Blockchain-Based SDN Security Guaranteeing Algorithm and Analysis Model

Zhedan Shao, Xiaorong Zhu(✉), Alexander M. M. Chikuvanyanga, and Hongbo Zhu

Nanjing University of Posts and Telecommunications, Nanjing 210003, China
xrzhu@njupt.edu.cn

Abstract. Although Software Defined Networking (SDN) has a lot of advantages, it also leads to some security issues such as DDoS/DoS attacks, unauthorized access, and single point of failure. To improve the security and efficiency of the SDN control plane, we propose a novel consensus algorithm–Simplified Practical Byzantine Fault Tolerance (SPBFT) to transfer messages between controllers and then establish an analysis model to analyze the security and performance of SPBFT based on game theory. In this paper, we apply blockchain technology in SDN to build a readable, addable, and unmodifiable distributed database which maintains a list of updated system activities and time stamps in each controller. The simplified three-step consensus algorithm SPBFT makes the message transfer and verification carry out efficiently in parallel. In addition, we use recovery mechanism and credibility assessment on the primary controller to increase the invulnerability of system. Simulation results show that compared with the PBFT algorithm, the proposed algorithm can significantly improve system performances in terms of security and efficiency.

Keywords: SDN · Security · Blockchain · Consensus algorithm

1 Introduction

Software Defined Networking (SDN) is a new technology that separates control planes from the data plane to support network virtualization. Compared with traditional networks, SDN has many advantages such as the separation of control and data planes, programmability, dynamic flow control, and centralized control management. However, it poses some threats in security such as DDoS/DoS attacks, illegal access, and single point of failure which can limit the large-scale deployment and application of SDN in many scenarios. The SDN control plane serves as the center of centralized control. For the attacker, attacking control plane is the most effective method to destroy the network. Therefore, the key measure lies in the effective security protection mechanism for the control level.

Foundation Items: National Natural Science Foundation of China (61871237), National Science and Technology Major Project (2017ZX03001008), and Natural Science Foundation of the Higher Education Institutions of Jiangsu Province (16KJA510005).

M. Jia et al. (Eds.): WiSATS 2019, LNICST 281, pp. 348–362, 2019.
https://doi.org/10.1007/978-3-030-19156-6_32

Blockchain refers to the technology of collectively maintaining a reliable database through decentralization and de-trust. It is actually a series of data blocks generated by using cryptography correlation. Each data block contains information for valid confirmation of multiple transactions. This storage method makes it difficult to tamper with information once it has been recorded.

The characteristics of the blockchain, such as de-trust, openness, and non-destructive information, coincide with the requirements of the distributed SDN architecture. Moreover, smart contracts of blockchain can be better configured under the programmability of SDN.

Regarding to the current situation of SDN security, the advantages and disadvantages of the SDN is analyzed from the perspective of security and possible attack behaviors is illustrated from three levels, including DDoS and illegal intrusion attacks for the control plane in [1]. These two points should be contained in the consideration of the security mechanism design in this paper. The authors of [2] summarize and analyze the security problems caused by the programmable data plane and point out possible loopholes in the future data plane. Security architecture platforms are proposed based on SDN [3, 4], however the more granular security scheme is not mentioned. Some authors are considering the methods to deal with specific DoS attacks in SDN which are useful for SDN security but lack of integrity [5–7]. Researches on the security of the control plane for the core of the SDN architecture is still relatively one-sided, and many are based on the scenario of a single controller or a specific attack method.

With respect to the application of blockchains in security, the authors of [8] establish a blockchain-based electronic medical record system that applies blockchains to data structures and source layers. The system monitors the data request actions from all requesters through smart contracts to prevent the medical data of patients from being stolen, making medical records traceable and checkable. The authors of [9] build a user-centric content distribution system with blockchain to implement a decentralized proxy mechanism which enables Content Providers (CPs) and Technology Enablers (TEs) to compete and negotiate with each other to instantiate the best content distribution session, but specific negotiation mechanism is not proposed. The authors of [10] establish a smart grid water network architecture that uses the blockchain as a link to store data to ensure the verification of information and the correctness of data. The use of basic cryptographic primitives makes the system more secure and user-friendly. It can input user data to complete multi-party security protocols and prevent privacy leakage and data abuse. [11] proposes a two-step protocol that securely shares health data to each node in the network in a pervasive social network (PSN). For access control and node verification, the author authorizes a security component as an access control provider and believes that blockchain technology can further enhance the security of this system. The authors of [12] argue that blockchains need to adapt to their own characteristics according to different systems. It is proposed that the direction in the future is to overcome blockchain restrictions such as most of the blockchains can only complete 7 transactions per second.

The literatures show that blockchain has been succeeded in security aspect [8–12]. The blockchain creates a good structural foundation for the architecture protection with its decentralized, smart and convenient use, information confidentiality, integrity and traceability. However, at present, there are still few literatures on the use of blockchain in SDN. [13] proposes a new type of blockchain-based distributed cloud architecture, including device layer, fog layer, and cloud layer, so that controller fog nodes at the edge of the network can meet the required design principles. However, the internal structure of controller proposed above still adopts the traditional SDN controller structure without adding the blockchain feature and putting forward a specific security mechanism.

Therefore, we propose an SDN security mechanism based on blockchain with distributed SDN architecture. The controllers are intelligently distributed in different geographical locations to alleviate "single-point failure" so to avoid serious widespread network crashes. We use blockchain technology to build a blockchain database together on SDN controllers. The block records the real-time system activities and the time stamps, thus making the entities activities traceable. A simplified consensus algorithm named Simplified Practical Byzantine Fault Tolerance (SPBFT) is proposed to verify the messages transmitted between blocks which contains the communication time restriction, the recovery mechanism and the credibility assessment. In addition, we establish an analysis model to evaluate the security and performance of consensus algorithms and make simulation on the proposed consensus algorithms.

2 Blockchain-Based SDN Security System Model and Guaranteeing Mechanism

This paper proposes a security SDN system model and guaranteeing mechanism based on blockchain. The system model adds block to each controller to store security verification information in the control plane and deals with different security issue with different contracts. Communication guaranteeing algorithm between controllers SPBFT and emergency strategy are proposed based on consensus algorithm and smart contract respectively, according to two kinds of attack, the illegal invasion and DDoS attacks.

2.1 Blockchain-Based SDN Security System Model

To guarantee the security of the SDN system model, we mainly take the following four aspects into consideration: system reliability, scalability, confidentiality of information, and non-repudiation. The system model is shown in Fig. 1.

We adopt a horizontal distributed controller architecture to divide the network into different independent areas. Each area is controlled by a separate controller. Different controllers construct the entire network view through a consensus algorithm to prevent single point of failure and improve the system reliability.

The network between controllers is equivalent to a P2P network. Each controller exists equally. In this case, System can add new controllers with some verification of other legal controllers. From the perspective of global security and load balancing, it is necessary to dynamically update the global topology and link state information, such as the switch port status, link utilization, and host CPU usage, to form a global network view. Real-time updated data is recorded in blockchains, and global views can be created in each controller.

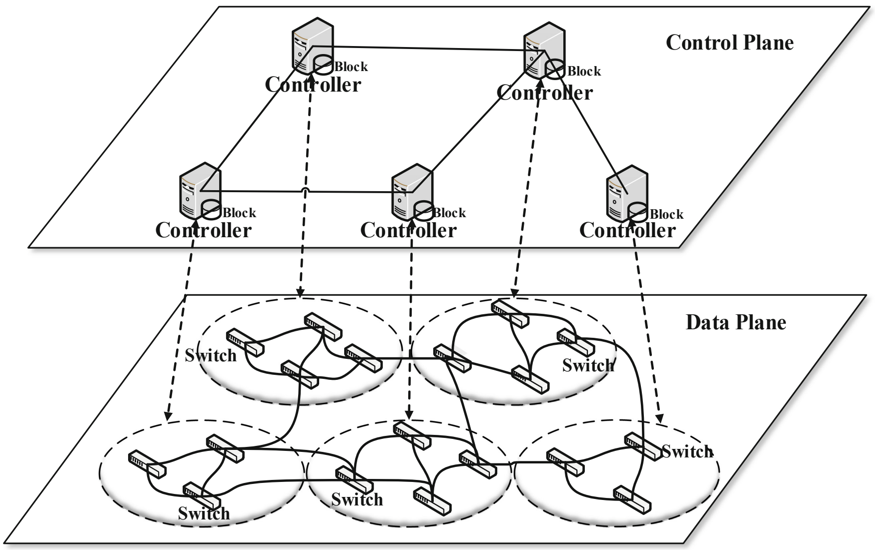

Fig. 1. Blockchain-based SDN security system model.

For a controller, the function can be divided into two phases, prevention phase and processing phase. In the prevention phase, the security mechanism of the SDN control plane mainly includes two aspects: preventing the access of unauthorized controllers and preventing attackers from falsifying shared database information. The main approach is to authenticate by exchanging data between SDN controllers. The controller has its own verification information and stores the verification information of other controllers. Through the encrypted data transmission, information verification and obtaining consensus to anti-falsification forgery. Uniform data distributed storage between controllers makes it difficult to tamper with the information of a single controller. During the processing phase, the controller detects abnormal traffic and determines whether there is an attack. Through the detection of the analysis center, the mitigation of smart contracts can automatically react to attacks.

2.2 Blockchain-Based SDN Control Plane Work-Flow

Controller Join Process: Based on the centralized control of SDN, the control plane needs to ensure the security of the controller node and the confidentiality of the information transmission. The generation of the controller does not depend on the "mining" model in the public chain, but up to the administrator. Therefore, adding the new controllers can be verified by the administrator which is the first protection of the control plane.

At the very beginning, the administrator randomly determines an initial controller A to receive connection requests from other controllers. The initial controller A has a private key and a public key. The controller B who wants to join the controller layer sends a request to A so as to obtain A's public key. Controller B uses A's public key to encrypt its own information and send it back to A. In the case of a successful verification, B will be added to the trust list. The next controller can request A or B's authentication to gain access. In this way, the entire controller network is connected.

In terms of security, public key encryption, digital signatures and hashing are used to guarantee message confidentiality. Controllers complete the encrypted communication through a pair of private/public keys. They are bound to the controller ID and accurately located to the corresponding hardware device so that the controller information can be traced from point to point.

Controller Normal Work-Flow: In the first phase, the controller verifies, records, and resolves messages sent from switches. As shown in Fig. 2, switches send messages such as Features_Reply, Stats_Reply and Packet_In to the controller, where Features_Reply is a one-time report to the features of switch itself. Stats_Reply can be used by the Analysis Center to analyze the status of the local network. Flow_Mod and Packet_In, as a pair of corresponding messages, are the signaling for requesting and delivering the flow table. The above three kinds of messages reflect the different aspect of the switches under the SDN controllers, so they are worth recording.

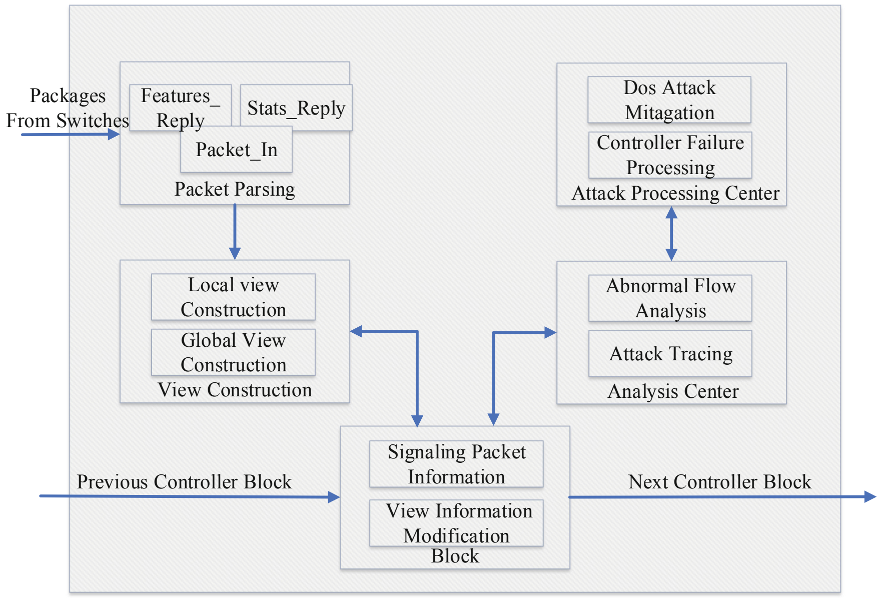

Fig. 2. Blockchain-based SDN controller internal structure.

In the second phase, the topology state of the route and the state set of the switch are extracted to construct a topology map of the network flow which dynamically display the local traffic. At the same time, the above four kinds of information are stored in the local block, so as to trace back to the source when problem occurs.

In the third phase, each data is uploaded to the block to form a global view. This process can be divided into periodicity and triggered modes. When the network runs smoothly, each controller independently manages its own domain. In this case, the global network topology is periodically updated. However, when a controller issues an alert message, the global view is immediately updated so that each controller can grasp the overall situation and provide feedback on the alert controller.

Requests for uploading global views need to be verified by other controllers. Thus, each operation for the management of the global view of the controller and the modification of the global view will be recorded in the block. For controllers with successful recording, the percentage of trust value is a larger percentage of each account.

In the fourth phase, the information and state data of each controller and switch recorded in the block will be analyzed to determine the type of attack and use intelligent contracts to automatically make response strategies.

2.3 Consensus Algorithm Based Security Guaranteeing Algorithm Among SDN Controllers

This paper proposes a Simplified Practical Byzantine Fault Tolerance consensus algorithm based on Practical Byzantine Fault Tolerance (PBFT) algorithm.

The SPBFT algorithm proposed in this paper has made up for the shortcomings of PBFT that low consensus efficiency, poor self-healing, and invulnerability. It simplifies the PBFT consensus process and makes the message transfer and verification carry out in parallel. It also grades requests to ensure that urgent requests can be handled quickly and efficiently. A credibility assessment that the controller with the highest degree of trust value is used as the primary controller to prevent invasion has been added. The malicious controller can be found with the guide of primary controller and measures will be taken to restore its availability. The changes above make SPBFT more suitable for the application of SDN.

The workflow of the SPBFT algorithm is as follows:

(a) A certain controller (named A) that wants to broadcast a request to other controllers.
(b) The other controllers verify the source of the request.
(c) If the others verify that controller A is valid, the execution of the request continues, and a reply is sent to controller A. Otherwise the system stops executing the request and enters the defense state. The controller with highest degree of trust value becomes primary and dominates the normal controller against the effects of the bad controller. After the defense phase the trust value is cleared to restart the next view.
(d) Controller A waits for $f + 1$ replies from different controllers (replies with the same result) and the reply is the result of the request operation.
(e) Operation results are recorded in the controller's block to calculate the trust value and weight of all controllers.

SPBFT uses a three-phase protocol to broadcast requests to other controllers, Prepare, Commit, Reply, as shown in Fig. 3. The red lines express the processes of

SPBFT and the black lines express that of PBFT. PBFT was originally applied to a distributed system mainly consisting of state machines. It needs to perform the same serialization processing on distributed nodes, which are not exhaustive in SDN. In PBFT, a five-phase protocol, there are two consensuses in Prepare and Commit phase to guarantee the same decision and operation in each node. However, In SPBFT, we need only one consensus to ensure the identity and decision at one time. Because there are many types of requests in SDN. Request processing with different priorities can be queued, so there is no need to sort the requests in a fixed manner in the SDN. The requests can be automatically executed according to request priority and the same-level request first-in-first-out principle.

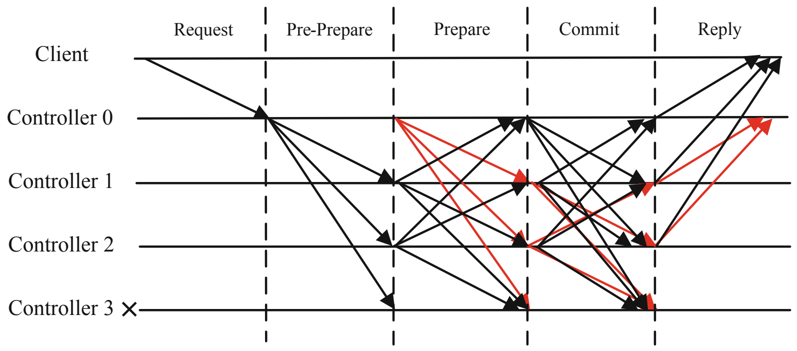

Fig. 3. SPBFT and PBFT normal working phases (Color figure online)

In SPBFT, the Prepare phase is mainly to confirm the validity of the request and the controller. In the subsequent Commit phase, the controllers reach an agreement and execute the request. In the Reply phase is mainly to inform the request controller of the completion of the request.

In the Prepare phase, we provide authentication mainly in the form of encryption and Q&A.

Controller A broadcasts a prepare message to other controllers, this information < $(prepare, v, n, h, a, d)_{en}$, m> contains the view v, request number n, controller A's own information h, request content m, the question answer a, and the content summary d. Wherein, v represents the view where the controller is located. n is the sequence number down ordered by the requesting controller A according to its own stored request. h is the information used by the controller to prove itself, and a represents that the answer to the problem set by the primary controller. d is a summary of the content m. The subscript en indicates the encryption process. The prepare message is sent to each controller and this information is inserted into the block.

When the controller i receives the prepare message from controller A, it determines whether the information matches. If it matches, the controller receives the prepare message and enters the following commit phase. Messages <$(commit, v, n, d, i)_{en}$> are broadcast to other controllers other than A, and prepare and commit messages are appended to the log.

When a controller has previously received a prepare message that is in the same view and number is also n, but the digest is different, the controller will reject the message and report the situation to the primary controller and record the exception in the blockchain.

In the Commit phase, the controller executes the request with the principle of the minority when the malicious controller is limited.

The controller collects the commit messages to determine whether the controller is prepared by checking the view, serial number *n*, and request summary. When there are *f* commits from different controllers that is prepared, it means that $<(commit, v, n, d, i)_{en}>$ is true, then the status of the request on this controller is considered to be executable. As a result, the operation is performed according to the request.

Once each controller receives different result of the request, it reports to the primary controller to check. If the controller has not received a certain controller information within a few request periods, it may be that the link between the two controllers is interrupted or there is a bad controller. At this point, the situation message is also sent to the primary controller for verification.

Thus, we also enhance system security by limiting communication time, adding recovery mechanisms, and assessing trustworthiness.

Limited Communication Time: In order to guarantee the security of information transmission and tamper-proof, it is necessary to ensure that the transmission time of each message is within a certain range. Therefore, we set a maximum delay Δ. If the transmission delay of the message is less than Δ, the message is considered to be valid. Otherwise if the transmission delay of the message is greater than Δ, the message may be intercepted and tampered or the communication failure occurred. In this case, the transmission has failed and the message will be ignored. The maximum delay Δ can be expressed by the maximum arrival delay counted on previous view.

Recovery Mechanism: After each synchronization, if the primary controller receives more than 1/3 of the replica controllers reporting a controller out of response time, the primary controller broadcasts a message to other replica controllers to contact with the controller. If the replica controller finds the controller working normally, it receives a short message $<(OK, v, n)_{en}>$ to confirm the status of the controller. If not, the reply message is sent to the primary controller and the unconnected controller is suspected to have been compromised. To avoid further damage, the primary controller generates a new request (isolation command) that requires the switch under the problematic controller to be assigned to other replica controllers. When $f + 1$ replica controllers agree to this request, the identity certificate is used to inform the switch to execute the instructions so that the problematic controller is effectively isolated. The Attack Processing Center performs subsequent checks and repairs on the problematic controller.

Credibility Assessment: We uses the service as a measure of the credibility of the evaluation criteria.

In our SDN network, services are used as the "transactions" stored in the blockchain. We define the credibility in SDN from two aspects, utility and time. For the unity, it can be considered from two parts, control plain and data plain. In control plain, each service should be verified before being stored. The number of valid services also means the number of controller that being verified successfully. In data plain, the more switches the controller manage, the more important the controller is. For the time aspect, the normal working hours and rounds to a certain extent reflect invulnerability of controllers. Therefore, we define *Trust* as the trust value of each controller calculated by following parameters. S, W, M, R represent the proportion of valid services, the number of managed switches, the duration and the rounds of normal work, respectively.

$$Trust = \alpha S + \beta W + \chi M + \delta R (0 < \alpha < 1, 0 < \beta < 1, 0 < \chi < 1, 0 < \delta < 1) \tag{1}$$

$$\alpha + \beta + \chi + \delta = 1 \tag{2}$$

In the above formula, α, β, χ, δ express the weight parameters for the above four indicators.

Adding an assessment standard to the blockchain to determine the master-slave relationship of the controller participating in the contract. In this way, the system can solve the security risks as soon as possible.

2.4 Smart Contract Based DDoS/DoS Attack Protection Mechanism

This paper uses the Analysis Center and Attack Processing Center in the SDN security system model to collaboratively complete the implementation of protection contracts for DDoS/DoS attacks.

The threat of the control plane is mainly caused by DDoS/DoS attacks from the data plane. Attackers send large amounts of meaningless packets to SDN switches, so that the switch needs to request the flow table frequently from the controller. There is another way that the attacker hijacks or forges a switch to allow malicious traffic to flow to the other controller. Excessive flow table requests and controller resource usage can be considered that there may be DDoS/DoS attacks.

Attack Detection and Mitigation: For DDoS/DoS attacks, smart contracts periodically check the status of the controller with traversing the states, transactions, and trigger conditions contained in each contract one by one. Then the conditional transaction is pushed to the queue to be verified and wait for consensus. Transactions that do not satisfy the trigger condition will continue to be stored on the blockchain. Once the Analysis Center detects abnormal traffic, it triggers the generation of a DDoS/DoS Alert Contract and broadcasts it. First step is to verify the security of each controller and ensure its validity. The verified controller will enter the consensus set. After most of the verification nodes reach a consensus, they select the most reliable primary controller according to the trust mechanism of the consensus algorithm. The primary controller generates the DDoS/DoS Mitigation Contract and distributes it to other controllers to alleviate the alarm controller's DDoS/DoS attack. For the sake of fairness, the primary controller cannot be re-elected in two consecutive views.

Follow-Up Protection: The smart contract implements a confidence system based on a consensus algorithm as a guarantee to ensure the correct execution of the incident resolution. After the alarm is eliminated, the alarming controller needs to perform a series of verification operations to eliminate the security risks at the control level. At the same time, each controller sends status information to the block to analyze the current status of the alarming controller, the mitigation effect, the impact on other controllers and the origin of this attack. Each attack interval event will also be recorded in the block for systematic analysis.

3 Security and Performance Analysis Model on SPBFT Algorithm

In this paper, we use game theory to analyze the SPBFT algorithm.

Definition: The game is defined as $\Gamma = \{I, S, U\}$. where $I = \{a, d\}$ is the participant, a denotes the attacker and d denotes the defender, i.e. the controller. $S = \{t_a, t_d\}$ defines the strategy set of attacker and controller. $U = \{u_a, u_d\}$ is the utility function of attacker and the controller.

Assuming that there are n controllers and the attacker attacks r controllers at the same time with the attack speed t_a which means the duration that attacker breaks through the firewall of the controller itself and tampers with the controller information. $N(t_a)$ means number of the requests that have been sent in the period t_a. The total time is T. The verification is initiated in the process of request processing, so request initiation interval equals verification interval. We assume that the request initiation time interval t_d obeys the exponential distribution with the parameter λ. The number of controllers recovered in each attack period is Y.

In the next attack time ta, the controller will send a request with a probability of

$$P(N(t_a) = 1) = \lambda t_a e^{-\lambda t_a} \tag{3}$$

In the next attack time ta, more than one request will be sent with a probability of

$$P(N(t_a) \geq 1) = 1 - P(N(t_a) = 0) = 1 - e^{-\lambda t_a} \tag{4}$$

In a certain attack time ta, the probability of a successful attack for attackers is

$$X = 1 - P(N(t_a) \geq 1) = e^{-\lambda t_a} \tag{5}$$

Therefore, during this time, there are $X \times r$ controllers that will be intruded.

During attack period i, we assume there are $M(i)$ controllers that will be intruded and $M(i)$ can be expressed by

$$M(i) = \sum_{k \in [1,i]} (X \times r) - \sum_{k \in [1,i-1]} Y \tag{6}$$

Therefore, the probability of a successful attack to a certain controller can be obtained by

$$PT = \frac{M(i)}{n} \tag{7}$$

For a certain attacker, we define the cost includes intrusion cost C_{a1} that is the cost of intruding controllers and the penalty cost C_{a2} that is the cost of the attack being detected. If the attack succeeds, it will get the successful intrusion gain B_a.

For a certain defender, we define the cost C_d that is the cost of the controller failing to resist intrusion. If the controller operates normally, it will get the daily gain B_{d1}. In the case that controller successfully discovered the attack, it will get the profit B_{d2}.

Therefore, the attacker's utility function can be expressed as

$$u_a = B_a \times PT - C_{a1} - C_{a2} \times (1 - PT) \tag{8}$$

And the defender's utility function is

$$u_d = B_{d1} \times (1 - PT) + B_{d2} \times \frac{Y}{r} - C_d \times PT \tag{9}$$

Table 1 shows the comparison of different consensus algorithms. Compared with PBFT, the proposed SPBFT has advantages over error node tolerance. It contributes to that even if at most one-third of nodes is invaded, only if the primary node has not been invaded, the system may still keep in normal state. In addition, we also see that SPBFT has good trading performances. PBFT cannot meet the demand for high message volume of SDN network. However, SPBFT can simplify the process of verification and consensus and reduce the number of signaling.

Table 1. Consensus parameter comparison.

Consensus algorithm	PBFT	SPBFT
Number of nodes	3f + 1	3f + 1
Error node tolerance	At most 1/3 error nodes, but also depends on the primary controller	At most 1/3 error nodes
Consensus efficiency	General	High
Invulnerability	General	High

In the aspect of invulnerability, SPBFT uses the trusted data in the blockchain for analysis to find the invaded controller and has the mechanism to handle with the suspected controller, while PBFT does not. In that case, SPBFT can obviously slow down the overall speed of the system being compromised. When attacked, the system can respond quickly and stop it in time. Each block records various activities of the system and records time stamps, making each entity's implement self-inspection and self-recovery under the certain attacks. After mitigation of attacks, activity analysis and source track of the entities that generate the activity are carried out.

Furthermore, in terms of architecture, distributed decentralized security architecture can avoid large-scale network anomalies which are brought by "single point of failure". Also, other normal controllers assist problem controllers to get out of trouble. Under this architecture, we use identity and traffic as the focus of outlier monitoring to ensure the normal operation of the controller. The programmability of SDN makes this architecture easy to extend.

In the aspect of confidentiality and integrity of information, each controller manages its own private key and distributes the public key using blockchain technology. Blocks store encrypted segments of controller information without any third party access and control so that privacy protection can easily be achieved. The blockchain consists of a readable, addable, and non-removable distributed database, maintaining the block's record list. With the containing of time stamp and links to the previous block, the blockchain provides a security guarantee that, once recorded, the data cannot be modified.

4 Simulation Results and Analysis

In this paper, PBFT is used as the baseline of the algorithm to compare with SPBFT proposed in this paper with Matlab2015b. The request arriving intervals and the controller processing request intervals are both assumed to be with exponential distributions. We use the shortest path algorithm to transmit the signaling. Simulation parameters are shown in Table 2.

Table 2. Simulation parameters.

Parameters	Distribution	Mean	Var
Request arriving interval (ms)	Exponential	1.1	1.23
Controller processing interval (ms)	Exponential	1.1	1.23
Link delay (ms)	Uniform	1	0.34
Number of periods	N/A	60	N/A

We compare consensus time of normal controller with the probability of 1 and 2/3. Figure 4 shows the simulation results. The abscissa is the number of controllers and the ordinate is the mean consensus time. From the simulation results, the consensus time of the two algorithms is not much different when the number of nodes is less than 10. The consensus time of the SPBFT algorithm saves about 1/3 of the PBFT when the number of nodes is higher than 10.

Figure 5 shows the comparison of signaling overhead between two algorithms. The abscissa is the number of controllers and the ordinate is the average signaling overhead. Signaling overhead increases as the number of controller nodes increases. The signaling overhead of the SPBFT algorithm is approximately 50% lower than that of the PBFT algorithm.

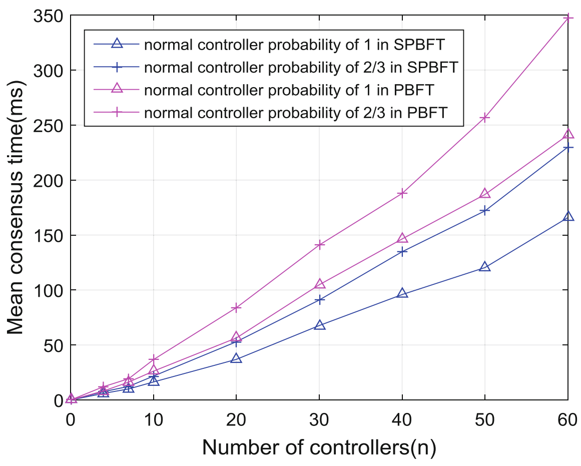

Fig. 4. Mean consensus time comparison.

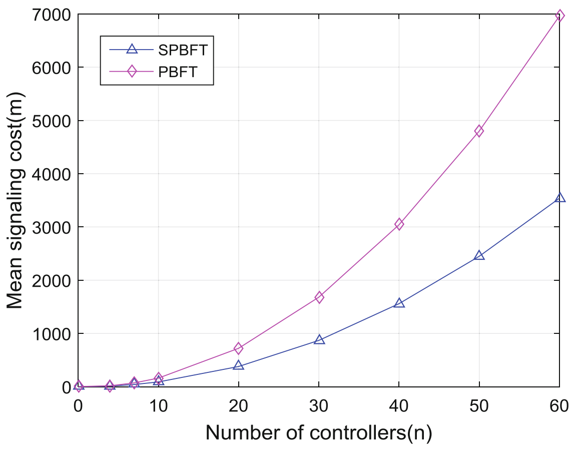

Fig. 5. Mean signaling overhead comparison.

This paper analyzes the complexity of the algorithm using the running time of the two algorithms in the program. A total of 20 runs were performed and averaged to obtain the algorithm complexity simulation results shown in Fig. 6. The abscissa is the number of controllers and the ordinate is the complexity. As shown, the complexity of the algorithm increases with the number of controllers. The complexity of the SPBFT algorithm is lower than that of the PBFT algorithm. Also, as the number of nodes increases, the complexity of the SPBFT algorithm rises more slowly than that of the PBFT algorithm.

In Fig. 7, we compare the invulnerability with and without using SPBFT algorithm under the same attack strength based on the analysis model proposed in this paper. The abscissa is the proportion of controllers being attacked, and the ordinate is the number

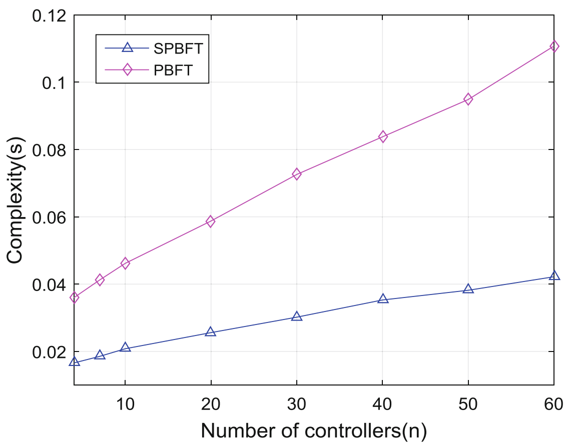

Fig. 6. Time complexity comparison

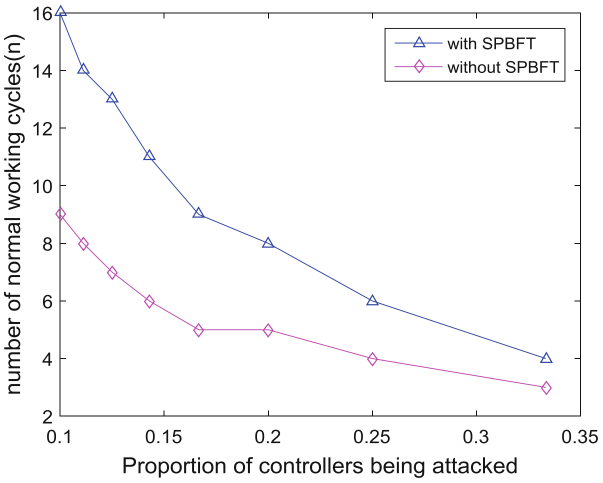

Fig. 7. Invulnerability comparison

of normal working cycles. As the proportion of attackers attacking the controller increases, the normal working cycle of the controller is shortened. The recovery strategy in this algorithm makes the control plane run normally for a longer time under the same intensity of attack.

Compared with the PBFT algorithm, the SPBFT algorithm reduces the one-time broadcast consensus, so the mean consensus time, signaling overhead, and time complexity are reduced, which is more in line with the SDN requirements of fast response. In addition, SPBFT uses the primary controller as the subject of the recovery measures implementation. Thus, under the same intensity of attack, the system can persist for longer with SPBFT algorithm.

5 Conclusion

In this paper, we propose a blockchain-based SDN security system model to improve the security and consensus efficiency of the SDN control plane. In order to avoid illegal access and DDoS/DoS attack, an efficient consensus algorithm SPBFT and smart contracts are proposed to transfer messages between controllers and execute requests. We analyze the system security and performance through a model based on game theory. Simulation results show that the SPBFT algorithm proposed in this paper can significantly reduce the consensus time, signaling overhead, time complexity and invulnerability, compared with the PBFT algorithm.

Our future work is to improve the consensus efficiency among SDN controllers under the blockchain-based system model, and to respond faster under attacks to ensure the security of the SDN control plane.

References

1. Dabbagh, M., Hamdaoui, B., Guizani, M., Rayes, A.: Software-defined networking security: pros and cons. IEEE Commun. Mag. **53**(6), 73–79 (2015)
2. Dargahi, T., Caponi, A., Ambrosin, M., Bianchi, G., Conti, M.: A Survey on the security of stateful SDN data planes. IEEE Commun. Surv. Tutorials **19**(3), 1701–1725 (2017)
3. Liang, X.D., Qiu, X.F.: A software defined security architecture for SDN-based 5G network. In: 2016 IEEE International Conference on Network Infrastructure and Digital Content (IC-NIDC), pp. 17–21 (2016)
4. Liyanage, M., Ahmed, I., Ylianttila, M., et al.: Security for future software defined mobile networks. In: 2015 9th International Conference on Next Generation Mobile Applications, Services and Technologies, pp. 256–264 (2015)
5. Zhao, Z., Liu, F.L., Gong, D.F.: An SDN based hopping multicast communication against DoS attack. KSII Trans. Internet Inf. Syst. **11**(4), 2196–2218 (2017)
6. Dridi, L., Zhani, M.F.: SDN-guard: DoS attacks mitigation in SDN networks. In: 2016 5th IEEE International Conference on Cloud Networking, pp. 212–217 (2016)
7. Macedo, R., Castro, R.D., Santos, A., Ghamri-Doudane, Y., Nogueira, M.: Self-organized SDN controller cluster conformations against DDoS attacks effects. In: 2016 IEEE Global Communications Conference (GLOBECOM), pp. 1–6 (2016)
8. Xia, Q., Sifah, E.B., Asamoah, K.O.: MeDShare: trust-less medical data sharing among cloud service providers via blockchain. IEEE Access **5**, 14757–14767 (2017)
9. Herbaut, N., Negru, N.: A model for collaborative blockchain-based video delivery relying on advanced network services chains. IEEE Commun. Mag. **55**(9), 70–76 (2017)
10. Rottondi, C., Verticale, G.: A privacy-friendly gaming framework in smart electricity and water grids. IEEE Access **5**, 14221–14233 (2017)
11. Zhang, J., Xue, N., Huang, X.: A secure system for pervasive social network-based healthcare. IEEE Access **4**(1), 9239–9250 (2016)
12. Anjum, M., Sporny, A.: Sill: blockchain standards for compliance and trust. IEEE Cloud Comput. **4**(4), 84–89 (2017)
13. Sharma, P.K., Chen, M.Y., Park, J.H.: A software defined fog node based distributed blockchain cloud architecture for IoT. IEEE Access **6**, 115–124 (2017)