BSec-NFVO: A Blockchain-based Security for Network Function Virtualization Orchestration

Gabriel Antonio F. Rebello, Igor D. Alvarenga, Igor J. Sanz, and Otto Carlos M. B. Duarte

Universidade Federal do Rio de Janeiro - GTA/COPPE/UFRJ

***Abstract*— Network Function Virtualization (NFV) and Service Function Chaining (SFC) offer flexible end-to-end services that deploy virtual network functions in clouds of competing providers. Orchestration of virtual network functions occurs in a distributed and trustless environment that must tolerate byzantine failures and collusion attacks. This paper proposes BSec-NFVO, a blockchain-based system that secures orchestration operations in virtualized networks, ensuring auditability, non-repudiation and integrity. We propose an NFV-tailored blockchain and a transaction model. BSec-NFVO provides a modular architecture to secure orchestration in a simple and agile way. We develop a prototype of BSec-NFVO for the Open Platform for Network Function Virtualization (OPNFV) with an adaptation of the normal-case of a collusion-resistant consensus protocol. The results show BSec-NFVO incurs low overhead to the cloud orchestrator and presents stable performance as the number of consensus participants increases.**

I. Introduction

The Network Function Virtualization (NFV) technology replaces hardware-based network functions by virtual network functions (VNF) that run in commodity machines. NFV reduces costs and offers flexible network management by reimplementing middleboxes in software that can be developed by a plethora of providers. Meanwhile, Software-Defined Networking (SDN) can be used with NFV to steer traffic through a sequence of virtual network functions[1]. The result, Service Function Chaining (SFC) [1], creates an end-to-end network function chain to provide on-demand services tailored for each application. NFV and SFC are fundamental to provide flexible traffic control and management for services in next generation networks.

Network function virtualization and service function chaining, however, incur new security challenges [2], [3]. As end-to-end service function chains may deploy virtual network functions in domains of competing cloud providers, we must ensure the service chain is built in a trustful manner while in a trustless environment [4]. Furthermore, in a virtualized scenario, tenants share the same cloud infrastructure. A multi-tenant and multi-domain environment increases the possibility of attacks inside the cloud, while also hindering accountability of service providers. The impacts of possible attacks become greater, since attacks to the host of VNFs can compromise thousands of users simultaneously [3].

The FCAPS (Fault, Configuration, Administration, Performance and Security) model is a standard for managing telecommunications networks in single-provider centralized environments. The NFV scenario, however, is a distributed environment in which trust between providers is uncertain. Providers offer distributed services through multiple clouds, rising new challenges such as: i) ensure correctness of service function chains; ii) identify the cloud provider, from the participants of a service function chain, that is accountable for a failure; iii) establish provenance, impact and total time a failure remained undetected.

In previous works, the authors of this paper evaluated the performance of service function chains in the Open Platform for Network Function Virtualization (OPNFV) [5] and analyzed the use of blockchain in OPNFV for securing configuration and migration of virtual network functions [6]. This paper proposes, develops, and evaluates BSec-NFVO, a blockchain-based system for agile and secure management of service function chain orchestration operations[2]. BSec-NFVO extends the premises of FCAPS to a multi-domain environment. Our proposal guarantees transparency of orchestration operations and accountability of its authors by immutably logging all instructions that manipulate a service chain. The use of blockchain and public-key cryptography ensures authenticity, integrity and non-repudiation of instructions, which allows confirming provenance. Therefore, BSec-NFVO guarantees auditability of the performed operations, which can be used for investigation and legal measures in the case of a security incident.

The main contribution of this paper is proposing blockchain and transaction models that provide traceability in a multi-tenant and multi-domain NFV environment. We implement a prototype of BSec-NFVO in the Open Platform for Network Function Virtualization (OPNFV) with a simplification of the Practical Byzantine Fault Tolerance (PBFT) consensus protocol. In our main findings, we discover that: i) the delay introduced by blockchain and transaction validation is not significant; ii) BSec-NFVO is more efficient in longer service chains; and iii) transaction throughput remains stable while increasing the number of consensus participants. The prototype offers the necessary infrastructure for performing a secure orchestration and creating service function chains.

II. Related Work

Previous works address the problem of security vulnerabilities in multi-tenant and multi-cloud NFV environments [7],

[1]In the scope of this paper, the terms virtual network function (VNF) and service function (SF) are used interchangeably.

[2]Available at https://github.com/gfrebello/bsec-nfvo

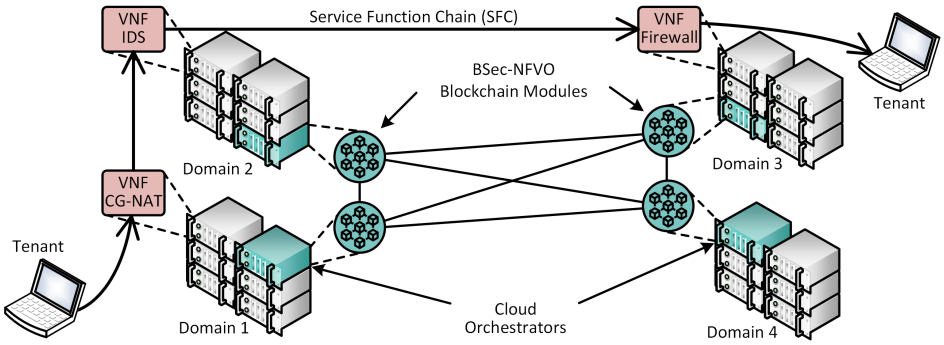


Fig. 1. An example of a BSec-NFVO use case scenario in which a service function chain comprises VNFs located in several domains. Blockchain modules immutably record operations issued by orchestrators of competing cloud providers.

[8]. The works indicate trust in cloud providers is uncertain and that compromising a single VNF at the network core endangers entire end-to-end services. Massonet *et al.* propose an architecture for global configuration of security VNFs in federated networks with automated deployment [9]. Khettab *et al.* propose to secure multi-domain NFV environments by dynamically instantiating security network functions such as firewalls and intrusion detection systems [10]. The aforementioned works assume the cloud is a trustful entity for provisioning and storing data. Our work assumes a trustless environment and uses the immutability and auditability properties of blockchain to prevent possible malicious behaviour by cloud providers in NFV.

Other works propose the use of blockchain to ensure transaction traceability in distributed trustless environments. Zaowat and Hasan proposed SECAP, a blockchain-based framework to securely store a provenance tree of cloud applications [11]. The framework, however, only protects the logs of application state changes. Bozic *et al.* propose an architecture for managing execution states of virtual machines using a blockchain-based system [12]. The system logs instructions sent to the virtualization hypervisor on the blockchain as transactions. The authors, however, only protect virtual machine creation instructions and provide no implementation of the proposed architecture.

We propose to secure and provide auditability of VNF orchestration operations using a private blockchain. A consensus protocol that prevents collusion attacks without compromising latency and throughput is mandatory in our proposal. Hence, solutions based on Proof of Work or that only tolerate crash failures do not attend the necessary demands. We adopt a byzantine fault tolerant (BFT) protocol called Practical Byzantine Fault Tolerance (PBFT) to promote low-latency consensus while tolerating malicious behavior of up to one third of participants [13], [14]. Unfortunately, to the best of our knowledge there is no open source implementation of the PBFT consensus for blockchains. The Hyperledger Fabric project aims to implement PBFT and other BFT-based blockchains in the future [15].

III. The BSec-NFVO system

The main goal of BSec-NFVO is to protect all creation, removal, and modification instructions of virtual machines, virtual network functions, and service chains. In our proposal, instructions are tenant-issued requests for orchestration operations. Our scenario is composed by orchestrators, tenants, and cloud providers. Orchestrators are entities that manipulate a service function chain according to the set of instructions received from a tenant. Tenants are domain clients that use a service provided by the service function chain. Cloud providers are providers that manage a set of data centers, also called a domain. BSec-NFVO allows all orchestrators, tenants and cloud providers to easily verify a local and immutable history of past instructions in a blockchain to identify malicious participants in the network.

The NFV scenario implies big data center communication and comprises four key aspects: i) limited number of identified providers, as each provider takes part in service level agreements with tenants and other providers; ii) low number of crash failures, due to the high availability of big data centers; iii) high throughput and low latency in end-to-end communication, as VNFs are implemented in the network core; and iv) tolerance to malicious behavior between competing providers and tenants. Furthermore, the geographical location of network functions in an end-to-end service may affect efficiency according to the VNF role in a service function chain. Figure 1 illustrates an example of a service function chain in an NFV scenario. In the example, a service function chain comprises VNFs hosted by competing cloud providers, thus creating a trustless environment. Each provider has a domain, which contains a single VNF orchestrator.

In BSec-NFVO, instructions that manipulate a service chain become signed transactions. Each issued instruction corresponds to a request transaction and a response transaction. Tenants sign request transactions and orchestrators sign response transactions, and both types are validated and agreed upon consensus. Thus, a signed transaction in the blockchain is an irrefutable proof that a specific tenant requested an operation and that an identifiable orchestrator executed it. Because our main goal is to provide auditability, the transaction content is visible to anyone in the network, including tenants, orchestrators and consensus participants. Other implementations could provide transaction confidentiality by using public-key cryptography. Each instance of BSec-NFVO contains a replica of the global blockchain, and

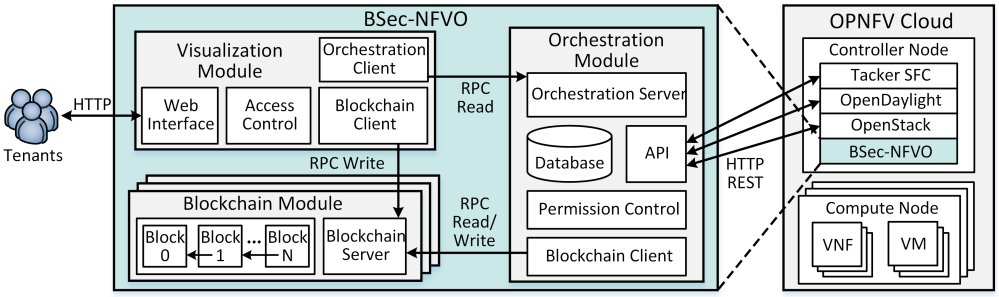


Fig. 2. The proposed system architecture. Tenants request orchestration operations through a web interface. The instructions are signed and sent to the blockchain module as request transactions. The orchestration module reads transactions from the blockchain and communicates with the OPNFV cloud orchestrator, which then returns the operation result to be signed and validated as a response transaction.

at least one instance must run in each domain.

The architecture of BSec-NFVO, depicted in Figure 2, comprises three modules: i) the visualization module, which serves as the interface between tenants and the NFV and SFC services; ii) the orchestration module, which executes instructions sent by tenants through the visualization module; and iii) the blockchain module, which validates transactions before execution by the orchestration module.

The **Visualization Module** aims to ease the specification of an end-to-end service. This module comprises four main components: i) a user-friendly web interface that allows a tenant to manage hired services and to issue instructions; ii) an access control manager that applies Service Level Agreements (SLA) to each tenant and limits its access to the hired services; iii) an orchestration client that communicates with the orchestration module to read the current state of services in the platform; and iv) a blockchain client that signs instructions and sends the resulting request transactions to the blockchain module. Client components communicate through remote procedure calls (RPC) protected by the Transport Layer Security (TLS) protocol.

The **Orchestration Module** executes instructions and responds to read requests received from the visualization module. The module comprises five main component: i) an orchestration server that receives RPC calls from the visualization module; ii) a database that records account information of each tenant; iii) a permission control system that verifies whether a tenant is authorized to issue the received instruction; iv) a blockchain client that communicates with the blockchain module to verify the existence of pending instructions and to send signed response transactions; and v) an Application Programming Interface (API) that connects to the OPNFV platform to execute authorized instructions.

The **Blockchain Module** performs instruction validation and comprises two components: a blockchain server and the blockchain itself. The blockchain server receives RPC calls for write and read requests to the blockchain. After each time interval, on the order of seconds, the blockchain module records every received transaction in a block, links it to the hash function output of the previous block and signs it with a key pair provided by its corresponding cloud provider. Each blockchain module keeps a replica of the blockchain that contains transactions approved in the consensus protocol.

IV. The Blockchain and Transaction Models

A blockchain is a replicated data structure that ensures trust and agreement in a distributed system while dismissing the need for a common central authority. The blockchain works as a permanent and immutable ledger of ordered transaction blocks that are identified by a digest of a cryptographic hash function. Because the blockchain is replicated in multiple network participants and every transaction is signed, non-repudiation of transactions is guaranteed.

We propose and develop blockchain and transaction models to address the needs of the NFV scenario. Figure 3 depicts the main difference between the BSec-NFVO blockchain and a traditional structure. A BSec-NFVO block comprises two sections: content, in which the public key of the blockchain module that proposed the block is stored along with the previous block signature, and content signature. The proposed content signature field provides auditability of the consensus process by clearly identifying the blockchain module that proposed the recorded block. We also provide the public key of the signer in the block content and, thus, any entity can verify the signature. Blockchain modules never emit transactions, but each blockchain module has an asymmetric key pair for signing blocks and consensus messages.

We define two types of transactions in the proposed architecture: request transactions, issued by tenants for recording instructions, and response transactions, issued by orchestrators for recording the execution output of the received instruction. Hence, a request transaction is signed by the tenant that proposes it and a response transaction is signed by the orchestrator that executes the corresponding instruction. BSec-NFVO transactions contain a header and a content section. The header of each transaction contains the signature of the content as generated by its emitter, which ensures authenticity and transaction integrity. The content fields are: type, which defines the transaction category; sender, which contains the public key of the sender; timestamp, which defines the moment in which the transaction was emitted. A request transaction also contains an instruction field, which defines the write instruction to be executed by an orchestrator. A response transaction has three additional fields: source transaction, which identifies the corresponding request transaction to which it responds by the header; result, which defines the response of the orchestrator to the requested instruction; and error, which identifies potential errors in

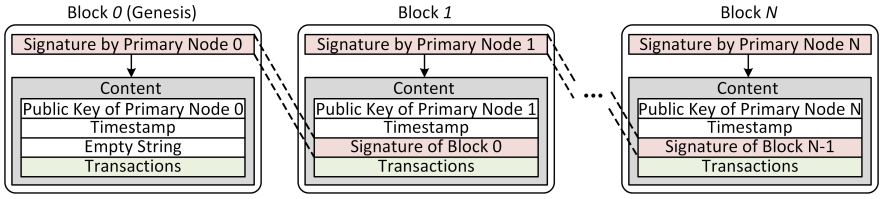


Fig. 3. The BSec-NFVO blockchain. The consensus primary node signs the current block to ensure authenticity.

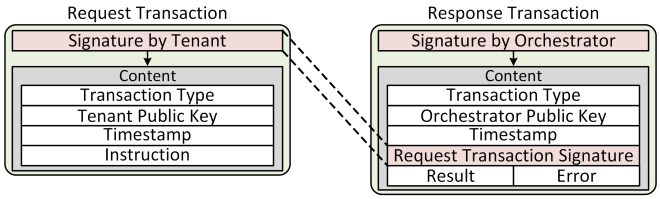


Fig. 4. The two types of transactions in BSec-NFVO. Tenants issue request transactions and orchestrators issue response transactions. Every response transaction is bound to a corresponding request transaction by the header.

the orchestration process. Every response transaction must identify its corresponding request transaction, proving communication between tenant and orchestrator occurred.

Blockchain modules validate transactions by verifying four aspects: i) the transaction signature, to match the public key of the sender; ii) the presence of every field as specified in each transaction type definition; iii) the timestamp, according to a predetermined threshold, to avoid the execution of future and past transactions; and iv) the presence of the candidate transaction in the blockchain, to avoid duplicates. For request transactions, the instruction field semantics is also verified to avoid the execution of arbitrary instructions. Invalid transactions are immediately discarded. Transaction validation is performed locally on each blockchain module.

A. Consensus in BSec-NFVO

In order to be efficient and robust, we select a Byzantine Fault Tolerant (BFT) consensus protocol instead of Proof of Work or a consensus that only tolerates crash failures. The chosen BFT consensus protocol presents low latency and tolerate malicious behavior of up to one third of participants [13], [14]. We develop a BSec-NFVO prototype that implements a simplified version of the Practical Byzantine Fault Tolerance (PBFT) consensus protocol. Our implementation does not account for exception conditions nor leader election, but behaves similar to the PBFT protocol in normal conditions. Our implementation is, thus, fit for a proof of concept and for the performed prototype evaluation.

We propose a three-phase operation sequence simplification of the conventional five-phase sequence of the normal-case PBFT. The three phases are depicted in Figure 5: i) pre-prepare, in which the primary node, also referred to as the leader, creates a signed block containing a new set of candidate transactions and broadcasts it to all participants; ii) prepare, in which each participant validates the transactions locally and broadcasts its decision through signed messages; and iii) commit, in which each participant broadcasts its final result through signed messages after receiving more than two thirds of signed votes. In the commit phase, every prepare message received by a participant is included in the final message as a proof of consensus.

It is important to note that we discard the traditional request and response phases of PBFT because they occur asynchronously when employed in a blockchain-based application. Due to the immutability characteristic of the blockchain, all states are verifiable by anyone at any time. Hence, the request and response phases are not necessary. Although the private blockchain network comprises tenants and cloud providers, only cloud providers participate in consensus, reducing overall consensus time and increasing throughput. If a blockchain module is not the primary node, it relays every received transaction to the primary node. The primary node proposes a new block that contains a set of all transactions received before the start of the current round.

V. Performance Evaluation of BSec-NFVO

BSec-NFVO uses the Open Platform for Network Function Virtualization (OPNFV) for creating service function chains. OPNFV implements service function chaining as described in RFC 7665 [1] and uses the Network Service Header protocol. The evaluation of the BSec-NFVO prototype aims to measure three key aspects: i) the overhead introduced by blockchain and transaction validation in the OPNFV environment; ii) the performance of the simplified consensus protocol; and iii) the blockchain size increase rate, which affects the cost of storing replicas.

To evaluate the system overhead, we install and configure BSec-NFVO in an Intel 16-core Xeon E5-2650 32 GB RAM that acts as the controller of an OPNFV environment deployed in the GTA/UFRJ laboratory. The compute nodes consist of three servers: one Intel 8-Core Xeon X5570 96 GB RAM (node 1), one Intel 8-Core i7-6700 64 GB RAM (node 2) and one Intel 8-Core i7-2600 32 GB RAM (node 3). A top of rack switch connects all servers in a LAN with 1 Gb/s network interfaces. A personal computer issues instructions to the visualization module through HTTP requests. Every signature in the prototype uses 2048-bit RSA keys and the PKCS#1-PSS signature standard as implemented by the PyCryptodome[3] cryptography library.

The first experiment evaluates the delay introduced by our BSec-NFVO proposal. First, we evaluate the time overhead caused by signing instructions and validating transactions without exchanging consensus messages in the network. Thus, we use a single-consensus blockchain module. We measure the tenant-platform communication delay between an HTTP request for creating a service function chain, issued

[3]The library is available at https://github.com/Legrandin/pycryptodome

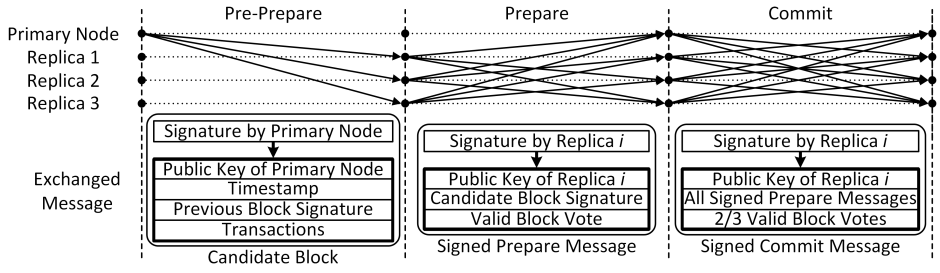


Fig. 5. The proposed three-phase operation sequence for the PBFT normal-case: pre-prepare, prepare and commit. After the commit phase, the new block is appended to the blockchain in each participant blockchain module.

by a tenant through the BSec-NFVO web interface, and its response, i.e. the confirmation that the instruction will be executed by the cloud orchestrator. Figure 6(c) shows that, with a 95% confidence interval, the blockchain-induced additional delay is about 0.06 s or 3%, indicating the time overhead introduced by the blockchain is not significant.

In a second experiment we aim to measure the memory overhead generated by public-key cryptography operations in the proposed blockchain and transaction models. Therefore, we issue empty transactions and empty blocks. The typical instruction in OPNFV includes the instruction itself and its optional parameters that may point to a configuration model, e.g. "`opnfv create-vnf -config=vnf.conf`". Hence, except for specific cases such as sending the configuration file content directly in the instruction, we expect an instruction to contain no more than dozens of bytes. The total size of empty transactions is 637 B for request transactions and 655 B for response transactions while the total size of an empty block is 859 B. These values indicate a high memory overhead for instructions and demonstrate the high cost of ensuring authenticity and integrity of transactions. The values, however, are plausible in cloud environments, in which network, memory and disk resources are notably abundant. Stressing the system to obtain the maximum transactions throughput, we obtain an average number of 3808 transactions per block. Therefore, the size of a block typically ranges from kB to MB and the block signature overhead is not significant.

In a third experiment, we aim to evaluate the performance of the simplified consensus protocol in creating a service function chain. We deploy all VNFs in the same compute node to minimize overheads introduced by the OPNFV platform [5]. Figure 6(b) shows the creation time of a service function chain versus an increasing number of component VNFs. The results show a direct relation between creation time and the number of chained VNFs. This is due to OPNFV, which sequentially creates each VNF from an image in a virtual machine before chaining. The average VNF creation time is 30.1 seconds and the introduction of a PBFT-based consensus incurs a delay of up to 20 seconds, for the 9-node case. The results demonstrate the overhead introduced by consensus is inversely proportional to the chain size and that the dominant term for long service chains is the instantiating time. Hence, BSec-NFVO is more effective to protect end-to-end services composed by many VNFs.

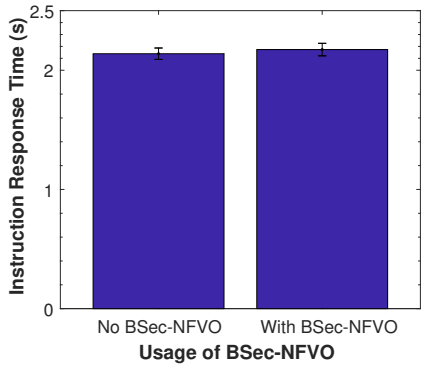
The fourth experiment evaluates the throughput of the proposed architecture. We deploy several blockchain and orchestration modules as independent processes in the same host with a dedicated CPU core and 4 GB memory each. The host machine is an Intel 32-core Xeon E5-2650 192 GB RAM. Figure 6(c) shows the impact on system throughput as instruction size increases. The evaluation considers only request transactions because both types of transactions have similar sizes. We select the sizes considering the typical length of an instruction in OPNFV, using one byte per character for encoding. Hence, we expect larger sizes to represent complex instructions, which require more parameters. The results show the prototype throughput remains stable when increasing the number of consensus participants for all instruction sizes, except when compared to the single-consensus case, in which no consensus messages are exchanged. Throughput also remains stable when increasing instruction size up to 64 B. The minimum processing capability between the primary node and the remaining participants determines an 803.3 transactions per second upper bound.

The last experiment, shown in Figure 6(d), evaluates the growth rate of a blockchain for increasing instruction sizes. We use a constant rate of 100 transactions per second to simulate a production environment in which transactions represent requests issued by tenants. Experiments using 10 and 500 transactions per second generated similar results. The results show that, for instructions of up to 64 B, the blockchain growth rate reaches 100 kB/s per consensus participant and grows significantly for instruction sizes above 256 B. The near-constant growth rate for transactions of less than 64 B is due to the overhead generated by signatures, which produces minimum transaction and block sizes.

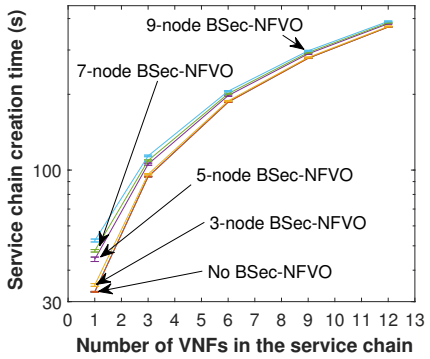
VI. Conclusion

The network function virtualization technology provides end-to-end services by chaining virtual network functions between competing cloud infrastructures in a trustless environment. In this multi-tenant and multi-domain scenario, it is important to define and locate failures to identify malicious agents that compromise the good behavior and quality of service of thousands of users simultaneously. This paper proposes BSec-NFVO, a blockchain-based system that provides the necessary security for orchestrating virtual network functions in a multi-domain and multi-tenant environment. BSec-NFVO offers secure services by ensuring auditability of all operations that manipulate a service function chain.

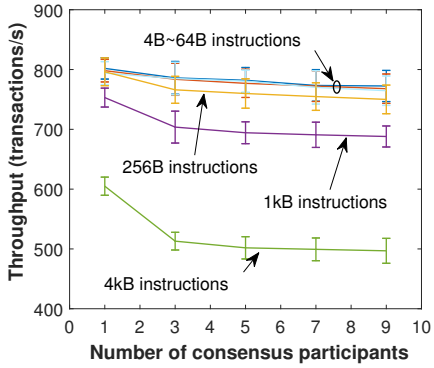
We propose and implement NFV-adapted blockchain and



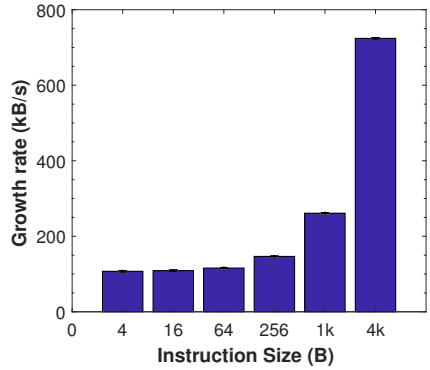
(a) Response time overhead introduced by using a single-node consensus blockchain.



(b) Creation time of a service function chain versus the number of component VNFs for multiple amounts of consensus participants.



(c) Impact on throughput as the number of consensus participants and instruction sizes increase.



(d) Blockchain growth rate when issuing 100 transactions per second.

Fig. 6. BSec-NFVO prototype performance evaluation: overhead, consensus protocol, and blockchains growth rate.

transaction models. We implement a prototype of BSec-NFVO in the Open Platform for Network Function Virtualization (OPNFV) with a simplification of a byzantine fault tolerant consensus protocol to provide low-latency consensus while tolerating collusion of up to one third of consensus participants. The results show the delay introduced by the blockchain is not significant, and that BSec-NFVO is more effective in longer service chains. The peak throughput of the prototype is 803.3 transactions per second. Throughput remains stable for an increasing number of consensus participants and for instruction sizes of up to 64 B. For future works, we will formalize the adapted consensus protocol and use blockchain to ensure consistency under byzantine faults for software defined networking controllers.

VII. Acknowledgment

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001 and by CNPq, FAPERJ, and FAPESP (2015/24514-9, 2015/24485-9 and 2014/50937-1).

References

[1] J. Halpern and C. Pignataro, "Service Function Chaining (SFC) architecture," Internet Requests for Comments, http://www.rfc-editor.org/rfc/rfc7665.txt. Accessed October 31, 2018.

[2] M. D. Firoozjaei, J. P. Jeong, H. Ko, and H. Kim, "Security challenges with network functions virtualization," *Future Generation Computer Systems*, vol. 67, pp. 315–324, 2017.

[3] S. Lal, T. Taleb, and A. Dutta, "NFV: Security threats and best practices," *IEEE Comm. Mag.*, vol. 55, no. 8, pp. 211–217, 2017.

[4] X. Li, J. Mangues-Bafalluy, I. Pascual, G. Landi, F. Moscatelli, K. Antevski, C. J. Bernardos, L. Valcarenghi, B. Martini, C. F. Chiasserini *et al.*, "Service orchestration and federation for verticals," in *IEEE WCNCW*, 2018, pp. 260–265.

[5] I. J. Sanz, D. M. F. Mattos, and O. C. M. B. Duarte, "SFCPerf: An automatic performance evaluation framework for service function chaining," in *IEEE/IFIP NOMS*, 2018.

[6] I. D. Alvarenga, G. A. F. Rebello, and O. C. M. B. Duarte, "Securing management, configuration, and migration of virtual network functions using blockchain," in *IEEE/IFIP NOMS*, 2018.

[7] M. Pattaranantakul, R. He, Q. Song, Z. Zhang, and A. Meddahi, "NFV security survey: From use case driven threat analysis to state-of-the-art countermeasures," *IEEE Communications Surveys & Tutorials*, 2018.

[8] N. Paladi, A. Michalas, and D. Hai-Van, "Towards secure cloud orchestration for multi-cloud deployments," in *EuroSys-CrossCloud*, 2018.

[9] P. Massonet, "BEACON project: Software defined security service function chaining in federated clouds," in *Workshops of ESOCC*, 2016.

[10] Y. Khettab, M. Bagaa, D. L. C. Dutra, T. Taleb, and N. Toumi, "Virtual security as a service for 5G verticals," in *IEEE WCNC*, 2018, pp. 1–6.

[11] S. Zawoad and R. Hasan, "SECAP: Towards securing application provenance in the cloud," in *9th IEEE CLOUD*, 2016, pp. 900–903.

[12] N. Bozic, G. Pujolle, and S. Secci, "Securing virtual machine orchestration with blockchains," in *1st CSNet*, 2017.

[13] M. Vukolić, "The quest for scalable blockchain fabric: Proof-of-work vs. BFT replication," in *International Workshop on Open Problems in Network Security*. Springer, 2015, pp. 112–125.

[14] C. Cachin and M. Vukolić, "Blockchains consensus protocols in the wild," *arXiv preprint arXiv:1707.01873*, 2017.

[15] Hyperledger, "Hyperledger Fabric FAQ," 2018, https://hyperledger-fabric.readthedocs.io/en/release-1.3/Fabric-FAQ.html#bft. Accessed October 31, 2018.