A Blockchain Enabled SLA Compliance for Crowdsourced Edge based Network Function Virtualization

Mohammad Saidur Rahman
School of Computing Technologies
RMIT University
Melbourne, Australia
mohammadsaidur.rahman@rmit.edu.au

Ibrahim Khalil
School of Computing Technologies
RMIT University
Melbourne, Australia
ibrahim.khalil@rmit.edu.au

Mohammed Atiquzzaman
School of Computer Science
University of Oklahoma
Oklahoma, USA
atiq@ou.edu

***Abstract*—Network Function Virtualization (NFV) allows Edge devices to host different types of Virtual Network Functions (VNFs) and support Computing in the Network (COIN). Therefore, an edge device can abstract multiple VNFs of different Infrastructure Providers (InPs) as services and serve them at different times to act as a tenant for different InP. Deploying edge devices are expensive and difficult to manage. Hence, crowdsourced edge devices can be used. To ensure the quality and availability of the services and the privacy of sensitive data of the service stakeholders, it is extremely important to establish a service level agreement (SLA) between an edge device owner and an InP. The monitoring of SLA compliance and enforcing accountability, a trusted platform is required. Traditional NFV management authority cannot be trusted due to their centralized characteristics. In this article, we propose a blockchain-powered framework for the SLA compliance of edge devices offering VNFs as services and introduce a smart contract-driven framework to establish trust in edge-based NFV. We analyze the performance of the proposed framework in the context of the private blockchain network.**

***Index Terms*—Edge-Cloud continuum, in-network computing, COIN, Network function virtualization, blockchain, Service level agreement, policy enforcement, SLA compliance, Edge-based NFV, crowdsourced edge.**

I. Introduction

The advancement of edge computing enables different computing tasks to push from cloud to edge devices. In edge computing, a user can only access resources that are hosted by the local edge devices deployed by an infrastructure provider (InP) [1]. On the other hand, user can access cloud resources different geographic locations across countries and continents. In general, edge computing can provide realtime computing tasks that demands low latency, high bandwidth, and low processing power. contrarily, cloud computing can host computing tasks that are delay tolerant and resource hungry. Therefore, edge and cloud computing platform form a computing continuum to build innovative types of applications [2], [3].

Network Function Virtualization (NFV) [4] can be considered as an example application that can be benefited from the Edge-Cloud continuum. In NFV, the execution of network operations are decoupled from hardware to software. The edge-based NFV allows an edge device to host one or more Virtual Network Functions (VNFs) such as Deep Packet Inspection (DPI), Network Address Translation (NAT), Access Control (AC), packet routing, security, and privacy. VNFs are distributed among multiple edge devices with the aim of load balancing and serving the local users only. However, such distribution of VNFs requires deployment of sufficient edge devices which can be considered expensive from the InP's point-of-view. Moreover, the management of the edge devices can be cumbersome for an InP. To avoid the deployment and managment issues of edge devices, an InP can exploit the idea of crowdsourcing [5] edge devices. With crowdsourcing, different individuals or organizations should be allowed to register their edge devices in the infrastructure of the InP and providing NFV services by giving some financial benefits to the edge device owners.

Though crowdsourced edge device can play a significant role in the NFV, we identify the following issues that would hamper the success of the edge-cloud continuum for NFV:

- An InP does not have direct control on the crowdsourced edge devices availability. For instance, an owner of the edge device may join or leave the infrastructure at any time.
- It is impossible to maintain the Quality-of-Services (QoS) of third-party owned edge devices by an InP. For example, the owner of an edge device promises to ensure a level of reliability, security, and privacy during the NFV service provisioning. However, the edge device has failed to provide that. In such case, InP cannot improve the QoS but relying on the action of the owner of the edge device.
- The existence of too many edge devices, providing the same network function at the same time for a particular local network, would require optimization which will introduce additional computational overhead. On the other hand, lack of required edge devices to complete a particular NFV, would hamper the NFV operation.

To ensure the quality of services offered by edge device owners, a Service Level Agreement (SLA) needs to be devel-

oped between an InP and corresponding edge device provider. The SLA is the summary of the performance requirements [6] (e.g., processing latency, throughput, storage and processing capacity, availability and reliability) and the network functionality offered by an edge device provider which need to be maintained during the service provisioning. A monitoring system continuously evaluates the SLA compliance for ensuring the uninterrupted service provisioning and required QoS.

The SLA compliance is extremely important in the crowdsourced edge computing for the success of edge-cloud continuum. The traditional edge-cloud continuum is mainly operated under a platform with centralized administrative control which opens the chance of SLA and QoS tampering by a dishonest InP and internal or external cyber attackers. Therefore, the SLA compliance should be done in a trustworthy manner. Though the amalgamation of the NFV, edge computing, and blockchain technology enables trust in the NFV [7], a trustworthy SLA compliance is not yet explored.

In this paper, we introduce a blockchain [8] powered trustworthy SLA compliance framework for the crowdsourced edge-based NFV. In the proposed framework, the decentralized blockchain platform is leveraged for distributing the SLA among multiple nodes and ensuring the compliance of SLA between edge device providers and InPs with the help of Smart Contract [8] technology. Overall, contributions of this paper are as follows:

- We introduce a crowdsourced edge-based NFV framework for the edge cloud continuum.
- The framework is improved by integrating with a private blockchain for ensuring trust in the edge SLA management process.
- Further, a smart contract-based SLA enforcement mechanism is developed to achieve compliance.

II. Background

Blockchain is a trustworthy platform that stores data into multiple nodes in a decentralized manner [9]. In other words, each node stores the exact same copy of the data. To make any authorized changes in the blockchain data, a consensus needs to be achieved. Moreover, blockchain uses heavy cryptographic mechanisms that makes the data immutable in the blockchain. Smart Contract is a computer program that can be executed in the blockchain platform [8]. The smart contract is replicated in all blockchain nodes containing a set of rules to execute blockchain transactions. Blockchain transactions are validated by smart contracts before executing in the blockchain network to make sure the execution of only valid blockchain transactions. Hence, controlling the operation of IoT edge devices via smart contracts has become a new norm [10]. From that point of view, blockchain is a good candidate platform to build a trustworthy SLA compliance framework for the crowdsourced Edge-based NFV.

Authors in [6], presents a trustworthy architecture for multi-domain edge orchestration using blockchain for solving the problem of multi-constraint QoS. The architecture also introduces the automation of the fulfillment of service-level agreements through smart contracts. A blockchain enabled trustworthy slicing mechanism is proposed for NFV in [11]. In [12], a blockchain-based secure orchestration system for NFV, called BSec-NFVO, is proposed that ensures non-repudiation and integrity. Another blockchain-enabled edge-based distributed NFV framework is proposed in [13] for achieving consensus among multiple management and orchestration systems for NFV. However, none of the aforementioned approaches provides blockchain based SLA compliance for the crowdsourced Edge devices in NFV.

III. A Practical Case Scenario: Crowdsourced Edge-based Network Function Virtualization

In this article, we consider the SLA compliance of a virtualized content delivery system (VMCDS) [14] as a case scenario. In our scenario, VMCDS delivers various types of multimedia contents across the network. This system utilizes crowdsourced edge devices and virtualization layers to dynamically deploy VMCDS virtual functions in crowdsourced edge devices. The deployment is done based on an agreement, called *Service Level Agreement (SLA)*, between edge device owners and the cloud in VMCDS. The key objective of the SLA is to ensure the required content delivery services to each type of users based on their location, service request time, and subscription type.

Assume that there are multiple edge device owners that want to provide Virtualized Multimedia Content Delivery (VMCD) services by hosting Virtual Network Functions (VNFs). Therefore, edge device owners register their devices in VMCDS. We assume that an edge device owner can register multiple edge devices to provide similar or different types of services. VNFs include storing multimedia contents, load balancing, security, privacy, and data analytics related to content delivery. The type of VNF is defined in the SLA between an edge device owner and VMCDS. To guaranty the desired quality of services, each edge device must comply with the SLA to get paid for their service.

IV. Importance of SLA compliance in Crowdsourced Edge-based NFV

In this section, we discuss different possible parameters of an SLA that are applicable to the crowdsourced Edge-based NFV. Most importantly, we present our thoughts on the importance of SLA compliance in the crowdsourced Edge-based NFV.

Assume that there are four crowdsourced edge devices in the EC2N that are denoted as: E_A, E_B, E_C, and E_D. V_1,V_2,V_3, and V_4 are VNFs are hosted by E_A, E_B, E_C, and E_D, respectively. At the time of device registration, each edge device agrees on the functionality that it would provide and on some performance requirements (i.e., QoS). For instance, the owner of E_A agrees that it would provide a VNF V_1 (e.g., NAT functionality to the local user devices) between time t_1 and t_2. The owner of E_A also agrees 99.99% availability, 99.9% reliability, a latency of maximum 1.02 milliseconds, using security protocol SP, and data privacy mechanism $Priv$.

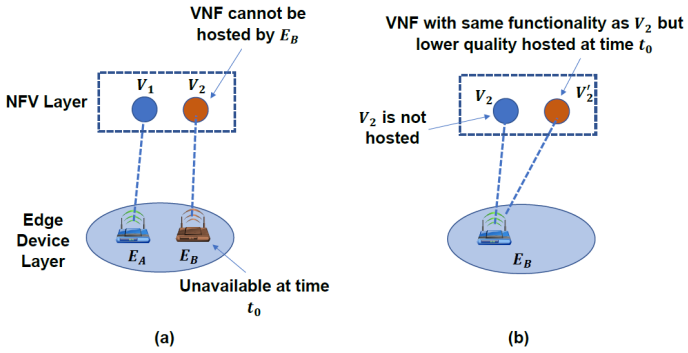


Fig. 1. Failure to comply with SLA: (a) Edge device is unavailable at time t_0 and (b) Edge device does not host the VNF as per SLA.

An SLA defines the name of the VNF (i.e., V_1) as the Network Function (NF) of E_A and the availability, reliability, latency, security protocol and privacy-preserving mechanism as the QoS parameters. The SLA for E_A is stored in the SLA repository and continuously monitored by the CM. Depending on the SLA, two or more edge devices are selected to provide a computing tasks with NFV.

If an edge device fails to comply with the SLA, i.e., fails to maintain the requirements according to SLA, the whole NFV system would be failed. Assume that an orchestrated computing service comprises of two VNFs: V_1 and V_2, that are hosted by E_A and E_B, respectively. Both E_A and E_B have SLA to serve their respective VNFs between time t_1 and t_2 daily. However, E_B is not available during the agreed time slot and the VNF cannot be executed (Fig. 1(a)). Consider another case where E_B is supposed to host VNF V_2. At a given time the E_B is not capable of providing V_2. Instead, E_B is offering $V_2^{'}$ (Fig. 1(b)). The later VNF has the same functionality (e.g., audio or video streaming) with comparatively low quality service. In all of the aforementioned cases, the device E_B failed to comply with SLA. As a result, the corresponding service and the whole NFV system are failed.

An SLA compliance methods is extremely important that would create several policy that would help to identify non-complying edge devices, enforce the policies to ensure compliance, and penalize the non-complying edge device owners.

V. Limitations of Traditional Edge-based NFV and Its SLA Compliance System

In this section, we discuss some of the key limitations of a traditional Edge-based NFV and Its SLA Compliance System that would fail the objective of crowdsourced edge-based NFV platform.

A. Risks of SLA Data Tampering

The traditional NFV systems are operated under the centralized authority of the infrastructure provider (InP). To make it more clear, SLA data are stored in the SLA repository that are manged by the InP. A dishonest InP can tamper the QoS data in the SLA to give benefit to one or more non-complying edge device owners or with the aim of sabotage. Linking with an earlier example given in Section IV, the device E_B hosts a different VNF instead of the agreed one and fails to comply with the SLA. The dishonest InP can alter SLA functional description data as well to hide the non-compliance of E_B and give it benefit. Overall, the InP has no obligation in the traditional Edge-based NFV and its SLA compliance system; hence, cannot be trusted.

B. Risks of Security

As crowdsourced Edge-based NFV involves untrusted edge device provider, the device registration module can be compromised by an internal or external attacker of the EC2N platform. Hence, any device can be compromised to host malicious VNFs in the compromised edge device or a malicious device can be deployed in the EC2N platform. For example, the edge device E_B is compromised by an attacker to host a VNF that would poison the NAT data. The NAT data poisoning would allow transferring network packets to a particular host owned by the attacker. As a result, the security of the EC2N platform would be at risk.

C. Risks of Privacy Breach

A registered edge device can take the advantage of accessing sensitive data that is stored in it. With the knowledge of sensitive data, the owner of the edge device can learn the business model of the InP and make money by revealing it to the other InPs. This is a serious privacy breach. Traditional SLA mechanism does not enforce any policy that would prevent the privacy breach in the EC2N platform.

VI. Toward a Blockchain Enabled SLA Compliance Framework for Crowdsourced EC2N

Though crowdsourcing can enhance the adaptation of edge devices in the NFV, it introduces several challenges in ensuring the SLA compliance. The SLA compliance mechanism should be made trustworthy to make the crowdsourced Edge-based NFV. In this section, we present our proposed blockchain enabled SLA compliance (BSC) framework for the Crowdsourced EC2N. At first, we present the overview of the proposed system. Later, we discuss the smart contract enabled SLA compliance for NFV.

A. Overview of the Blockchain Enabled Crowdsourced Edge-based Network Function Virtualization

In this section, we discuss our proposed blockchain-enabled crowdsourced edge-based NFV framework. The primary objective of the framework is to allow different individuals and organizations to register their edge devices in a Edge-Cloud continuum based NFV (EC2N) platform. Fig. 2 shows the overview of the framework. The framework consists of five layers: *Edge Device layer*, *SLA Compliance layer*, *NVF layer*, *orchestration layer*, and *service layer*.

The Edge Device (ED) layer consists of heterogeneous edge devices owned by different individuals and organizations. ED layer is the physical layer of Edge Cloud Continuum and resides at the bottom of the framework. Edge devices are

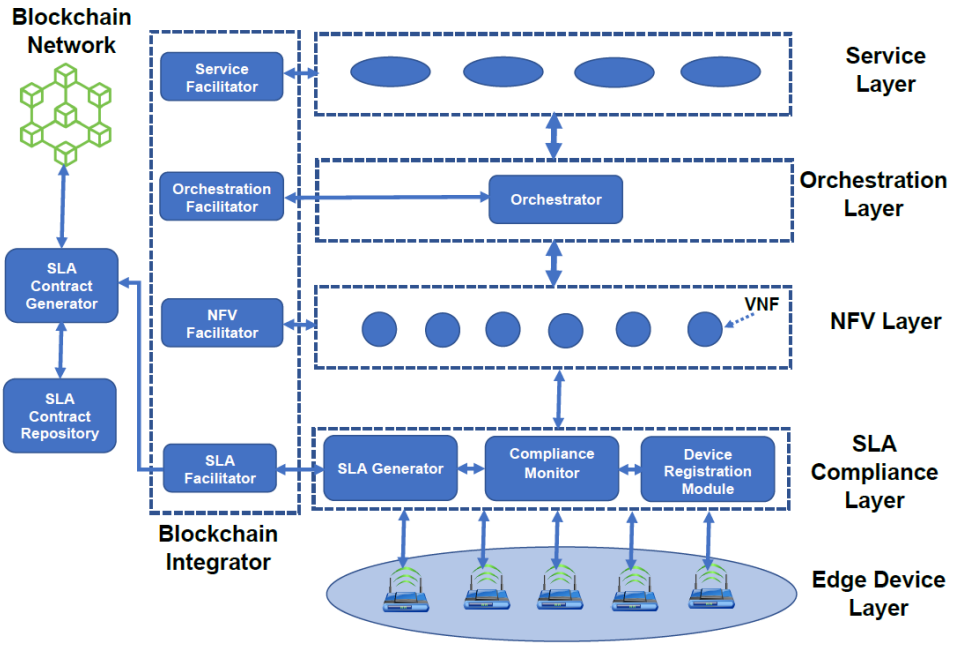


Fig. 2. Overview of the proposed blockchain enabled SLA compliance (BSC) framework.

mainly geographically distributed under the control of the infrastructure provider. SLA Compliance (SC) layer is the second layer of the framework that is responsible for edge device registration, SLA generation, and continuous monitoring of the SLA compliance. When an edge device owner joins the EC2N platform, it has to register itself into a module, called *Device Registration Module (DRM)* of the SC layer. During the registration process, DRM dynamically generates an SLA between the EC2N platform and the device using a pre-defined template. Once the SLA is generated, it is stored in the *SLA repository* within the SC layer. SC layer contains a *Compliance Monitor (CM)* that continuously observe the compliance of edge devices in the ED layer with the help of SLA stored in the SLA repository. Here, the CM works in a centralized fashion under the authority of the EC2N infrastructure provider.

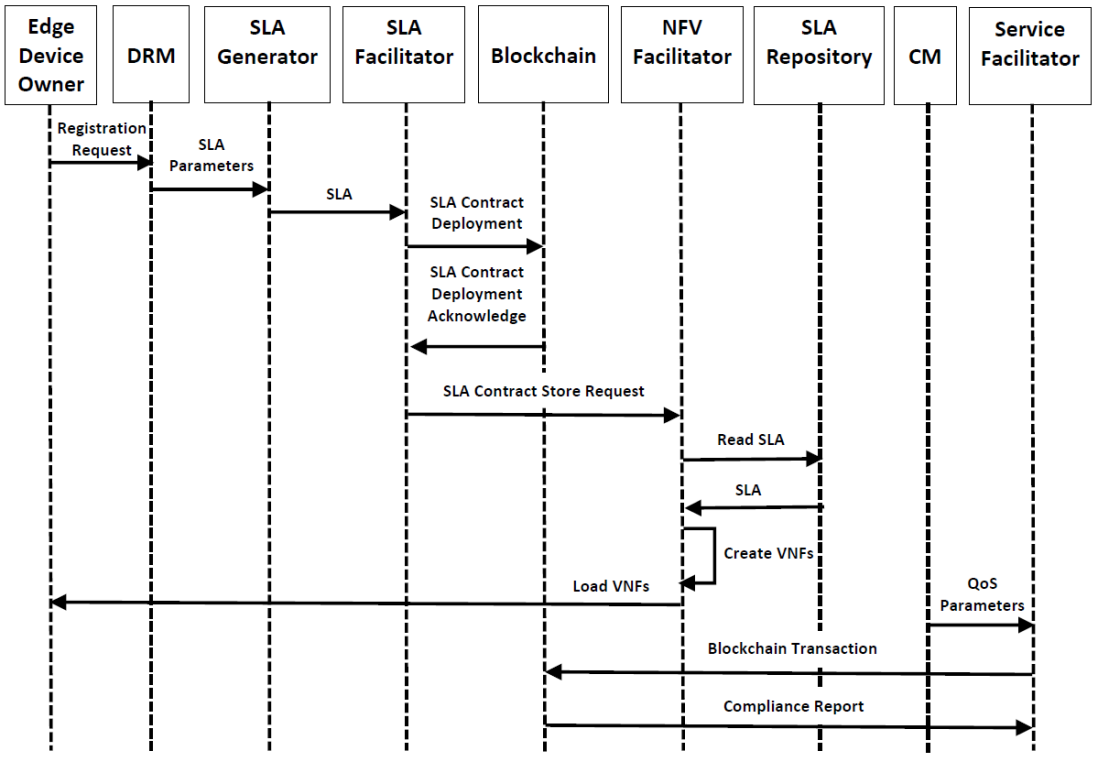
The third layer of the framework is the NFV layer that abstracts each hardware-based computing task as a Virtual Network Function (VNF). In this framework, it is assumed that a VNF alone cannot fulfil a complete computing task. Therefore, the NFV layer contains multiple VNF instances with the same and different functionality. The objective of this redundancy is to ensure the execution of any computing tasks at the close proximity of the geographically distributed users of the platform. Each VNF is hosted by an edge device in the ED layer and associated with an SLA in the SC layer. Therefore, an edge device acts as a Virtual Machine (VM) for the corresponding VNF.

The fourth layer is the orchestration layer that performs the VNF orchestration to generate a computing service. The orchestration layer, using the SLA stored in the SLA repository of SC layer, combines multiple VNFs that are available in the NFV layer. The fifth and final layer is the service layer that advertises the orchestrated computing services to the users.

In addition to the components of CEN framework, the BSC framework consists of four core components: *blockchain integrator*, *SLA contract generator*, *SLA contract repository*, and *blockchain network*. The SLA contract generator receives a contract generation request from the SLA generator as blockchain transaction. The transaction consists of functional and QoS parameters related to the SLA. SLA contract repository stores the SLA contracts for all of the SLAs in EC2N platform. The blockchain network stores SLA data in an immutable manner that is used by the SLA contracts for enforcing SLA compliance.

The blockchain integrator (BI) facilitates the communication among the components of the CEN framework. The BI is a software module that consists of multiple sub-components,referred as *facilitators*. Facilitator sub-components include *SLA facilitator*, *NFV facilitator*, *Orchestration facilitator*, and *Service facilitator*. The SLA facilitator ensures trustworthy SLA generation by interfacing among the SLA generator, SLA contract generator, and blockchain network. The NFV facilitator creates VNFs through the guidance of SLA contract generator. The orchestration facilitator interfaces between the orchestrator and the blockchain network. Overall, the facilitator sub-components facilitates the integration of the NFV operations.

In our proposed framework, we use a private blockchain network as we assume that the VMCDS is owned by a single entity. However, a consortium blockchain network can also be chosen if the VMCDS is a combination of multiple VMCD service providers. The private blockchain network of the proposed framework consists of multiple authorized nodes. In



(a)

```
Contract SLA{
   Address addr;
   String serviceType;
   DateRange dRange;
   DateTime dTime;
   function createSLAContract(Address addr, String serviceType, DateRange dRange, DateTime dTime, QoSParam qos){
      setAddress(addr);
      setServiceType (serviceType);
      setDateRange(dRange);
      setTime(dTime);
      setQoS(qos);
   }
}

Contract SLACompliance{
   function checkCompliance(addressOfEdgeDevice, serviceToDeply){
     Address addr = addressOfEdgeDevice;
     if(addr is in ValidDeviceList) then
        SLA s = readSLA(addr);
        QoS qValues = computeQoS(addr);
           if ( qValues comply with to s.QoS) then
              if (serviceToDeploy.type == s.serviceType && serviceToDeploy.time comply with s. dRange and s.dTime) then
                   Allow LOAD_NFV for addr
              else
                   Discard request
              endif
           else
              Discard request
           endif
       Discard request
       endif
   }

}
```

(b)

Fig. 3. (a) Flow diagram of the proposed blockchain enabled SLA compliance. (b) Smart Contract pseudocodes SLA Compliance.

our proposed framework, authorized nodes are selected by the VMCDS owner and nodes are geographically distributed. For example, a cloud-based multimedia content server can be an example of a blockchain node. The number of the blockchain nodes depends on the number of geographically distributed content servers of the VMCDS. We use an Ethereum based

private blockchain in this framework. Hence, the blockchain network uses Proof-of-Work (PoW) [15] based consensus protocol which is the default consensus mechanism of Ethereum platform. The PoW consensus mechanism is widely used in different existing blockchain networks. In PoW, the blockchain nodes can be devided into two classes: *prover* and *verifier*. Each prover node votes by solving proof of work instances (e.g., a complex mathematical challenge). On the other hand, the verifier node verifies if a prover node has spent sufficient computational power or not to solve the mathematical challenge. The voting requires extensive computation power which makes it hard to generate false vote. Therefore, only legitimate blockchain nodes can validate transactions and construct the appropriate blocks. The block with the majority votes is added to the blockchain. Our proposed framework stores an edge device's Quality-of-Service (QoS) data in the blockchain once validated by SLA contracts.

B. Blockchain Enabled SLA Compliance for NFV

In the proposed BSC framework, the SLA compliance is done by leveraging the blockchain and smart contract technology. Fig. 3(a) shows the overview of the compliance process. The SLA compliance process has the following sub-processes:

1) SLA contract generation: At the beginning, an edge device owner sends a device registration request to the DRM of the SC layer. Next, the DRM sends a SLA generation request to the SLA generator. After that, the SLA generator defines the functional and QoS parameters of the SLA between the edge device owner and InP and send the parameters to the SLA facilitator. The SLA facilitator generates a SLA smart contract, called *SLA contract* (see Fig. 3(b)). The SLA contract mainly sets the SLA information such as the *unique address of the edge device*, *proposed service type*, and *promised service date range and time*.

Finally, the SLA contract is deployed in the blockchain. During the deployment process, the SLA contract is distributed among all nodes in the blockchain network. Hence, an SLA contract cannot be changed by an internal or external attacker. Once the SLA contract is deployed in the blockchain, the edge device is registered in the DRM.

2) Creation of VNFs for Edge Devices: Once an edge device is registered, the SLA facilitator sends a notification to the NFV facilitator. The NFV facilitator reads the functional information of the edge device in its SLA and generates one or more VNFs for the edge device. The VNFs are loaded in the edge device.

3) SLA compliance via SLA contracts: During the execution of the VNFs by edge devices, the CM takes the performance parameters of an edge device and sends them to the service facilitator. The service facilitator generates a blockchain transaction with the performance data and send it to the blockchain network for the compliance check. The blockchain network execute the corresponding SLA contract of the edge device and validates the compliance of the SLA. The compliance report is notified to the CM. Fig. 3(b) illustrates the pseudocode of the $SLACompliance$ smart contract that validates the compliance of an edge device in the proposed framework.

C. System Analysis

In this section, we analyze the risk mitigation of the proposed system. The following analysis justifies how well the proposed method mitigates the risks of traditional edge-based NFV and its SLA compliance system mentioned in Section V.

1) SLA Data Tamper Resistance: In our proposed blockchain enabled SLA compliance framework, the SLA data and QoS values of crowdsourced edge devices are stored in the blockchain as a chain of hash values [8]. If an internal attacker (e.g., a dishonest InP) or external attacker want to tamper data in the blockchain, hashes of all the subsequent data needs to be recomputed and verified by the majority of the blockchain nodes. As the blockcahin of SLA and QoS data are distributed among multiple blockchain nodes, an attacker must change the hash values in the blockchain of SLA data and QoS values of majority of the blockchain nodes. Therefore, an attacker cannot tamper the SLA data and QoS values and SLA data becomes tamper resistance in our proposed framework.

2) Secure Device Registration and SLA compliance: The proposed framework allows the registration of edge devices via smart contracts. A smart contract is generated for each edge device containing the SLA and QoS information and the byte code of smart contract is distributed among all the blockchain nodes [8]. The smart contract for an edge device is generated only if the device is an authorized participant and can fulfil the network requirement. For example, an authorized edge node must have required number of coins to apply for a registration. At the first point, a malicious device cannot register itself in the network. If a registered edge is compromised by an attacker later, the attacker must change the SLA data in all blockchain node to compromise the SLA validation process which is extremely difficult for an attacker. Thus, our proposed framework is secure from unauthorized device registration and SLA smart contract alteration.

3) Privacy Preservation during SLA Compliance: The compliance of SLA is verified via the SLA smart contract which is generated for a device. The SLA smart contract ensures that a device can only access the QoS and SLA information for which it is entitled to. Hence, the sensitive information related to SLA dnd QoS are not revealed to any party including the InP and edge devices. Hence, a privacy-preserving SLA compliance is ensured in this proposed framework.

VII. Experimental Results and Performance Evaluation

In this section, we conduct several experiments to evaluate the performance of the proposed blockchain enabled SLA compliance framework. The main objective of the performance evaluation is to check the computational overhead introduced by the blockchain technology.

We consider different number of virtual edge devices (100 to 500). For the sake of simplicity, we assume that each edge

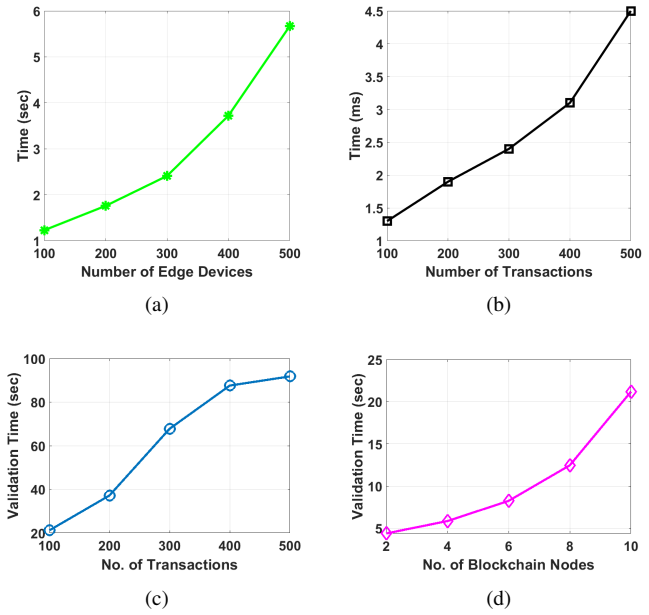


Fig. 4. (a) Time required for generating SLA contracts at the time of registration for different number of edge devices (number of VNFs = 5). (b) Time required for validating SLA of Edge devices without smart contract. (c) Time required for validating simultaneous transactions with SLA contract for different number of transactions and a fixed number of nodes (number of blockchain nodes = 2). (d) Time required for validating simultaneous transactions with SLA contract for a fixed number of transactions (number of transactions = 100) and different number of blockchain nodes.

device hosts only one VNF and only one SLA is required for each edge device. The experiments are performed on Amazon AWS EC2 (c4.2xlarge instance) with Ubuntu 16.04 operating system, 16GB memory, and Intel E5-1650 8 core CPU. Experiments are conducted in Ethereum based private blockchain network [15] using default network topology of the Ethereum. Smart contracts are written using *solidity* [15]. Java-based programs are used to compute the execution times for blockchain based SLA compliance. As the integration of the blockchain is at the core of validating the SLA compliance, the experiments are designed to evaluate the execution time for the edge device registration and compliance tasks with different number of blockchain transactions and blockchain nodes.

At first, we show the time required for generating smart contracts (i.e., SLA contracts) in Fig. 4(a). We consider different number of edge devices while keeping the number of VNFS=5. The required time increases exponentially if the number of simultaneous edge devices increase. Next, we show the SLA compliance of edge devices in the traditional NFV based VMCDS that does not include blockchain and smart contract technology. Fig. 4(a) shows that the SLA validation time is very fast in traditional NFV based VMCDS.

We evaluate the performance of the SLA Compliance of edge devices via smart contracts (SLA contracts) in terms of the required times for validating different number of simultaneous blockchain transactions to check the SLA compliance. While keeping the number of blockchain nodes equal to 2, the validation time increases almost linearly when simultaneous transactions increase (see in Fig. 4(c)). Next, the required times are computed for different number of blockchain nodes while keeping the number of simultaneous transactions equal to 100. Results show that the validation time increases exponentially if the number of blockchain nodes is increased (Fig. 4(d)). Hence the number of blockchain nodes should be kept minimal in the proposed Edge SLA compliance mechanism for better scalability.

This paper introduces the crowdsourced edge devices for NFV. To the best of our knowledge, the proposed method is the first attempt to enforce the SLA compliance of crowdsourced edge-based NFV. Existing research work on edge-based NFV [6], [11]–[13] do not consider the SLA compliance techniques of crowdsourced edge devices. Therefore, the performance of our proposed work cannot be compared with existing results. The experimental results exhibit the feasibility of the proposed blockchain enabled SLA compliance framework for the crowdsourced edge-based NFV.

VIII. Conclusion

This paper addresses the Service Level Agreement (SLA) compliance issue in the Network Function Virtualization (NFV). A Virtualized Multimedia Content Delivery (VMCD) system is considered as a practical case scenario in this paper. SLA compliance in edge-based NFV is very important as any device can join an edge network to compromise an edge device and behave maliciously. Therefore, it is extremely important to ensure secure registration of edge devices and monitoring their compliance to SLA in NFV. State-of-the-art techniques, such as cryptography, can ensure the secrecy of communication; however, fails to handle inside attack in most of the cases. From that point of view, this paper proposes a blockchain and smart contract enabled SLA compliance technique. SLA data and compliance verification codes, respectively for SLA data storage and SLA compliance verification, can be stored in a decentralized manner in the blockchain network. Hence, changing SLA agreement and SLA smart contract become infeasible for an attacker. Therefore, the proposed framework ensures trustworthy SLA compliance.

This paper introduces a novel edge device SLA compliance approach for NFVs using the blockchain and smart contract technology. Initially, a framework of the crowdsourced edge-based NFV framework is proposed. As the crowdsourced edge devices cannot be fully trusted, an SLA compliance mechanism is introduced. To ensure the trust of SLA compliance, the blockchain and smart contract technologies are used to extend the proposed framework. The SLA compliance framework is implemented in a Ethereum based private blockchain to exhibit the feasibility. We consider the scalability as the feasibility parameter. The experimental results shows that the scalability of the SLA compliance framework is good. However, the number of nodes in the blockchain should be minimal. As the proposed framework uses PoW consensus algorithm, which is the Ethereum blockchain's default consensus algorithm, the validation times are higher due to its complexity. In our future work, we will investigate the scalability of the proposed SLA

compliance framework with other consensus mechanism in both private and consortium blockchain networks.

REFERENCES

[1] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE internet of things journal*, vol. 3, no. 5, pp. 637–646, 2016.

[2] L. Baresi, D. F. Mendonça, M. Garriga, S. Guinea, and G. Quattrocchi, "A unified model for the mobile-edge-cloud continuum," *ACM Transactions on Internet Technology (TOIT)*, vol. 19, no. 2, pp. 1–21, 2019.

[3] L. Bittencourt, R. Immich, R. Sakellariou, N. Fonseca, E. Madeira, M. Curado, L. Villas, L. DaSilva, C. Lee, and O. Rana, "The internet of things, fog and cloud continuum: Integration and challenges," *Internet of Things*, vol. 3, pp. 134–155, 2018.

[4] N. F. Virtualisation, "An introduction, benefits, enablers, challenges & call for action," in *White Paper, SDN and OpenFlow World Congress*, 2012.

[5] S. Yu, X. Chen, S. Wang, L. Pu, and D. Wu, "An edge computing-based photo crowdsourcing framework for real-time 3d reconstruction," *IEEE Transactions on Mobile Computing*, 2020.

[6] V. K. Rathi, V. Chaudhary, N. K. Rajput, B. Ahuja, A. K. Jaiswal, D. Gupta, M. Elhoseny, and M. Hammoudeh, "A blockchain-enabled multi domain edge computing orchestrator," *IEEE Internet of Things Magazine*, vol. 3, no. 2, pp. 30–36, 2020.

[7] D. B. Rawat, "Fusion of software defined networking, edge computing, and blockchain technology for wireless network virtualization," *IEEE Communications Magazine*, vol. 57, no. 10, pp. 50–55, 2019.

[8] V. Buterin *et al.*, "A next-generation smart contract and decentralized application platform," *white paper*, vol. 3, no. 37, 2014.

[9] Z. Zheng, S. Xie, H. Dai, X. Chen, and H. Wang, "An overview of blockchain technology: Architecture, consensus, and future trends," in *2017 IEEE International Congress on Big Data (BigData Congress)*. IEEE, 2017, pp. 557–564.

[10] J. Wu, M. Dong, K. Ota, J. Li, and W. Yang, "Application-aware consensus management for software-defined intelligent blockchain in iot," *IEEE Network*, vol. 34, no. 1, pp. 69–75, 2020.

[11] J. Backman, S. Yrjölä, K. Valtanen, and O. Mämmelä, "Blockchain network slice broker in 5g: Slice leasing in factory of the future use case," in *2017 Internet of Things Business Models, Users, and Networks*. IEEE, 2017, pp. 1–8.

[12] G. A. F. Rebello, I. D. Alvarenga, I. J. Sanz, and O. C. M. Duarte, "Bsec-NFVO: A blockchain-based security for network function virtualization orchestration," in *ICC 2019-2019 IEEE International Conference on Communications (ICC)*. IEEE, 2019, pp. 1–6.

[13] X. Fu, F. R. Yu, J. Wang, Q. Qi, and J. Liao, "Performance optimization for blockchain-enabled distributed network function virtualization management and orchestration," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 6, pp. 6670–6679, 2020.

[14] S. Fu, J. Liu, and W. Zhu, "Multimedia content delivery with network function virtualization: The energy perspective," *IEEE MultiMedia*, vol. 24, no. 3, pp. 38–47, 2017.

[15] "Ethereum repository." [Online]. Available: https://github.com/ethereum/go-ethereum (Accessed on: March 15, 2020)

Mohammad Saidur Rahman is a research associate in School of Computing Technologies, RMIT University, Melbourne, Australia. He has received his Ph.D. degree in Computer Science from School of Science, RMIT University, Melbourne, Australia. Rahman received B.Sc. and M.Sc. degrees from American International University-Bangladesh (AIUB) of Dhaka, Bangladesh, in 2007 and 2009. He worked as a faculty member in AIUB before starting his Ph.D. in RMIT University. His current research interests include blockchain, data privacy, lossless data hiding, Internet-of-Things (IoT), and service computing.

Ibrahim Khalil is an associate professor in School of Computing Technologies, RMIT University, Melbourne, Australia. He received the Ph.D. degree in Computer Science from the University of Berne, Switzerland, in 2003. He has several years of experience in Silicon Valley Companies. Ibrahim also worked with EPFL and the University of Berne in Switzerland, and Osaka University in Japan. His research interests include scalable computing in distributed systems, e-health, wireless and body sensor networks, biomedical signal processing, remote health care, network and data security, secure data analytics and privacy.

Mohammad Atiquzzaman received the MS and PhD degrees in electrical engineering and electronics from the University of Manchester, United Kingdom. He currently holds the Edith Kinney Gaylord Presidential professorship in the School of Computer Science at the University of Oklahoma. He is the editor-inchief of Journal of Networks and Computer Applications, founding editor-in-chief of Vehicular Communications and has served/serving on the editorial boards of various IEEE journals and co-chaired numerous IEEE international conferences including IEEE Globecom. His research interests are in communications switching, transport protocols, wireless and mobile networks, satellite networks, and optical communications. He is a senior member of the IEEE.