A Permissioned ISPs Blockchain For The End to End Quality of Service Enhancement

Michele Scarlato, Cristian Perra, *Heung No Lee,
e-mail : michele.scarlato@diee.unica.it, cperra@ieee.org, heungno@gist.ac.kr.

Department of Electrical and Electronics Engineering Cagliari University,
****Gwangju Institute of Science and Technology.***

Abstract

In this work we designed a permissioned ISP blockchain for the end to end (e2e) quality of service (QoS) enhancement that shall be framed in an Internet Service Provider (ISP) cooperative scenario. The designed blockchain is able to store users' connection data, in those connections where a lack of Quality of Service is experienced. Those data will be retrieved from the Networks Management System (NMS), and in some specific cases can be retrieved from the Software Defined Networks (SDN) controllers, of the ISPs chaining scenario that we recently proposed.

We designed the blockchain and we defined the smart contract by using Hyperledger Fabric.

I. Introduction

The need for a cooperation among ISPs, in order to provide the OTT's final users with a better QoS, has been highlighted in several works.

Ahmad et al. [1] proposed a collaboration model where a strategy is described, to increase the revenue generation and QoE for the OTTs and the ISPs.

In our previous work [2] a mechanism to provide a communication among OTT and ISPs has been provided. REST APIs have been proposed in order to create a channel of communication among OTT and ISPs, with the purpose of improving the end to end QoS of the final users.

In this work we designed a permissioned blockchain for an ISPs cooperative scenario where user connection data is stored. We chose to implement the designed blockchain by using Hyperledger Fabric \cite{cachin2016architecture}, above all because it permits the use of channels and the creation of separate ledgers.

The paper is organized as follows. The design of the blockchain is provided in Section II. In section III some aspects are discussed related to the embryonic deployment of the Blockchain. Section IV draws the conclusions of this paper and the future direction of the research.

II. Design of the permissioned Blockchain

In our design we considered that certain data can be seen by all the participants, in order to provide the required improvement in the QoS delivered to the final users, but other data can be confidential and hence has to be exchanged just by the OTT and some specific ISPs. To do this, we used the channels approach architecture of Hyperledger Fabric.

This approach makes use of established channels to a subset of participants in which only a determined set of transactions can be visualized, namely those related to the participants of a certain channel. This is thought as a network overlay, above the underlying blockchain.

Our architecture is composed of a network consisting mainly of: clients, peers and ordering services nodes, smart contracts, channels, ledgers, and Fabric Certificate Authorities.

Nodes, in general, are the entities that are

communicating in the blockchain. They can be considered just as a logical function, being possible to run multiple instances of them in the same physical server. It is important to consider how they are grouped in trusted domains, and how they are associated to the logical entities that are able to control them.

The clients, also called submitting clients, are designed to be the representation of the end-users. They are in fact those who submit an actual transaction invocation to the endorsers, and those in charge of broadcasting transaction proposals to the ordering service. In order to interact with the blockchain they are connected to a peer node.

The peer nodes are network entities owned and managed by the members of the blockchain, that maintain the state and a copy of the ledger, besides performing read/write operation. This particular kind of node receives ordered state updates in forms of blocks from the ordering service. They also can play the special role of endorsers. This special function is performed towards a particular portion of a smart contract and consists into endorsing a transaction before it is committed. Inside the smart contract an endorsement policy can be defined, which refers to a specific set of endorsing peers, where the necessary and sufficient conditions for a valid transaction endorsement are defined.

In the designed architecture the SDN Controllers are the peer nodes, to which it is also delegated the role of endorsers, being themselves the entities able to enforce policies inside their network.

The ordering services are the ones in charge of providing a shared communication channel to peers and node, and are able to broadcast messages to all the peers, in the same logical order in which they have been output.

The access to the channels is permissioned, and they can be considered as a further private blockchain overlaid to the main blockchain, where data isolation and confidentiality are provided. Their definition is made by the configuration block, that resides in the channel specific ledger itself.

The smart contracts, also called chaincode, are programmable containers where it is possible to define the functions that will permit the interaction with the ledgers.

The number of ledgers will vary depending on the number of the channels, considering one per channel. Every ledger is composed of the blockchain, where the immutable and sequenced records are stored in blocks, and the state database, where the current fabric state is maintained.

The Hyperledger Fabric Certificate Authority (CA) issues the certificates that will authenticate OTT, ISPs and users to the network.

This work proposes the basis and a first approach to a chained collaboration between OTTs and ISPs, focused on the enhancement of QoS by exchanging and balancing connection parameter values, so that the OTTs provide their customers with enhanced service by a reduced price or even free, the ISPs provide the customer with better QoE while reducing parts of their SLAs as a counterpart, and finally the customer solves a problem with the service being consumed almost transparently.

The RESTful methods proposed in table 1 [2] are envisioned for the use case in which RTT is requested to be reduced by accepting a downgrade in the available bandwidth as a counterpart. We find these calls interesting for services like online videogames or videoconferencing systems, among others.

2.1 The Membership Operation Architecture

The access to the system is allowed through a trusted Membership Service Provider (MSP), whose task is to deploy a membership operation architecture performing all the cryptographic mechanisms and protocols requested for the release and the validation of the certificates and the user authentication.

Two or more nodes are able to create a channel, and only these nodes are able to access to the data transacted, thus providing a certain level of privacy and confidentiality that is bounded to the security of the channel. Upon every created channel, the participants will be able to create a separate ledger of transactions.

REST Call	Description
POST /qos/rttreduction?network=IPv4/netmask&maxrtt=maxdesiredrtt	With this POST the gaming OTT asks the ISP to reduce the Round Trip Time, passing the IP of the host or the network of hosts, and passing the maximum RTT desired.
GET $/qos/minmaxrtt?network = IPv4/netmask$	With this GET it is requested to the ISP the minimum and the maximum RTT for one particular host, or network of hosts.
GET $/qos/availablebw?network = IPv4/netmask$	With this GET it is requested to the ISP the bandwidth available for one particular host, or network of hosts.
POST $/qos/exch - av - bw4rtt - red$	With this POST the OTT wants to exchange the available bandwidth with a reduction of the RTT.

Table 1 A rest API for ISP communication in QoS

The importance of this feature is highlighted by the particular nature of the participants to our Blockchain, where all of them are potential competitors, but where their collaboration with the other entities involved is crucial for the delivery of a better QoS to the final users.

2.2 The Channel Configuration.

Every channel contains its own shared ledger that has to be considered as an overlay used for data isolation and confidentiality. Only authenticated entities can access to the information transacted inside the channel.

The configuration block defines the channel, and contains a single configuration. When the configuration of the channel changes, a new configuration block will be forged, this process is called configuration transaction. The first of these blocks, where the initial configuration is used for the bootstrapping of the channel, is called Genesis block.

The configuration blocks store data related to which organizations are members of the channel and which channel access policies have to be applied. Also the block batch size is defined inside of it.

2.3 The Shared Ledger

The Hyperledger Fabric ledger subsystem is composed of two elements, named the world state and the transaction log. Every node owns a copy of the ledger to which it is belonging.

The world state is the database of the ledger where it is described its state at a given point in time, while in the transaction log, all the transactions belonging to the current value of the world state will be recorded. It can be considered the update history registry for the world state. The whole ledger results to be a combination of these two components.

2.4. The Smart Contract

The smart contract that we defined, will be used by the External Service Application (ESA) [2] where our APIs will be implemented or by the SDN controller itself whenever the APIs will be integrated in its Northbound interface. By calling the functions defined in the smart contract, the exchanged information will be stored as a transaction into a block of the blockchain. In Hyperledger Fabric, nowadays the chaincode can be written in Go language or in Node.js. For our development we chose to use Go.

The smart contract that we designed will be used to store the responses to REST calls shown in table 1, where 4 methods are described, which can be used by OTTs in order to retrieve information regarding the user connection in a certain ISP's network. We designed our smart contract to store in the ledger the responses to these calls.

The logic of functioning can be represented, as shown in figure 1, by this example: the call to the method GET*/qos/minmaxrtt?network= IPv4/netmask* is triggered by an OTT towards a certain ISP, asking for information regarding the minimum or the maximum RTT for a particular host, or network of hosts, whose traffic is crossing the ISP's network.

This call will be performed by an application that has to be developed with the Hyperledger Fabric SDK, that will interact via a network endpoint with the smart contract. The application will propose an update transaction.

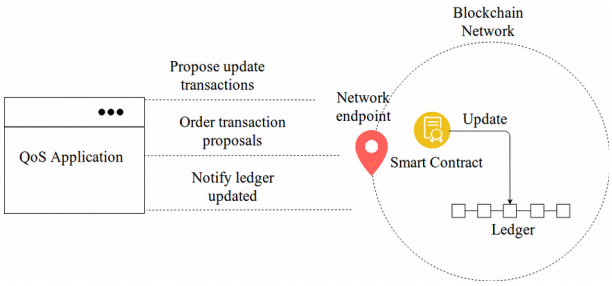

Figure 1: QoS Triggering

The smart contract, through its functions, will update the specific ledger for the channel. The access to this ledger will be granted at least to the OTT that is requiring the information and to the ISP that is giving back the response, in case the ISPs along the path among OTT and final user are not collaborating each other.

In case they have stipulated an agreement, the channel will be accessed by more than one ISP, facilitating the communication among the peers and user's connection related information will be shared with all the participants.

In these specific cases, the ISP's ESAs, or their SDN Controller, will perform this action through an application that has to be written using one of the released Software Development Kits (SDKs).

Nowadays, Hyperledger Fabric is offering two official SDKs, one for Node.js and another for Java, but unofficially, the SDKs for the languages Python, REST and Go, also are available for the download and testing.

The development of this application will be covered in future works, together with a testbed deployment, where the implemented Blockchain will be used in a simulated ISPs cooperative SDN scenario.

2.5 The Ordering Services

The ordering services are a defined collective of nodes that orders transactions in a block. The transactions are ordered basing on the first come first serve principle and are independent on the peer processes. Members are tied by cryptographic material which is contained in this element.

Due to the heterogeneity of the participants, more CAs will be deployed, in order to provide a choice to the entities involved in the communication. In our architecture the certificates are also used by the organizations to authenticate the transaction proposals generated by their applications. If we consider that the transactions committed are valid, the peers will use their certificates to endorse them.

Ordering services are a crucial point in the Hyperledger Fabric architecture because the channel that they provide supports the atomic delivery of all the messages. We may refer to this technique also as total-order broadcast, atomic broadcast, or consensus.

Through the ordering service it is also possible to provide support for multiple channels, getting close to the messaging system of publishing and subscribe.

2.6 The Fabric Certification Authority (CA)

The Fabric CA provides a server and a client components. The server component is the one that releases the certificates, while the client one is used to register new identities and enrolling new peers. In figure 2 it is shown how the server fits in the overall architecture. In particular it is possible to interact with the server via the client component or via one of the provided SDKs. All the communications towards the server are implemented via REST APIs.

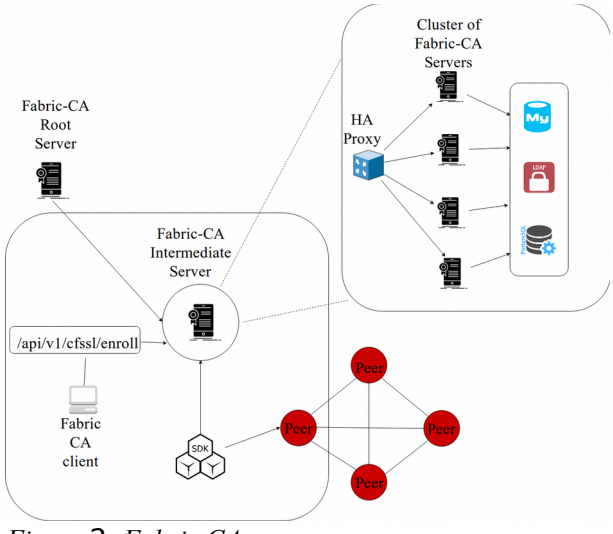

Figure 2: Fabric CA

As shown in the figure, the architecture made use of a HA Proxy load balancer, to distribute the requests from a Fabric-CA intermediate server, towards a cluster of Fabric-CA servers. This latter will then

communicate with the databases, among them we can find the support for MySQL and PostgreSQL, but if LDAP is configured, it is possible to integrate it, and the identity information will be kept in the LDAP instead of in a database. Being our first implementation still in a very embryonal phase, we preferred to use only MySQL.

2.7 Specific use case: e2e QoS delivery enhancement.

The OTT requests a reduction of the RTT for a certain user. The OTT, in this specific use case, represents one of the client nodes that is submitting a transaction invocation to some endorsers. The enhancement of the QoS for a particular final user is the object of the smart contract. This request will be endorsed by the peer nodes, that in this case are represented by the SDN controllers along the path. We assumed that all the ISPs along the path are implementing an SDN ruled by a controller, or a set of them, that is interacting with the permissioned ISP blockchain. In case the ISPs are members of the same channel, the OTT's request will be made just once. In case the ISPs along the path are not all members of the same channel, the OTT request will be repeated more times, until the requested QoS for the final user will be achieved. It is obvious that, choosing a channel where the ISPs are connected together, namely where already the trust among them has been established, can be more profitable because the ISPs along the paths have more interest in delivering to the OTTs and the final users the QoS requested.

This approach brings advantages for everyone, from the OTT's point of view, the quality of the provisioning of its specific service is related to some QoS parameters that a specific channel may already have demonstrated to be able to deliver.

This disruptive blockchain broadens the horizons for adopting new, specific and well defined policies when choosing routing paths to utilize in order to provide a good end to end (e2e) QoS for low latencies applications. By nature, the Blockchain is used as a leading technology in cryptocurrencies, and

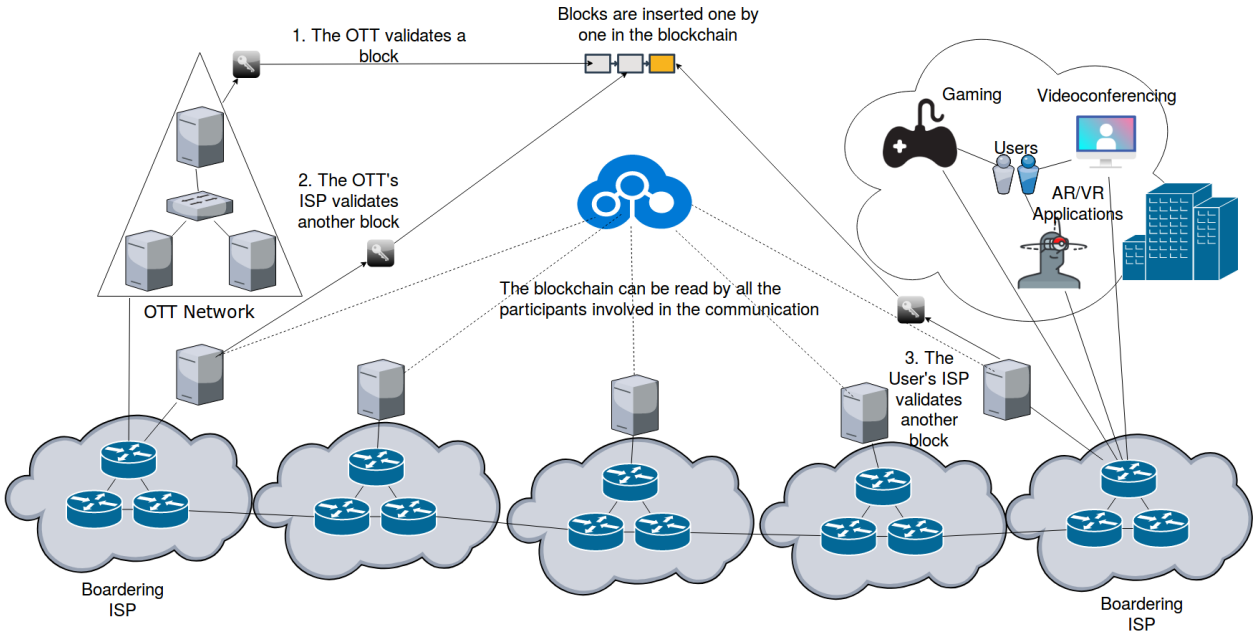

Figure 3: ISP blockchain simple scenario

the scenario of cooperation among ISPs is highly suitable to the definition of a new cryptocurrency, or to the use of an already existing cryptocurrency, through which it would be possible to bill feasible improvements in the QoS.

This last approach would lead to a further increase in the efficiency of the management of the network resources, such as for example the bandwidth dedicated for each user, optimizing its usage and increasing the routing efficiency for those applications which require a low latency. However, it

is important to underline that the employment of the SDN controllers proves fundamental in this approach, inasmuch they have demonstrated to be able to adopt a reactive behavior with respect to the information encapsulated in every new block added to the blockchain.

Ⅲ.Implementation of the Blockchain

3.1 The Membership Service Provider (MSP)

Each instance of an MSP has to be configured locally in each peer, orderer and on the channel where they are communicating, in order to enable the identity validation and the authentication, performed by the verification of the signature.

The network is created starting from the Ordering service, the green square in the figure, in which the configuration of the channel is contained. Each channel is configured via policies and membership information, using X509 root certificates.

A consortium of two or more organizations is defined by ISPs that need to coordinate their QoS parameters related to a certain user, agreeing on policies that rule the network. In the cooperative ISP scenario described in[1] a consortium is represented by the ISP chaining that connects the OTT to the final user. The policies will be applied by the SDN controllers of the ISP networks providing a Round Trip Time (RTT) within the desired thresholds, in order to provision a good QoS to the final user.

After the creation of the consortium, a channel is created, representing the communication layer that will be used to connect the entities involved in the communication.

3.2 The Channel Configuration

The channel configuration is done creating a configuration block that, being the first, is called the genesis block. In this block no other transactions will be stored. The channel configuration owns three important properties: it is versioned, it is permissioned and it is hierarchical.

In case we are adding a new organization to a channel, the necessary steps are the following: to generate its crypto material by using the cryptogen tool, to prepare the command line interface (CLI) environment by running a docker instance and exporting the ORDERER_CA and CHANNEL NAME variables. Then a Protocol Buffer binary file will be saved, where the channel configuration block will be contained. This block is generated by fetching the configuration, using the peer channel fetch command.

The previously created protobuf binary file containing the configuration block, has to be converted into a JSON format, by using the configtxlator tool. Furthermore, the block will be cleaned from those headers, creator signatures and metadata that are not relevant to the change that has been done, by using the jq tool.

Then the crypto material that had been previously generated has to be added to the new organization configuration definition. After this step, the configuration update has to be signed and submitted. Finally, before a new organization is joining the channel, the leader election process has to be configured. The last part of the joining process is the updating and the invocation of the chaincode, during which its new version will be installed.

The generation of the crypto material is done by means of the yaml files contained inside the artifacts directory of the new organization. Information such as name and domain name of the new organization is contained in the related YAML file.

3.3 Smart Contract

The invocation of the function RTTreduction of the smart contract that we defined, containing the request to reduce an RTT for a certain user, is done as consequence of the validation of a transaction proposed by the OTT application towards a particular ISP, or group of ISPs.

ISPs are able to reduce the RTT for a certain user giving priority to its flows, and adopting best path routing techniques. In our previous work, we also considered the possibility to exchange the exceeding bandwidth with a reduction of the RTT.

The lag experienced in certain real time communications is related to the RTT, and it is

possible that the bandwidth contracted by the user with its own ISP is not totally used. The application of smart contracts to this situation will provide advantages to all the participants, namely, will give the OTT and the final user the desired QoS, while for the ISPs it will be possible to know exactly which amount of bandwidth a certain user is not using at all, and employ the unused one in other ways, for example improving the connections of those users that are not using low latencies services.

IV. Conclusions and future directions

In the conclusion of this first step towards the full deployment of a permissioned ISP blockchain, we found very important for our scope several native features offered by Hyperledger Fabric. In particular, the possibility to create channels and to use a specific ledger for every channel created, made us further reflect about the need to create consortia of ISPs with the purpose of cooperating towards the same direction, namely the improvement of the QoS for the final users.

The next step in our research is related to the deployment in a real testbed of a simulated ISPs network, where the ISPs cooperative scenario will be proposed. In order to reach that step we need to configure the SDN controller to be able to read inside every new block of the ledger, and to program them to have a reactive behavior depending on the information contained in the blocks. The usage of a blockchain in a chaotic scenario such as the ISPs network traffic, will permit to create a new generation of network administrators more and more oriented to the user's flows optimization, bringing this new generation to be skilled network software defined programmers.

A further step in our research is the design and the implementation of a permissioned ISP blockchain ICN oriented. Namely, a hybrid ICN blockchain, able to exploit the native in-network cache and multicast functionalities offered by several ICN implementations, such as Named Data Networking and Community information Centric Networks, that are the focus of our studies.

Acknowledgements

This work was supported by the PhD scholarship from Region Sardinia (P.O.R. Sardegna, European Social Fund 2007- 2013 - Axis IV Human Resources, Line of Activity l.3.1.).This work was partly supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2019-0-00465, Scalable Decentralized Secure (DeSecure) ECCPoW Blockchain) and work was partly supported by the "Practical Research and Development support program supervised by the GTI (GIST Technology Institute)" grant funded by GIST in 2019 (Development of Error Correction Codes PoW and Bitcoin/Ethereum Hardfork 1.0).

References

[1] A. Ahmad, A. Floris, and L. Atzori, "QoE-centric service delivery: A collaborative approach among OTTs and ISPs," Computer Networks, vol.110, pp. 168–179, 2016.

[2] M. Scarlato, J. Ortiz, C. Perra, and A. Skarmeta, "Leveraging OTT and ISP cooperation to enhance End to End QoS by exchanging valuable resources," in 2018 Tenth International Conference on Ubiquitous and Future Networks (ICUFN). IEEE, 2018, pp. 118–120.

[3] C. Cachin, "Architecture of the hyperledger blockchain fabric," in Workshop on distributed cryptocurrencies and consensus ledgers , vol. 310, 2016.