BRAIN: Blockchain-based Reverse Auction for Infrastructure Supply in Virtual Network Functions-as-a-Service

Muriel Figueredo Franco[1], Eder John Scheid[1], Lisandro Zambenedetti Granville[2], Burkhard Stiller[1]
[1] *Communication Systems Group CSG, Department of Informatics IfI, University of Zurich UZH*
Binzmhlestrasse 14, CH-8050 Zrich, Switzerland
[2]*Computer Networks Group, Institute of Informatics INF, Federal University of Rio Grande do Sul UFRGS*
Av. Bento Gonçalves, 9500, Porto Alegre, Brazil
[franco|scheid|stiller]@ifi.uzh.ch, granville@inf.ufrgs.br

***Abstract*—Network Functions Virtualization (NFV) is transforming the way in which network operators acquire and manage network services. By using virtualization technologies to move packet processing from dedicated hardware to software, NFV has introduced a new market focused on the offer and distribution of Virtual Network Functions (VNF). Infrastructure Providers (InP) can benefit from an NFV market by providing their infrastructures to fulfill demands of end-users that, in turn, acquire VNFs-as-a-Service (VNFaaS). In this context, solutions that promote the competition between InPs can lead to lower prices, while increasing VNF performance to accommodate specific demands of end-users. In this paper, BRAIN, a blockchain-based reverse auction is presented to introduce an auditable solution in which InPs can compete to host VNFs taking into account the demands of each particular end-user. Such a solution helps reduce costs involved in VNF's commercialization and also monetize NFV-enabled infrastructures. BRAIN is supported by a case study that provides evidence of the solution's feasibility and effectiveness. A discussion regarding blockchain advantages and drawbacks in this use-case (*e.g.*, , additional costs and time) concludes this paper.**

Keywords – Network Functions Virtualization, Virtual Network Functions-as-a-Service, Infrastructure Supply, Blockchain, Smart Contract.

I. Introduction

Network Functions Virtualization (NFV) transforms the way that computer networks are built and operated by decoupling packet processing from proprietary hardware middleboxes to Virtual Network Functions (VNF) running on standard off-the-shelf devices [1]. Through the use of virtualization technologies in networks, operators can deploy custom network services composed of VNFs (*e.g.*, firewalls, load balancers, and routers) to meet specific user demands and to minimize network disruption. As a consequence, NFV helps provide service delivery with an accelerated time-to-market and agility, while reducing both capital and operational expenditures (CAPEX and OPEX) [2].

NFV has been attracting the attention from both academy and industry, but not only because of its technological potential, but also because of economic opportunities around the NFV market, which include the offering, distribution, and execution of VNFs [3] by third party companies. As such, NFV also creates new businesses. That includes, for example, opportunities for service providers to offer VNF-as-a-Service (VNFaaS) to end-users [4]. In that model, end-users can quickly contract and deploy VNFs from public/private catalogs (*i.e.*, marketplaces) to support end-user demands. Also, VNF developers can profit by announcing their VNF implementations in marketplaces so that interested end-users can purchase them. Such marketplaces can be as simple as only providing VNFs to be downloaded and instantiated at end-users' infrastructure, or it can also be as complex as, besides offering VNFs, providing physical substrate and lifecycle management of VNFs. Efforts in such a direction have been made [5] to propose solutions that simplify the process of VNFs offering, acquisition, and management, thus, contributing to the broad adoption of NFV, while introducing business opportunities. However, the research on models to efficiently host, audit, and improve revenue from VNFaaS and NFV-enabled infrastructures is still in its early days [6].

Once the end-user acquires desired VNFs, they need to be deployed in an NFV-enabled infrastructure. If the end-user does not own a physical substrate to host acquired VNFs, the user can still choose between hiring generic infrastructures (*e.g.*, Amazon AWS or Microsoft Azure) or marketplaces' infrastructures. In both cases, companies expose their services conditions and prices usually in an open fashion, *i.e.*, any user can observe the market and pick the more affordable or suitable infrastructure. Competition, in this case, is open but rather static: prices are not tailored according to end-users' demands. This paper argues, however, that the introduction of strategies enabling real-time, user-tailored competition between Infrastructure Providers (InP) favors lower prices and also contributes to the expansion of marketplaces for VNF while meeting specific demands of end-users [7]. As such, both established and new InPs can achieve a large audience and offer their infrastructures by estimating costs based on each particular demand according to their current business model (*e.g.*, lower prices for a firewall with larger memory than one with CPU core demands). To address such issues, a reverse auction mechanism [8] was proposed, in order to enable an efficient and flexible approach to support fair and auditable competition between providers. Such an approach has been used along the years, for example, by governments to enable a fair and transparent competition between suppliers.

978-3-903176-16-4 © 2019 IFIP

Reverse auction is a helpful tool in the process of choosing the infrastructure candidate to host a VNF. In a reverse auction, distinctly from traditional auctions, sellers compete for buyers, and the auction winner will be the seller that provides the lowest price for the better-tailored service [9]. Such a strategy allows InPs to monetize their infrastructure by receiving requests to host VNFs and give a sealed-bid based on their business algorithms, *i.e.*, InPs can estimate, based on the VNF's and user's demands, the costs and efforts, according to their configurations, to supply infrastructures and resources for each request. Hence, after the bidding process, the auction owner can recommend, based on the best price, the appropriate infrastructure to host VNFs in order to supply the request demands. To support and provide trust for such reverse auctions for VNFs, blockchains and Smart Contracts (SC) can be employed, since they offer immutable and permanent records, which allow interested parties to audit and trust in the data persisted [10]. Thus, an SC can be defined to describe end-users and VNFs requirements that can automate and enforce obligations [11] agreed upon by stakeholders (*i.e.*, infrastructure providers, end-users, and marketplaces) before a VNF deployment.

In this paper, BRAIN, a **B**lockchain-based **R**everse **A**uction solution for **I**nfrastructure supply in **N**FV scenarios, is introduced to address the challenge of discovery and selection of infrastructures being able to efficiently host a VNF regarding user-specific demands. This solution facilitates InPs' competition and allows them to monetize their computational resources by hosting end-users' acquired VNFs. End-users also benefit from lower prices and high availability of heterogeneous InPs to supply their dynamic demands. Such a solution correlates aspects of the infrastructure, such as resource availability, price, and capacity of fulfilling specific end-user and VNF requirements (*e.g.*, minimum resources, geolocalization, and maximum latency) to create an SC that allows InPs to obtain these requirements information and place bids. Based on this, and after the bidding process, the solution computes the best candidate infrastructure (in terms of costs) to run a VNF regarding the criteria specified in the SC. The process of determining and placing bid values of each competitor, based on their resources and business models, is also detailed. Furthermore, relevant information from the auction (*e.g.*, who contracts and which infrastructure is hosting the VNF) are stored in a blockchain to maintain the content of the immutable record of contract clauses (*i.e.*, SC) and to support future auditing.

The remainder of this paper is organized as follows. Background on blockchain and SCs is presented in Section II. Related work is reviewed in Section III. Section IV introduces the BRAIN solution, and architectural details are provided. Section V presents a prototype designed and implemented to demonstrate the feasibility of the proposed solution, including a case study. Finally, Section V draws conclusions and comments on future work.

II. Background

Blockchain was initially conceived as a distributed ledger to be the backbone of the Bitcoin cryptocurrency [12]. However, its capacity to provide a trustworthy, decentralized, and immutable records has attracted the attention of both industry and academia [10]. Blockchain has several benefits, which include: *(i) decentralization*, which results in transactions validation without the need of trusted intermediaries, *(ii) transparency*, to everyone observe what is on the blockchain and thus allowing auditing the records, *(iii) immutability*, which represents that once a data has been recorded into the blockchain, it is almost impossible to be changed without leaving traces, and *(iv) high availability*, ensured by a blockchain replication on thousands of nodes in a peer-to-peer network.

Cryptography is key to blockchain implementation. The blockchain is an ordered list of blocks, where its cryptographic hash identifies blocks. Each block is chained to the block that came before it, which creates a dependency between each block. If an attacker wants to change a data or transaction, he/she must spend a massive amount of computational resources to change every block back to the beginning of the chain (*i.e.*, the genesis block). Thus, once a block is created and appended to the blockchain, the transactions in that block virtually cannot be changed or reverted. That ensures the integrity of the data and transactions occurred in a blockchain.

The second generation of blockchains (*e.g.*, Ethereum [13]) implements the concept of Smart Contracts (SC), which are executable codes that run inside a blockchain to facilitate, execute, and enforce the terms of an agreement [11]. The fees involved in SC are lower than traditional systems that require a trusted intermediary (*e.g.*, approaches based on third-party regulators). In an SC, data structures and algorithms can be developed to store information and execute tasks when the specified conditions are met, such as transferring a determined value between users as a punishment when any part of an agreement is not accomplished. Different programming languages are available to build SC (*e.g.*, Solidity and Liquidity) [14]. The choice of a language depends directly on the objectives and blockchain support. For example, in this work, Solidity – a high-level Turing language for implementing SC – is used because it has been designed focusing on Ethereum, which is a popular blockchain platform for using SCs.

After being implemented, the code of an SC for Ethereum is transformed into a stack-based bytecode language code and executed in an Ethereum Virtual Machine (EVM). Next, the blockchain users can, for example, interact on the SC via an Ethereum's Remote Procedure Call (RPC) server to request and insert new information. In order to ensure immutable records provided by the blockchain, every input must be recorded as a transaction in the Ethereum blockchain. Therefore, after being recorded, the information can be validated on the blockchain by interested parties. In such a direction, SCs can be used to implement reverse auction mechanisms as proposed in this work, but also to support any solution that requires trust and automatic execution of contract clauses without the requirement of all involved nodes trusting on one another.

The concept of reverse auctions is also important to this work. Reverse auctions can be described as a real-time competitive bidding process where the buyer sends a request for quotation and suppliers issue bids according to the buyers' demands [15]. The winning bidder turns to be the seller (*i.e.*, provider) that offers the lowest price. Thus, in a reverse auction, buyers and sellers reverse their traditional roles from conventional auctions, *i.e.*, sellers react to the buyer's needs.

This kind of auction is widely used, for example, by public sector agencies and companies to find suppliers [16].

III. Related Work

Marketplaces for network softwarization motivate the introduction of new network paradigms. In the context of NFV, Bondan *et al.* [6] introduced a marketplace and federated ecosystem for the distribution and execution of VNFs. Similarly, Xilouris *et al.* VNFaaS [4] proposed an integrated architecture to provide. Also, D'oro *et al.* [5] used game theory in a distributed solution for NFV that accounts for monetary costs, communication latency and congestion of computational resources. However, ecosystems to support such marketplaces demands but also to support efficient business models becomes imperative.

Although previous work covers important aspects related to VNFaaS and NFV-enabled business, they do not provide discussions for accounting, pricing, and resources provisioning. Furthermore, infrastructure providers are not yet entirely included inside any business model. Regarding business opportunities, the authors of this paper argue that the broad deployment of future network architectures based on NFV depends mostly on the success of resource allocation [17]; several works have been exploring such a problem. Tasiopoulos *et al.* [18] presented a resource management framework for NFV based service function chaining. Similarly, Jakaria *et al.* [19] introduced a solution for resource management in VNFaaS scenarios. Others solutions employed applied artificial intelligence [20] and queuing network models [21] also to address the resource issue. Nevertheless, none of them can be fully applied in the context of future marketplaces because of their business particularities and demands. Such solutions focus on optimized management of resources location, but they do not worry about infrastructure supply regarding the VNFaaS landscape.

One helpful approach for the NFV infrastructure supply is the mechanisms for auctions. Several authors have studied such mechanisms in areas as distinct as economics and computer science. In computer networks, auction mechanisms are already applied as an enabler to provide competition between providers to supply distinct technologies (*e.g.*, in wireless networks [22] and cloud computing [23]). In the context of NFV, Gu *et al.* [24] were the first to design an auction mechanism for the dynamic provisioning and pricing of NFV service chains in a data center. Also, Zhang *et al.* [25] proposed an online stochastic auction mechanism for on-demand service chain provisioning and pricing at an NFV provider. Both solutions explore auctions to introduce features for NFV-enabled networks. However, none of the noticed solutions focus on end-users particular demands, on specifications of each VNF (*e.g.*, information available in such descriptors), nor in providing reliable records about such auctions and the stakeholders' behaviors.

In such a direction, several works advocate in favor of blockchain and its positive impact on society and industry [26] by providing a trustworthy, decentralized, and permanent data storage. Few solutions have been exploiting blockchain to tackle NFV issues. In one of them, Alvarenga *et al.* [27] proposes a blockchain-based architecture for secure management, configuration, and migration of VNFs. That work is part of an open source project [28] for providing safe chaining of VNFs by using blockchain. In another context, Scheid *et al.* [29] used SCs for automatic Service Level Agreement (SLA) compensation in NFV-enabled networks. Other NFV issues such as accounting, resource allocation, and pricing of VNFs have not been addressed yet. Besides, no work examines blockchain and SCs as business models enablers for NFV scenarios.

IV. Design of the BRAIN solution

The new blockchain-based reverse auction solution proposed (*i.e.*, BRAIN), uses SCs to provide competition and reliable records. Such records contain agreements between end-users and InPs during the auction and even bids' history. Thus, end-users can benefit from lower prices, while InPs can offer their infrastructure for a broader audience. One feature provided by such solution determines a straightforward path to integrate it with generic marketplaces. To employ such a solution successfully, few components should be inserted inside VNFs catalogs, while the remainder of the solution is running in a distributed way. Basically, marketplaces need to implement a parser to create SCs regarding available information and run a component to control the auction (*i.e.*, auctioneer).

BRAIN explores the concept of the English reverse auction [30] to define how bids are processed. Bidders place their bid in a sealed envelope and simultaneously hand them to the *Auctioneer*. Envelops are then opened and the bidder with the lowest bid wins, paying the amount bid. Based on this, concerning a VNFaaS scenario, InPs will send their bids without knowing the amounts placed by other bidders. All of such bids will be recorded in the SC, which will reveal the winning bid only when the auction ends. At the conclusion of the auction, the information about the auction will be available to the other bidders for audition purposes. Such reverse auction is supported by a blockchain implementation that guarantees a joint trust assumption between InPs and end-users. Therefore, each bid is processed as a blockchain transaction from the InP to the SC defining the correspondent auction.

Figure 1 presents the architecture of the BRAIN solution. White boxes represent optional components that should be implemented by marketplaces or InPs that desire to use the proposed auction solution, while the dark gray boxes highlight the BRAIN components. The *NFV broker* the entity that manages and offers VNFs. Thus, current marketplaces can be viewed as a broker implementing the auction components inside of their architecture to support BRAIN. Additionally, every candidate NFV-enabled infrastructure is implemented following the European Telecommunications Standards Institute (ETSI) standards [31] for NFV management and orchestration. Thus, InPs in such approach are able to host VNFs and end-users to sustain their demands when renting/deploying VNFs, since own infrastructure is not available. In addition, this may simplify VNF deployment after the auction conclusion, the chosen InP is aware of the VNF configurations, and code can be quickly deployed to offer the service according to end-user preferences.

The flow of the architecture is described as end-users access a catalog (*i.e.*, marketplace) firstly and acquire a VNF. Next, priorities defined by end-users during the acquisition

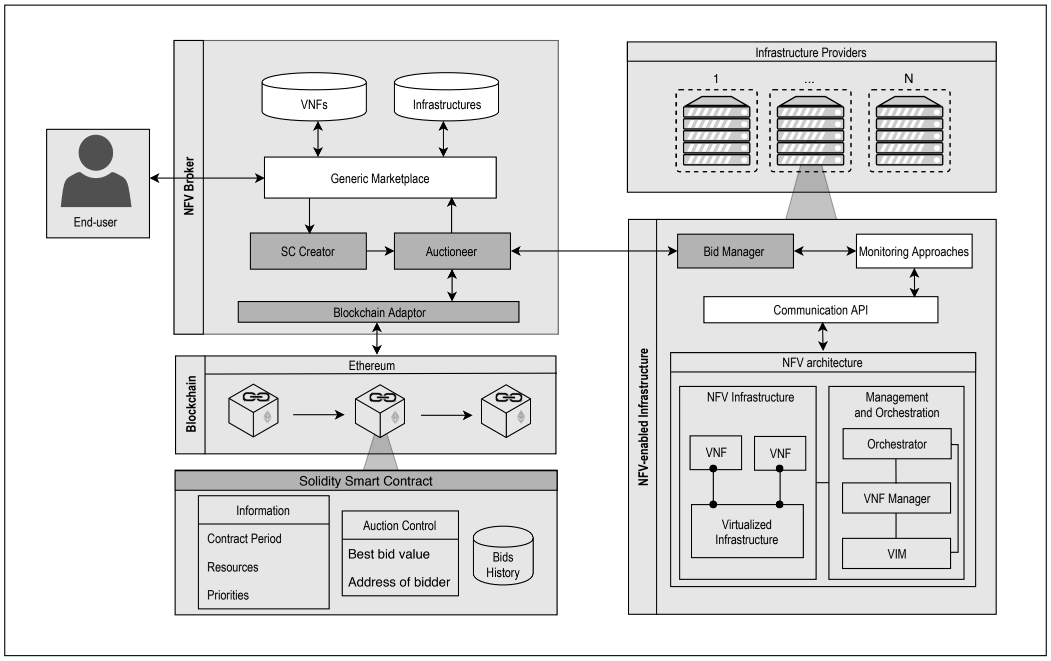

Fig. 1: BRAIN Architecture

process and requirements of the selected VNF are sent to be processed by the *SC Creator*. Then, the *SC Creator* processes the information received and creates an SC according to the specifications. After its creation, the SC is sent to the *Auctioneer*, which deploys the SC in the blockchain and communicates the InPs, via an interface provided by the *Bid Manager*, that an auction is opened. In addition to establishing communication, the *Bid Manager* also implements algorithms that will issue automated bids according to their configuration and the infrastructure status (*e.g.*, resources available and costs). The *Auctioneer* can send a command to the SC to finish the auction and return the best bid. Finally, after knowing which is the best infrastructure, the *Auctioneer* forwards it to the marketplace and the process of VNF deployment can be started as implemented in the NFV Broker.

A. Smart Contract (SC) Creator

The *SC Creator* component is in charge of mapping VNFs requirements and end-users' priorities into an SC format that is deployed in a blockchain to be available for InPs to obtain information related to the auction and bids issued. After the definition and creation of the SC, it is forwarded to the *Auctioneer* to move on with the deployment and the auction call phase.

The creation of the SC takes into account information extracted from two description files: *(i)* VNF Descriptor (VNFD), describing a VNF with regard to its deployment and operational requirements (*e.g.*, minimum resources), and *(ii)* Priorities Descriptor (PD), a JSON file that is generated by the *NFV broker* to describe the particular conditions defined by end-users, *e.g.*, contract time, additional resources, and geolocalization preferences. VNFD follows the ETSI standard [31], while the concept of PD files is proposed here as a JSON file representing these particular priorities of the end-user.

The SC provides functions that can be called by InPs to obtain information and submit bids. In the same way, the *Auctioneer* can terminate the auction, knowing which is the address in the blockchain (*i.e.*, InP address) and also what value of the bid won the auction. Table I presents an overview of each function available to interact with the SC. In addition to its parameters and data returned, each function is also classified in terms of being payable or not, which describes whether there is a monetary cost to call the function (*i.e.*, payable) or not according to what is defined by the blockchain (*e.g.*, get functions are free of charges by default to obtain information from the auction).

TABLE I: Overview of smart contract functions

Call	Parameters	Payable	Data Returned
submitBid	SC address bid value bidder address	Yes	-
getInfo	SC address	No	VNF name VNF type VNF developer Contract period
getResources	SC address	No	Memory Disk CPU cores
getPriorities	SC address	No	Additional memory Additional disk Additional CPU cores Latecy max Packets per second capacity
getWinner	SC address	No	Bid value Winner bidder address

Two address parameters are required to execute the operations. An SC address is a unique ID identifying the contract and its information. InPs must also have a bidder address of the blockchain to issue auto-signed bids. Such addresses are an SHA-256 hash format generated from a cryptography public key representing one address inside the blockchain. Thus, in each auction, all bids and information will be securely recorded inside the blockchain for decisions related to auctions or future audition. The information returned by each function presents end-users' demands that were transformed into an SC. All related information is extracted from descriptor files provided by the marketplace. The VNF type and contract period, for example, are to be defined during the VNF acquisition inside of a PD file and can be used by InPs to promote discounts (*e.g.*, one month contract for a Firewall can benefit from a price reduction). VNFD can be available inside the VNF Package representing each VNF offered within the catalog.

Besides minimum resources, additional resource requests and specific capacities (*e.g.*, processes determined number of packets per second) can be described inside PD files to ensure that the VNF will support periods with a highly unusual demand. Further, a maximum latency can be defined to enforce that only InPs with a low latency will be able to provide the infrastructure. It is important to highlight that the *SC Creator* can deal with any information in a JSON format. Based on this, descriptors can be extended to provide additional information to be used during bid decision.

B. Auctioneer

The *Auctioneer* is the component responsible for supporting the communication between InPs and the SC. It announces to interested InPs that an auction is open and also decides when the auction will be finished based on the previous configuration. The *Auctioneer* must know the communication interface (*i.e.*, RESTful API) of each InP during the call phase. Such an address can be stored and accessed using two different models: *(i)* a database maintained by the *NFV broker* (*e.g.*, a marketplace for VNFs), where InPs can register for free to participate in the auctions, or *(ii)* a database of infrastructures, where InPs need to pay a fee to be part of these auctions. In the case of *(i)*, the operational costs of the auction should be absorbed by the *NFV broker* mainly, while *(ii)* allows the *Auctioneer* to make revenue.

After knowing the communication interface of InPs candidates, the *Auctioneer* sends the SC address and the respective Application Binary Interface (ABI) to the *Bid Manager* running on the InP side. Either, the *Bid Manager* can issue information requests (see Table I) or submit bids as auto-signed transactions that will be sent to the SC via an RPC server provided by the blockchain. It is also important to highlight that the *Auctioneer* component can deal with a large number of SCs at the same time. Different auctions can be identified by their unique SC address allocated within the blockchain. Each SC created has three distinct phases that describe the *Auctioneer*'s behavior along the time.

1) Opening and Call: The auction starts when the SC is deployed into the blockchain. Henceforth, the call phase is defined as the period in which the *Auctioneer* announces to each InPs that an SC is available to receive bids. Such an announcement can be directed to selected candidates or sent as a broadcast announcement. It only depends on the business model defined by the marketplace.

2) Bidding Phase: The bidder requests the auction information via RPC server and issues its bid. Then, the SC compares the bid with the current best bid, thus detecting that a new best bid is received and who is the bidder. If a bidder is not able to provide a bid (*e.g.*, based on a blacklist), the SC will automatically revoke it. During this phase, the competitors do not know how many providers are issuing bids nor the value of their bids.

3) Closing and Decision: As the *Auctioneer* is the owner of the SC, it is also in charge of terminating the auction after on a predefined period (*e.g.*, time, a minimum number of bids, or a price goal). Such a request is recorded as a transaction in the blockchain to the SC address and defines the auction as closed. Finally, the winner is revealed by the SC, and such information is sent to the *NFV broker* to proceed with the VNF deployment process.

C. Bid Manager

The *Bid Manager* runs inside each infrastructure and provides a RESTful API for receiving information when a new SC is available. The *Bid Manager* can make RPC calls to the SC interface, requesting for information (*e.g.*, minimum resources and priorities) and mapping the data returned to be used as input to the bid calculation.

A bid algorithm implemented as part of the *Bid Manager* is responsible for calculating and issuing the bid according to business variables. It maintains crucial business information such as price per unit of each resource, discounts for additional resources, and which type of VNF are considered a business priority (*e.g.*, some InPs can have optimized infrastructures to host firewalls). InP should configure all of such information according to its business directions. If one infrastructure has sufficient memory available, for example, it will reflect a lower value for memory in bid calculations.

The current bid algorithm supports three contract periods that can be specified inside the PD file. In this way, special prices can be defined for different contract periods of time. These periods are described as: *(i) hour* to contract less than 24 hours, *(ii) day* to contract greater than 24 hours but less than seven days, and *(iii) week* that represents each period larger than seven days. The bid algorithm calculates the final bid integrating risks and advantages of the contract (*e.g.*, contracts with higher periods are most attractive to some InPs) according to infrastructure policies and business models. As mentioned before, customizations can be made in the bidding algorithm according to InP preferences. In such a direction, an InP can also implement its bid algorithm. Thus, such an implementation only should be able to make RPC calls to obtain information and issue bids to the SC interface.

After bid calculations, each bid is auto-signed by the InP with its private key and the signed transaction is sent to the SC. Thus, one bid cannot be repudiated by an InP, and contract clauses can be executed to punish the InP that do not meet the agreements (*e.g.*, pay a reward or be added in a blacklist). Always, all bids are recorded inside the SC and can be audited to validate the competition. Arbitrary users can see the history of the auction containing bids and each bidder address to understand how the winner was decided. It is important to mention that the data flow is the same to sustain multiple users requests. Thus, a new SC has to be created to address each user request separately.

V. Prototype and Evaluation

A prototype was implemented to demonstrate the feasibility and effectiveness of the BRAIN solution. The code for the *SC Creator*, the *Auctioneer*, and an example of an algorithm to calculate the bids are available online [32], including examples of the PD file and SCs. The Python programming language was chosen in its latest version 3.7.1 to implement the auction. This decision was due to the possible support of multiple programming paradigms and focuses on code readability. The development integrates well-defined libraries to deal with blockchains and SCs. Flask 1.0.2 was used to implement the RESTful APIs supporting the communication between components.

The Ethereum blockchain platform runs an SC written in diverse languages [13]. SCs were coded using Solidity 0.4.24, Ethereum's contract-oriented programming language. In order to validate such implementations, the Ganache Framework in its latest version was configured and executed to simulate the blockchain where the SC will runs. It allows for the creation of a development blockchain to run tests, to execute commands, and to inspect states. Ganache itself provides the RPC server. Thus, a behavior similar to the Ethereum blockchain and its operations can be emulated to support calls from all auction components. The case study scenario applies five 20 byte addresses, which were generated in Ganache to be allocated to distributed InPs. One address is automatically generated when an SC is deployed in the blockchain. Every call and transaction can be executed in the Ethereum blockchain through functions implemented in the SC. This case study considers a public catalog of InPs, which allows InPs to register without fees to be part of auctions (*i.e.*, the Auctioneer does not make revenue).

```
function getInfo() public constant returns (string,
     string, string, string) {
  vnf_name = "Firewall";
        vnf_type = "Security";
        vnf_developer = "Muriel Figueredo Franco";
        contract_period = "month";
        return (vnf_name, vnf_type, vnf_developer,
             contract_period);
}

function getResources() public constant returns (int,
     int, int) {
        int mem = 4;
        int disk = 2;
        int cpu = 2;
        return (mem, disk, cpu);
}

function getPriorities() public constant returns (
     string, int, int, int, int, int) {
        country = "Any";
        int additional_memory = 0;
        int additional_disk = 0;
        int additional_cpu = 2;
        int latency_max = 20;
        int packets_per_second = 0;
        return (country, additional_memory,
             additional_disk, additional_cpu,
             latency_max, packets_per_second);
 }
```

Listing 1: Functions used by InPs to request information from SCs

There are three main functions available inside the SC that allows InPs to obtain user requirements information. The code of these functions, as presented on Listing 1, defines *getInfo()*, which returns information related to the VNF development and the contract period. In another function, details on resources specified by the VNFD of the acquired VNF can be accessed. Such a function is *getResources()* and it returns the memory RAM (GB), disk usage (GB), and CPU cores required. Finally, *getPriorities()* returns customized configurations required by the end-user. Thus, end-users' particular demands include specific country requests in which the VNF must be deployed or additional resources that must be available to supply high-demand periods.

InPs can also submit via *submitBid(bid_value)* to the RPC server. This call must be sent as a signed-transaction to be officially processed by the SC. Hence, the InP will calculate its bid value and send it to the SC. The SC will receive such a value and with the address of the InP that signed the transaction. The SC will compare the value of the bid with the current best bid. If the new bid is lower, the best bid will be updated with the new bid and the address of the InP that signed the bid. Thus, the SC will maintain the best bid value and the owner of the bid. These variables can be retrieved by the *Auctioneer*, when the auction terminates. For auditing purposes, each change in the best bid is persisted inside the blockchain. Listing 2 presents the function code performing the bid submission.

```
function submitBid(uint bid) public returns (bool) {
     if (winner == false) {
            if (bid < bestBid) {
                    bestBid = bid;
                    bid_owner = msg.sender;
                    return true;
            }
            else {
                    return false;
            }
     }
     else {
            return false;
     }
}
```

Listing 2: Function to InPs submit a bid

```
function endAuction() private returns (bool) {
        if !(winner) {
                winner = true;
        }
        else {
                return false;
        }
            return true;
    }
function getWinner() private constant returns (uint,
     address) {
        if (winner) {
                return (bestBid, bid_owner);
        }
        else {
                return false;
        }
}
```

Listing 3: Functions to control the auction

Since an auction starts when the *Auctioneer* deploys the SC, the *Auctioneer* offers two private functions to control and obtain information about the auction: *endAuction()* and *getWinner()*. The first function (*cf.* Listing 3) changes a boolean variable responsible for defining whether there is a winner (when true) or that the auction is still open (when false). Hence, the *Auctioneer* can be in charge of invoking a function when the auction should be terminated, or it can be implemented to follow a policy to automatically terminate the auction (*e.g.*, based on time or bid values). Finally, the *getWinner()* function can be called to return the value of the best bid and also the blockchain address of the InP that gives the lower bid.

A. *Case Study*

Let us suppose a marketplace that provides a Web-based interface, where the end-user can interact with a catalog of VNFs. The end-user decides to acquire and instantiate a VNF, but he/she does not operate his/her own NFV-enabled infrastructure. Such a marketplace concept is similar to the one found in literature [6]. Hence, one customer (*i.e.*, end-user) wants to contract for more than one month a VNF that implements a state-of-the-art firewall, which is available to be purchased in the marketplace. The desired VNF has a descriptor (*i.e.*, VNFD) defining as minimum resources for the firewall the following: 4 GB RAM, 2 GB disk space, and 2 CPU cores. Additionally, the end-user requests two additional GB of memory to be used only in unusually high demand periods and also defines a maximum latency of 20 ms. With such information, the contracted VNF is instantiated in an NFV-enabled infrastructure. Thus, the marketplace should know which is the best infrastructure available that supports these demands and provides a cost-efficient solution. For this, the BRAIN solution is executed.

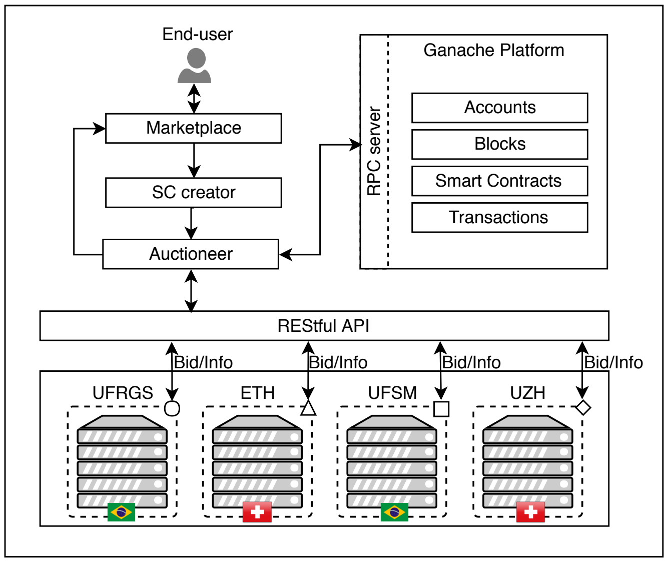

Fig. 2: Case Study Scenario

After VNF acquisition by the end-user via the marketplace (*cf.* Figure 2), an SC is created by the *SC Creator* component based on the VNFD (provided by the marketplace) and end-user particular demands that were defined during the acquisition. After that, the *Auctioneer* identifies, based on a database request, that there are four infrastructure candidates available to supply the VNF demand. The *Auctioneer* deploys the SC inside the blockchain and sends the SC address to each InP via the RESTful API. Thus, they can obtain information about the end-user request and also issue bids based on their own algorithms. Table II presents the information defined in the PD file, which represents priorities previously indicated by the end-user. Note that only the contract period input is mandatory, other fields are optional for users. Besides this information, the VNFD containing the description of the network service is used as an input to the bid calculation as well.

TABLE II: Priorities defined by end-users in PD file

Parameter	Value	Required
Contract period	weekly	Yes
Geolocalization preference	South America and Europe	No
Additional CPU	+2 core	No
Additional memory	+2 GB	No
Additional disk space	+20 GB	No
Maximum latency	10 ms	No
Traffic supported	10.000 packets/s	No

The bid is calculated by using a customized algorithm that is part of the *Bid Manager* and runs inside each infrastructure. This algorithm obtains information from the SC and calculates the final bid based on resources required, VNF type, and end-users priorities (*e.g.*, maximum latency and geolocalization). Each one of these InPs has its algorithm configured with actual values. Thus, based on the input containing information from the auction, an output with the final bid is provided and recorded on the blockchain. Finally, the *Auctioneer* sends a last transaction to the SC in order to terminate the auction. Hence, the final bid and auction winner is revealed to the marketplace, and the process continues with the deployment of the VNF.

TABLE III: Infrastructure Providers Overview

Provider	Country	Free Resources	Final Bid
University of Zurich (UZH)	Switzerland	40 CPU cores 128 GB memory 15 TB disk	$49.22 USD/month
Federal University of Rio Grande do Sul (UFRGS)	Brazil	20 CPU cores 3 2GB memory 1 TB disk	$43.63 USD/month
Federal University of Santa Maria (UFSM)	Brazil	4 CPU cores 16 GB memory 1 TB disk	$55.02 USD/month
Swiss Federal Institute of Technology (ETH)	Switzerland	5 CPU cores 32 GB memory 2 TB disk	$59.18 USD/month

Table III shows each candidate InP and their available resources. The value calculated for each bid algorithm is shown, while UFRGS and UZH provide lower prices to supply VNF demands. These lower prices are due to the number of resources available and their cheaper operational costs (*e.g.*, infrastructure optimized to deal with firewalls with high demands) to host the VNF. Note that it is not guaranteed that the infrastructure with more resources will provide the best price. The final bid depends on several factors that can be defined inside the bidding algorithm, presenting the business model. In such a case, UFRGS provides the best bid. Therefore, the *Auctioneer* will let the marketplace know the bid value and the UFRGS address of the blockchain. Thus, the communication between the marketplace and UFRGS to effectively deploy the VNF starts. UFRGS has to deliver the promised performance and resources; if that is not delivered, mechanisms to punish such an InP can be defined based on the SC (*e.g.*, compensations or reputation decreased).

B. *Discussion*

Despite the benefits introduced by BRAIN, drawbacks have to be considered when deploying such a solution. The drawbacks are mainly regarding additional fees and time. The fees are not high but should be considered in order to deploy the auction into a large scale. An analysis of the current state of Ethereum blockchain was conducted to investigate costs. For this, Gas used was calculated by each auction call, and it was analyzed regarding the Gas price and average time. Gas defines the internal pricing to run a transaction or a contract in Ethereum blockchain, which is efficient to measure the computational usage in terms of monetary costs (*e.g.*, Gas per United States Dollar or Swiss Franc).

The deployment of the SC requires two minutes to be mined by the blockchain and the cost is US$0.59, which must be paid by the *Auctioneer* during the opening and call phase. The time can be decreased to 40 seconds but the fee to deploy it in the blockchain is US$1.51. On the InPs side, there is a cost of US$0.37 to submit a bid. This value should be paid by each bidder to participate in the auction and guarantees that the bid will be processed in at least 40 seconds. There is no significant additional cost and time to handle the requests for resources and priorities information.

Based on the time that a transaction requires to be processed (*i.e.*, mined) in the blockchain, the auction time is more than 2 minutes. Thus, such additional time should be considered by the marketplace to avoid a negative impact in the end-user's experience nor during the VNF deployment phases. BRAIN can be used to supply a large variety of services based on VNFaaS. However, since the blockchain cannot achieve real-time transactions, there are specific scenarios (*e.g.*, hosting VNFs to mitigate imminent cyber attacks) where other approaches should be considered to reduce the deployment time. Solutions could try to reduce costs (*e.g.*, Gas usage optimization) and also the time required for the auction. Moreover, according to the evolution of blockchains and the next-generation blockchains, new opportunities can be expected to deal with such issues. The authors are researching the viability of Distributed Ledger Technology (DTL) solutions to address such limitations.

Regarding scaling properties of the proposed solution, they depend on the Ethereum blockchain. The number of providers that can participate is as large as an SC can support according to the current state of the blockchain, which is, basically, limited by the max Gas per block on Ethereum. In addition, the number of transactions secured on the Ethereum blockchain and, consequently, inherited by the proposed solution, is close to 20 transactions per second. It is important to mention that the availability of resources announced by InPs can be forged, because the solution cannot provide trust due to the monitoring system, thus, it is not possible to validate monitored information yet.

Payment issues are out of the work scope. However, BRAIN was designed in a generic manner in order to be integrated with innovative payment methods that address the auction phases and also a VNF billing. Automatic payments can be implemented by using the Ethereum cryptocurrency Ether. To this end, the payment can be processed inside the blockchain and transactions containing the respective value can be done between distinct wallets (*e.g.*, end-users pay directly to InPs). Monitoring tools (*e.g.*, blockchain-based SLAs monitors and malicious behavior detectors) can be helpful to measure if stakeholders (*i.e.*, end-users and InPs) involved in the auction are complying to contract clauses, thus, executing the full payment or applying monetary penalties to the parties that not follow the deal, *e.g.*, when an InP promises to host a VNF with a specific requirement that cannot be achieved by using the provided infrastructure.

VI. Summary and Future work

This paper presented BRAIN a solution that introduces a reverse auction mechanism and fosters the competition among InPs to supply infrastructures to host VNFs acquired by end-users. BRAIN builds on blockchain-based SCs to provide trust to all stakeholders involved and to handle bid submissions. Moreover, it explores aspects related to ETSI standards (*i.e.*, VNF descriptors) and proposes a way to create SCs to request an infrastructure based on end-users preferences and demands. Thus, InPs can be aware of requirements and can issue bids according to their capacities (*e.g.*, resources available and costs). This solution provides immutable and trustful records about each interaction between InPs, marketplaces, and end-users. This allows for a trust, fair, and auditable auction. In summary, the contributions of this work include: *(i)* the technical basis for a lucrative competition-based business model for NFV-enabled infrastructure providers, *(ii)* a simplified way to end-users to find the best, in terms of costs and performance, infrastructure to host their VNFs, and *(iii)* a trustworthy agreement between stakeholders (*i.e.*, end-users and InPs) regarding resources contracted and configurations required.

The feasibility of the solution was evaluated in a prototype implementation and the dedicated case study discussion. The scenario discussed four InPs competing to supply a VNF acquired by the end-user via a marketplace. It starts with the deployment of an SC based on the VNFD and custom information provided by the end-user. InPs provide their bids without knowing other competitive bids. The winner is the InP that provided the lower bid to host the VNF while meeting the defined demands. The evaluation was also conducted concerning the practical application of the solution. Based on this evaluation it was found additional costs to the *Auctioneer* for the SC deployment and also for each InPs issue bids. Also, the time increased by the auction are about 2 minutes according to the experiments performed. These additional costs and time can be absorbed both by marketplaces and InPs without significant losses on their profits and performance. However, specific scenarios may require real-time performance, where approaches to optimize the auction must be investigated.

Future work includes: *(i)* the proposal of a new algorithm based on the immutable records provided to deal with the placement and migration of VNFs, *(ii)* investigation of machine learning techniques to predict the behavior of InPs and their bids during the auction, *(iii)* design and development of a reputation and compensation system for stakeholders involved in the reverse auction, and *(iv)* conduction of studies related to the economic impact of BRAIN. Finally, other distributed ledger technologies will be investigated to address limitations discussed in this work. In addition, further real-world evaluations will be conducted to demonstrate benefits and drawbacks of BRAIN in terms of performance, scalability, and robustness.

References

[1] ETSI NFV ISG, "Network Functions Virtualisation," June 2012, White Paper. Accessed on March 20, 2019. [On-line] https://portal.etsi.org/nfv/nfv_white_paper.pdf

[2] B. Yi, X. Wang, K. Li, S. k. Das, and M. Huang, "A Comprehensive Survey of Network Function Virtualization," *Computer Networks*, Vol. 133, March 2018, pp. 212–262.

[3] B. Han, V. Gopalakrishnan, L. Ji, and S. Lee, "Network Function Virtualization: Challenges and Opportunities for Innovations," *IEEE Communications Magazine*, Vol. 53, No. 2, February 2015, pp. 90–97.

[4] G. Xilouris, M. A. Kourtis, M. J. McGrath, V. Riccobene, G. Petralia, E. Markakis, E. Palis, A. Georgios, G. Gardikis, J. F. Riera, A. Ramos, and J. Bonnet, "T-NOVA: Network Functions as-a-Service over Virtualised Infrastructures," in *Network Function Virtualization and Software Defined Network (NFV-SDN), 2015 IEEE Conference on*, San Francisco, CA, USA, November 2015, pp. 13–14.

[5] S. D'Oro, L. Galluccio, S. Palazzo, and G. Schembra, "A Marketplace as a Scalable Solution to the Orchestration Problem in SDN/NFV Networks," in *IEEE Conference on Network Softwarization (NetSoft)*, Bologna, Italy, July 2017, pp. 1–5.

[6] L. Bondan, M. F. Franco, L. Marcuzzo, G. Venancio, R. L. Santos, R. J. Pfitscher, E. J. Scheid, B. Stiller, F. De Turck, E. P. Duarte, A. E. Schaeffer-Filho, C. R. P. d. Santos, and L. Z. Granville, "FENDE: Marketplace-Based Distribution, Execution, and Life Cycle Management of VNFs," *IEEE Communications Magazine*, Vol. 57, No. 1, January 2019, pp. 13–19.

[7] U. Habiba and E. Hossain, "Auction Mechanisms for Virtualization in 5G Cellular Networks: Basics, Trends, and Open Challenges," *IEEE Communications Surveys Tutorials*, Vol. 20, No. 3, March 2018, pp. 2264–2293.

[8] S. M. Wagner and A. P. Schwab, "Setting the stage for successful electronic reverse auctions," *Journal of Purchasing and Supply Management*, Vol. 10, No. 1, January 2004, pp. 11–26.

[9] Y. Zhang, C. Lee, D. Niyato, and P. Wang, "Auction Approaches for Resource Allocation in Wireless Systems: A Survey," *IEEE Communications Surveys Tutorials*, Vol. 15, No. 3, November 2013, pp. 1020–1041.

[10] T. Bocek and B. Stiller, "Smart Contracts - Blockchains in the Wings," Digital Marketplaces Unleashed, Heidelberg, Germany, January 2017.

[11] M. Alharby and A. van Moorsel, "Blockchain-based Smart Contracts: A Systematic Mapping Study," in *International Conference on Computer Science and Information Technology (CoSIT)*, Geneva, Switzerland, March 2017.

[12] Satoshi Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," 2014, White Paper. Accessed on March 20, 2019. [On-line] https://bitcoin.org/bitcoin.pdf

[13] Vitalik Buterin, "Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform," 2014, White Paper. Accessed on March 20, 2019. [On-line] https://cryptorating.eu/whitepapers/Ethereum/Ethereum_white_paper.pdf

[14] R. M. Parizi and A. Dehghantanha, "Smart Contract Programming Languages on Blockchains: An Empirical Evaluation of Usability and Security," in *International Conference on Blockchain*. Springer, June 2018, pp. 75–91.

[15] D. C. Wyld, "Current Research on Reverse Auctions: Understanding the Nature of Reverse Auctions and the Price and Process Savings Associated with Competitive Bidding," *International Journal of Managing Value and Supply Chains*, Vol. 2, No. 3, September 2011, pp. 11–23.

[16] Glenn Wheaton, "Government Reverse Auctions," June 2010, Accessed on March 20, 2019. [On-line] http://www.epiqtech.com/government-reverse-auctions.htm

[17] J. G. Herrera and J. F. Botero, "Resource Allocation in NFV: A Comprehensive Survey," *IEEE Transactions on Network and Service Management*, Vol. 13, No. 3, August 2016, pp. 518–532.

[18] A. G. Tasiopoulos, S. G. Kulkarni, M. Arumaithurai, I. Psaras, K. K. Ramakrishnan, X. Fu, and G. Pavlou, "DRENCH: A Semi-Distributed Resource Management Framework for NFV based Service Function Chaining," in *IFIP Networking Conference (IFIP Networking) and Workshops*, Stockholm, Sweden, 2017, pp. 1–9.

[19] A. H. M. Jakaria and M. A. Rahman, "A Formal Framework of Resource Management for VNFaaS in Cloud," in *IEEE 10th International Conference on Cloud Computing (CLOUD)*, Honolulu, HI, USA, June 2017, pp. 254–261.

[20] N. Ma, J. Zhang, and T. Huang, "A Model based on Genetic Algorithm for Service Chain Resource Allocation in NFV," in *3rd IEEE International Conference on Computer and Communications (ICCC)*, Paris, France, December 2017, pp. 607–611.

[21] M. S. Yoon and A. E. Kamal, "NFV Resource Allocation Using Mixed Queuing Network Model," in *IEEE Global Communications Conference (GLOBECOM)*, Washington DC, USA, December 2016, pp. 1–6.

[22] L. Gao, P. Li, Z. Pan, N. Liu, and X. You, "Virtualization Framework and VCG Based Resource Block Allocation Scheme for LTE Virtualization," in *IEEE 83rd Vehicular Technology Conference (VTC Spring)*, Nanjing, China, May 2016, pp. 1–6.

[23] E. I. Nehru, J. I. S. Shyni, and R. Balakrishnan, "Auction based Dynamic Resource Allocation in Cloud," in *International Conference on Circuit, Power and Computing Technologies (ICCPCT)*, Nagercoil Andaman and Nicobar Islands, India, March 2016, pp. 1–4.

[24] S. Gu, Z. Li, C. Wu, and C. Huang, "An Efficient Auction Mechanism for Service Chains in the NFV Market," in *The 35th Annual IEEE International Conference on Computer Communications (INFOCOM)*, San Francisco, CA, USA, April 2016, pp. 1–9.

[25] X. Zhang, Z. Huang, C. Wu, Z. Li, and F. C. M. Lau, "Online Stochastic Buy-Sell Mechanism for VNF Chains in the NFV Market," *IEEE Journal on Selected Areas in Communications*, Vol. 35, No. 2, February 2017, pp. 392–406.

[26] T. Aste, P. Tasca, and T. D. Matteo, "Blockchain Technologies: The Foreseeable Impact on Society and Industry," *Computer*, Vol. 50, No. 9, September 2017, pp. 18–28.

[27] I. D. Alvarenga, G. A. F. Rebello, and O. C. M. B. Duarte, "Securing Configuration Management and Migration of Virtual Network Functions using Blockchain," in *IEEE/IFIP Network Operations and Management Symposium (NOMS)*, Taipei, Taiwan, April 2018, pp. 1–9.

[28] Sinfonia Team, "Sinfonia - A Secure Virtual Network Function Orchestrator for Non-Repudiation, Integrity, and Auditability," 2017, Grupo de Teleinformática e Automação GTA, Federal University of Rio de Janeiro, Brazil. Accessed on March 20, 2019. [On-line] https://www.gta.ufrj.br/sinfonia/

[29] E. J. Scheid, B. Rodrigues, L. Z. Granville, and B. Stiller, "Enabling Dynamic SLA Compensation Using Blockchain-based Smart Contracts," in *IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, Washington DC, USA, April 2019, pp. 1–9.

[30] Z. Neeman, "The effectiveness of English auctions," *Games and Economic Behavior*, Vol. 43, No. 2, May 2003, pp. 214–238.

[31] ETSI GS NFV-MAN, "Network Functions Virtualisation (NFV); Management and Orchestration," December 2014, white Paper. Accessed on March 20, 2019. [On-line] http://www.etsi.org/deliver/etsi_gs/NFV-MAN/001_099/001/01.01.01_60/gs_NFV-MAN001v010101p.pdf

[32] Muriel Franco and Burkhard Stiller, "NFV4all: Simplifying the Adoption and Provision of Virtual Network Functions," Communication Systems Group CSG, Department of Informatics IfI, University of Zurich UZH, Switzerland. Accessed on March 26, 2019. [On-line] https://www.csg.uzh.ch/csg/en/research/NFV4all.html