layer, that supports connection between these peers for exchange of messages. So, that is through constellation and there is a state Patricia trie, that split into 2 right.

So specifically, they support both public state as well as private state, we will talk about that again in the next slide on how privacy of contract and transaction information is supported in Quorum. And this transaction creation has been modified to allow only the hash or maybe even the encrypted hash of data to be recorded in the transaction and the private information is in fact, kept private.

S, in Quorum, there is no pricing involved, but gas itself remains right. So, they have kept gas itself. So, that not to make too many code changes and the transaction manager is responsible for transaction privacy again, I will when I will talk about that when we talk about the public state and the private state and the transaction manager does not hold any secret information.

Quorum – Transaction Processing

NPTEL

Quorum supports public and private Transactions

Public Transactions

- When a transaction is public, each participant will execute the same contract code and their underlying public StateDBs will be updated accordingly.

Transaction Privacy

- Private transactions are only visible to participants whose public keys are specified in the `privateFor` field
- The Quorum Node propagates the transaction to the rest of the network, after replacing payload with a hash of the encrypted payload it receives from Constellation.
- Participants party to the private transaction will replace the hash with the original payload before calling EVM for execution, and their private StateDBs is updated accordingly.

The diagram illustrates the transaction processing flow in Quorum across three parties (Party A, Party B, and Party C) and a Constellation node. Each party has a Client Node and a Transaction Manager. The process starts with a client sending a 'Send Tx' message to its local Transaction Manager. The Transaction Manager then sends a 'Tx Hash' to the Constellation node. The Constellation node responds with a 'TXID Confirmation' and a 'TX Hash'. The Transaction Manager then sends a 'Tx Payload Request' to the Client Node. The Client Node responds with a 'Tx Payload' (which may be encrypted or hashed). The Transaction Manager then sends a 'Tx Payload' to the Constellation node. The Constellation node responds with a 'TXID Confirmation' and a 'TX Hash'. Finally, the Transaction Manager sends a 'Tx Response' back to the Client Node. The diagram shows how the transaction is propagated through the network and how the final response is received by the client.

8

But let me give you the (Refer Time: 12:15) of it right, how are they achieving transaction and transaction data privacy? Quorum supports 2 kinds of transactions,

public transactions and private transactions, the 2 cannot be mixed right. For instance, I cannot have a transaction that handles both public state as well as private state. Now the public transactions are similar to a theorem right, when a transaction is public each participant will execute the same contract on the same data. And all the data, modified data is going to get recorded in the state database and will be recorded on the blockchain.

So, the state data, the public state data will be in the transaction and is recorded on the blockchain. So, that is public for everyone to view, everyone in the network there is a Quorum does not have the notion of channels similar to fabric. So, in the network everyone sees all the transactions, the only thing you can keep private is the data itself, there is some data that you can hold private, but the transactions themselves there is no notion. So, all the network participants will see the public transactions.

Now, the separately from the public transactions, you can also have private transactions, that are only visible to a subset of the participants in the network right. So, the other nodes in the network will see that some transaction exists, but they would not see the details of it right.

So, it will have information about, it will basically, have a private for field and I will mention the entities that the transaction is private for. So, one privacy one notion of privacy you do not get with Quorum is that private for field itself is public. So, people will know. So, let us say there are 15 participants in the network, there are 3 participants, who are trying to do a private transaction. The entire network of 15 participants will know that these 3 entities are doing something privately and they do not want to disclose that to the rest of the network.

What they are doing? Will be kept private, but the fact that they are doing something privately is known to everyone. So, that cannot be of scatted fabric for instance, it is possible to hide that also, but with Quorum it is not possible to hide that.

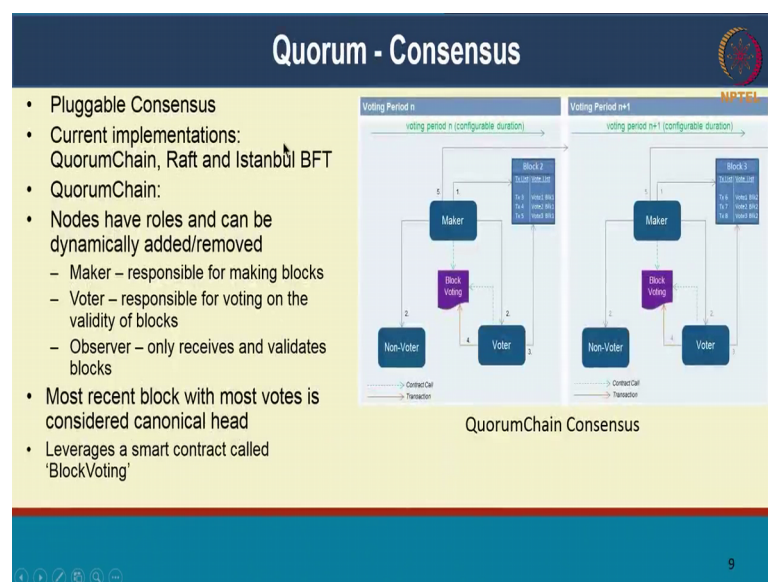
So, what how does this private transactions work? So, in the private transactions as I mentioned there is a private for field that mentions a list of entities to which this transaction is private. So, only those entities will get the transaction data and the core of node propagates the transaction to the rest of the network. So, what it does? It replaces the payload, which is intended to be private in this transaction, it replaces it with just a hash and the encrypted payload is sent to the constellation.

So, the constellation will then constellation in each node. So, if you are an authorized part is the entity of this private transaction. So, if you are part of the private far field, then within your constellation you will be able to decrypt it, because you have the key, you will be able to decrypt it and execute the transaction and see the decrypted content.

But all the others will only receive a hash, in some sense this is similar to site DB and fabric, because you can now in fabric, you can have private collections and you can have a subset of entities in a channel part of a collection. This is similar to that, except that side DB in fabric allows you to handle data pertaining to multiple. So, one transaction can actually have data pertaining to multiple collections, each collection having a different set of participants and you can also have public data within it.

Whereas here, a transaction has to be either public or private and if it is private, all the data within that transaction is private for that set of entities. So, you cannot have multiple collections, one transaction of operating on multiple collections that is the really the difference between Quorum, the privacy notion in Quorum and the privacy notion in fabric.

(Refer Slide Time: 16:42)



But in some sense because, there is a restricted functionality you could also imagine that maybe, it might be easier to achieve this privacy you do not have to work with the many policies that are there in fabric, but I have not personally tried out Quorum for myself.


So, I am probably not a good judge of that. Let us let us all we looked at the transaction processing in Quorum, how public and private transactions are supported. So now, let us look at consensus, how does consensus work? So like I mentioned they have removed the proof of work consensus in Ethereum and they have replaced it with a pluggable notion of consensus. \

So, today there are some implementations of this consensus that are that is available, one of them is Quorum chain. So, it is their own consensus algorithm. So, it works based on voting. So, each peer can have one or more rules it can be a maker, which is the peer is responsible for creating the block, there could be a voter, which is responsible for voting on, whether a block is valid or not. And there could be observers, who are just verifying or just validating whether block is valid.

So, the most recent block with the most votes is considered the head of the chain right. So, there could be (Refer Time: 17:41) of course, but as people vote on it, the head will get determined and hopefully a single chain of transactions will emerge. So, there is a inherently though, this whole voting mechanism is also implemented as a smart contract, it is called block voting and that smart contract is loaded in a pre specified address and that is part of the genesis block..

(Refer Slide Time: 18:15)

Fun Reading



- Ethereum developer tools: <https://github.com/ethereum/homestead-guide/blob/master/source/contracts-and-transactions/developer-tools.rst>
- Quorum Whitepaper: [https://github.com/jpmorganchase/quorum-docs/raw/master/Quorum Whitepaper v0.1.pdf](https://github.com/jpmorganchase/quorum-docs/raw/master/Quorum%20Whitepaper%20v0.1.pdf)
- Quorum overview documentation: <https://github.com/jpmorganchase/quorum/wiki/Quorum-Overview>
- RAFT Consensus: In Search of an Understandable Consensus Algorithm, <https://ramcloud.stanford.edu/wiki/download/attachments/11370504/raft.pdf>

10

So, it is already the genesis block itself comes with a particular transaction sorry, particular smart contract that is the loaded and that smart contract is responsible for consensus.

So, that is a quick overview of Quorum and some of the developer tools available with Ethereum and those developed to most of those developer tools also work with Quorum. So, for a good list of the developer tools for Ethereum, you can look up the Ethereum Github repo itself, there is a lot of these reb developer tools you can check those out. To the Quorum, white paper is an interesting read it is very high level overview and you can check that out.

The overview documentation has a good bit of information. So, I would encourage you to read that is well. So, I briefly mentioned raft consensus. So, raft is really a fairly old protocol, that was built as a alternative to Praxis. So, if you have seen Praxis, Praxis is a crash fault tolerant algorithm, consensus algorithm and people will believe that Praxes is very hard to understand.

So, raft actually was devised as an alternative to praxes. So, that it can actually people can actually understand, how this consensus a consensus algorithm works? So, you can check out that the raft paper itself. So, it is a good read on how that consensus works? Ok. So, that brings us to the end of the 2 part series on Ethereum and Quorum. So, in the next couple of lectures, we look at corda and some of the interesting properties and capabilities corda provides.

Thank you; see you soon in the next lecture.