Distributed network slicing management using blockchains in E-Health environments

Noname manuscript No.
(will be inserted by the editor)

Distributed Network Slicing Management Using Blockchains in E-Health Environments

João Paulo de Brito Gonçalves · Henrique Carvalho de Resende · Rodolfo da Silva Villaca · Esteban Municio · Cristiano B. Both · Johann M. Marquez-Barja

Received: date / Accepted: date

Abstract The new 5G networks must face the challenge to fulfill multiple requirements in different use cases, applications and verticals. These networks are designed to provide strict Quality of Service (QoS) compliance independently of the network conditions in each use case. Some use cases are deployed in Public Land Mobile Network (PLMN), infrastructures that are managed by mobile network operators. However, for other use cases, devices are connected to services in private networks named Non-Public Networks (NPNs). Network slicing is the 5G concept that addresses the compliance of multiple services requirements on the same network by virtualizing the physical network infrastructure. Network Slices (NSs) are created to manage the different requirements above and to ensure coexistence between these different requirements. In this work, we avail from network slicing concept to create a solution for NPNs in e-health environments which maintains QoS and privacy requirements among slices. Moreover, we also propose a blockchain mechanism to secure the NS management layer. This mechanism ensures data integrity and reliability for the NSs settings. We validate our approach by demonstrating our comprehensive solution to secure e-health environments sensitive data and the management of the slices by the e-health environments. We deployed a Proof of Concept (PoC) using the 5G EmPOWER system and a public blockchain and we evaluate some performance metrics in different scenarios.

Acknowledgements This research has received funding from the European Union's Horizon 2020 Research and innovation program under grant agreement No. 826284 (Protego).

Conflict of interest

The authors declare that they have no conflict of interest.

Keywords network slicing · quality of service · blockchain · smart contracts · oracles · e-health

1 Introduction

Providing guaranteed network requirements when owning the network infrastructure is already a challenge by itself. However, creating a mechanism to allow third-party private networks to maintain the network requirements is as far as challenging. The 3rd Generation Partnership Project (3GPP), the organization that assesses the requirements for new network infrastructures in the second phase of 5G networks (3GPP Rel-16 and beyond) [1], classifies the future 5G networks in two types: Public Land Mobile Networks (PLMNs) and Non-Public Networks (NPNs). The first type is the network provided directly by the Mobile Network Operator (MNO) infrastructure and it will be able to address many service requirements. The second type is composed of net-

João Paulo de Brito Gonçalves, Henrique Carvalho de Resende, Esteban Municio, Johann M. Marquez-Barja
University of Antwerpen - imec, Antwerpen, Belgium
E-mail: joaopaulo.britogoncalves@uantwerpen.be, henrique.carvalhoderesende@uantwerpen.be, esteban.municio@uantwerpen.be, johann.marquez-barja@uantwerpen.be

Rodolfo da Silva Villaca
Federal University of Espirito Santo, Vitoria-ES, Brazil
E-mail: rodolfo.villaca@ufes.br

Cristiano B. Both
University of Vale do Rio dos Sinos, Porto Alegre-RS, Brazil
E-mail: cbboth@unisinos.br

works deployed for private reasons, such the ones created for indoor connectivity of sensors, robots, autoguided vehicles, remote workers, IoT devices, wearable devices, etc [1].

NPNs are characterized by being more secure for who deploy it, but harder for the MNO that provides the network connectivity to maintain it.

Therefore, mechanisms are needed to maintain the network requirements inside NPNs and also the reliability and security of the network settings. One concept that focus on delivering the expected network requirements for specific use cases is network slicing. Network slicing makes use from virtualization to flexibilize the underlying physical infrastructure, enabling the isolation of network flows and differentiation of the network traffic in a dynamic manner [2].

Therefore, the deployment of Network Slices (NSs) in a NPN is a solution to continue delivering the expected Quality of Service (QoS) to the users, and a mechanism to maintain the reliability of the configuration of this NSs even when deployed in NPNs would give more security to MNO. The assignment of NSs to clients is a process that evolves dynamically, according to the demand variations of each client. At the same time, the chain of network resource loans must be negotiated in a secure, transparent and fast way, such that the life cycle of each slice is not affected. Due to its decentralized nature, the blockchain technology suits well these requirements as a secure, robust and transparent management solution. A distributed ledger allows all members of the system to be aware of the current (and past) network resource availability. A secure resource exchange is guaranteed by smart contracts and distributed consensus algorithms, allowing the system to evolve autonomously without the need of centralized authorities [3]. The greatest benefits that would be achieved by applying blockchain in network slicing are:

- Secure, dynamic, and distributed consensus on network management issues;
- Reliable ledge for forensic security analysis.

Hospital networks are examples of NPNs, because these networks are the medium to transmit, in-site, highly sensitive information every day. Even more, to provide connectivity to in-site sensors and health services, hospitals will need to deploy new technologies to enhance their existing private network infrastructure. This enhanced private infrastructure will allow hospitals to prioritize network traffic and secure the information in the very edge of the network. In an European context, the use of a NPN is also mandatory in the use case of hospitals because these e-health networks must comply with the General Data Protection Regulatio (GDPR) [4]. However, to deploy an NPN, research on how to enable network slicing is needed to evaluate heterogeneous technologies and their integration with PLMNs. In this work we propose a network slicing solution for Non-Public Networks (NPNs) hospital environments. The solution integrates state-of-the-art technologies to provide performance and privacy isolation in a hospital network for the NS data plane, and reliability/integrity checking to the NS control plane using blockchain. Thereby, our main contributions are: (*i*) to design an architecture for network slicing in hospital environments; and (*ii*) a decentralized slice management layer using blockchain and smart contracts in e-health environments.

The rest of this paper is organized as follows: Section 2 provides the related work and discuss the proposal regarding the state-of-the-art of the use of blockchain for e-health services and network slicing. In Section 3, the background about the main technologies used in our paper is presented. In Section 4 we present our proposal while Section 5 presents the implementation and evaluation. Finally, in Section 6, we conclude the paper and discuss future work proposals.

2 State of Art

Buzachis et al. [5] and Azaria et al. [6] propose solutions using smart contracts and blockchain to manage Electronic Health Records (EHR) and patient identities in e-health environments. A Proof of Concept (PoC) was designed using the Ganache tool [7] that simulates a blockchain to simplify application deployment and tests, but without deployment in a real environment.

MeDShare [8] and My Health My Data [9] are healthcare data systems that share data via cloud computing and blockchain technology. In these systems, the authors have used smart contract and authentication permission on data access, aiming at encouraging hospitals to start making anonymous data available for open research, while prompting citizens to become the ultimate owners and controllers of their health data.

Rosa et al. [10] propose the use of smart contracts in the context of operational phases in support of multi-administrative domain networking and slices management and the proof of concept is implemented using simulation using the Mininet simulator [11]. Moreover using simulation, with no deployment in a real environment, Scheid et al. [12] presented the design and implementation of a smart contract that simplifies and automates the compensation process in Service Level Agreements (SLA) violations.

Pascale et al. [13] propose a smart contract to automate Small-Cell-as-a-Service (SCaaS) agreements between the small-cell owners and network operators but there are no evaluations or real deployments of the proposal as in the work of Backman et al. [14] where a Blockchain Network Slice Broker is proposed to reduce the service creation time for dynamically slice acquisition and for verifiable charging and billing in service level agreements, but no implementation or performance analysis is provided.

Zanzi et al. [15] proposed NSBchain, a novel Network Slicing Brokering (NSB) solution, which leverages the blockchain technology to address the new business models needs beyond traditional network slicing agreements. They implemented NSBchain on top of Hyperledger Fabric and its benchmark tool, namely Hyperledger Caliper [16], but the deployment is done on a private and local blockchain. Afraz et al. [17] also propose a 5G Network Slice Brokering using Hyperledger Fabric and using Hyperledger Caliper as benchmark tool. The idea is to use a distributed process to replace the conventional centralized approach to slice brokering, where a single authority does not control the entire conduct of the market, but also without a deployment on a public blockchain.

Finally, Zhou et al. [18] introduce the concept of witness, based on the John Nash's Equilibrium Principle [19], to create a mechanism using Smart Contracts to report violations of compliance regarding SLAs between providers and consumers of cloud computing services. This mechanism would dispense the use of Oracles, commonly used to obtain reliable data external to the blockchain.

Unlike previous works that use blockchain technology at the application level in e-health, our proposal is to use this technology to provide security at the network layer, at the control plane. Besides that, we presented works using network slicing and blockchain, but generally oriented to telecom applications such as [20] and none oriented to e-health. Therefore, to the best of our knowledge, we are one of the first works to apply these technologies together in the e-health context and besides that, we are using a real and public blockchain for deployment, allowing access to data from anywhere. We have presented our initial approach in [21] and in this paper we present an evolved system with a comprehensive description of our solution.

3 Background

In the next subsections, we describe technologies used in our work to provide performance, flexibility, security and also monitoring and audition to network slices creation and management.

3.1 5G-EmPOWER Framework

5G-EmPOWER framework [22] is the component in the Protego's network slicing solution for performance isolation. 5G-EmPOWER is a Software-Defined Networking (SDN) framework for wireless networks that provides slicing techniques. Therefore, for hospital environments, the wireless slicing feature provided by 5G-EmPOWER allows the performance isolation of different services at the radio-side. 5G-EmPOWER utilizes techniques and specific hardware to virtualize WiFi access points and offer different QoS to its connected users. To achieve traffic prioritization the operator must configure a parameter called quantum and the greater the quantum of a slice, the higher its priority in relation to the others and is directly related with the airtime reserved for a particular slice.

The overall 5G-EmPOWER Framework architecture is presented in Fig. 1, where we can see a division in three layers: Management Plane, Control Plane and User Plane. In the Control Plane, The 5G-EmPOWER Operating System is divided into 3 components: (*i*) controller, (*ii*) Backhaul controller, and (*iii*) the Wireless Termination Point (WTP). This components can be translated to our ProTego's network slicing architecture, as SDR controller, SDN controller and Access Point, respectively. The controller is the component responsible for the management of the Radio Access Network (RAN) by deploying the necessary configuration in WTP, prioritizing the network traffic. The Backhaul controller manages SDN switches, which identify and tag the network packets that need to be prioritized. The WTPs are deployed in the WiFi access points where the clients connects, and the network bandwidth is monitored and managed. For more information about 5G-EmPOWER, please refer to [23] and [22].

3.2 Blockchain

Despite its initial financial original application [24], blockchain technology has grown to a multiplicity of different applications, such as distributed computing [25], Internet of Things [26], file storage [27], prediction [28], among many others. The blockchain is a distributed ledger, where each participant has a copy of the database with all the validated information. Besides, a consensus protocol is implemented among the participants in order to allow them to agree about the global state of the blockchain.

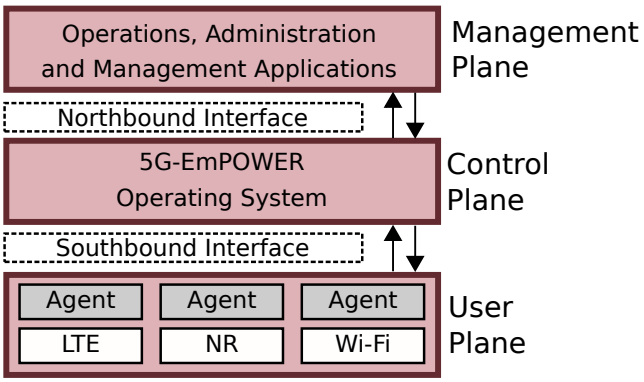

Fig. 1 5G-EmPOWER Architecture

In a blockchain, each block is a set of transactions chained through hash addresses. Each block includes, among other information, a timestamp, its hash, and the hash of the previous block, so that once the block is created, it cannot be tampered under the penalty of the stored hash not matching to the hash of the modified block.

In public blockchains, i.e., where access is not controlled by a central authority, validation of transactions and blocks is often based on the Proof of Work (PoW) consensus protocol. In PoW, a cryptic challenge is proposed in order to create a valid block, once solved, the block is propagated over the network.

At this point, it is important to differ between two basic types of blockchains: public and permissioned. In public blockchains, any computer can join the network and have full access to it. Because of this anonymous character of computers, measures to mitigate attacks must be adopted which results in performance degradation, as the PoW. Bitcoin and Ethereum [29] are examples of public blockchains. Permissioned blockchains are mainly used in corporate environments. This means that a user must have a certain level of access to interact on this network, read transactions, and participate in the consensus process. Hyperledger Fabric [16] is an example of a permissioned blockchain.

Data writing in public blockchains usually are expensive operations due the limited space in the nodes. For this case, off-chain storage using system as the Interplanetary File System (IPFS) [30] is a interesting option to store files that would be too expensive to write in the blockchain. In IPFS, the file storage address is the hash, providing a unique identification that is tightly linked to the file itself.

3.2.1 Ethereum and Smart Contracts

Several blockchain platforms have emerged in recent years and among the most popular are those based on the Ethereum platform. Ethereum is a platform for executing blockchain applications that are modeled as smart contracts and has its own cryptocurrency, the ether. Smart contracts capture and translate traditional legal contract clauses into a series of computational rules which are executed automatically and, once validated, don't require additional legal instruments [31].

Another important Ethereum concept is gas. Gas is a way of decoupling the cost of transactions in the Ethereum from the floating exchange rate of the ether cryptocurrency, establishing a cost for each fluctuating computational job in the financial market. Gas is also a mechanism to prevent a smart contract with infinite loops from running indefinitely on the blockchain. Once the maximum amount of gas allocated to a contract expires, the contract finish its execution. On the Ethereum platform, applications run on the Ethereum Virtual Machine (EVM), which executes smart contract instructions, allowing you to enter and query stored data. Ethereum uses PoW as its current consensus mechanism, but it is in a transition phase towards Ethereum 2.0. These two main changes should enable the processing of up to 10000 transactions per second [32].

The Ethereum platform is currently the largest general purpose public blockchain exponent on the Internet. It is a very flexible alternative to the development of dApps (Decentralized Applications) as it provides a complete programming language. A dApp is a decentralized application that uses a smart contract in the blockchain as a back-end, and a web interface as front-end, allowing users to insert and receive data from the blockchain in a friendly way.

3.3 Oracles

Usually smart contracts need information that is processed outside their computational logic and the availability of this information is crucial to the achievement of smart contracts full potential. However, this is challenging since smart contracts can only access and write information that is stored on the blockchain, which is an enclosed network without direct interfaces to the real world. Oracles bridge the gap between the blockchain and the real-world by feeding data from outside the blockchain to smart contracts. They are usually application's APIs which produce data that can be consumed by smart contracts. They are used to report events and data created after the smart contract has been programmed.

There are two types of oracles: centralized and decentralized. The centralized oracles suffer from the same problem of centralized networks, they have a single point

of failure, because have one data source. The decentralized oracles are better suited to deal with data sources where the user has no control over, such as system APIs available on the Web. Examples of decentralized oracles are Chainlink [33].

3.3.1 Chainlink

Chainlink [33] provides real-world data to smart contracts on the blockchain and provides the security guarantees by allowing multiple links to the same data. The process starts when a smart contract requires data and this smart contract puts out a request for information. The Chainlink protocol registers this request as an event and creates a corresponding smart contract (Chainlink Service Level Agreement Contract) on the blockchain to get this off-chain data. The Chainlink SLA generates three sub contracts: a Chainlink Reputation Contract, a Chainlink Order-Matching Contract and a Chainlink Aggregating Contract.

The Chainlink Reputation Contract checks an oracle provider's track record to verify its authenticity and performance history, then evaluates and discards disreputable or unreliable nodes. The Chainlink Order-Matching Contract delivers the Requesting Contract's request to Chainlink nodes and takes their bids on the request (when the Requesting Contract does not choose a specific set of nodes) and selects the right number and type of nodes to fulfill the request.

Finally, the Chainlink Aggregating Contract takes all the data from the chosen oracles and validates and/or reconciles it for an accurate result. In ChainLink, a K-out-of-M threshold signature is used by multiple oracles to reach a consensus on the answer to be accepted. For example, a 3-out-of-5 signature scheme requires at least three or more oracles out of five oracles to sign on the same value for the value to be accepted as the answer.

4 Protego's Solution for Secure Network Slicing on E-Health Environments

ProTego project[1] is developing a security toolkit to be applied in e-health environments such as key management encryption, secure data storage, and networks slicing. ProTego's network slicing is essential for the security and flexibility of NPNs since it isolates the services traffic in order to dynamically setup the right QoS and security to a specific service. To achieve such objectives, the ProTego network slicing is composed by three modules: the Performance Isolation, the Privacy Isolation, and the Blockchain Application. The performance and privacy isolation have a direct influence in the data plane by managing QoS and encryption of the sliced services, while the Blockchain Application focus on the control plane by securing the integrity of slices configuration.

[1] https://protego-project.eu/

4.1 Architecture

The overall architecture can be seen in Fig. 2, where the components in red are from the performance isolation module, the blue components are from the privacy isolation module, and the gray components performs parsing of external communications. The yellow components are for blockchain slice management.

In the first layer, the component called Slice Processor, is designed to process input from external operators. These operators can be mobile network companies or hospitals, etc. This input is configured by inserting network parameters or providing a template. This file or commands sequel is processed by the Slice Processor and generates performance and privacy rules, which are sent to the second layer, called Performance Isolation Engine and the Privacy Isolation Engine. The engines can map the requirements to Application Programming Interface (API) calls to the different components of the third layer, *i.e.*, Software-defined Radio (SDR) controller, SDN controller, and the Secure Interface Setup.

4.2 Perfomance Isolation and Privacy Isolation Modules

The Performance and Privacy Isolation modules have the objectives of manage the traffic prioritization in the wireless-side and encrypt the different service flows from the Access Network (AN) to the core of the network. The combination of these two features enables the network to prioritize the critical data coming from medical devices and encrypt high-sensitive data, while non-critical data can be delayed and sent through a less secure communication channel.

To differentiate services in the network, 5G-EmPOWER uses Open vSwitch (OvS) [34] to tag network packets, which will enable the WiFi scheduler to apply the right prioritization rule for that flow, as these tagged network packets will be identified by a packet processor responsible for prioritizing the wireless network traffic. For the privacy isolation, OvS is used to redirect the traffic among different network interfaces that can tunnel this traffic applying encryption techniques, and sending the data to the network. These op-

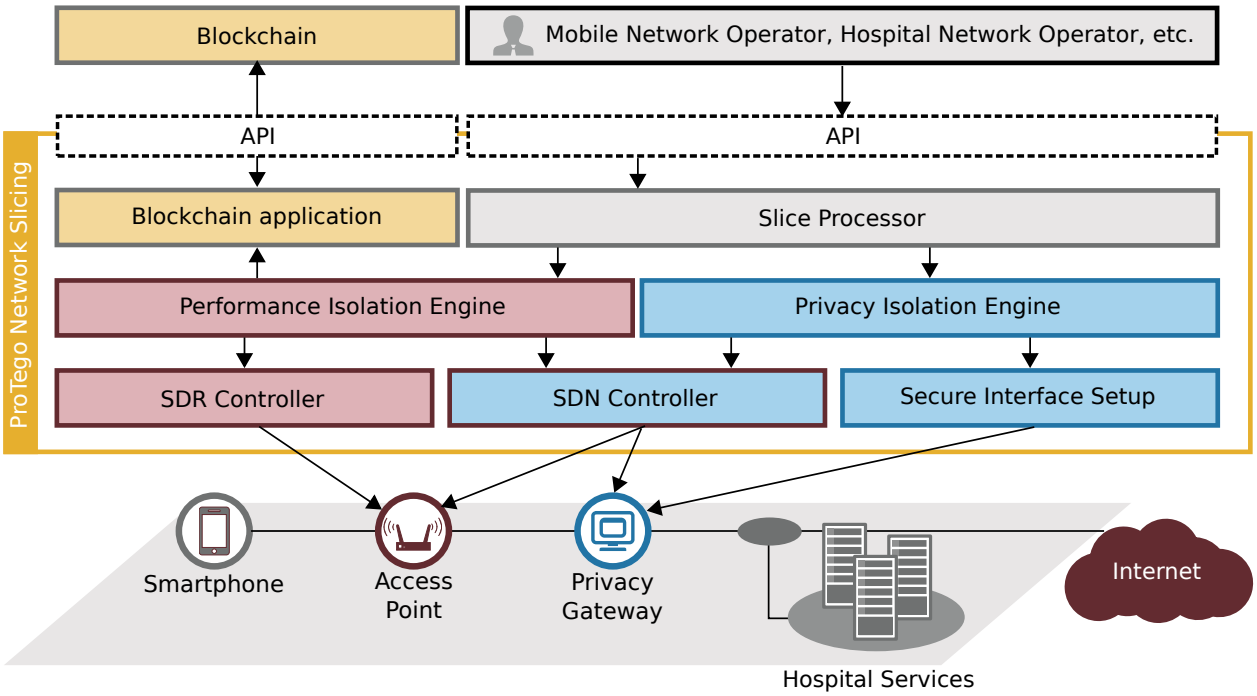

Fig. 2 ProTego's Network Slicing architecture.

tions enable the dynamic privacy isolation for different levels of private information.

The SDR controller is responsible for the configuration of wireless parameters and priority queues to provide the necessary requirements for the specified service. It will set the available SDN network to redirect the traffic securely and without performance degradation inside the hospital network. The Secure Interface Setup provides secure network interfaces with encryption techniques to enable the confidentiality of private data. These secure network interfaces will be available for the SDN switches to redirect the critical service flow through them and perform an extra and custom security layer for hospital services.

4.3 Blockchain Application Module

We identify slice logging as an interesting use case to use blockchain technology within Protego's network slicing approach. By storing and logging several slicing metrics as WiFi priority queues parameters or quantum, network slices can be securely managed and audited, ensuring higher reliability to the control plane. This approach may prevent malicious users from manipulating QoS parameters to gain access to secured slices. Furthermore, the logs attached to the distributed ledger are immutable, which guarantees that none of the parties can modify it. For instance, to avoid that a compromised node requests illegitimate network slices (e.g., to perform a DoS attack on a client), the 5GEmPOWER controller and WTPs must always double-check the blockchain if the given operation is allowed. Due to the cost of storing data on the blockchain, log reports (e.g., slice bandwidth) can be saved in a hash-based distributed file system, such as IPFS [30].

Fig. 3 shows the proposed extension for the Protego's Network Slicing where a dApp connected to the 5G-EmPOWER Controller container executing in a server within the hospital intranet, receives the logs and processes them, sending relevant data to the blockchain. It is worth to mention that we do not store confidential data about health status of patients on the blockchain, only technical data related to slices updates.

We developed a smart contract in Solidity, connected to a Web interface, composing a dApp. We created the dApp interface using React framework [35], a JavaScript library for building user interfaces and we deployed the smart contract in the Ropsten network [36], a public Ethereum test network that was chosen among all others because it is the only one that implements the PoW verification approach, being closer to the real Ethereum's current network, but with no real monetary expenses to execute transactions. In Ropsten, faucets are used to create ethers with no real value.

To visualize the transactions submitted to the blockchain, we used Etherscan [37], a block explorer and analytics platform for Ethereum blockchain. In its dashboard we can visualize all the transactions details: status, block number, timestamp, gas used, gas price as

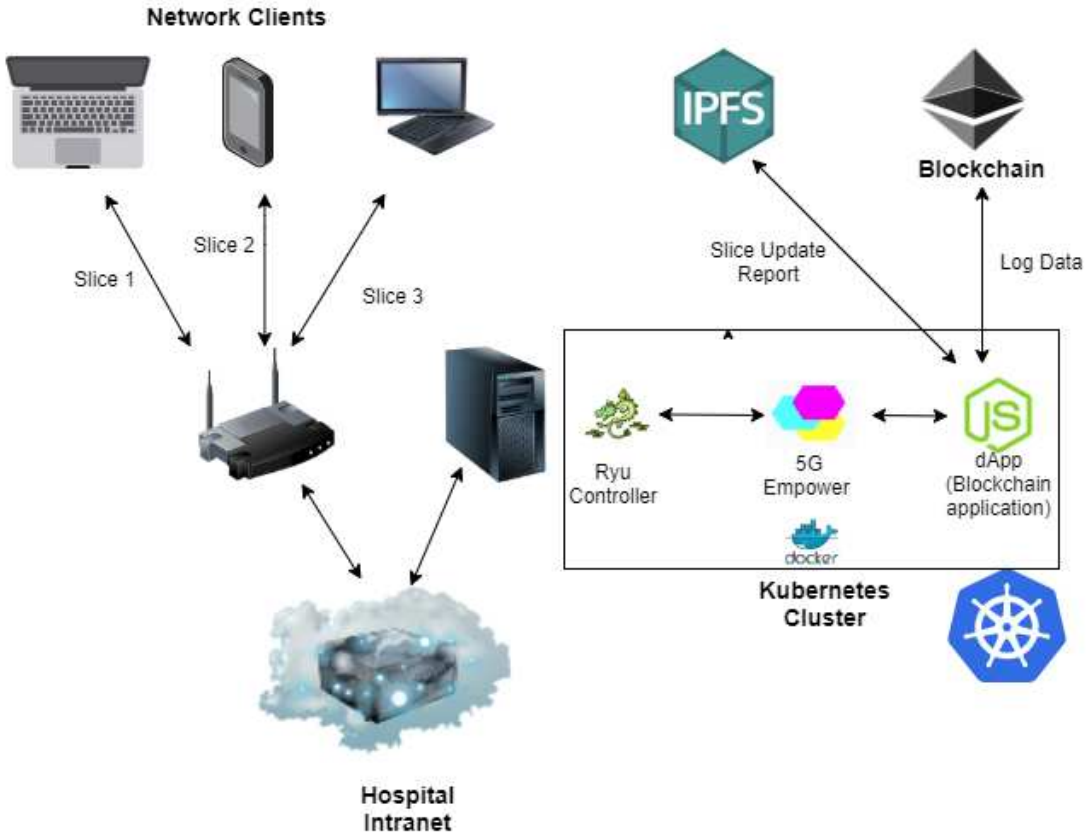

Fig. 3 Blockchain Application

well as the transaction content itself. Moreover we used the Metamask plugin in the Web browser to handle users' Ethereum accounts, and to enable the interaction with the blockchain without the need to locally run a full Ethereum node.

Streamr is a framework that enables live storing and sharing of data streams [38] and is build on top of Ethereum blockchain. Among its components, one of the most important is the Streamr Engine which is in charge of user authentication and payment. We used the Streamr Engine [38] to capture data produced by the 5G-EmPOWER tool, process and send it to the blockchain in an automatic way. To receive slices updates we created a microservice which consumes and acts upon real-time data, using a canvas in Streamr. This real-time data channel will act as a Oracle, inserting real-world data in the blockchain. To receive slices QoS parameters from the 5G-EmPOWER API we used Chainlink Distributed Oracle.

IPFS is the storage system selected to store the reports generated by the solution proposed in this work. To connect to IPFS, we used Infura, a scalable backend infrastructure for building dApps, to connect to both blockchain network and decentralized storage. A feasible use case is showed in Fig. 4, when a malicious user with privileges to change the priority on a slice to benefit an application can have this action discovered due to the log data that is saved on the blockchain, exposing the fraud attempt.

5 Validation and Experimentation

In this section, we provide a PoC on NPN slicing. Our testbed for PoC includes a PC Engines APU2D4, a Intel-NUCi7, and an MacBook Pro. The PC Engines APU2D4, or just APU, provides the wireless access network, the Intel NUC works as privacy gateway and also host for the ProTego network slice control plane, and the MacBook Pro is used as a wireless client.

It is important to mention that, as a 5G-EmPOWER requirement, the APU needs to support specialized requirements for slicing, for example, exposing the access point Basic Service Set Identifier (BSSID) register so that can be changed on-demand following the service requirements of QoS and handover [23]. By customizing the BSSID register, it enables a centralized controller to manage the handover for IEEE 802.11 networks, directly affecting QoS. Therefore, we utilized the Compex WLE200NX miniPCIe as the network card for the access point.

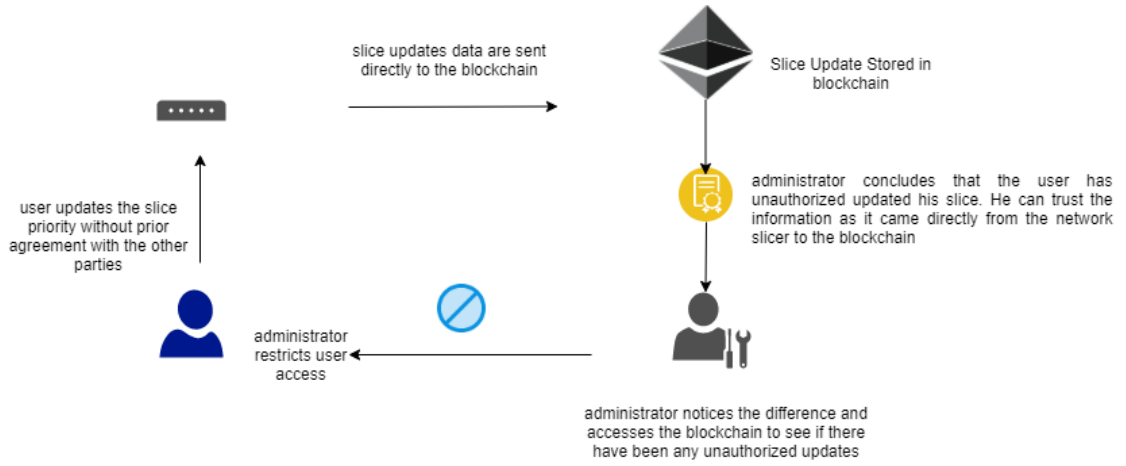

Fig. 4 Blockchain Use Case

In the envisioned scenario, the wireless client connects to the access point with the 5G-EmPOWER WTP enabled and the client has two services requesting data from a hospital service server located in the processing node. Before getting to the processing node, the service traffic passes through the privacy gateway, which encrypts the traffic for this specific service flow. In our scenario, the controllers were deployed in the privacy gateway, which was already configured with a container engine. The container engine supports the deployment of the 5G-EmPOWER controller, Backhaul controller, and the Secure Interface Setup in different containers, isolating the control services from one another. The container management tool used for the PoC is Docker Engine [39] and for integration with the rest of the project, the use of Kubernetes [39] is foreseen. We could analyze the behavior patterns of NPNs with different QoS requirements with this testbed setup.

5.1 Performance Isolation

The experiment for performance isolation aims to validate the expected software behavior for different protocols such as TCP and UDP when using the traffic prioritization of network slicing. Validation using both protocols was chosen to analyze how network slicing would behave with services that will be used in the hospital and that use both transport protocols. Due to the distributed characteristic of WiFi, it is already known beforehand that only the client downlink will be sliced. 5G-EmPOWER, the tool used for performance isolation, does not modify the client only the access point. The summary of the experiment protocols and airtime changes can be seen in Table 1.

The performance isolation experiment is deployed using two iPerf3 flows emulating two different services.

Protocol	Period	Slice 1 airtime	Slice 2 airtime
UDP	0-90	1/2	1/2
	90-180	1/3	2/3
TCP	0-90	3/4	1/4
	90-180	1/2	1/2

Table 1 Performance isolation experiment description

We have two slices, one for each service. The services' flows will be steered using OpenFlow (OF) rules based on service port, service 1 connected to port 55333 and service 2 connected to port 55444. Having two different services, we configure in 5G-EmPOWER one slice for each service enabling the customization of the utilized airtime by them. For both experiments of UDP and TCP, the bitrate for the iPerf traffic was set to 20Mbps from the processing node to the wireless client (downlink). Moreover, for the UDP, the experiment starts distributing the traffic equally, and at the 90th second, we give only a third of the airtime to the slice 1, as can be seen in Fig. 5, indicated by the arrow.

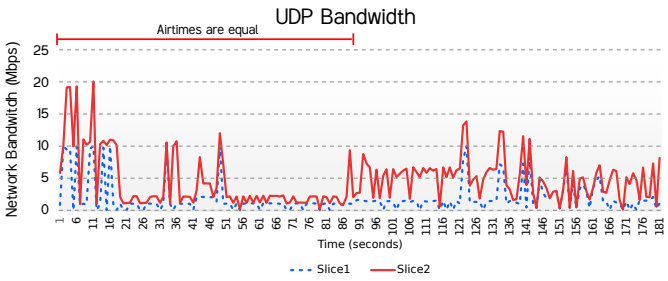

Fig. 5 Performance UDP

In general, the traffic through the wireless link has a lot of variation due to the interference. In the first half of the UDP experiment, we can see that the traffic for both slices maintains the same throughput, confirm-

ing the network's expected behavior given the airtime configuration. Furthermore, after the second 90, slice 1 was changed to 1/3 of the airtime of slice 2. The result of this change can be explicitly seen in the second half of the chart when the throughput of slice 2 increases. Besides some network spikes due to wireless connectivity variation, slice 1 throughput keeps at 1/3 of the network throughput of slice 2, demonstrating that the UDP downlink for the application can be customized to prioritize critical hospital services.

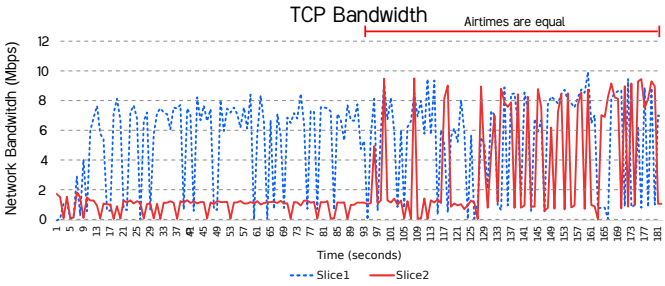

Fig. 6 Performance TCP

The experiment for the performance isolation of TCP traffic started with different airtimes for the slices. After the second 90, the airtimes were equally distributed between the slices, as shown in Fig. 6. The distribution of airtime started with slice 2 having 1/4 of the airtime of slice 1. After we changed the airtime for slice 2 to the same as slice 1, we can see that the throughput of slice 2 does not adapt immediately, having an adaptation time before consuming the same amount of bandwidth as slice 1. This delay is caused by the normal behavior of the TCP protocol, which increases the amount of data sent gradually. After assessing the outcome of using performance isolation with downlink TCP traffic, we conclude that any Web-based application most of the time the downlink traffic is higher than the uplink traffic will benefit from the performance isolation provided by 5G-EmPOWER and the ProTego network slicing tool.

5.2 Privacy Isolation

The privacy isolation is characterized by the separation and encryption of given traffic, *i.e.*, service or client. In ProTego, the encryption method must be utilized to protect the confidentiality of patients and staff data. The patient's and staff's service data are encrypted when passing through a virtual secure network interface that utilizes Virtual Extensible LAN (VXLAN) to create a virtual network and Internet Protocol Security (IPsec) [40] as a secure protocol. It is fundamental to understand how security affects the QoS of the network, so a trade-off between security-enabled slices and non-secure slices can be researched. Therefore, to assess the behavior of the proposed privacy isolation tool, we deployed two iPerf3 clients and two iPerf3 servers. The first iPerf3 connection uses the UDP protocol, and the second uses TCP. The traffic in this experiment is separated by protocol, but later can be enhanced to separate by destination port, for example. The default virtual network interface is tagged with VXLAN Network Identifier (VNI) 10, and the secure network slice is tagged with VNI 20. We highlight that in this experiment, the traffic steering among different virtual interfaces was assessed.

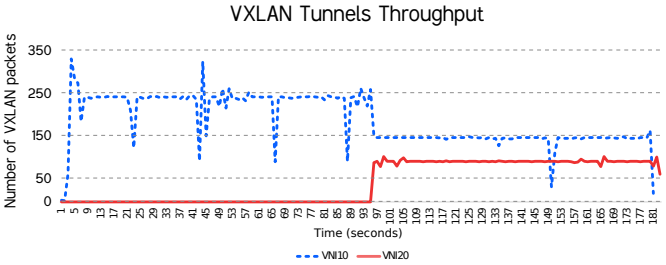

Fig. 7 Privacy isolation

We started the two traffic generator, as shown in Fig. 7, in the second 0. The data were collected by sniffing the physical network interface where both traffic, secure, and not secure, pass-through. At first, the secure network slice was not created, due to all the traffic was passing through the default virtual interface with the VNI 10. After the second 90, the slice was created, and the UDP traffic was redirected to the secure virtual interface, tagged with the VNI 20. The deployment of the OF rule for traffic steering was proven to be useful for our purposes, having less than 5 seconds of delay on changing the traffic from the default interface to the secure interface. This tool is currently being enhancement with the integration of the IPsec protocol over OvS, to provide point-to-point encryption for several slices. In the future, it will be possible to enable different slices with different encryption parameters, increasing communication security.

5.3 Blockchain Application

The Blockchain Application is the module in charge of monitoring and audit the slices updates in the architecture. The PoC works as follows: a bash script running in the 5G- EmPOWER Docker container filters the log file from the controller in a hourly basis, searching for slices updates. Every slice update found is send to a Streamr

Client implemented in the container, that via a TLS connection sends the data to the Streamr Engine, thus creating an Oracle. After that, the microservice running on Streamr sends this information automatically to the blockchain, where it can be audited later, to check if this slice update was previously authorized.

Using a dApp we access the latest update recorded on the blockchain as well as generate reports with the latest registered updates and send them to IPFS in a web-based interface. We also visualize using the 5G-EmPOWER API the QoS parameters of the slices and the changes made to these parameters after a slice update. As this API was exposed on the Internet, it is accessed by the Chainlink Distributed Oracle that through its distributed consensus mechanism guarantees data consistency. In this way, whenever an unauthorized slice update happens, we check using the Oracle via API and send to the blockchain which are the QoS parameters of the updated slice. Any slice update described in the previous sections as part of the experiments was captured from 5G-EmPOWER controller log and API and sent to the blockchain.

As the analysis was focused on blockchain behavior, the results collected focus on blockchain-related metrics. The main differences of blockchains compared to traditional programs and databases, are the occurring monetary cost of transactions as well as the high latency of processing due to the creation time of blocks. The performance of a blockchain application/network can be measured using the following metrics:

- Transaction Throughput: measured in transactions per second (TPS) and represents the number of transactions that are processed by the blockchain and written on the ledger in a given second.
- Transaction Latency: The amount of time taken from the moment when a transaction is submitted until the moment when it is confirmed and available on the blockchain.
- Operation Costs: The amount of computing resources and monetary costs consumed by the blockchain throughout the operating time. Therefore, the computing intensity would also affect the operation costs of the blockchain as the used gas of a transaction is expected to be closely correlated to the computational cost (CPU time) required.

Regarding the operation costs, the higher the gas price the more expensive the transaction costs but it also leads to faster validation of the transaction. It is essential to find a balanced point between the cost and the speed for the proposal. The deployment and interactions with the smart contract that alters its state required the payment of a certain gas fee and the transaction fee is calculated by the amount of gas used in a transaction multiplied by the gas price. Using Etherscan it is possible to see the transactions and its corresponding block history, providing information such as the amount of gas consumed per transaction and per block. The transactions were executed at a fixed gas price of around 0.0000000015 ether per unit and transaction fees that oscillate between 0.000039103 to 0.0174515 ether as we can see in Fig. 8, that shows the consumption in ethers of the transactions executed in the application in the interval of 1 month.

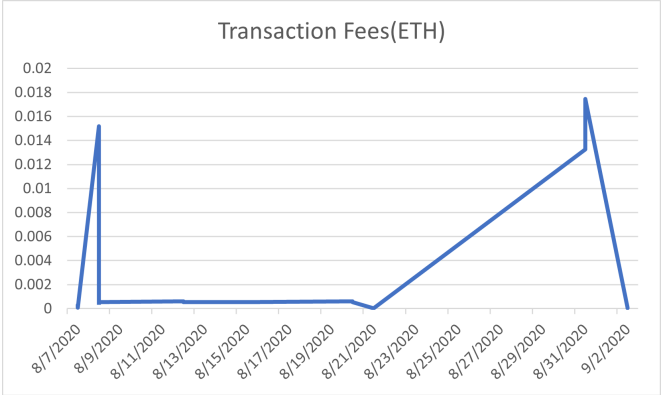

Fig. 8 Transaction Fees

It is noticeable that the creation of the smart contract is the most expensive function, since it is the operation that writes the greater amount of data on the blockchain. This emerges from the way how transaction costs are composed in Ethereum, namely by a fixed and a variable part. The EVM demands a fixed cost of 32.000 gas units for the creation of a smart contract in addition to the 21.000 gas units for each transaction. The remainder is the variable part which depends on the size of the contract code. Each byte of code consumes 200 gas units, the more code a contract has the more expensive its creation is [29]. The other operations in the application are slice data and IPFS hashes writing. These are composed of a fixed gas fee of 21.000 plus a certain fee for each operation. In a scenario without IPFS integration, the storage of the availability reports would be the most costly operation, but with IPFS integrated to the system, only hash addresses are written via the smart contract.

Since every transaction interacting with the smart contract needs to be included into a block to be validated, this leads to a blockchain-dependent latency. In practice, participants in a blockchain network are geographically distributed and such distribution introduces additional latency. In the Ethereum blockchain, the transaction latency was 15 seconds, which is the average block time creation [29]. This small latency to update variables in the smart contract is acceptable for

the applications proposed in this paper, where the interval between slice updates does not last less than 15 seconds, so there are low probability of having inconsistencies due to the validation delay, and this delay does not affect the slice creation process, which is independent of the blockchain transactions.

Regarding the metric number of transactions per second, the application is subject to the maximum throughput of the Ethereum blockchain, that is 15 transactions per second [29].

6 Conclusions and Future Work

In this paper we present a NPN architecture to base the research on experimental NPNs slicing and a PoC focused on network slicing for hospital environments. Up next, to provide precise results for this initial research a testbed setup was implemented and described. We also propose a blockchain aplication to provide a decentralized management layer to the system, that receives data from the the network slicing solution using oracles and stores data both on the blockchain and on a decentralized file system.

We achieved the objectives described at the introduction of this article: (i) design an architecture for network slicing in hospital environments with an open-source prototype (ii) provide an extra layer of reliability by integrating the prototype with the blockchain technology. This extra layer stores data regarding slice updates to prevent an unauthorized privilege escalation.

The network slicing PoC proved to provide both performance isolation over WiFi networks and privacy isolating to the service traffic, providing data confidentiality. The blockchain application PoC evaluation shows that the costs, expressed in ethers, to execute the transactions on the blockchain was not too excessive and the response time of the validated transactions, even in a public blockchain, is acceptable. It is important to mention that the chosen blockchain, even being published, does not impact the privacy of patient data in the hospital environment, as only technical information about the slices is recorded on the blockchain and the patient data will be recorded in other applications of the Protego project not related to network slicing module.

Given the results evaluation, our solution provides with the required performance isolation in current 5G NPN networks and provides enough security because both the privacy isolation and the blockchain application proposed. As future work we plan the full integration of the IPSec protocol with the Privacy isolation module providing point-to-point encryption.

References

1. G. (2019), "Service requirements for the 5G system," 3rd Generation Partnership Project (3GPP), Technical Specification (TS) 22.261, 12, version 16.10.0. [Online]. Available: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails .aspx?specificationId=3107
2. A. Barakabitze, A. Ahmad, R. Mijumbi, and A. Hines (2020), "5g network slicing using sdn and nfv: A survey of taxonomy, architectures and future challenges," *Computer Networks*, vol. 167. [Online]. Available: https://doi.org/10.1016/j.comnet.2019.106984
3. J. P. de Brito Gonçalves, R. L. Gomes, R. da Silva Villaca, E. Municio, and J. Marquez-Barja (2020), "A service level agreement verification system using blockchains," in *2020 IEEE 11th International Conference on Software Engineering and Service Science (ICSESS)*, pp. 541–544. [Online]. Available: https://doi.org/10.1109/ICSESS49938.2020.9237735
4. P. Voigt and A. Von dem Bussche (2017), "The eu general data protection regulation (gdpr)," *A Practical Guide, 1st Ed., Cham: Springer International Publishing*. [Online]. Available: https://doi.org/10.1007/978-3-319-57959-7
5. A. Buzachis, A. Celesti, M. Fazio, and M. Villari (2019), "On the design of a blockchain-as-a-service-based health information exchange (baas-hie) system for patient monitoring," in *2019 IEEE Symposium on Computers and Communications (ISCC)*. IEEE, pp. 1–6. [Online]. Available: https://doi.org/10.1109/ISCC47284.2019.8969718
6. A. Azaria, A. Ekblaw, T. Vieira, and A. Lippman (2016), "Medrec: Using blockchain for medical data access and permission management," in *2016 2nd International Conference on Open and Big Data (OBD)*. IEEE, pp. 25–30. [Online]. Available: https://doi.org/10.1109/OBD.2016.11
7. W.-M. Lee (2019), "Testing smart contracts using ganache," in *Beginning Ethereum Smart Contracts Programming*. Springer, pp. 147–167. [Online]. Available: https://doi.org/10.1007/978-1-4842-5086-0_7
8. Q. Xia, E. B. Sifah, K. O. Asamoah, J. Gao, X. Du, and M. Guizani (2017), "Medshare: Trust-less medical data sharing among cloud service providers via blockchain," *IEEE Access*, vol. 5, pp. 14 757–14 767. [Online]. Available: https://doi.org/10.1109/ACCESS.2017.2730843
9. E. Morley-Fletcher (2017), "Mhmd: My health, my data." in *EDBT/ICDT Workshops*. [Online]. Available: http://www.myhealthmydata.eu/blockchain/
10. R. Rosa and C. E. Rothenberg (2018), "Blockchain-based decentralized applications for multiple administrative domain networking," *IEEE Communications Standards Magazine*, vol. 2, no. 3, pp. 29–37. [Online]. Available: http://doi.org/10.1109/MCOMSTD.2018.1800015
11. F. Keti and S. Askar (2015), "Emulation of software defined networks using mininet in different simulation environments," in *2015 6th International Conference on Intelligent Systems, Modelling and Simulation*. IEEE, pp. 205–210. [Online]. Available: https://doi.org/10.1109/ISMS.2015.46
12. E. J. Scheid, B. B. Rodrigues, L. Z. Granville, and B. Stiller (2019), "Enabling dynamic sla compensation using blockchain-based smart contracts," in *2019 IFIP/IEEE Symposium on Integrated Network and Service Manage-*

ment (IM). IEEE, pp. 53–61. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8717859
13. E. D. Pascale, J. McMenamy, I. Macaluso, and L. D. (2017), "Smart contract slas for dense small-cell-as-a-service," *CoRR*, vol. abs/1703.04502. [Online]. Available: http://arxiv.org/abs/1703.04502
14. J. Backman, S. Yrjölä, K. Valtanen, and O. Mämmelä (2017), "Blockchain network slice broker in 5g: Slice leasing in factory of the future use case," in *2017 Internet of Things Business Models, Users, and Networks*. IEEE, pp. 1–8. [Online]. Available: https://doi.org/10.1109/CTTE.2017.8260929
15. L. Zanzi, A. Albanese, V. Sciancalepore, and X. Costa-Pérez (2020), "Nsbchain: A secure blockchain framework for network slicing brokerage," *arXiv preprint arXiv:2003.07748*.
16. V. Dhillon, D. Metcalf, and M. Hooper (2017), "The hyperledger project," in *Blockchain enabled applications*. Springer, pp. 139–149. [Online]. Available: https://doi.org/10.1007/978-1-4842-3081-7_10
17. N. Afraz and M. Ruffini (2020), "5g network slice brokering: A distributed blockchain-based market," in *2020 European Conference on Networks and Communications (EuCNC)*. IEEE, pp. 23–27. [Online]. Available: https://doi.org/10.1109/EuCNC48522.2020.9200915
18. H. Zhou, X. Ouyang, Z. Ren, J. Su, C. de Laat, and Z. Zhao (2019), "A blockchain based witness model for trustworthy cloud service level agreement enforcement," in *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*. IEEE, pp. 1567–1575. [Online]. Available: https://doi.org/10.1109/INFOCOM.2019.8737580
19. K. Binmore (2007) *et al.*, *Game theory: a very short introduction*, vol. 173. [Online]. Available: https://ssrn.com/abstract=1284255
20. P. H. Isolani, N. Cardona, C. Donato, J. Marquez-Barja, L. Z. Granville, and S. Latré (2019), "Sdn-based slice orchestration and mac management for qos delivery in ieee 802.11 networks," in *2019 Sixth International Conference on Software Defined Systems (SDS)*, June, pp. 260–265. [Online]. Available: https://doi.org/10.1109/SDS.2019.8768642
21. H. C. C. de Resende, J. P. de Brito Gonçalves, C. B. Both, and J. M. Marquez-Barja (2020), "Enabling qos-secured enhanced non-public network slices for health environments," in *Proceedings of the 6th EAI International Conference on Smart Objects and Technologies for Social Good*, pp. 18–23. [Online]. Available: https://doi.org/10.1145/3411170.3411244
22. E. Coronado, S. N. Khan, and R. Riggio (2019), "5g-empower: A software-defined networking platform for 5g radio access networks," *IEEE Transactions on Network and Service Management*, vol. 16, no. 2, pp. 715–728. [Online]. Available: https://doi.org/10.1109/TNSM.2019.2908675
23. E. Coronado, R. Riggio, J. Villa1ón, and A. Garrido (2018), "Lasagna: Programming abstractions for end-to-end slicing in software-defined wlans," pp. 14–15. [Online]. Available: https://doi.org/10.1109/WoWMoM.2018.8449797
24. S. Nakamoto (2008), "Bitcoin: A peer-to-peer electronic cash system." [Online]. Available: http://bitcoin.org/bitcoin.pdf
25. J. B. Pollack and H. Lipson (2000), "The golem project: Evolving hardware bodies and brains," in *Proceedings of the 2nd NASA/DoD Workshop on Evolvable Hardware*. IEEE, pp. 37–42. [Online]. Available: https://doi.org/10.1109/EH.2000.869340
26. A. D. Dwivedi, G. Srivastava, S. Dhar, and R. Singh (2019), "A decentralized privacy-preserving healthcare blockchain for iot," *Sensors*, vol. 19, no. 2, p. 326. [Online]. Available: https://doi.org/10.3390/s19020326
27. S. Wilkinson, T. Boshevski, J. Brandoff, and V. Buterin (2014), "Storj a peer-to-peer cloud storage network." [Online]. Available: https://storj.io/storj2014.pdf
28. J. Peterson and J. Krug (2015), "Augur: a decentralized, open-source platform for prediction markets," *arXiv preprint arXiv:1501.01042*.
29. G. Wood (2014), "Ethereum: A secure decentralised generalised transaction ledger," *Ethereum Project Yellow Paper*, vol. 151. [Online]. Available: http://gavwood.com/paper.pdf
30. J. Benet (2014), "Ipfs-content addressed, versioned, p2p file system (draft 3)," *arXiv preprint arXiv:1407.3561*.
31. N. Szabo (1997), "Formalizing and securing relationships on public networks," *First Monday*, vol. 2, no. 9. [Online]. Available: https://doi.org/10.5210/fm.v2i9.548
32. A. Skidanov and I. Polosukhin (2019), "Nightshade: Near protocol sharding design," *URL: https://nearprotocol.com/downloads/Nightshade. pdf*, p. 39.
33. S. Ellis, A. Juels, and S. Nazarov (2017), "Chainlink a decentralized oracle network," *Retrieved March*, vol. 11, p. 2018. [Online]. Available: https://link.smartcontract.com/whitepaper
34. B. Pfaff, J. Pettit, T. Koponen, E. Jackson, A. Zhou, J. Rajahalme, J. Gross, A. Wang, J. Stringer, P. Shelar (2015) *et al.*, "The design and implementation of open vswitch," in *12th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 15)*, pp. 117–130. [Online]. Available: https://www.usenix.org/conference/nsdi15/technical-sessions/presentation/pfaff
35. H. Brito, A. Gomes, Á. Santos, and J. Bernardino (2018), "Javascript in mobile applications: React native vs ionic vs nativescript vs native development," in *2018 13th Iberian Conference on Information Systems and Technologies (CISTI)*. IEEE, pp. 1–6. [Online]. Available: https://doi.org/10.23919/CISTI.2018.8399283
36. Y.-C. Hu, T.-T. Lee, D. Chatzopoulos, and P. Hui (2018), "Hierarchical interactions between ethereum smart contracts across testnets," in *Proceedings of the 1st Workshop on Cryptocurrencies and Blockchains for Distributed Systems*, pp. 7–12. [Online]. Available: https://doi.org/10.1145/3211933.3211935
37. E. Team (2017), "Etherscan: The ethereum block explorer." [Online]. Available: https://etherscan.io/
38. M. Spiekermann (2019), "Data marketplaces: Trends and monetisation of data goods," *Intereconomics*, vol. 54, no. 4, pp. 208–216. [Online]. Available: https://link.springer.com/article/10.1007/s10272-019-0826-z
39. D. Bernstein (2014), "Containers and cloud: From lxc to docker to kubernetes," *IEEE Cloud Computing*, vol. 1, no. 3, pp. 81–84. [Online]. Available: https://doi.org/10.1109/MCC.2014.51
40. N. Ferguson and B. Schneier (1999), "A cryptographic evaluation of ipsec." [Online]. Available: http://www.csci.csusb.edu/ykarant/courses/sp2002/csci594/papers/counterpane-ipsec.pdf