Blockchain-Based Secure Distributed Control for Software Defined Optical Networking

Hui Yang[1,*], Yongshen Liang[1], Qiuyan Yao[1], Shaoyong Guo[2], Ao Yu[1], Jie Zhang[1]

[1] State Key Laboratory of Information Photonics and Optical Communications, Beijing University of Posts and Telecommunications, Beijing 100876, China

[2] State Key Laboratory of Networking and Switch Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China

* The corresponding author, email: yang.hui.y@126.com

Abstract: Software defined optical networking (SDON) is a critical technology for the next generation network with the advantages of programmable control and etc. As one of the key issues of SDON, the security of control plane has also received extensive attention, especially in certain network scenarios with high security requirement. Due to the existence of vulnerabilities and heavy overhead, the existing firewalls and distributed control technologies cannot solve the control plane security problem well. In this paper, we propose a distributed control architecture for SDON using the blockchain technique (BlockCtrl). The proposed BlockCtrl model introduces the advantages of blockchain into SDON to achieve a high-efficiency fault tolerant control. We have evaluated the performance of our proposed architecture and compared it to the existing models with respect to various metrics including processing rate, recovery latency and etc. The numerical results show that the BlockCtrl is capable of attacks detection and fault tolerant control in SDON with high performance on resource utilization and service correlation.

Keywords: blockchain; fault tolerant control; network security; software defined networking

Received: Oct. 25, 2018
Editor: Erwu Liu

I. INTRODUCTION

WITH the characteristics of programmable control, open standards and etc., software defined networking (SDN) has received increasing attention in recent research and industry [1, 2]. The integration of SDN and other technologies (e.g., mobile network and data centre network) as an innovative architecture has likewise become an important direction of current network evolution. As a research hotspot in recent years, the software defined optical networking (SDON) can better adapt to the intelligent development of large-capacity optical networks by introducing an SDN control plane model at the optical layer. SDON expands the SDN-related protocols and develops object-oriented interactive control interfaces, to achieve heterogeneous network information abstraction and cross-stratum network control integration [3]. Accordingly, SDON becomes a novel heterogeneous network architecture with unified control capabilities between wired and wireless networks, access and core networks as well as data and the optical networks [4, 5]. In addition, the network resources virtualization of SDON can better utilize the network infrastructure resources, and optimize the uti-

lization of network resources through a software defined resources management platform [6-10]. Based on the network architecture of virtualization, it is possible to quickly and efficiently control network resources under the premise of ensuring the quality of service according to the application needs of different services [11, 12]. With the above advantages, SDON has already become a new development direction for future optical network technology.

With the development of SDON, control plane becomes the key to realize optical network intelligence which is also a bridge connecting the underlying switching devices and the upper application [13, 14]. Since the controller plays an important role in SDON, it has already become the primary goal of network attacks. Attackers can launch massive DoS attacks to a controller, whose process resources will be exhausted by these useless requests. As a result, the controller can no longer respond to normal network services [15, 16]. Furthermore, attackers will even deploy their own controllers to spoof those underlying switching devices, confusing them to regard the attacker-controlled "disguised" controller as their master controller and give it the highest level of network control permissions. A single point of failure, such as the downtime or malicious action of controllers will degrade the network performance and even lead to a network paralysis. The existing firewall technologies are designed to prevent illegal requests from appending to the controller processing sequence by checking and filtering network packets. However, the firewall cannot guarantee the absolute filtering of illegal packets because of the vulnerability such as the flow table being easily modified and the firewall policy conflicts. Once the number of DoS attack messages is sufficient to exhaust the I/O or computing resources of the target controller, the single point of failure will be used by attackers to tamper with the flow rules and push illegal flow tables as their wishes. As a result, malicious traffic can easily enter the network for data theft and malicious modification, which pose a great threat to network security.

OpenFlow, developed and maintained by the Open Networking Foundation [17], has already become the mainstream standard of SDON in recent years. In OpenFlow, all the data plane forwarding functions are performed by optical switching nodes while the control plane is represented by a (logically) centralized controller. In order to protect such a centralized control architecture from being a single point of failure, a high cost will be spent in the packet filtering of controllers, which is yet difficult to guarantee that it will never be compromised. Therefore, the distributed control architectures have recently attracted the attention of researchers. Distributed control architectures have the advantages of fault tolerant control, redundant backup, and scalability, enabling the highly available SDON [13, 14]. In the distributed control architecture, switching nodes can recover to a slave controller once the original master controller fails, so that the traffic can be continuously processed regardless of the single points of failure. Note that the data consistency between multiple controllers of distributed control architecture is the basis of the above failure recovery. However, most of the current consistency studies are only about controller state information, not taking into account the state information of the switching nodes, which will cause the switching nodes to repeatedly execute commands after recovery [18]. The problem of command repeatedly execution cannot be ignored since many operations are not idempotent, and repeated execution will bring more problems.

Due to its excellent performance in distributed systems, the application of blockchain technology to the integration of various distributed scenario has become a hot research topic recently [19-21]. Based on the complexity and security threats of current distributed networks, the blockchain puts forward three technological innovations to solve the trust and security problems in distributed networks. *Distributed ledger* is built by all members in a distributed network and ensures the data

consistency through consensus, which is different from traditional distributed storage synchronizing data by backing up the central node to other nodes. *Consensus mechanism* is a process that uses special message interaction and voting to make a transaction endorsed by the entire network, even though the participating members do not trust each other. *Smart contract* is a rule and term embedded in nodes that are automatically executed through the pre-configured script code, based on those trust and untamable data provided by distributed ledgers. Taking the above special advantages for data consistency of blockchain, the combination of blockchain and distributed control architecture can solve the problem of command repeatedly execution in failure recovery and provide the trusted collaboration of SDON controllers as well as leverage smart contracts for efficient traffic processing in SDON.

Contribution: In this paper, we design a blockchain-based distributed SDON control architecture that deals with the single point of failure caused by DoS or spoofing attacks and data consistency problem mentioned above. Our architecture allows the controllers to use consensus mechanism for traffic collaboration processing in distributed control architecture to achieve fault tolerant control. In addition, when the initial master controller being faulty in the attacked/failure scenario, the presented architecture enables the SDON switching nodes to promptly recover to the optimal slave controller using distributed ledgers and smart contracts. The performances of the proposed architecture are evaluated and compared with the existing models with respect to service processing speed in attack scenarios, fault recovery delay in faulty tolerant scenarios, and network performances including resource utilization and service correlation.

Organization: The rest of this paper is structured as follows. Section II discusses the related work of distributed control architecture and the combination with blockchain in SDON. Section III discusses the design principles required for a distributed control architecture in SDON and presents the proposed blockchain-based secure distributed control architecture and high- efficiency failure recovery strategy of SDON. Section IV describes the specific implementation of our architecture in details. Section V presents the evaluation of the proposed architecture based on different performance metrics. Section VI gives the conclusions of this research.

II. Related Work

Recent several efforts have been made for distributed SDON control architecture. These studies can be divided into two major types of redundancy-based and consensus-based, which will be discussed in this section. Besides, we also introduce several application cases of blockchain combined with communication networks.

2.1 Redundancy-based distributed control architecture

The major of traditional distributed control architectures achieve the distributed fault tolerant control through redundant replicas. SMaRtLight [22] realized the controller state redundancy through state replication of master and slave controllers. The master controller takes control of all network decisions, while the slave controller replicas need to back up the synchronized network view in order to take over the global network control in case of a single point of failure occurs on the master controller. All controller replicas guarantee a consistent data store for fault tolerance and strongly consistency using an implementation of Replicated State Machine (RSM). Ravana [23] also extended the RSM approach to deal with switch state consistency under failures. It uses a two-step replication protocol that separates the logging of the event delivery information from the logging of the event processing transaction completion information in the consistent store in order to guarantee consistency under the failure scenario. However, both SMaRtLight and Ravana implement failure recovery by means of redundancy-based

recovery which does not provide the data consistency about switching nodes. These solutions are unable to address the problem of commands repeating command execution mentioned above.

2.2 Consensus-based distributed control architecture

Recently, most distributed control designs rely on consensus mechanisms to ensure the data real-time consistency of multiple controllers. Onix [24] is a distributed control architecture based on Paxos consensus mechanism for data consistency. It offers two data-store options already implemented by Onix: a replicated transactional database that guarantees strong consistency at the cost of good performance for persistent but slowly-changing data (state), and a high performance memory-only distributed hash table (DHT) for volatile data that does not require strict consistency. Eldefrawy et al. [25] designed the Byzantine fault tolerant control architecture using BFT consensus which can obviously reduce threats of single points of failure to the SDON. Among this architecture, the proxy servers used to achieve sorting, receiving and sending of transactions in BFT consensus existing between the control plane and the data plane will become the new attack targets. However, the distributed control architecture based on consensus mechanisms as described above will bring up a lot of communication overheads that only achieve the purpose of data consistency.

2.3 Combination of blockchain and communications network

Although there are still few cases in which the blockchain is used for SDON, it has already shown extraordinary talents in many network scenarios involving security and distributed systems. Broadly considering the needs of future cloud infrastructure, Sharma et al. [26] proposed a blockchain-based architecture for cloud computing that integrated SDN and edge computing to reduce the need for trusted third party platforms and cloud computing costs. This architecture outlined a blockchain-based service provisioning in the cloud, blockchain-connected SDN controllers in fog clusters, SDN-enabled base stations at the edge, and a 2-hop blockchain consensus called Proof-of-Service. In our previous research [27-30], we proposed a blockchain-based trusted cloud radio over optical fiber network architecture (BlockONet) with anonymous access identification for Internet of Things (IoT), which uses a global load balancing strategy and a zero-knowledge-proof algorithm to implement the secure access of IoT devices with superior network performance. In general, as an emerging technology, there are still many blockchain-related network applications to be explored, which will be a hot research spot in the future.

III. Distributed Blockchain-Based Trusted Edge Computing Cooperation

This paper tries to integrate the blockchain technology into SDON to design an efficient and secure distributed control architecture. In this section, we firstly analyze the problems faced in designing a distributed control architecture and then describes the blockchain-based efficient secure distributed control architecture in detail.

3.1 Problems analysis

To design a feasible distributed control architecture for SDON, the following problems must be addressed into consideration:

1) Fault Tolerant Transaction Processing

As mentioned above, firewalls cannot guarantee the absolute blocking of illegal packets. When illegal flows successfully enter the controller network and cause a single point of failure of the controller, it is particularly important how to implement fault-tolerant processing of the service.

2) Failure Recovery

When a controller failure occurs, in order to ensure the continued service process, there must be an available failure recovery mechanism to enable the switching nodes connected

to the faulty controller to recover to other normal controllers.

3) Data/State Consistency

Consistency is the basis of distributed control. A feasible failure recovery strategy requires not only a consistent network view, but a synchronization state of underlying network devices. A failure recovery without considering the real-time state of underlying devices may cause problems such as repetitive execution of commands.

4) Network Performance

Strong consistency is mostly achieved through a consensus algorithm, which brings a lot of communication overheads to the network. Although these costs are difficult to eliminate, it is an alternative approach to use consensus results to improve network performance from another perspective.

Based on the analysis presented above, we find that the existing architectures are unable to completely address the problems faced by distributed control for their executions. Meanwhile, many technical components of the blockchain such as consensus mechanism and distributed ledger can meet the above design requirements of the distributed control architecture. Distributed ledgers enable consistent logging and data management for control plane transactions, while consensus mechanisms ensure fault tolerant processing of transactions and provide a foundation for data consistency. In this section, we propose a novel blockchain-based distributed control architecture (BlockCtrl) to meet the requirement for security and efficiency in SDON.

3.2 Blockchain-based distributed control architecture

Figure 1 presents an overview of the proposed architecture, which is categorized into four planes, i.e. data, control, consensus and contract. The data plane provides underlying data forwarding function which is software defined with OpenFlow protocol and controlled by the SDON controllers for simplicity. In the control plane, by using the blockchain technique, all the controllers are connected in a distributed manner within different control domains, each of which contains multiple switching devices. Meanwhile, at the software level, each controller in control plane is loaded with the identical distributed ledger maintaining by consensus plane and smart contracts using the consistent data in the distributed ledger to provide the customized network function. The consensus plane performs multi-controller consensus for the pending-process services sent to the con-

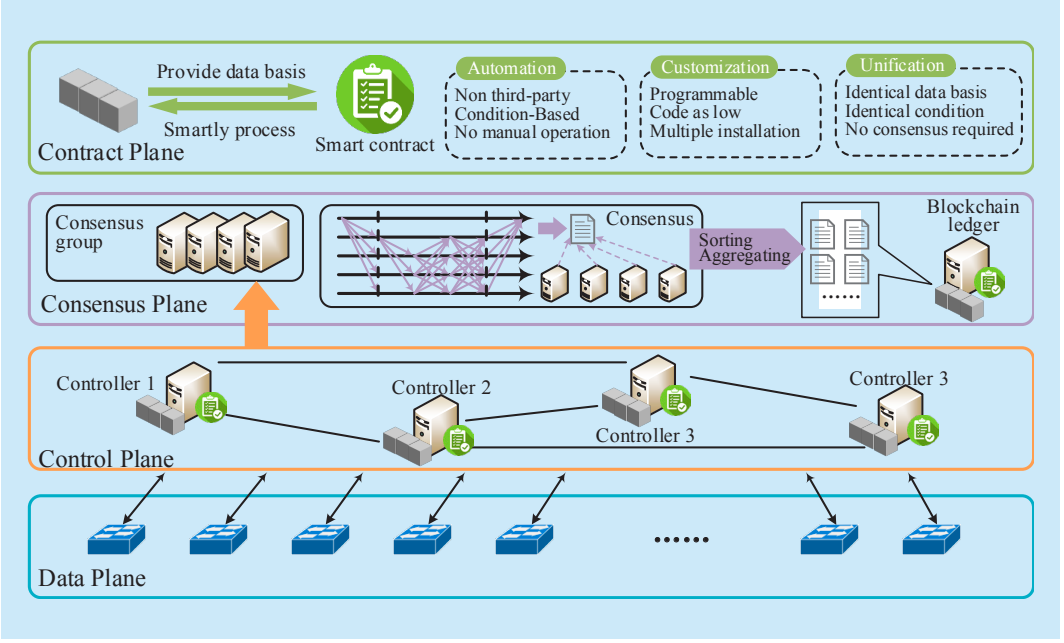

Fig. 1. *The architecture of proposed BlockCtrl.*

troller to generate consistent process results of the entire network and sorts the results into a block data structure to be appended to the distributed ledger of multi-controller. The contract plane contains scripts, algorithms, and smart contracts which can be automatically invoked when the constraint is reached without manual operation or third-party participation, and can also be automatically cancelled when the conditions are not met.

The proposed model guarantees data consistency with respect to switching equipment and controllers, and allows for the wider implantation of advanced network functions (e.g., failure recovery and data analysis) while ensuring secure control. The characteristics of BlockCtrl mainly include three aspects which are described as follow.

1) Fault Tolerance based on Consensus

With the introduction of a fault-tolerant consensus mechanism, the distributed controllers invariably get the correct traffic processing results, in the case where the number of faulty controllers is below the fault-tolerance threshold, whether the fault controller is down or controlled by attackers.

2) Data Consistency based on Distributed Ledger

Each replica of the consensus group formed by the distributed controllers can aggregate the consensus results in a block through ordering by the primary controller, which is appended to the identical blockchain ledger. Storing the network topology information as well as the real-time configuration of switches, blockchain ledgers ensure a consistent status of the switches to all the distributed controllers for more accurate distributed services processing.

3) Smart Contract

Based on the blockchain ledger, a customizable smart contract is installed in each controller, which automatically performs the network functions loaded in the smart contract that stipulates the execution conditions without the participation of third-parties. With smart contracts, each controller is able to perform consistent operations based on the identical blockchain ledger, without third-party or additional communication to prove the consistency of operations.

3.3 Blockchain ledger-based efficient recovery strategy

Although the complex consensus mechanism brings a large communication overhead, it also ensures the strong consistency of the device state. On the other hand, the failure recovery strategies of the existing architecture are static, and the master-slave relationship of controllers is preset in the system. When a single point of failure occurs, controllers functioned well would take over the work of the faulty controller by the determined master-slave relationship. However, the current network traffic is dynamic and huge. Such a static recovery strategy not only reduces the service correlation of the switching nodes to the controllers after recovery but also increases the bandwidth load of the network link causing service blocking, which greatly degrades the network performance of SDON.

Based on BlockCtrl described above, we propose a blockchain ledger-based efficient recovery (BlockRec) strategy for the contract plane in failure scenarios. This strategy exploits the blockchain distributed storage feature, enabling the switching nodes to quickly recover to the optimal master controller with highest service correlation and resource utilization when the initial master controller goes down or does evil, without the repeated interactions as in previous schemes. In this strategy, the smart contract stipulates that one block will be generated for each n traffic process results and isolate the faulty controller from consensus group immediately after discovering through consensus.

To enhance the network performance of BlockCtrl, we introduce the global load balancing mechanism [4, 5] and establish an evaluate factor α to calculate the optimal slave controller for switching nodes while failure recovery is performed. The normalization factor contains the processing resources of the control plane, link resources of data plane and service correlation parameters. In each SDON

controller, three time variable processing parameters, including RAM utilization *R(t) R*, CPU utilization *R(t) C*, and I/O scheduling condition *R(t) I*, describe the processing resources. The network links parameters contain the occupied bandwidth B_l and transmission delay τ_l on each link, as well as the number of hops H_p on each candidate path. Besides, we also define *r s C* to indicate the services relationship between switching nodes and controllers.

In the expression of controller resource utilization, we define three adjustable estimation levels k_C, k_R and k_I to describe the relative weight between CPU, RAM utilization, and I/O scheduling rate. In order to simplify calculations, we take the values of k_C, k_R and k_I as three fixed values Q_1, Q_2, and Q_3 which satisfy $Q_1 > Q_2 > Q_3$ and $Q_1 + Q_2 + Q_3 = 1$. According to the rule of high utilization corresponding to high priority, three parameters select the corresponding priority according to the respective mathematical expectation in the past t_0 time. Specifically, when $E_C(t_0) \geq E_R(t_0) \geq E_I(t_0)$, the estimation levels will change to $k_C = Q_1$, $k_R = Q_2$, and $k_I = Q_3$. The expectation of CPU utilization in the past t_0 time $E_C(t_0)$ is used to estimate the occupancy of the recent CPU resources, as described in (1), where t_c and *f (t) c* represent the current time and the probability of CPU utilization. According to the same principle, $E_R(t_0)$ and $E_I(t_0)$ described in (2) and (3) represent the expectations of RAM resource utilization and I/O scheduling ratio respectively. Therefore, the resource occupancy rate of each controller has the above three resource parameters and is described as (4).

$$E_C(t_0) = \sum_{t=t_c-t_0}^{t_c} R_C^{(t)} f_C^{(t)} \Big/ \sum_{t=t_c-t_0}^{t_c} f_C^{(t)}, t \in [t_c - t_0, t_c]. \quad (1)$$

$$E_R(t_0) = \sum_{t=t_c-t_0}^{t_c} R_R^{(t)} f_R^{(t)} \Big/ \sum_{t=t_c-t_0}^{t_c} f_R^{(t)}, t \in [t_c - t_0, t_c]. \quad (2)$$

$$E_I(t_0) = \sum_{t=t_c-t_0}^{t_c} R_I^{(t)} f_I^{(t)} \Big/ \sum_{t=t_c-t_0}^{t_c} f_I^{(t)}, t \in [t_c - t_0, t_c]. \quad (3)$$

$$f_{ac}\left[R_C^{(t)}, R_R^{(t)}, R_I^{(t)}, k\right] = \frac{k_C \times R_C^{(t)} + k_R \times R_R^{(t)} + k_I \times R_I^{(t)}}{k_C + k_R + k_I}. \quad (4)$$

In addition, the adjustable estimation levels k_B, k_τ between the bandwidth and delay of links can also be dynamically adjusted at the data plane according to the above scheme. Thus the network resource parameter expression for each candidate path is represented as a dimensionless factor shown as(5), where *B* is the total bandwidth of the link.

$$f_{bc}\left[B_l, \tau_l, H_p\right] = k_B \sum_{l=1}^{H_p} \frac{B_l}{H_p B} + k_\tau \sum_{l=1}^{H_p} \frac{\tau_l}{H_p \tau}. \quad (5)$$

For service correlation parameters, we use (6) to describe the degree of service correlation of controller *C* with the switching node affected by the failure. Among the *n* services stored in the latest block, there are *p* services related to the affected switching nodes while *r s C* represents the relationship between each service *s* and the controller *C*. A value of 1 for *r s C* means that the service is related to the controller, and a value of 0 indicates the opposite meaning.

$$f_{cc}\left[m, r_C^s\right] = \sum_{s=1}^{m} r_C^s \Big/ m. \quad (6)$$

To sum up, the evaluate factor α with *k* candidate slave controllers is described as (7), whereβ and γ are the adjustable weight among the controller, link resource and service correlation parameters in different application scenarios.

$$\alpha = \frac{\beta f_{ac}}{\max\{f_{a1}, f_{a2} \cdots f_{ak}\}} + \frac{\gamma f_{bc}}{\max\{f_{b1}, f_{b2} \cdots f_{bk}\}} + \frac{(1-\beta-\gamma) f_{cc}}{\max\{f_{c1}, f_{c2} \cdots f_{ck}\}}. \quad (7)$$

This evaluation factor considers both the global resource utilization and service correlation which can illustrate the optimal slave controller with the highest traffic correlativity in failure recovery scenarios. When a single point of failure occurs on the controller that causes switching nodes to connect to other controllers, the smart contract loaded BlockRec will automatically query the block data and calculate the best slave controller with the

smallest α.

IV. Implementation

In this section, the implementation of the BlockCtrl architecture with BlockRec strategy will be described in detail for normal traffic processing scenario and failure recovery scenarios.

4.1 Implementation of normal traffic process

The BlockCtrl architecture implements fault tolerance processing through the consensus mechanism within blockchain technology. The reason why fault the consensus can complete fault-tolerant processing is that it sets a fault tolerance threshold according to the principle of subordination of the minority to the majority, while the error messages below the threshold will be ignored. Therefore, as long as the number of faulty controllers is below the threshold, the validity of processing results will always be guaranteed, and the faulty controllers will not continue to do evil because it will be cleared immediately after the error message is detected.

Fig. 2 depicts the specific process of fault-tolerant service request processing in the BlockCtrl architecture from the perspective of the controller. The switching node sends an identical service request to each controller. After a controller receives the service request and processes it, the processing result will be sent to other controllers for consensus, while the controller will also receive processing results from other controllers. Since other controllers hold the same blockchain and network view, all controllers will receive an identical result, under normal circumstances and the correct processing result will also be sent to the underlying switching node. Certainly, the switching node will execute the command only after receiving the reply above the fault tolerance threshold, so that the error message below the threshold will not affect the final result.

4.2 Implementation of failure recovery

In the BlockCtrl architecture, the existence of the consensus group can minimize the impact of faulty controllers on the traffic process, but there is still a demand for effective failure recovery function to ensure the Continuous availability of the architecture.

As shown in the figure 3, using the smart contract installed in each controller, BlockRec realizes the rapid handover of the switching node after a single point of failure occurs on a controller, while the failure detection is achieved by consensus mechanisms. Meanwhile, the normal controllers will isolate the faulty controller from the consensus group until it proves that it has returned to function

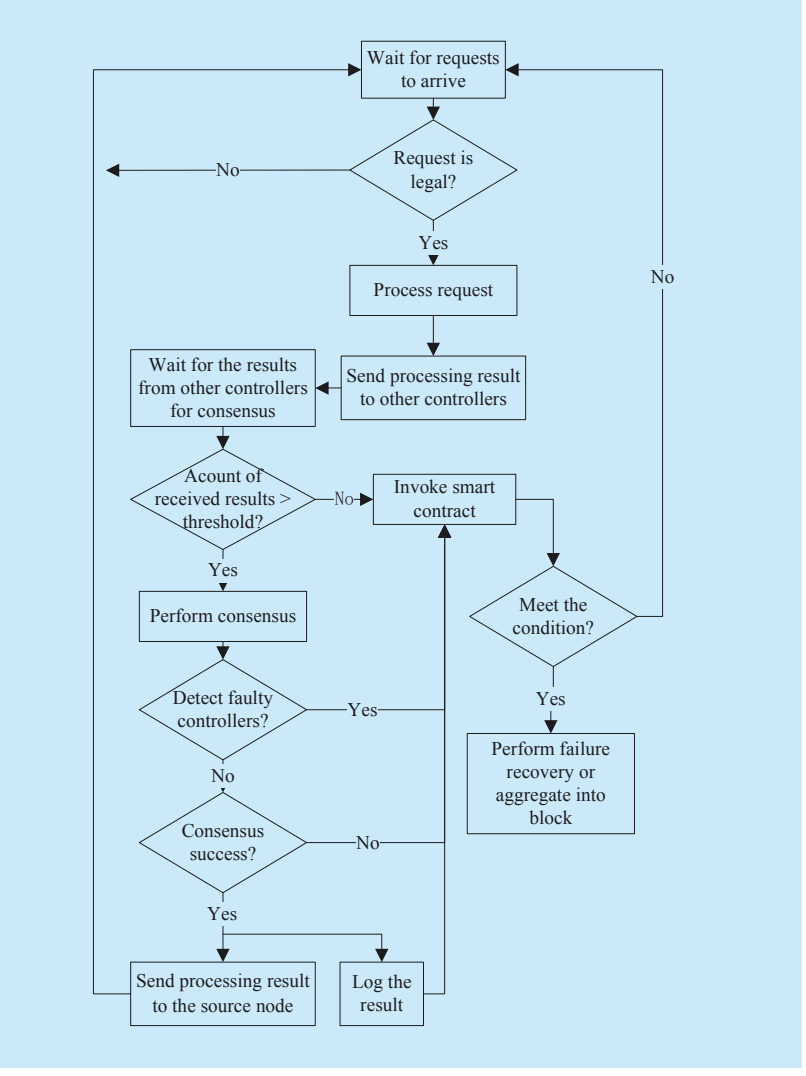

Fig. 2. *Procedure of normal traffic process in BlockCtrl.*

well.

The specific implementation procedure of BlockRec is shown in figure 4. When the switching node receives an unrecognized packet, the Packet-in messages are sent from the switching node to all controllers in the consensus group. After receiving the messages, each controller will process the message according to the network information stored and a consensus on the processing result in the consensus group. At this point, a single point of failure occurs in controller 0, while other controllers detect this fault based on the error messages of controller 0 in multiple consensus interactions. According to the BlockRec strategy compiled in the smart contract, normal controllers automatically execute the contract terms after the fault is detected, which reset the composition of the consensus group and select the optimal slave controller for switching node X. Subsequently, each controller sends a corresponding Role-request message to switching node X according to the execution result of smart contract. Particularly in BlockRec strategy, switching nodes will execute the message after the number of identical received messages reaches the threshold, whether it is Packet-out, Role-request or other messages, so that the switching node X can continuously process service regardless of the failure of controller 0.

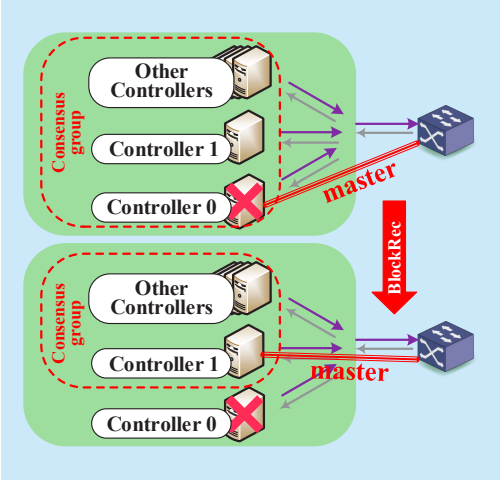

Fig. 3. *Schematic diagram of BlockRec in failure recovery scenarios.*

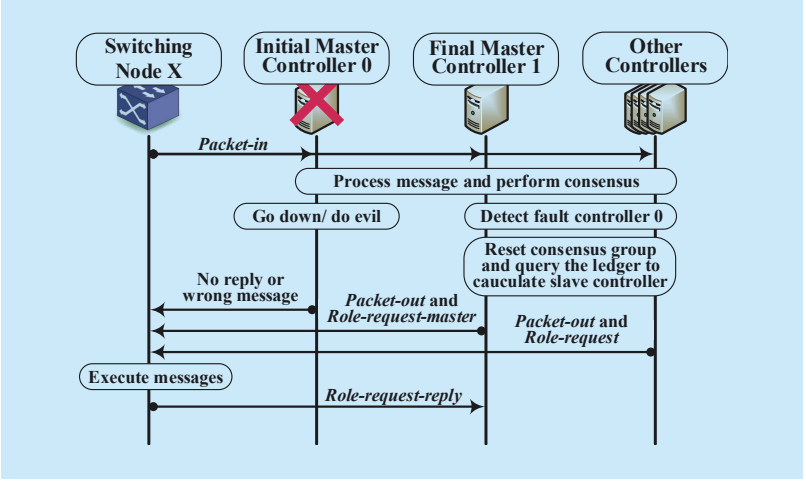

Fig. 4. *Procedure of BlockRec.*

V. Evaluation

In this section, we present results based on BlockCtrl architecture with BlockRec strategy to demonstrate the safety, failure recovery and network performance of BlockCtrl in different network environments. We demonstrate the feasibility of the proposed architecture in three aspects, including defense effects, failure recovery performance, and network performance.

We set up the BlockRec-embed BlockCtrl architecture with SDON based on our testbed, which is a multi-core server with 16 physical 2.90GHz CPU cores and 80GB RAM running Linux 4.15. In our simulation, the consensus group consists of 8 controllers, each of which can be attacked and stop working after the process resources are exhausted. In the data plane, 20 switching nodes are connected to the consensus group and software defined based on the OpenFlow protocol. Services arrive at the network following a Poisson process, which are setup with bandwidth randomly distributed from 10Mbps to 1Gbps. In addition, the functions of the consensus layer and the contract plane are implemented by Hyperledger Fabric 1.2 based on Docker, where 8 controllers are set up to realize modules of consensus, distributed ledgers, and smart contracts.

5.1 Defense effects

In order to prove that the proposed architec-

ture can effectively defend against attacks, we evaluated it at different attack rates and compared it with the existing OpenFlow architecture. In our simulation, multiple DoS attacks of different attack rates will be sent to a controller. When the volume of attacks per second is enough to exhaust the processing resources of the controller, a single point of failure will occur on the controller and invoke the failure recovery strategy. However, the remaining attack messages will point to other controllers, so that it is important to complete failure recovery before the failure expands further.

As shown in figure 5, the processing speed of both BlockCtrl and OpenFlow starts at 8 Mbps without any attack exists. As the attack rate increases, the processing speed of OpenFlow decreases significantly while the speed of BlockCtrl does not change much. BlockCtrl implements fault tolerant processing through a consensus mechanism, which cannot be affected while the number of failures is below the fault tolerant threshold.

5.2 Failure recovery performance

Failure recovery is an important function of distributed control architectures, which can highly degrade the impact of single points of failure on network operations. Accurately, the performance of failure recovery includes the performance of failure detection and failure recovery. From the perspective of latency, we evaluated the performance of these two parts of the proposed architecture and compared it with the BFT [25] and AR2C2 [31].

Fig. 6 and figure 7 show the latencies of the three strategies, respectively under zero traffic load and heavy traffic load. We note that BlockCtrl has lower detection latency at different traffic load. This is because the redundancy-based control architectures mostly use the send-wait heartbeat signals to passively detect failures, which costs a lot of time for waiting. However, the consensus-based control architectures usually proactively detect failure during the consensus process, so the latency will be lower. On the other hand, different from other architectures, BlockCtrl gets the result of the failure recovery based on blockchain ledgers and smart contracts, which also significantly reduces the latency for failure recovery. Figure 8 depicts the recovery latency of BlockCtrl at different traffic load. It can be seen that BlockCtrl has an excellent performance of failure recovery when the traffic load is not high.

5.3 Network performance

Compared to other consensus-based control architectures, BlockCtrl has better network performances because of the global load balancing mechanism introduced in the smart contract. With the smart contract, controllers can select the controller with best network

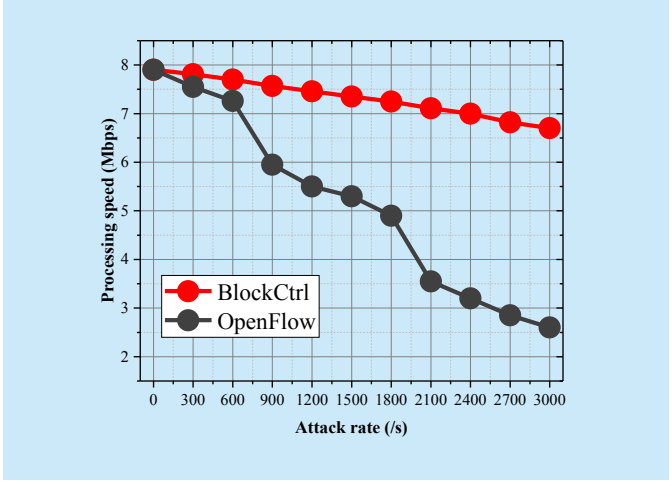

Fig. 5. *Effects on processing speed during different attack rate.*

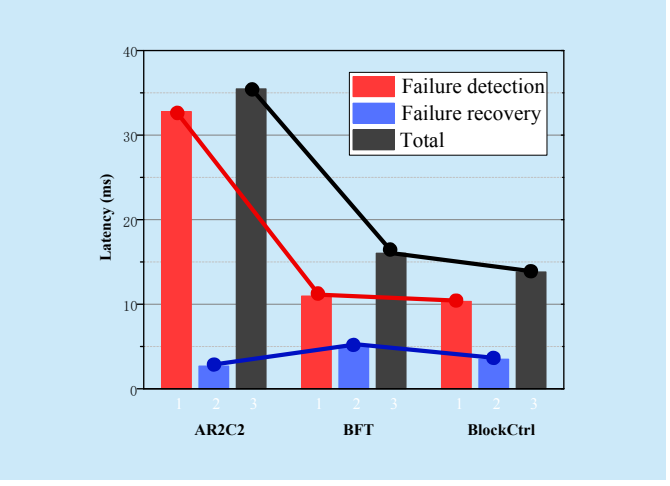

Fig. 6. *Failure detection and recovery latency under zero traffic load.*

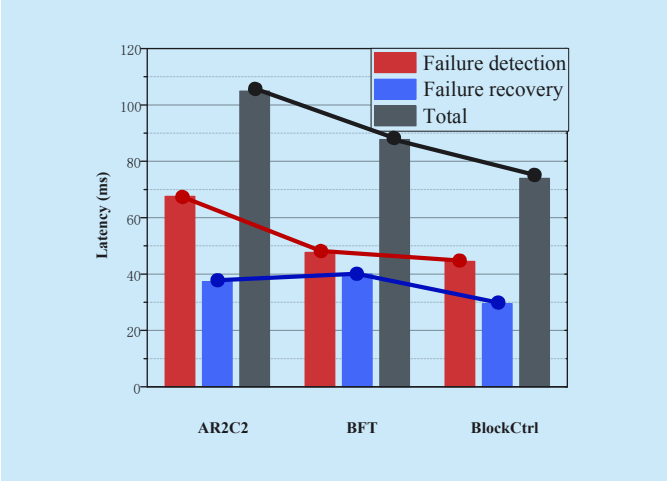

Fig. 7. *Failure detection and recovery latency under traffic load of 750packets/s.*

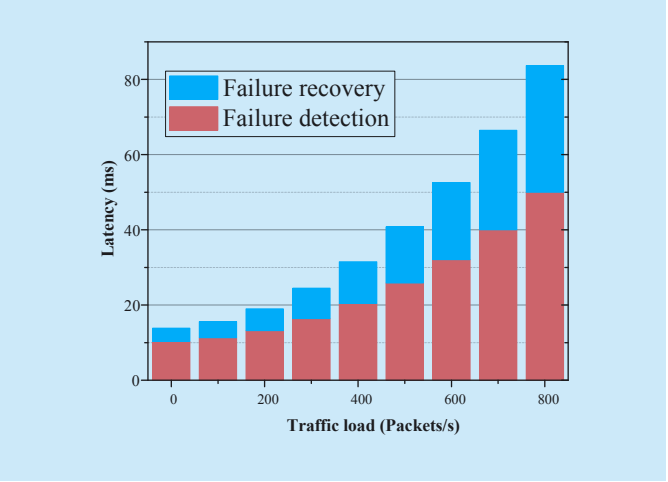

Fig. 8. *Failure detection and recovery of BlockCtrl.*

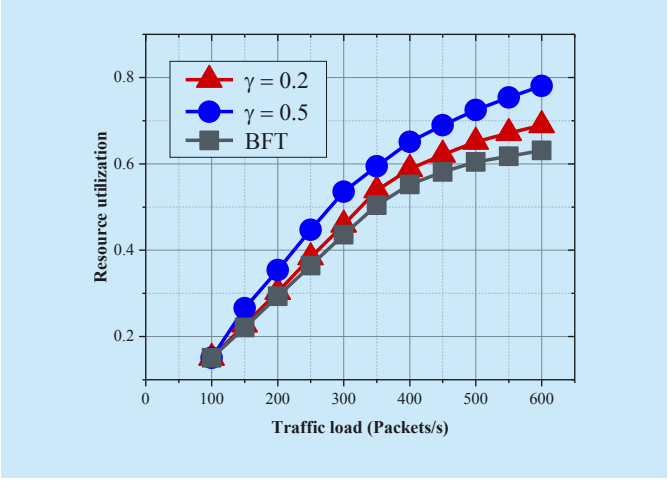

Fig. 9. *Service correlation of BlockCtrl.*

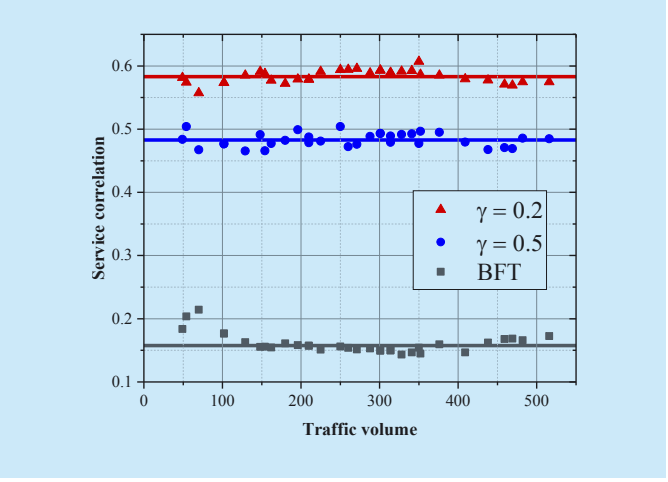

Fig. 10. *Resource utilization of BlockCtrl.*

performances for the switching node based on the dynamic network environment instead of simply connecting it to the default controller.

In our simulation, we set β to 0.3 and compare the BlockCtrl architecture with BFT by variable weights of γ on resource utilization and service correlation. As shown in figure 9, the service correlation of BlockCtrl is better than which of BFT. The reason is that BlockCtrl would consider the latest traffic stored in the blockchain ledger when selecting the newly master controller of switching nodes, which makes it more convenient for the controllers to monitor the traffic of switches and to manage it more efficiently. Figure 10 shows that the BlockCtrl outperforms BFT in resource utilization, especially when the network is heavily loaded. Since it integrates the processing resources of controllers, the bandwidth resources of links as well as the service correlation of switching nodes at a global view to achieve the global load balance to enable the fault tolerant processing and as such can yield higher resource efficiency. Besides, we also notice that BlockCtrl can represent different network performances according to the adjustable weights, which makes it more convenient for network managers to adjust network performance on the basis of different network environments and service requirements.

VI. Conclusion

In this paper, aiming at the single point of failure within SDON control plane, we proposed BlockCtrl architecture, a new distributed secure control architecture, using blockchain technique in SDON to address the security and efficiency issues faced by SDON in recent researches including fault tolerant processing, failure recovery, data consistency and network performance. Based on BlockCtrl, we take the advantages of smart contracts and blockchain ledgers to design a block ledger-based efficient fault recovery strategy, which is applied to rapid recovery in controller failure scenarios. The simulation shows that the secure control architecture we proposed can greatly degrade the impact of DoS attacks on SDON. Besides, based on the automatic smart contracts and global synchronous blockchain ledgers, BlockCtrl can also realize the effective failure recovery with advantages of high network performance, low recovery latency and so on.

Acknowledgment

This work is supported in part by NSFC project (61871056), Young Elite Scientists Sponsorship Program by CAST (2018QNRC001), Fundamental Research Funds for the Central Universities (2018XKJC06) and Open Fund of SKL of IPOC (BUPT) (IPOC2018A001).

References

[1] Y. Qiao, et al., "Software-defined networking (SDN) and distributed denial of service (DDoS) attacks in cloud computing environments: A survey, some research issues, and challenges," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, 2016, pp. 602-622.

[2] L. Cui, et al., "When big data meets software-defined networking: SDN for big data and big data for SDN," *IEEE network*, vol. 30, no. 1, 2016, pp. 58-65.

[3] P. Qin, et al., "A novel stateful PCE-cloud based control architecture of optical networks for cloud services," *China Communications*, vol. 12, no.10, 2015, pp. 117-127.

[4] H. Yang, et al., "C-RoFN: multi-stratum resources optimization for cloud-based radio over optical fiber networks," *IEEE Communications Magazine*, vol. 54, no. 8, 2016, pp. 118-125.

[5] H. Yang, et al., "Multi-dimensional resources allocation based on reconfigurable radio-wavelength selective switch in cloud radio over fiber networks," *Optics Express*, vol. 26, no. 26, 2018, pp. 34719-34733.

[6] W. Fang, et al. "Joint defragmentation of optical spectrum and IT resources in elastic optical datacenter interconnections," *IEEE/OSA Journal of Optical Communications and Networking*, vol. 7, no. 4, 2015, pp. 314-324.

[7] H. Yang, et al., "Experimental demonstration of multi-dimensional resources integration for service provisioning in cloud radio over fiber network," *Scientific Reports*, vol. 6, 30678, 2016.

[8] H. Xu, et al., "Differential game based link resource management for next generation optical network," *China Communications*, vol. 14, no. 9, 2017, pp. 72-79.

[9] H. Yang, et al., "Performance evaluation of multi-stratum resources optimization with network functions virtualization for cloud-based radio over optical fiber networks," *Optics Express*, vol. 24, no. 8, 2016, pp. 8666-8678.

[10] H. Yang, et al., "Resource Assignment based on Dynamic Fuzzy Clustering in Elastic Optical Networks with Multi-core Fibers," *IEEE Transactions on Communications*, vol. X, no. X, 2019, pp. 1-1. DOI: 10.1109/TCOMM.2019.2894711

[11] Y. Zhao, et al., "Multi-core virtual concatenation scheme considering inter-core crosstalk in spatial division multiplexing enabled elastic optical networks," *China Communications*, vol. 14, no. 10, 2017, pp. 108-117.

[12] R. Nejabati, et al., "SDN and NFV convergence a technology enabler for abstracting and virtualising hardware and control of optical networks," *Proc. OFC*, 2015, pp. 1-3.

[13] T. Hu, et al., "Reliable and load balance-aware multi-controller deployment in SDN," *China Communications*, vol. 15, no. 11, 2018, pp. 184-198.

[14] T. Hu, et al., "A distributed decision mechanism for controller load balancing based on switch migration in SDN," *China Communications*, vol. 15, no. 10, 2018, pp. 129-142.

[15] T. Wang, et al., "SGuard: A lightweight SDN safe-guard architecture for DoS attacks," *China Communications*, vol. 14, no. 6, 2017, pp. 113-125.

[16] H. Zhang, et al., "Exploring machine-learning-based control plane intrusion detection techniques in software defined optical networks," *Optical Fiber Technology*, vol. 39, 2017, pp. 37-42.

[17] Open Networking Foundation, ONF, Palo Alto, CA, USA, 2014. [Online]. Available: https://www.opennetworking.org/

[18] F. Bannour, et al., "Distributed SDN control: Survey, taxonomy, and challenges," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 1,

2018, pp. 333-354.
[19] W. Gao, et al., "A Survey of Blockchain: Techniques, Applications, and Challenges," *Proc. ICCCN*, 2018, pp. 1-11.
[20] Z. Ma, et al., "A master-slave blockchain paradigm and application in digital rights management," *China Communications*, vol. 15, no. 8, 2018, pp. 174-188.
[21] S. Huh, et al., "Managing IoT devices using blockchain platform," *Proc. ICACT*, 2017, pp. 464-467.
[22] F. Botelho, et al., "On the design of practical fault-tolerant SDN controllers," *Proc. EWSDN*, 2014, pp. 73–78.
[23] N. Katta, et al., "Ravana: Controller fault-tolerance in software-defined networking," *Proc. ACM SIGCOMM SOSR*, 2015, pp. 1–12.
[24] T. Koponen, et al., "Onix: A distributed control platform for large-scale production networks," *Proc. USENIX OSDI*, 2010, pp. 351–364.
[25] K. ElDefrawy, et al., "Byzantine fault tolerant software-defined networking (SDN) controllers." *Proc. of COMPSAC*, 2016. Pp. 208-213.
[26] P. Sharma, et al., "A software defined fog node based distributed blockchain cloud architecture for IoT," *IEEE Access*, 2018, vol. 6, pp. 115–124.
[27] H. Yang, et al., "Distributed Blockchain-based Trusted Control with Multi-Controller Collaboration for Software Defined Data Center Optical Networks in 5G and Beyond," *Proc. OFC*, 2019, pp. 1-3.
[28] Y. Liang, et al., "Blockchain-based efficient recovery for secure distributed control in software defined optical networks," *Proc. OFC*, 2019, pp. 1-3.
[29] H. Yang, et al., "Blockchain-based Trusted Authentication in Cloud Radio over Fiber Network for 5G," *Proc. ICOCN*, 2017, pp. 1-3.
[30] S. Kou, et al., "Blockchain Mechanism Based on Enhancing Consensus for Trusted Optical Networks," *Proc. ACP*, 2017, pp. 1-3.
[31] E. Spalla, et al., "AR2C2: Actively replicated controllers for SDN resilient control plane," *Proc. NOMS*, 2016, pp. 189-196.

Biographies

Hui Yang, is an associate professor at Beijing University of Posts and Telecommunications (BUPT). His research interests include Blockchain, SDN, 5G, cross-stratum optimization, and so on. He has authored or coauthored 100 papers in prestigious journals and conferences, and is the first author of more than 40 of them. He served as Session Chair of CHINACOM '14, and received the Best Paper Award at NCCA '15. He is an active reviewer or TPC member for several journals and conferences.

Yongshen Liang, was born in Guangdong Province, China, in 1994. He received the B.S. degree (2017) from South China Normal University (SCNU), Guangdong, China. He is a M.S. candidate in Electronics and Communication Engineering at Beijing University of Posts and Telecommunications (BUPT). His research interests include blockchain, software defined optical networking, 5G.

Qiuyan Yao, is currently pursuing her PhD at Beijing University of Posts and Telecommunications (BUPT), Beijing, China. She received her MS degree in computer science and technology from Hebei University of Engineering, Handan, China, in 2015. Her research mainly focuses on the routing and spectrum assignment strategy in elastic optical networks and software-defined optical networks. She is a member of SPIE.

Shaoyong Guo, born in 1985, PhD. Lecturer of State Key Laboratory of Networking and Switch Technology, Beijing University of Posts and Telecommunications, China. His current research interests include smart grid, network management and terminal management.

Ao Yu, was born in Shandong Province, China, in 1992. He was received a B.S. degree from China University of Petroleum (UPC), Shandong, China, in 2014. He is a Ph.D. candidate in information and communication engineering at Beijing University of Posts and Telecommunications (BUPT). His main research interests include resource allocation, data center optical networks, and deep learning.

Jie Zhang, is a professor and dean of the Institute of Information Photonics and Optical Communications at BUPT. He is sponsored by more than 10 projects of the Chinese government. He has published eight books and more than 100 articles. Seven patents have also been granted. He served as a TPC member for ACP 2009, PS 2009, ONDM 2010 and so on. His research focuses on optical transport networks, packet transport networks and so on.