Antevski, K. y Bernardos, C. J. (2020). Federation of 5G services using distributed ledger technologies. *Internet Technology Letters*, 3 (6).

DOI: https://doi.org/10.1002/itl2.193

Federation of 5G services using Distributed Ledger Technologies

Kiril Antevski, Carlos J. Bernardos
University Carlos III of Madrid, Spain

***Abstract*—Federation of services, as a 5G networks concept, aims to provide orchestration of services across multiple administrative domains. In this paper, we are exploring a solution of applying distributed ledger technologies, precisely the combination of blockchain and smart contracts, to enable highly secure, private, fast and distributed interaction between administrative domains in the federation process. Along with the designed solution, we developed an experimental prototype that requires simple one-time setup and fast simultaneous registration time for multiple administrative domains. Obtained results show single service deployment times (without considering the deployment time) of around 5 seconds.**

***Index Terms*—blockchain, federation, multi-domain, DLT, NFV**

I. Introduction

5G targets different vertical industries with the goal to enhance and optimize network connectivity. The ITU-R classified these services in three different groups: enhanced mobile broadband (eMBB), massive machine-type communications (mMTC), and ultra-reliable and low-latency communications (URLLC) [1]. This initial classification was further evolved towards vertical specific services.

By leveraging technologies such as as the Network Function Virtualization (NFV) and Network Slicing, vertical specific services can be created to satisfy the requirements for each vertical industry. The H2020 5Growth project [2], as an evolution of the H2020 5G-TRANSFORMER project [3], intents to improve the usability, automation, performance and security in providing services to verticals. Vertical specific services are translated into network services (NFV-NS). The generated NFV-NSs are deployed and orchestrated over the local domain infrastructure and over external multiple domains. The orchestration of services across multiple administrative domains [4] is one of the key features of the 5Growth platform, referred to as *federation*.

With the introduction of distributed ledger technologies (DLT), the application of smart contracts to blockchain technologies paves a way for fully distributed solutions that involve high level of trust, privacy and security among users without the need of central authority to regulate their interactions.

The main goal of the paper is to (*i*) apply efficient, distributed, secure and private DLT solutions for the federation of 5G services and interactions among hosting administrative domains; (*ii*) evaluate the solution through experimental setup, and (*iii*) compare the work with similar existing efforts.

The rest of the paper is organized as follows. Sec. II provides background on the 5Growth platform and DLT solution functions. Sec. III explains the federation concept and how DLT can be applied in a solution. Sec. IV introduces some validation experiments, while Sec. V summarizes obtained results. Finally, in Sec. VI we compare our work to related approaches, before concluding the paper in Sec. VII.

II. Background

A. 5Growth

5Growth is an EU H2020 project with the goal of providing automated deployment and orchestration of customized slices with fulfilled requirements for specific vertical industries (e.g., Industry 4.0, Transportation and Energy, etc.). The project proposes a modular 5Growth architecture shown on Fig. 1, using the 5G-TRANSFORMER platform [3] [5] as a basis. Top-down, the 5Growth architecture is composed of the 5Growth Vertical Slicer (5Gr-VS), the 5Growth Service Orchestrator (5Gr-SO) and the 5Growth Resource Layer (5Gr-RL) with additional components for automation and monitoring processes.

The 5Gr-VS is the entry point for verticals to request their services providing a range of customized parameters through a simple interface with a vertical OSS/BSS. Verticals focus only on the service logic, all network-related and resource orchestration is left to the lower modules of the 5Growth architecture. The 5Gr-VS maps and translates the specific request to a network service (NFV-NS), which is directed to the underlying 5Gr-SO.

The 5Growth Service Orchestrator (5Gr-SO), upon receiving a NFV-NS request, is responsible for instantiation and end-to-end orchestration of the NFV-NS over the single (local) domain or multiple (external) domains while satisfying the requirements of the service. The 5Gr-SO is an extension of the 5G-TRANSFORMER Service Orchestrator [6]. First, the 5Gr-SO runs internal logic for decomposing the requested NFV-NS into smaller segments. Then the 5Gr-SO runs placement and decision algorithms [7] [8] which, based on resource availability and service requirements, can decide for local instantiation or external segment deployment by using *federation*. This work explores the interaction mechanisms between peering domains while using federation. Upon instantiation of all NFV-NS segments in local or external domains, the 5Gr-SO is responsible for life-cycle management of the complete end-to-end NFV-NS. Internal 5Gr-SO procedures are not visible to the 5Gr-VS.

The 5Growth Resource Layer (5Gr-RL) is in charge of managing the local domain resource infrastructure and transport network. As an extension of the 5G-TRANSFORMER Mobile Transport and Computing Platform (5GT-MTP), the 5Gr-RL receives and executes the instantiation or life-cycle management instructions from the 5Gr-SO. At same time, the 5Gr-RL provides to the 5Gr-SO an abstracted view of the underlying infrastructure.

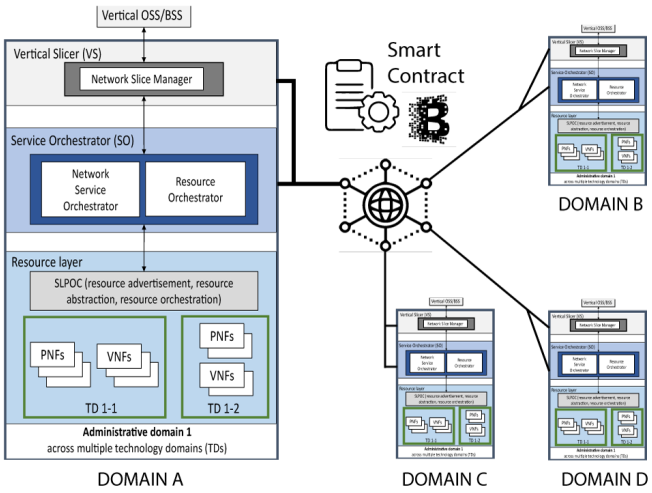

Fig. 1: 5Growth Architecture: Federation using Blockchain

B. Blockchain & Smart-contracts overview

The *blockchain* technology originated as a driving mechanism for Bitcoin [9], providing a distributed and secure system that records every transaction between anonymous users (i.e., distributed ledger). In general, the blockchain technology provides users with distributed time-stamped blocks filled with transactions that can contain any data. Each new block points to the previous block. The block pointers create a chain of blocks in history, referred to as *blockchain*. The blocks are generated by nodes that are interconnected between each other and form a blockchain network. A node can be any (powerful enough) computing device.

Only a single block is created at a time. To maintain single block creation, the nodes run consensus mechanism that validates the addition of a block created by a node in the blockchain. In Bitcoin, the consensus mechanism is called Proof-of-Work (PoW), where a node *mines* a block and the rest of the nodes (miners) validate that the block can be added to the blockchain. The mining process is a continuous calculation of cryptographic hash function until matching target is found. The consensus mechanism plays a key role to enable and maintain distributed and secure way of adding blocks to the chain, without the need of 3rd-party centralized entity. More consensus mechanisms can be applied, such as Proof-of-Stake (PoS), Byzantine Fault Tolerance (BFT), and etc. [10] [11].

There are two types of blockchain networks: permissionless and permissioned. Permissionless blockachain networks are open and referred to as public blockchain networks. Participants are anonymous users that can send cryptographically signed transactions to any node and be included in the immutable blockchain, and publicly visible upon signature verification. Most popular persmissionless blockchain networks are Bitcoin, Ethereum [12] [13], etc. Permissioned blockchain networks are private networks where the participating nodes and the users are known to each other or belong to a central organization, group or etc. Most popular permissioned blockchains are Hyperledger [14] and Corda [15]

Ethereum as a near Turing-complete blockchain, adapts a concept of *Smart Contracts* introduced by Nick Szabo [16]. The smart contract is a set of binary code, similar to a computer application, that runs on top of blockchain. The code contains a set of pre-defined rules that are created by the smart contract creator. Once the smart contract is deployed on the blockchain, it is immutable and operates independently (from its creator) with its own blockchain address. Similar to applications, users send transactions to a smart contract address with an input data included. The smart contract executes itself (bytecode) to produce an output data, which can be permanently stored on the blockchain.

With the adoption of smart contracts, the application of the blockchain technology significantly expands in solving lots of general problems that require a middle-man in the process.

III. Federation concept

A. Federation in 5Growth

As described in Sec. II-A the 5Growth platform is able to deploy and orchestrate NFV-NS over multiple administrative domains (ADs) through the *federation* mechanism [6] [17]. The federation mechanism is used by a consumer domain to deploy network services or allocate of resources over an external provider domain, through a mutual pre-determined trust agreement. Depending on the state of the underlying infrastructure, current orchestration capabilities, specific service geo-footprint (i.e. specific location for service deployment) etc., one AD can decide to deploy part of the service in an external domain. The decision to trigger federation can be executed by the 5Gr-VS or the 5Gr-SO.

Federation requires all involved administrative domains to have mutual cooperation agreements. Typically, these agreements are signed on a business round tables. Upon agreement, the agreed parties trust each other and establish a peer-to-peer connection between themselves or define a trusted centralized entity to manage their interactions.

Federation interactions can be executed in a centralized, decentralized or distributed manner. A **centralized solution** means that all involved ADs have to establish mutual agreements and trust a centralized entity located in a neutral location. The centralized entity is administrated by the joint community of ADs and oversees the federation interaction, acting as neutral ”middle-man”. The positive point is that it is a highly trusty and scalable solution, however the drawback is that setting it up takes significant time, joint effort, continuous maintenance and it is a single-point of failure system. A **decentralized solution** is the most simple one, at the cost of the lowest scalability. An AD establishes peer-to-peer connectivity with each external AD. This implies each new connection is followed by business agreement and connectivity set-up (e.g., 50 connections would take at least 50 or more days). A **distributed solution** is a hybrid approach, more similar to the centralized solution, where the central entity is distributed in each AD. The joint set-up effort is approximately the same as in the centralized solution with same benefits and without having a single-point of failure.

For all of the previously defined interactions, federation has several procedures that are executed to successfully federate services or resources:

- **Registration** - initial procedure through which the administrative domains establish their peer-to-peer interconnectivity or register to a central entity. The registration procedure characterizes the type of federation, which can be relatively open or strictly closed. As an open federation can be considered when external new domains can more easily register to the peer-to-peer or centralized interaction. Interaction between registered ADs in open federation requires high-level of security. The closed federation includes pre-defined participants with strict policies and rules, manually set and defined by the ADs. In this case, the trust level between participants is higher which requires lower security policies employed.
- **Discovery** - in this procedure the participating ADs periodically broadcast or exchange among themselves information on their capabilities to provide services or (computing/networking) resources. Since the exchanged information is abstracted and more descriptive then precise (e.g., total number of CPUs, catalogue of NFV-NSs, etc.), each AD can create a global view of the available service or resources at the external ADs. The discovery procedure is constantly running and generating the updated view of externally available services/resources.
- **Announcement** - this procedure is triggered by the 5Gr-VS or 5Gr-OS, once it has been decided the need to federate part of a service (or allocate resources) in an external peering domain. In that case, the AD acting as a consumer broadcasts an announcement offer to all potential provider ADs. The announcement conveys the requirements for a given service or set of resources. In the centralized case, the central entity is used as a proxy.
- **Negotiation** - the potential provider ADs receive the announced offer, analyze if they can satisfy the requirements and send back a positive or negative answer. The positive answer includes the pricing of the service.
- **Acceptance & deployment** - the consumer AD collects all positive answers and chooses a single provider domain. The selection process is entirely left to the consumer AD's internal policies and preferences. The consumer domain sends an acceptance reply to the chosen provider AD. The provider AD begins the deployment of the requested *federated* service.
- **Usage & Charging** - once the provider AD deploys the federated service, notifies the consumer AD and sends all necessary information for the consumer AD to include the federated service as part of the end-to-end service deployment. From that on, the provider AD starts charging for the federated service during its life-cycle, until it is terminated.

Please note that the security/privacy and trust among the participating ADs is vital in all the aforementioned procedures. Actually, due to competitive reasons, any AD (e.g., mobile operators, cloud providers, etc.,) would not reveal much information regarding the underlying internal infrastructure or the full capabilities for service deployments.

B. Federation using DLT (Blockchain + Smart Contract)

Establishing a centralized entity for maintaining the federation process can be costly and can run into several issues. Questions such as: *where would the central entity be placed?; who would govern it?; how to avoid the single-point of failure system?;* are relevant and the solution is leaning towards a peer-to-peer federation solution. In a peer-to-peer federation approach, the main impediments lie in realization of the business-level agreements, establishing and maintaining the inter-connection with a number of ADs. The dynamics of establishing federation would be more time consuming in the preparation period (e.g., number of business meetings, table sit-downs, negotiating agreements and establishing rules) than the actual usage and benefit from the federation.

The introduction of distributed ledger technologies (i.e., blockchain + smart contracts) opens an opportunity for a fast, secure, trusty and efficient setup of highly distributed and scalable federation solution. Our idea (shown on Fig. 1) is to use a permissioned blockchain where each of the administrative domains runs a single node as part of the permissioned blockchain network. A single generic Federation Smart Contract (SC) is installed to the blockchain to act as a distributed authority. The federation using blockchain provides high level of security and trust. Each AD running a single node is running the same instance of the Federation SC and each line of code is executed at the same time in all nodes involved in the permissioned blockchain network. This adds significant security and trust among the participants.

The design of the Federation SC is essential for guaranteeing privacy of sensitive information to each AD while overseeing the federation procedures that involve all ADs. Every new AD that joins the blockchain network would need to register to the Federation SC with its unique blockchain address for participation in the federation processes. The registration process includes administrative information of the AD itself and service footprint. This way the registration procedure explained in Sec. III-A is satisfied with highly scalable, fast and anonymous solution (shown in Sec. V).

Fig. 2 presents the interactions between the ADs with the Federation SC for a single service federation process. Once all ADs are registered, they can participate in the federation processes as consumers or providers. When a consumer AD creates an announcement or federation offer, it is sent to the Federation SC which records the offer as a new auction on the blockchain. Then, the Federation SC broadcasts the auction to all registered ADs. Note that the address of the consumer AD is hidden in the broadcast announcement. This is done to protect the AD's privacy, preventing the rest of the ADs to passively collect information. Thus, having in mind that all participating ADs would not have any incentive to fully reveal the federation capabilities, the discovery phase is omitted in the design. Instead, we are using a single-blinded reverse

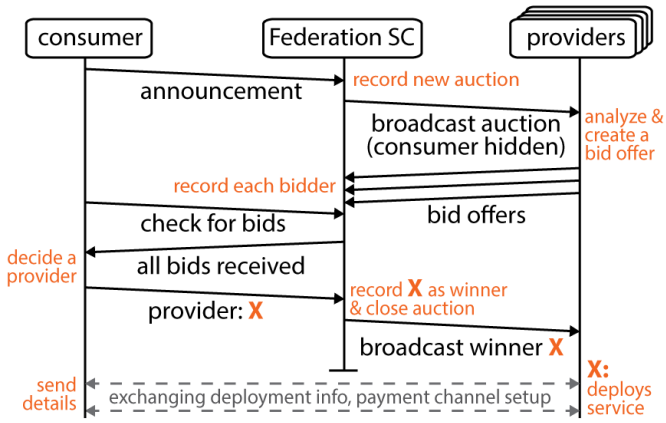

Fig. 2: Sequence message diagram for Federation Smart Contract and administrative domains during federation

auction [18], where a consumer AD anonymously creates an announcement offer and the rest of the potential provider ADs are bidding for it. Therefore, once the broadcast announcement is received, the potential providers analyze the requirements and place a bid offer to the Federation SC. Each received offer is mapped and recorded by the Federation SC.

Our viewpoint is that the Federation SC should be used as a tool, instead as an authority in the process. As a consequence, the bidding process can only closed only by the consumer AD. In this way the consumer AD has the full control and freedom to apply any selection policies (e.g., prioritize given providers, select the lowest price offer, etc.). Hence the consumer AD periodically polls the Federation SC and caches the bidding offers. Once the consumer AD selects a provider AD (e.g., selects the provider X), it closes the auction in the Federation SC. The selected provider X is recorded as winner by the Federation SC, which then sends a notification to the selected provider X and broadcasts notification to all ADs that the specific auction has finished. At this point the negotiation and acceptance phases are completed.

The consumer AD and the selected provider AD directly communicate and share information. The federated service is deployed and included in the end-to-end service by the consumer AD. The procedure of federated service deployment and inclusion to the end-to-end service is similar to the one explained in [19] and it is out of scope for this work.

Upon concluded deployment, the provider AD initiates charging for the federated service. The same permissioned blockchain network can be reused with applying micropayment channels [20] [21]. The micropayment channel applied on the blockchain can enable single non-bias charging record that is immutable for both the consumer AD and provider AD.

IV. Experimental scenario

To evaluate our solution, we developed an experimental system consisting of two machines: client and node. The node machine is an Ubuntu 16.04 with 8 GB RAM and Intel®Core™i5-4430 CPU @ 3.00GHz. The client machine is a Ubuntu 16.04 with 16 GB RAM and Intel®DualCore™i7-7560 CPU @ 2.40GHz.

The client machine emulates from 2 to 100 administrative domains through a nodeJS/React [22] web server application. On the node machine there is an Ethereum private chain. We have developed the Federation SC using Solidity [23]. The node and the client machines run in a LAN network. The client machine is accessing the Ethereum network using the web3 library [24]. The Ethereum network is using the default Proof-of-Work (PoW) consensus protocol with default difficulty adaptation.

We executed four sets of experiments:

1) We measured the time it takes for the administrative domains to register to the Federation SC.
2) We measured the federation time for a single federation announcement offer illustrated in Fig. 2. The measured time is from the start of the experiment until the selected provider receives the notification to deploy a service. We repeated the experiment for set of [2, 5, 10, 20, 50, 100] bidding ADs, competing to win the reverse auction process i.e. in the first trial 2 bidding ADs (bidders) compete for the single service, while in the last trial 50 bidders are competing.
3) We repeated the 2nd experiment while creating two federation announcements in sequence. We measured the total time it takes to resolve the two federation announcements while using the set of [2, 5, 10, 20, 50] bidders.
4) To justify the results from the previous experiment and stress the system, we measured the total time it takes to resolve set of [2, 5, 10, 20, 30] announcement offers posted. We used constant number of 50 bidders for each trial.

The policy of the consumer AD choosing the bidder is based on taking the lowest offered price, after the last bidder arrives. As mentioned in Sec. III-B, the consumer AD polls the Federation SC to cache the bids. For example in the case of 50 bidders, there are fifty sequential reading operations to the Federation SC from the consumer AD. Next section presents some experimental results.

V. Results

Fig. 3 shows the results from the first experiment. From the given results, it is clear that the registration time for multiple AD registering at the same time can range from ~ 0.6 seconds up to ~ 3.8 seconds. In reality, the extreme case of 100 ADs registering at the same time is quite uncommon, however it illustrates that even in such a stressed scenario, the setup time takes less than 4 seconds.

Fig. 4 presents the results from the second experiment, showing that a single service federation can be performed within 5 seconds. It also shows that the number of bidders have low impact on the time consumption. For example, the case of 10 bidders performs more than a second better than the cases of 2 or 100 bidders. Mainly this is due to the time trade-off between mining a transaction (i.e. included in Ethereum

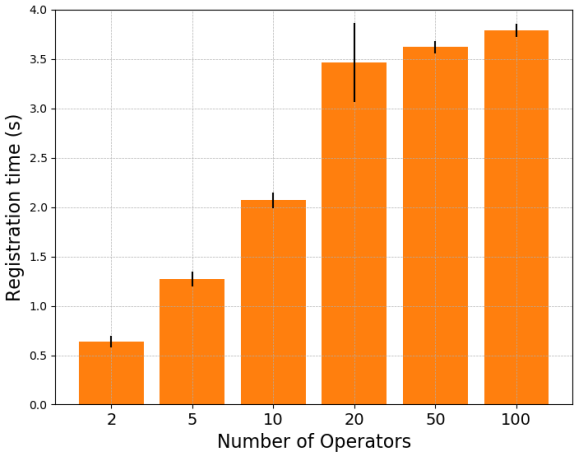

Fig. 3: Registration time for multiple operators registering at the same time instance $(t = 0)$

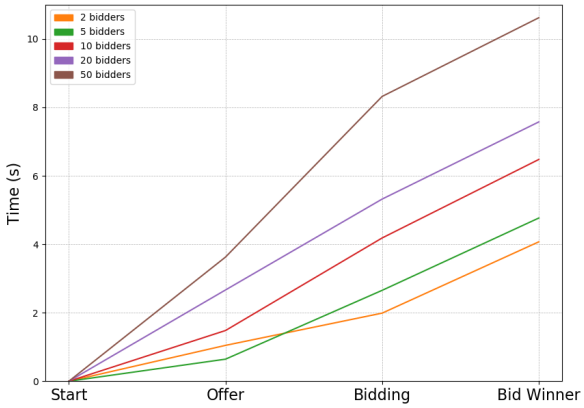

Fig. 5: Federation events for two announcements with multiple bidders

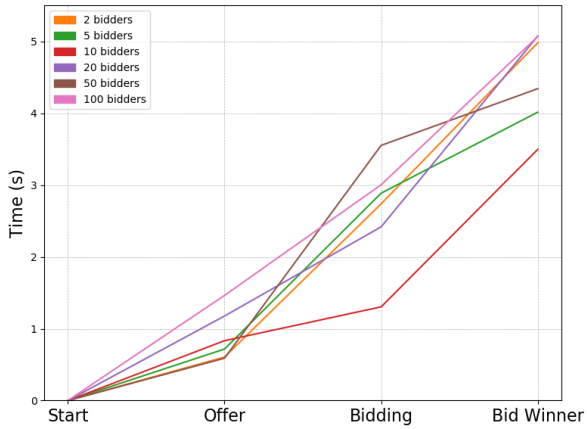

Fig. 4: Federation events for single announcement with multiple bidders

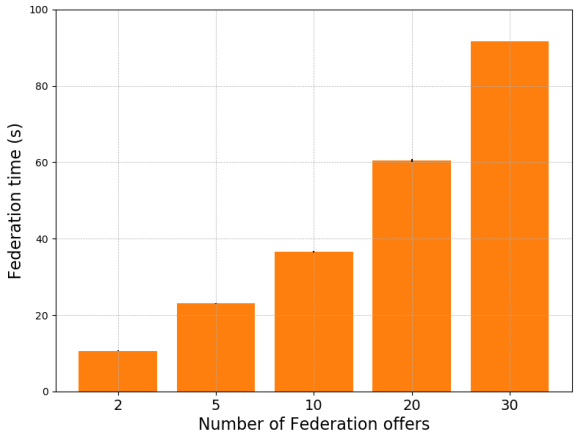

Fig. 6: Federation time for multiple offers with 50 bidders

block), the reading operations (i.e. API calls to read from blocks) and client machine processing. Note that the miner process by default has more incentive in mining faster a block with higher number of transactions (due to higher reward). Thus, the mining time for 2 or 100 transactions can be the same. In the higher bidders case, the time is mostly consumed by the reading operations and client machine processing. This is observed in the next experiments.

The results of the third experiment (Fig. 5) show the federation time of two sequentially announced offers. The shown graph represents the dependency of the reading operations, client machine processing and sequential order. Starting from 2 bidders per federation announcement the total completion time linearly increases, with maximum ~ 11 seconds for 50 bidders per each federation announcement.

If we look at how the federation time increases with an increase of the federation offers, obtained results (Fig. 6) show that there is a monotonic increase of the total federation time as the number of announced offers increases. The system manages to handle 30 sequentially announced offers with 50 bidders per announcement within total of ~ 90 seconds. The monotonic increase of the federation time is due to sequential announcements, the linear increase of reading operations and client side processing, while the mining time is constant.

VI. Related Works

The work in [25] presents similar a approach for orchestrating services in a multi-domain scenario. The work answers the question raised by the work in [26]. Differently from our work, [25] focuses on showcasing only the instantiation of multiple services over multiple domains, while leaving out rest of the federation procedures. Therefore it does not provide experimental results regarding the discovery, announcement, negotiation and acceptance phases.

In [27], authors propose a centralized marketplace where a broker authority, on vertical users' request, creates reverse auctions as new smart contracts. Then the infrastructure providers bid for it off-chain to the broker, then the broker records the bids and elects the winner. Compared to our work, [27] provides a fully centralized solution where the broker is the central point hosting auction processes and delegating service requests to infrastructure providers. In our work, the authority is distributed at the ADs, the smart contract has the goal of providing private and trusty communication/auction process without selective jurisdiction. In [27], a smart contract is setup per auction process by the broker, where in our case it is a one-time setup. The use of public Ethereum network generates a high operational cost.

The authors of [28] present a similar work as [27] plus using additional permissioned blockchain for deployment communication to all infrastructure providers. Compared to our work, [28] the solution is quite complex. The announcements are created on a permissionless network as new smart contract per network sub-slice, which means a single slice or a single network service would have one or more smart contracts deployed. This involves a high cost operation for the permissionless blockchain. Each subslice smart contract collects bids. There is a lack of registration process, although the selected providers exchange deployment information using a permissioned blockchain.

VII. Conclusion and future work

This work proposes a DLT solution (blockchain + smart contract) applied to federation of 5G services. The proposed solution satisfies the need for private, secure, efficient and distributed interaction among administrative domains that federate network services. In addition to the solution design, we have experimentally evaluated it. From the obtained results, it is clear that the one-time system setup and registration time is fast and simple (less than 4 seconds in the worst case). Regardless of the number of bidding provider domains, a single service federation (which involves announcement offer, negotiation and acceptance) is executed within 5 seconds.

As a future work, we are looking how to improve the solution by applying different consensus protocol or different solution (e.g., Hyperledger) and adding full service federation deployment on a real use-case scenario.

Acknowledgment

This work has been partially funded by the EU H2020 5GROWTH Project (grant no. 856709).

References

[1] ITU-R, "IMT Vision – Framework and overall objectives of the future development of IMT for 2020 and beyond M Series Mobile," International Telecommunication Union (ITU), Tech. Rep. REPORT ITU-R M.2410-0, November 2017.

[2] "5growth," accessed on: 28 Feb 2020. [Online]. Available: http://5growth.eu/

[3] A. Oliva et al., "5G-TRANSFORMER: Slicing and Orchestrating Transport Networks for Industry Verticals," *IEEE Communications Magazine*, vol. 56, no. 8, pp. 78 – 84, 2018.

[4] ETSI, "Network Functions Virtualisation (NFV) Release 3; Management and Orchestration; Report on architecture options to support multiple administrative domains," European Telecommunications Standards Institute (ETSI), Group Specification (GS) NFV 028 v3.1.1, January 2018.

[5] L. Valcarenghi et al., "A framework for orchestration and federation of 5g services in a multi-domain scenario," in *Proceedings of the Workshop on Experimentation and Measurements in 5G*, ser. EM-5G'18. New York, NY, USA: ACM, 2018, pp. 19–24. [Online]. Available: http://doi.acm.org/10.1145/3286680.3286684

[6] J. M. et al., "5g-transformer service orchestrator: design, implementation, and evaluation," in *2019 European Conference on Networks and Communications (EuCNC)*, June 2019, pp. 31–36.

[7] K. A. et al., "Resource orchestration of 5g transport networks for vertical industries," in *2018 IEEE 29th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, Sep. 2018, pp. 158–163.

[8] ——, "A q-learning strategy for federation of 5g services," in *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, June 2020.

[9] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," Manubot, Tech. Rep., 2019.

[10] G.-T. Nguyen and K. Kim, "A survey about consensus algorithms used in blockchain." *Journal of Information processing systems*, vol. 14, no. 1, 2018.

[11] L. Bach, B. Mihaljevic, and M. Zagar, "Comparative analysis of blockchain consensus algorithms," in *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*. IEEE, 2018, pp. 1545–1550.

[12] V. Buterin *et al.*, "Ethereum: A next-generation smart contract and decentralized application platform," *URL https://github. com/ethereum/wiki/wiki/% 5BEnglish% 5D-White-Paper*, 2014.

[13] G. Wood *et al.*, "Ethereum: A secure decentralised generalised transaction ledger," *Ethereum project yellow paper*, vol. 151, no. 2014, pp. 1–32, 2014.

[14] E. A. et al., "Hyperledger fabric: a distributed operating system for permissioned blockchains," in *Proceedings of the Thirteenth EuroSys Conference*, 2018, pp. 1–15.

[15] R. B. et al., "Corda: an introduction," *R3 CEV, August*, vol. 1, p. 15, 2016.

[16] N. Szabo, "The idea of smart contracts," *Nick Szabo's Papers and Concise Tutorials*, vol. 6, 1997.

[17] X. L. et al., "Service orchestration and federation for verticals," in *2018 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, April 2018, pp. 260–265.

[18] S. D. Jap, "The impact of online reverse auction design on buyer–supplier relationships," *Journal of Marketing*, vol. 71, no. 1, pp. 146–159, 2007. [Online]. Available: https://doi.org/10.1509/jmkg.71.1.146

[19] J. B. et al., "Composing services in 5g-transformer," in *Proceedings of the Twentieth ACM International Symposium on Mobile Ad Hoc Networking and Computing*, ser. Mobihoc '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 407–408.

[20] G. Avarikioti, F. Laufenberg, J. Sliwinski, Y. Wang, and R. Wattenhofer, "Towards secure and efficient payment channels," *arXiv preprint arXiv:1811.12740*, 2018.

[21] H. G. et al., "An efficient micropayment channel on ethereum," in *Data Privacy Management, Cryptocurrencies and Blockchain Technology*. Springer International Publishing, 2019, pp. 211–218.

[22] F. OpenSource, "React – a javascript library for building user interfaces." [Online]. Available: https://reactjs.org/

[23] "Solidity," accessed on: 27 Feb 2020. [Online]. Available: https://solidity.readthedocs.io/en/v0.6.3/

[24] "web3," accessed on: 27 Feb 2020. [Online]. Available: https://web3js.readthedocs.io/en/v1.2.6/

[25] R. V. R. et al., "Blockchain-based decentralized applications for multiple administrative domain networking," *IEEE Communications Standards Magazine*, vol. 2, no. 3, pp. 29–37, SEPTEMBER 2018.

[26] K. Wüst and A. Gervais, "Do you need a blockchain?" in *2018 Crypto Valley Conference on Blockchain Technology (CVCBT)*, June 2018, pp. 45–54.

[27] M. F. F. et al., "Brain: Blockchain-based reverse auction for infrastructure supply in virtual network functions-as-a -service," in *2019 IFIP Networking Conference (IFIP Networking)*, May 2019, pp. 1–9.

[28] N. B. et al., "A blockchain-based network slice broker for 5g services," *IEEE Networking Letters*, vol. 1, no. 3, pp. 99–102, 2019.