Computers and Electrical Engineering 66 (2018) 388–406

Contents lists available at ScienceDirect

Computers and Electrical Engineering

journal homepage: www.elsevier.com/locate/compeleceng

Distributed controllers multi-granularity security communication mechanism for software-defined networking☆

Fengjun Shang*, Yan Li, Qiang Fu, Wenkai Wang, Jiangfan Feng, Li He

College of Computer Science and Technology, Chongqing University of Posts and Telecommunications, Chongqing, 400065, China

ARTICLE INFO

Article history:
Received 30 December 2016
Revised 4 July 2017
Accepted 4 July 2017
Available online 8 July 2017

Keywords:
Software defined network
Security architecture
Secure communication

ABSTRACT

For the multi-domain software defined network (SDN), different controllers are not able to directly communicate with each other due to the different distances among control planes. Therefore, the exchange of information among different domains is generally unsecure. The main contribution of this paper can be summarized into two parts. Firstly, architecture of multi-granularity security controller is proposed, which includes a basic control module and a multi-granularity security customized module. Secondly, a secure communication mechanism is proposed for distributed controller, where a prototype of this mechanism is implemented. In particular, this mechanism can make use of the border switch as inter-domain agents, where special packets are used by the controller to send messages to the secure tunnel. A two-step authentication of the controller can be provided by inter-domain agents and digital certificates. The experimental results demonstrate that the distributed controller secure communication mechanism is capable of effectively improving the security of SDN domain.

1. Introduction

After the development of OpenFlow [2] protocol by researchers from Stanford University, the software defined networking (SDN) [1] has emerged as a novel network structure. Due to requirements of the new architecture, researchers have proposed RCP [3], 4D [4], SANE [5] and Ethane [6] architectures by utilizing decoupling control plane and data plane. With the successful application of the OpenFlow protocol, the open network foundation (ONF) was established in 2011, where the SDN white paper is published [7]. By using SDN, the protocol no longer completely depends on equipment manufacturers to deploy and develop. Notably, the network is capable of obtaining highly customizable and intelligent network management in SDN. Currently, the inter-domain controller communication standard is missing in SDN. However, it is of significant importance to secure the inter-domain information, where the stability and reliability of multi-domain SDN can be achieved.

This paper is structured as follows. In Section 2, previous works are reviewed on the intra-domain security and inter-domain interconnection. In Section 3, the Multi-granularity security controller architecture is proposed. Moreover, secure communication mechanism of the distributed controller is proposed, where the prototype of this mechanism is implemented in Section 4. Finally, the conclusion is presented in Section 5.

☆ Reviews processed and recommended for publication to the Editor-in-Chief by Guest Editor Dr. Zhihan Lu.

* Corresponding author.
E-mail address: shangfj@cqupt.edu.cn (F. Shang).

http://dx.doi.org/10.1016/j.compeleceng.2017.07.003
0045-7906/

2. Related work

2.1. Intra-domain security

The research of SDN security mainly includes security enhancement of the SDN architecture and application of SDN. In [8], an independent trust management or abnormal device detection is proposed to prevent malicious device attacks. Moreover, multi-trust anchor authentication center of the oligopoly or threshold cryptographic system is proposed to resolve security issues of the control plane communication including TSL/SSL defects, which may result in DoS and DDoS attacks. In [9], the security of the controller can be obtained by using replication, diversity, recovery mechanism, important information protection and priority based safety rules.

In [10], two architectures are proposed to protecting the security in SDN network, which can decouple and reconstruct the security control plane and data plane.

According to the investigation, researchers have proposed frameworks of security application development for SDN. However, the specific security issues in SDN are not adequately addressed. Therefore, a security controller architecture which has more comprehensive security features and customized security services are required to be designed, where a more flexible and effective SDN Intra-domain security can be upgraded.

2.2. Inter-domain interconnection

It should be noted that it is impossible to control the entire network due to limited resources of a single controller. Researchers have proposed a number of distributed controller solutions to improve the scalability of the control plane. Currently, researchers have investigated the interconnection mechanism of multi-domain SDN. In [11], a route transition scheme from traditional network to SDN is proposed, which is known as BTSDN. In BTSDN, the boundary of the SDN autonomous domain can adopt the BGP border router. In particular, the inter-domain runs the EBGP protocol, and the intra-domain runs the IBGP protocol based on the software router in the controller. The controller is able to indirectly control the border router by the OpenFlow switch. In [12], a BGP architecture, known as OFBGP, is proposed for the SDN, which is an upper layer application in the distributed controller. As a SDN software router, it can manage and control the intra domain and inter domain. In [13], Inter-domain Management Layer (IML) is proposed to horizontally divide the SDN resources, where management of network boundary and information sharing between domains can be facilitated. In [14], a new routing and switching scheme SDX is designed for SDN. In SDX, the exchange of routes is performed by Internet exchange point (IXP). All the SDN autonomous domains can send routing information to the IXP, where the IXP is able to obtain routing information from other domains. SDX can achieve fine-grained forwarding within the same domain in the multi-domain SDN by learning the whole network routing information. Due to the limited number of IXP, there is no access to IXP SDN domain, where the program is not be able to solve the routing exchange. In [15], SDN inter-domain interconnection mechanism is investigated. A east-west bridging scheme for heterogeneous NOS is proposed in SDN, where the information to be interacted is also designed. In our scheme, the inter-domain controller employs the point-to-point communication mode. In addition to routing information exchange, the network view can be shared with other autonomous domains to obtain a global view. In [16], a controller architecture, DISCO, is proposed in a distributed SDN multi-domain network to establish inter-domain interconnection with the east-west interface of controller.

Based on the OpenFlow protocol, a multi-granularity security controller architecture is designed for multi-granularity security controller. The security interconnection requirements for inter-domain controllers are analyzed in multi-domain environments, and the secure communication mechanism for distributed controllers is also designed. Based on granular computing in the controller, security services can be flexibly customized in the intra-domain and inter-domain. Therefore, a multi-granularity security architecture can be established.

3. Multi-granularity security controller architecture

3.1. Controller architecture design

3.1.1. Security scheme

3.1.1.1. Controller defense scheme. In general, OpenFlow Packet-In messages is used by the denial of service attacks to send unknown flow. If the controllers receive Packet-In messages, the controller can obtain the source switch information, the header of data packet and so on. A Packet-In message is defined as PacketIn = (in_sw, in_port, in_res, src_mac, dst_mac, src_ip, dst_ip, src_port, dst_port). In particular, in_sw represents the switches sending the message, in_port represents the switch port sending the message, in_res represents the reason of sensing the messages, src_mac represents the source physical address of the packet, dst_mac represents the destination physical address of the packet, src_ip represents the source IP address of the packet, dst_ip represents the destination IP address of the packet, src_port represents the source port of the transport layer, and dst_port represents the destination port of the transport layer. Based on the definition of the Packet-In message, Packet-In flow-based detection or machine-based detection can be used to identify and block malicious Packet-In messages. As a result, the controller can be protected against DoS attacks.

3.1.1.2. Flow table management scheme. The configuration consistency is the main cause of the flow table security. It is required that there is no logical conflict with the existing flow table entries when a new entry is going to be written. In SDN, the flow table configuration will generally consist of a large number of switches. If as the only management unit for the flow table management, the switch becomes incapable of completing the logic detection. Therefore, all the networks are required to be considered as a whole, where each forwarding policy (FlowPolicy) is managed as a unit. The forwarding strategy consists of three parts, namely, source, destination, and action. For instance, FlowPolicy consists of src, dst and action. For the controller, each forwarding strategy is stored and updated sequentially. The conflict detection algorithm is used by the controller to detect new strategies. Subsequently, the flow table is written on the switch by detecting the policy, where the flow table logic confusion could be desirably avoided.

3.1.1.3. Application management scheme. The openness of the northbound interface is the foundation of innovation for SDN application. In other words, the network management is authorized to the application developers in an extent manner. The application running over the controller should be restricted to the operation of the network. Firstly, the application needs to manage the calling permission of the northbound interface. Therefore, the controller can obtain required permissions of the application and notify the network administrator. Since malicious behaviors cannot be detected by the simple authority management, applications should detect malicious code before running a program. If necessary, the behaviors of the program should be simultaneously tested in a sandbox environment. For the application of the sensitive call permissions, their behaviors can be detected in real time and network operations can be prevented from being damaged.

3.1.1.4. Security scheme extension. As the centralized controller for the network, the reliability of the controller is of significant importance. The reliability of network operations can be improved by maintaining redundant controllers and letting them take over the network when the master controller is down. Therefore, important information backup on the redundant controller should be carried out by the main controller and real-time synchronization should be performed.

It is also necessary for controllers to protect security of the data plane. Moreover, the controller should also improve the detection of attacks on the network hosts in addition to attacks filtering of Packet-In messages. Meanwhile, access control can be used as a safety management tool for data plane to run in the controller and provide more network protection. To prevent the network forwarding policy from violating the admission control policy, the security detection of the corresponding increased policy is also required to be considered in the flow table management process.

3.1.2. The SDN security controller architecture

As the key control core in the SDN network architecture, the basic requirements of the SDN architecture should be satisfied for the control layer. However, a perfect design standard is not available yet, where various types of OpenFlow controllers have emerged. In this paper, a SDN security controller architecture is proposed based on the SDN security scheme, which is shown in Fig. 1. The architecture can be divided into a basic control module and a multi-granularity security customization module. More specifically, the basic control module can realize basic functions according to the SDN architecture, and the multi-granularity security customization module can realize the customizable security functions in the controller.

3.1.3. Module design

Threat defense module can integrate intrusion detector, access controller and other security features. The intrusion detector sub-module can provide the controller host and flow detection defense based on the SDN network. The sub-module of access controller can establish the traffic rule of the whole network and execute for the network administrator. The threat defense module can identify packets on the network and filter out illegal traffic by the southbound interface. If the data packet is released by the sub-module of intrusion detector, the sub-module of access controller can further filter the data packets based on access rules defined by the network manager.

The purpose of flow table manager module is to deal with the security problem of the flow table entries in the SDN network, which can be used to detect and resolve the collision of the flow table. It can also identify and process the security issues that might be triggered in flow table entries to be written. The flow table manager module can desirably cooperate with the routing service module and application manager module. When a flow table is generated by a routing service or application, it should be detected by the flow table manager module. If there is any conflict between the flow table entry and an existing flow table, a new security flow table entry can be generated in an automatical manner. The flow table that can pass the detection of the flow table manager module would be sent to the switch.

To facilitate fast recovery in presence of a single-point controller failure, the backup module can backup and continually update important data in controllers such as northbound applications, entire network flow tables and security policies. The redundant controller can also synchronize information with the master controller by using the backup module to take charge of the network in case of a master controller failure.

Application manager module can manage security for controllers of the network, including code auditor, authority manager of the program request interface and behavior detector in program running process and so on. The security function manager module manages the threat defense, flow table manager, backup and application manager modules. The network manager is able to make use of these modules to configure the network protection function, which can create a customizable security environment.

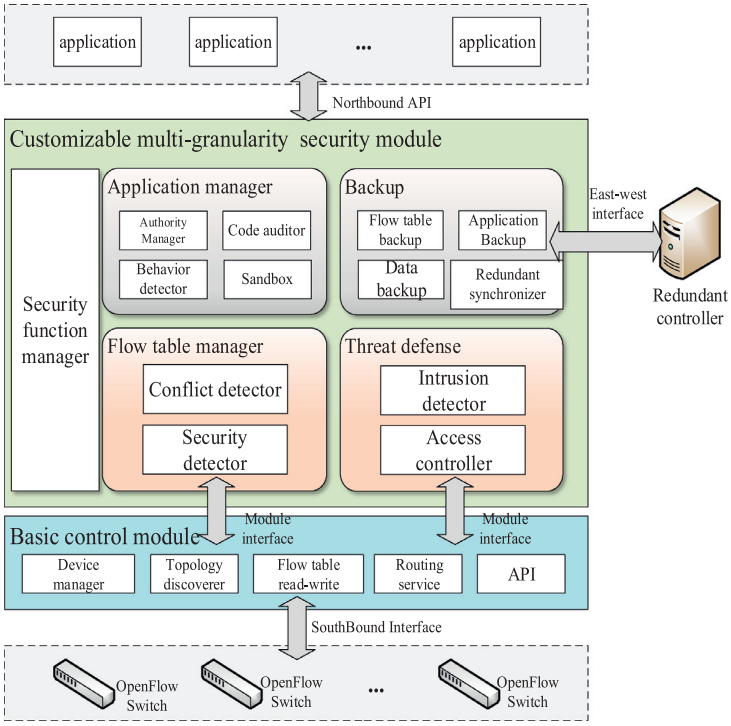

Fig. 1. SDN security controller architecture.

Table 1
Module interaction message types.

The message type	The message flow	The description
PACKET_INFO	Routing service–>Threat defense	Carrying packet information
PACKET_SEC	Threat defense –> Routing service	The packet detection result is allowed to be forwarded
PACKET_RFS	Threat defense –> Routing service	The packet detection result is denied forwarding
FLOW_INFO	Routing service–>Flow table manager	Carry the flow table information
FLOW_RSLT	Flow table manager –> Routing service	The flow table is allowed to be delivered and the result is contained in the information
FLOW_RFS	Flow table manager –> Routing service	The flow table is not allowed to be delivered

3.1.4. Operating mechanism

When the data packet is received, the OpenFlow switch will send a message to the controller, where the routing service of the controller is responsible for processing the destination of the packet. It represents the major security operation mechanism of the network. The routing service sends a message, which contains the packet information to the threat defense module before the packet table is generated. The intrusion detector module and the access controller module can sequentially identify the packet security. After the message of identification results is received, the flow table entry would be generated by the routing service. The generated flow table items can be collided and detected by the flow table manager module. Finally, flow table is written to the switch by the routing service according to its processing results.

In Table 1, the types of messages that interact with modules are listed. As shown in Fig. 2, the interactions among modules are described as follows. The switch can send PACKET_IN message when receiving no packet matching the packet controller, routing service can send PACKET_INFO message to the threat defense module to request identification packets, threat defense module can transmit the identification result PACKET_SEC or message PACKET_RFS to the routing service module, the routing service can generate the forwarding flow table for the data packet and send the FLOW_INFO message containing the flow table information to the flow table manager module, the flow table manager module transmits the FLOW_RSLT message or FLOW_RFS message containing processing results to the routing service module, the discarded flow table is transferred to the flow table read-write module if the PINGET_RFS message or the FLOW_RFS message is received by the routing service module, the message of the flow table is transferred to the table of read-write module if the FLOW_RSLT message is received, and the flow table of read-write module sends the FLOW_MOD message with the flow table to the switch.

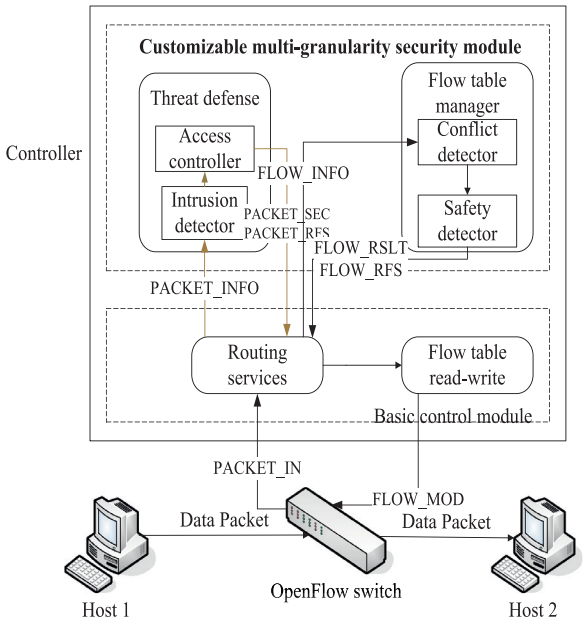

Fig. 2. Module interactions.

3.2. Multi-granularity security services

Granular computing can divide a large problem into many small problems. The concept of "granularity" can be formulated by the granularity of information proposed by Zadeh [17] and Hohss [18]. Granular computing uses granularity to measure information of a computer, where specific problems with different sizes of granularity can be desirably resolved. Granular calculation is mainly composed of particles, granular layers and granular structure [19]. Since the SDN network operates with a highly centralized control system, security features can also become "software-defined". Based on the idea of granular computing, refinement is proposed for SDN security services in this paper.

3.2.1. Multi-granularity service definition

Multi-granularity security service consists of different granularity security services, which can implement variable services with variable granularity. With different security requirements, multi-granularity security services should be adequately flexible to provide appropriate service granularity. If multi-granularity security service is denoted by S and the secure granularity space is denoted by Q with $S=(Q1, Q2, Q3, \ldots)$, Q1, Q2, Q3 are different granularities of the security levels.

3.2.2. Service granularity partitioning

To meet the requirements of multi-granular security services, security services are required to be granulated. In this paper, the security services of the SDN security architecture are capable of resolving many different security issues, where each type of problems also contains a number of refined security features. In this paper, the security service is classified into three granular layers according to the classification of security issues as shown in Fig. 3. There are different granularity security functions for each granular layer.

The security service can be divided into two coarse-grained particles by the top granular layer, which are intra-domain and inter-domain security services, respectively. More specifically, the SDN autonomous domain is considered by the granularity of the top layer and partition of the service resource with intra-domain and inter-domain is considered. For intra-domain security services, the security requirements to be satisfied are considered in a single autonomous domain. For inter-domain security services, the security requirements of inter-domain controller communication are mainly considered.

In the middle layer, the granularity of security services is refined, where different security levels of each particle can be classified based on the granularity of the top layer. According to the intra-domain security services, it can be divided into north to the security services, south to the security services, data plane security services and fault-tolerant security services.

Notably, the bottom granular layer is the finest granularity division. According to the intermediate granular layer, a finer granularity safety service can be formulated. Multi-granularity security services consist of fine grains at the bottom granular layer ultimately, where different security requirements can be satisfied by the combination of different service grains.

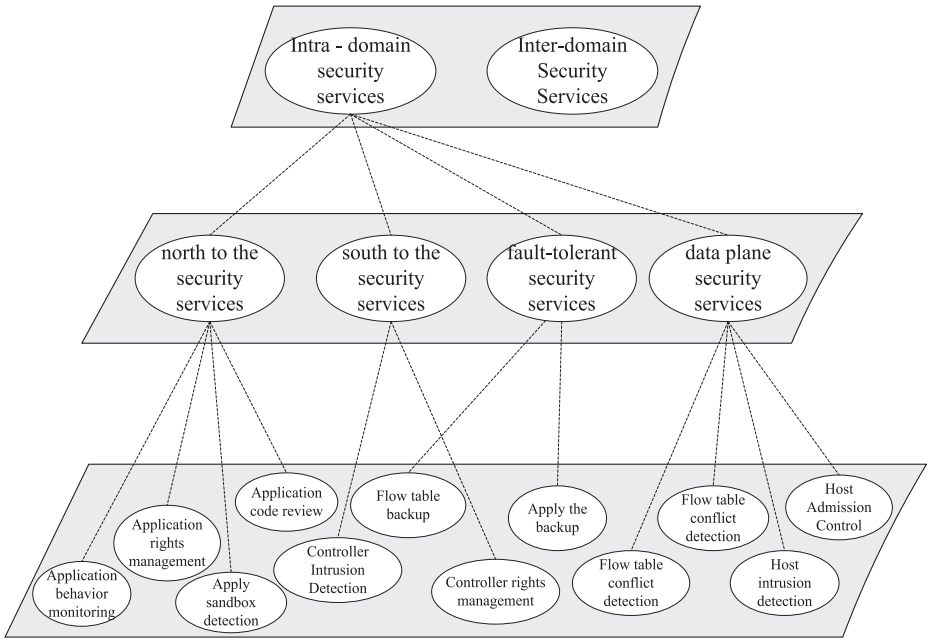

Fig. 3. Partition of security service granularity.

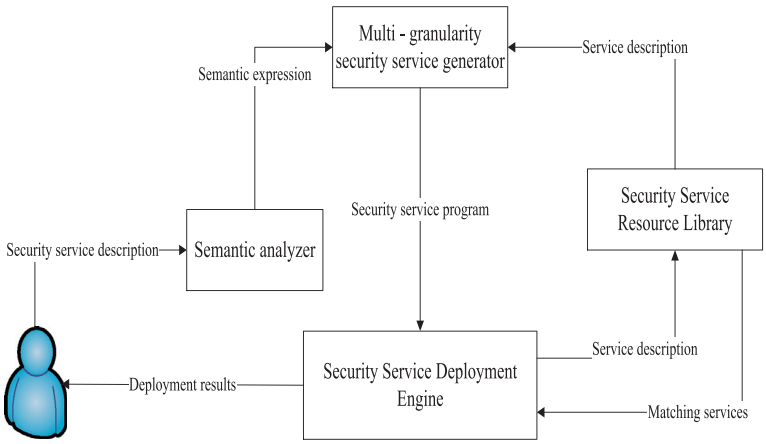

Fig. 4. Multi-granularity security service model.

3.2.3. Multi-granularity security manager model

Multi-granularity security service not only depends on the granulation of security service resources, but also the security requirements of users. The SDN controller is required to complete the combination of security services. In this paper, the controller architecture and security function manager module can complete the multi-granularity security service generation. For the generation of security services, the security requirements of users are required to be analyzed. Based on the analysis, the security services can be combined with different granular layers. Based on the security function manager module, the multi-granularity security service model can be shown in Fig. 4.

The SDN manager provides the security requirements on the semantic analyzer in a standardized form, which is by a security service description language. The semantic analyzer can obtain analysis on the semantic parser by investigating the security service description. According to the results of the semantic analyzer and description of the security service repository, the multi-granularity security service generator can obtain the matching and combining of services, where the generated service composition scheme can be sent to the security service deployment engine. The security service deployment engine can search over the security service repository for the services contained in the security service plan. Subsequently, the service would be deployed. If the security services are deployed, the results of the deployment are transmitted back to the network managers, where managers can easily achieve the realization of the security requirements.

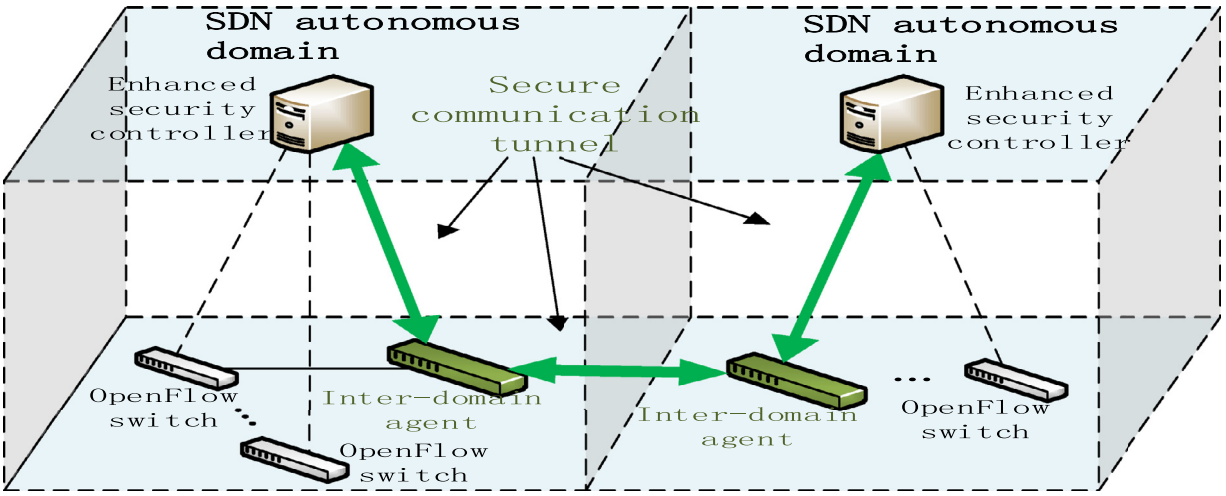

Fig. 5. Distributed controller communication architecture.

The round trip delay between the two hosts is 0.065 ms in absence of a security protection, and the average round trip delay is 2.219 ms when the security feature is turned on. The increase in time is 2.154 ms. These results show that the network performance would be influenced to a certain extent by introducing security modules. However, the experimental environment has not been adequately optimized, the network device is a virtual switch, and the time delay of introducing security modules is only in millisecond scale. Performance caused by introducing the security protection module would not be largely influenced. In [10], the latency of the two security architectures is different due to different design patterns. The delay of the VSA architecture increases around 0.58 ms, and the SDS architecture increases around 4.2 ms. The test results of delay for our architecture can increase 2.154 ms, which is between VSA and SDS. Since the network traffic is directly imported into the security device in the VSA architecture, almost all data forwarding can be completed in the data plane with small the processing delay. However, additional application of identification services in the controller would be by the SDS architecture. Therefore, the processing overhead would be increased. Although there is fewer overheads in the VSA architecture than that in the proposed architecture in this paper, it is advantageous that the correlation between controller and security device is low for VSA architecture, and it is not easy to implement flexible security management. Comparing the proposed architecture with the SDS architecture, security features of the proposed architecture are more desirable and flexible in security service customization with reduced processing overhead, although customizable security services can be achieved by both the SDS architecture and the proposed one.

4. Distributed controller secure communication mechanism

4.1. Secure communications architecture

4.1.1. Basic communication architecture

The SDN multi-domain architecture can use the software-border routing in the interconnection mode as an application of the control plane. An OpenFlow border switch, which can connect with two SDN autonomous domains, is used as an inter-domain agent to establish a secure communication tunnel between controllers of different autonomous domains by the connection of controllers and border switches. The basic architecture is presented in Fig. 5.

For this architecture, the controllers can use the security controller architecture designed in the previous section in every SDN autonomous domain. The network security can be improved by a security controller when the OpenFlow switches are only managed in their own domains. Since the previous security controller architecture only considers a security problem in the intra-domain, the enhanced security controller is adopted in the SDN multi-domain network environment, where the inter-domain routing and secure communication between the controllers would be realized by adding the east-west interface in the basic security architecture. Details of the design will be described in next section.

As a boundary switch in the SDN multi-domain network, the inter-domain agent is not only required to complete the inter-domain packet-forwarding task, but also required to establish secure communication tunnels among the inter-domain controllers. Since the SDN can decouple the control plane and the data plane, the existing OpenFlow switch can only realize the packet forwarding function based on the flow table and the information interaction with the controller. As a result, requirements of the inter-domain secure communication channel with the controller cannot be satisfied. Therefore, additional modules are introduced for the inter-domain agent in the general OpenFlow switch, where the security of inter-domain communication can be improved.

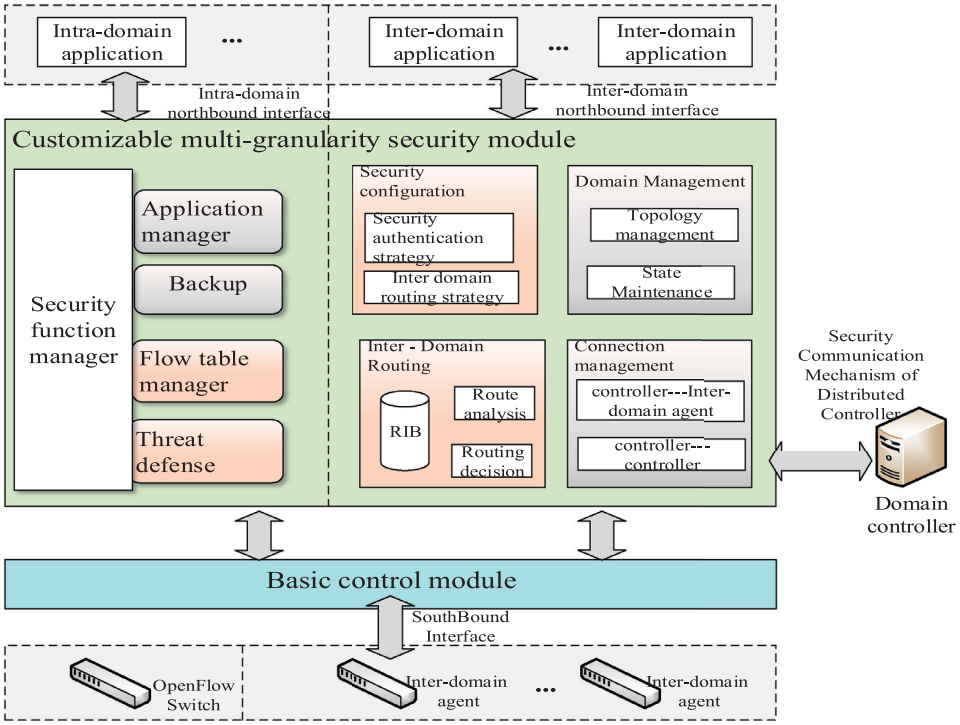

Fig. 6. Enhanced safety controller.

4.1.2. Enhanced security controller

Based on the requirements of controller interconnection in SDN multi-domain network, the inter-domain module is added on the basis of security controller architecture. More importantly, an enhanced security controller architecture is proposed. Inter-domain modules consist of security configuration, connection management, domain management and inter-domain routing, where the basic architecture can be shown in Fig. 6.

The security configuration module can provide a customizable inter-domain information sharing policy and can manage the security authentication policies among the controllers. In this module, the information sharing policy can affect contents of the information that the controller send to neighboring domains. While the security authentication strategy combines the connection management module to complete the authentication of the adjacent autonomous domain controller, the security configuration module can be managed by the security function manager module in common with the security module in the domain.

The connection management module can handle the network connections between the controller and the inter-domain agents. Moreover, it handles the network connections between the controller and its neighboring autonomous domain controller. If the border switch of an autonomous domain does not have the inter-domain agent function, the connection between the controllers of the two autonomous domains can be directly established and handled by the sub-module of “controller-controller” connection management. Otherwise, the controller can establish a connection with the border switch, which has the inter-domain agent function. Consequently, their communication will be handled by the sub-module of inter-domain agent connection management for the controller.

The domain management module is in charge of the information of all neighboring autonomous domains. It should be noted that the topology information requires to be parsed, stored and updated if it is obtained from the neighboring autonomous domain. Meanwhile, the module also requires to maintain the authentication status of the neighboring autonomous domain and operating status of the network. Not only the intuitive network management, but also more useful information for the higher-level application and other more innovative modules can be provided by the information of neighboring autonomous domains.

The BGP idea is employed in the inter-domain routing module. Moreover, the routing packets are processed by the inter-domain controller through a software router. The functions mainly include route advertisement generation, analysis and decision.

Fig. 7 demonstrates that in the middle granular layer, the inter-domain security services can be divided into controller communication security services and information sharing security services. Furthermore, the bottom granular layer can be divided into four sub elements.

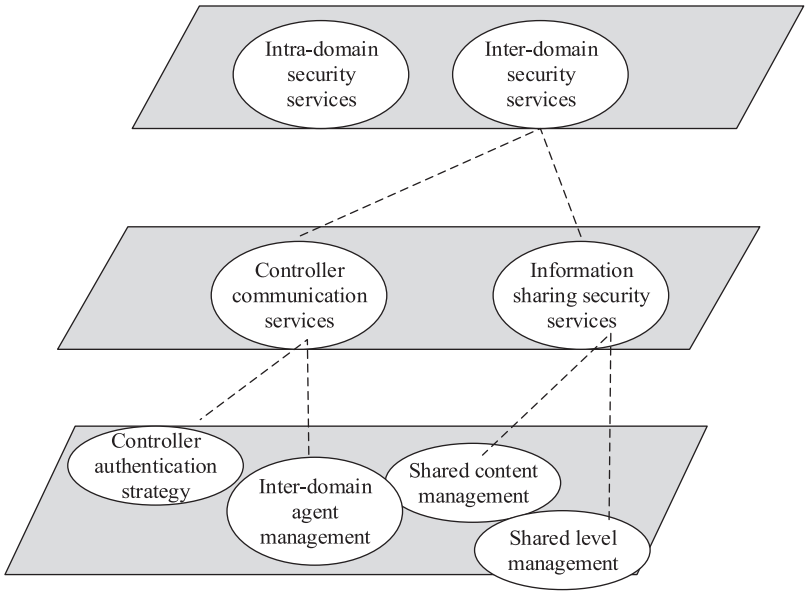

Fig. 7. The inter-domain security service granularity partitioning.

4.2. The secure communication mechanism

For the secure communication mechanism of the controller, there are three steps in the construction of the secure connection. In particular, they are neighbor discovery, credible authentication and tunnel establishment respectively. After the controller broadcasts information to its neighbors, the secure tunnel connection is requested by the controller. Subsequently, the secure communication tunnel will be established after the two parties are authenticated.

4.2.1. Message format

It should be noted that a proprietary Ethernet packet type identifier called 0xEFEF is used to transfer data between controllers. The payload portion preserves the IP packet format, and the UDP protocol is used in the transport layer. Moreover, the dedicated Ethernet identification is mainly used to distinguish between the general data packets and the controller data packets. The format of the reserved IP packets is intended to facilitate the communication between controllers at the network layer. The UDP is only a short link between controllers and its speed will not become the transmission bottleneck even though it is a connectionless but unreliable transport protocol. Thus, the reliability is not a critical issue in this paper. Compared with the TCP, there is no complex congestion control, retransmission policy and other additional overhead characteristic in the UDP. Substantially, the fast packet transmission and reception can be completed so that real-time information transfer between domains can be improved.

Fig. 8 demonstrates that the payload portion of the UDP packet is defined as the data format for the inter-domain message. In particular, the first 8 bits are message type and then the next 32 bits represent the length of message (the entire UDP payload portion, and bits are in bytes). The rest shows the message content. It should be noted that the message type includes the neighbor discovery message, Keepalive message and secure tunnel message with type IDs of 0×01, 0×02 and 0×03 respectively.

4.2.2. Neighbor discovery

In the traditional network, routers between autonomous domains are configured in order to establish a connection directly. In the multi-domain SDN environments, it is necessary for the control plane to master information of its neighboring autonomous domain. However, the control plane cannot be connected directly, so the controller needs to discover and learn information of its neighboring autonomous domains by the inter-domain agent.

It should be noted that the broadcast discovery is used to discover neighbors. In particular, each autonomous domain controller broadcasts own information by the inter-domain agent. In [15], the discovery of controller is designed, but information of other controllers can be acquired from the central registration server. In a large-scale of the SDN multi-domain network, the centralized registration method becomes invalid due to the widespread distribution of the controller.

In this paper, the controller sends the AS number of the autonomous domains, the supporting information of the secure tunnel and the port number of the secure tunnel service to the inter-domain agent. Subsequently, the inter-domain agent encapsulates the autonomous domain packets into Ethernet packets and forwards them to the neighboring autonomous domain. Fig. 9 shows the format of neighbor discovery message.

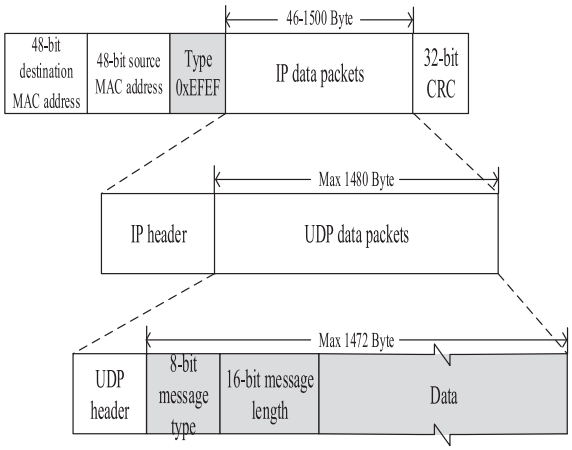

Fig. 8. The packet format between controller domains.

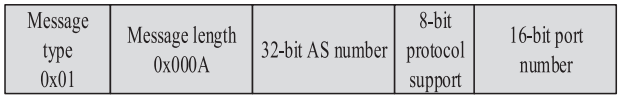

Fig. 9. The format of neighbor discovery message.

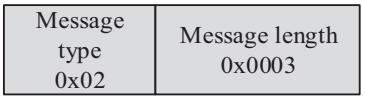

Fig. 10. The format of Keep alive message.

In general, the message type field is 0×01 and length is 10 bytes ($0 \times 000A$). The rest of the fields indicate the AS number (32 bits), the supporting information of the secure tunnel (8 bits, 0×00 means not support, 0×01 means support), and port number (16 bits).

When neighbor discovery has been performed, the controller keeps updated and maintains the information of the neighboring autonomous domain. If no message from neighboring autonomous domains is received by the controller within a specified period of time, Keepalive messages will be sent to check whether the neighbor exists or not. The controller returns a Keepalive message in a neighboring autonomous domain after a message is received. If the neighbor controller (Keepalive Timeout) does not receive a message within a specified period of time, the neighbor will be considered not to exist. In addition, the Message Timeout and Keepalive Timeout are set by each autonomous domain, which depends on the situation. It should be noted that only message type and length are contained in the Keepalive message. Fig. 10 demonstrates the message format.

4.2.3. Two-step authentication

In the OpenFlow protocol, there exists a TLS connection option between the controller and OpenFlow exchange, which is not used strictly. For the inter-domain controller communication, it is necessary to guarantee the security of the controller communication between inter-domain agents. Thus, the authentication between the controllers must be carried out. For the inter-domain communication architecture in this paper, two-step authentication protocol is proposed according to Client Puzzle [20] and DTLS to perform the identity authentication and encrypted transmission between controllers.

4.2.3.1. Authentication process. The two-step authentication involves inter-domain agent authentication and certificate authentication. After a handshake message is received from the controller who initiates the request, the inter-domain agent will be cached first instead of forwarding the request to the controller of the local autonomous domain. Subsequently, an authentication request will be imitated to the requester from the inter-domain agent, which is used to verify the non-attack intent of the requestor. After the requested controller finds the solution and passes the authentication, the inter-domain agent starts to forward the packet to the autonomous domain controller. Next, it will perform the handshake process to complete the certificate-based authentication. It should be noted that only the server-side authentication is in the DTLS handshake process. However, both sides need to clarify whether the other side is credible or not in the framework of this paper. Therefore, the strict two-way authentication is employed in this paper and the two-step authentication process is shown in Fig. 11.

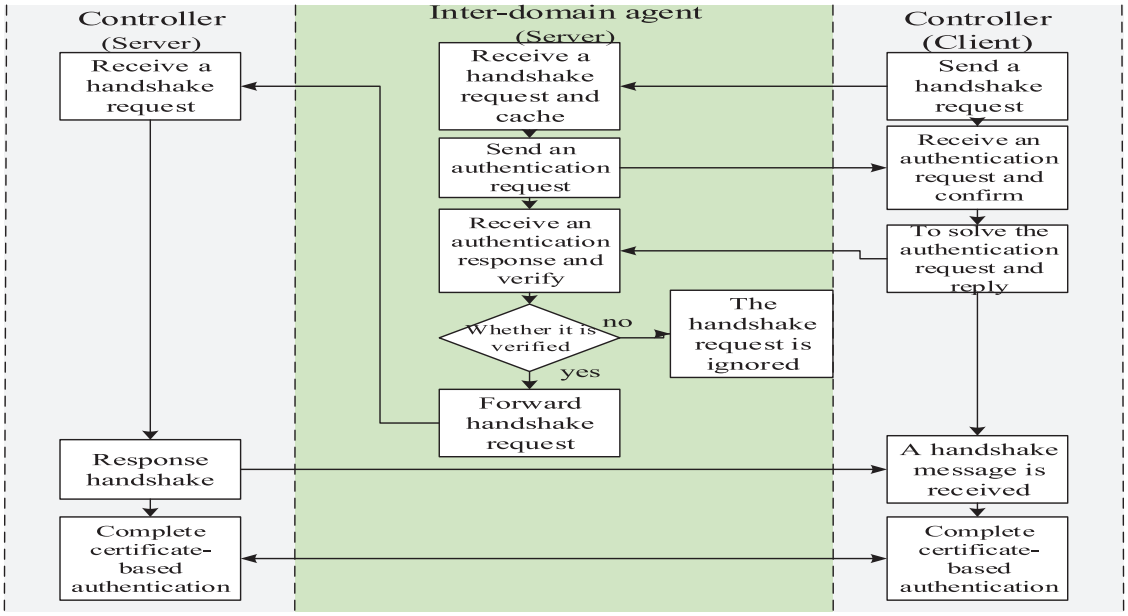

Fig. 11. The two-step authentication process.

4.2.3.2. The authentication scheme for the inter-domain agent. The authentication for inter-domain agent depends on three key elements: the random number R, degree of difficulty D and solution A. Thus, the authentication scheme focuses on the generation of the random number R and the validation of the solution A in this paper.

In this scheme, the random number R is generated using a single hash function and the SHA-256 algorithm is employed in this paper. Moreover, the generation of random numbers should depend on the autonomous domain AS_1 which initiates the interrogation, the physical address MAC_1, the AS_R number of the receiver, the physical MAC_R address of the network card, and a random number M.

$$R = SHA - 256(MAC_I \,||\, MAC_R \,||\, AS_I \,||\, AS_R \,||\, M) \tag{1}$$

The solution function is used repeatedly by the responder to answer the authentication request. In particular, the SHA-1 algorithm is implemented in the solution function to constantly obtain the correct answer A, which is based on the difficulty coefficient D, the random number R, and both AS numbers (AS_R and AS_1).

$$Psolve(\ SHA - 1(R \,||\, AS_I \,||\, AS_R \,||\ \ A),\ \ D\ ) = 0 \tag{2}$$

In the authentication solution, it should be determined whether the random number R can meet the requirements of the corresponding request or not. If R does not meet the requirements of the corresponding request,, validation of Answer A should be verified based on the solution algorithm.

The steps of the requestor to generate an authentication request can be summarized as follows:

1) Generate the difficulty coefficient D;
2) Extract the responder AS_R number from the neighbor list and generate a random number M;
3) Calculate the random number R according to (1);
4) Fill R and D in the authentication request packet and send them to the responder.

The steps to answer the authentication request can be summarized as follows:

1) Extract the random number R, the difficulty coefficient D and the solution A in the response message.
2) Exhaustively list the solution A meeting the requirements according to the solution function (2);
3) Fill R, D, and A in the authentication response packet and send them back to the supplicant.

The steps to verify authentication answers can be summarized as follows:

1) Extract the random number R, degree of difficulty coefficient D and solution A in the reply message;
2) Verify whether the random number R is consistent with the issue or not. If not, ignore the response; otherwise, continue;
3) Verify the solution according to (2).

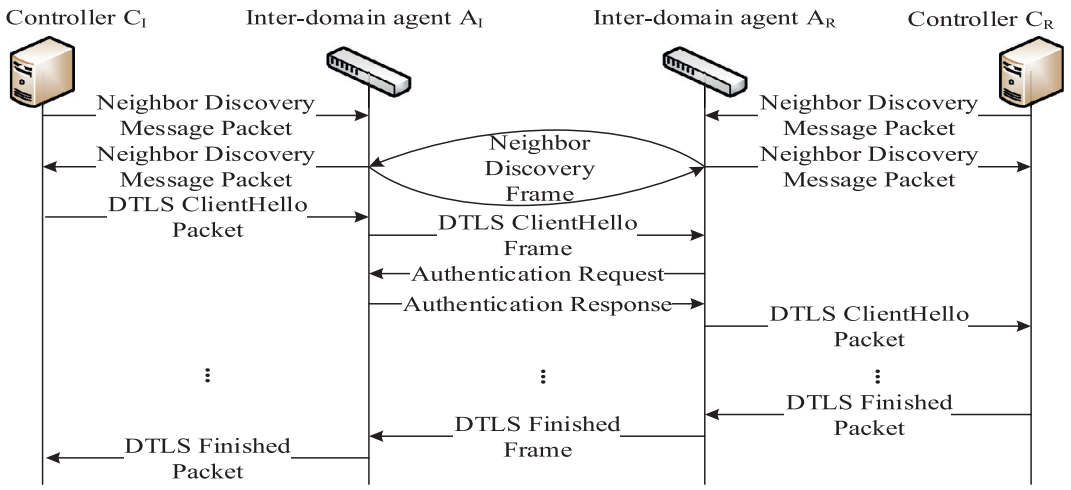

Fig. 12. The establishment process of the secure communication tunnel.

4.2.4. Tunnel establishment

During the operation of the SDN multi-domain network, the controller of each autonomous domain continuously broadcasts its own information. After the broadcasting message is received, the controller of the other autonomous domain will initiate the connection to the controller who sends the message.

The process of secure communication tunnel setup can be summarized as follows.

1) The controller constructs the neighbor discovery packet, and writes its own AS number, the supporting information of the secure tunnel and service port number into the packet. Subsequently, it sends the packet to all inter-domain agents. The inter-domain agent encapsulates the packet as an identifier 0×EFEF for the Ethernet frame, which will be forwarded.
2) Based on the broadcasting message of the sender, the controller decodes the AS number and the supporting condition of the secure tunnel of the neighbor autonomous domain after it receives the broadcast. If the secure tunnel is not supported by the sender device, the normal UDP data transmission method will be employed to send the information that will be shared according to the security configuration. Moreover, the original connection can be initiated by anyone. If no connection request is sent from the sender, a tunnel handshake message will be configured by one acted as the client to initiate a secure tunnel connection request to the server side.
3) After the server side receives the secure tunnel handshake message, it will no longer be a client to initiate a request to the other side. The two parties will perform the two-step authentication and encryption negotiation according to the inter-domain agent authentication and the digital certificate. After the two sides confirm the identities, the secure communication tunnel can be established. Otherwise, the tunnel setup fails and the normal UDP protocol is employed.
4) When the controller receives a broadcasting message from a neighboring autonomous domain, the timer for message timeout (any message from the neighbor is not received for a long time) will be activated. Subsequently, the controller will send Keepalive messages to the other side after timeout. If the other side does not return the Keepalive message within the Keepalive timeout period, the other side will be considered not to exist. Therefore, the secure tunnel will no longer be maintained.

Fig. 12 demonstrates the process of establishing a secure tunnel.

4.3. Safety analysis

In this paper, the controller secure communication mechanism is based on the inter-domain agent authentication mechanism, which can prevent attacks, mitigate the controller request pressure and perform the certificate-based authentication and encrypted transmission. The secure communication mechanism will be analyzed from two aspects: the controller security and message security.

4.3.1. Security authentication

4.3.1.1. Safety analysis. In the processes of communication among the controllers, the security of the controller is mainly based on the two-step authentication, which can defend against the controller attacks. For convenience, the two stages of the authentication mechanism are denoted by δ and φ, respectively. In particular, δ is the inter-domain agent authentication and φ is the certificate identity authentication.

The steps of the authentication protocol δ can be described as follows.

1) Connection request: C_I sends a request to C_R to establish the connection.

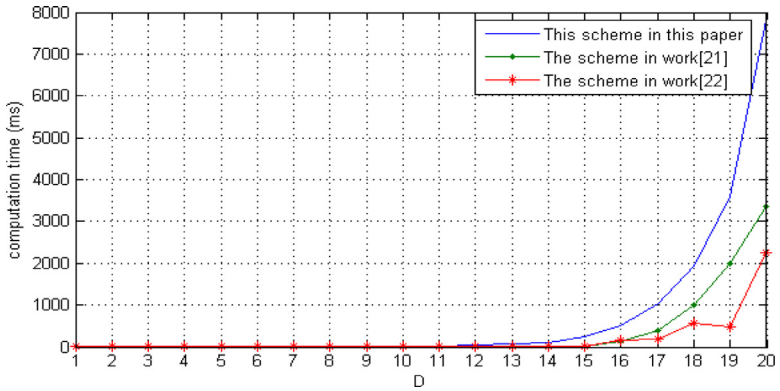

Fig. 13. Solution time for the authentication scheme.

2) Authentication Request: A_R generates a random number M after receiving the request and calculates $R = SHA-256(MAC_I||MAC_R||AS_I||AS_R||M)$. Next, it sends a message (R, D) according to the difficulty D to the C_I.
3) Authentication Response: C_I calculates the equation $Psolve(\ SHA-1(R||AS_I||AS_R||\ A), D\ ) = 0$, where solution A meets the above equation. Next, C_I sends the message (R, D, A) to A_R.
4) Verification: A_R verifies the consistency of R and D and the validation of A.

The steps of the authentication protocol φ can be described as follows:

1) Certificate transfer: C_R sends a digital certificate to C_I with $Cert_R = (Info_R||Sign_R)$, and $Sign_R = Enc(SHA(Info_R), PriK_R)$.
2) Identity authentication: C_I requests the public key $PubK_R$ from CA_I to verify whether the decrypted signatures $Dec(Sign_R, PubK_R)$ and the certificate hash $SHA(Info_R)$ are matched.
3) Certificate transfer: C_I sends a digital certificate to C_R with $Cert_I = (Info_I||Sign_I)$, and $Sign_I = Enc(SHA(Info_I), PriK_I)$.
4) Identify authentication: C_R requests the public key $PubK_I$ from CA_I to verify whether the decrypted signatures $Dec(Sign_I, \mathrm{PubK}_I)$ and the certificate hash $SHA(Info_I)$ are matched.

In the protocol φ, the input of random number R is based on the connection operation of AS_I, MAC_I, AS_R, MAC_R, and M. It is assumed that AS_I, MAC_I, AS_R and MAC_R are 32 bits and M is the random N-bit string in the IPv4 environment. Meanwhile, the binary length of (MAC_I || MAC_R || AS_I || AS_R || M) is $2\times32+2\times48+\mathrm{N}=160+\mathrm{N}$ bits. Since there are 2^{D} possibilities for the value of A, the probabilities of success is only $2^{(\mathrm{D}-(160+\mathrm{N}))}$ for the attacker in the use of random crack. According to the current SHA-256 security analysis, it can be concluded that δ meets the safety requirements since the complete collision attack is very difficult.

In the protocol φ, the reliability of the Cert depends on the confidentiality of the CA private key *PriK* and the encrypted signature *Sign*. Since *Sign* is based on asymmetric encryption, it is possible to use *PubK* to decrypt the *Sign* without *PriK* being exposed. Since *Dec*(*Sign, PubK*) cannot be matched with *SHA*(*Info*) due to the change of *SHA*(*Info*) by malicious tampering, the reliability of φ is very high when the *PriK* is strictly confidential.

In summary, the anti-attack ability of the protocol δ and the reliability of the protocol φ can guarantee the validity of the two-step authentication mechanism. Meanwhile, they can also ensure the security of the mechanism.

4.3.1.2. Comparison between the authentication time. In [21], the Client Puzzle scheme is designed for the TLS protocol. In particular, it uses the server ID, the client ID and random number to authenticate. In this scheme, the generation of the authentication response depends on $\mathrm{Hash}(C, N_S, N_C, X) = (000\ldots000||Y)$, where C, N_S, N_C, and X represent the client ID, the server random number, the client random number and puzzle solution respectively. In [22], the Client Puzzle scheme is designed for the IEEE 802.11i protocol, which can be expressed as $puzzle = \mathrm{Hash}(X||r||Ni||\ mac\_add||L)$, where $X, r, Ni,\ mac\_add$, and L represent the authentication solution, the access side random number, AP random number, AP physical address and degree of difficulty respectively.

It should be noted that D represents the degree of difficulty. In this paper, the authentication time for Q and the scheme of [21] and [22] is tested when D takes different values. In particular, D ranges from 1 to 20, with 100 tests for each value. Ultimately, it takes the average time of calculation. The number of bits for the random number M in δ is in the range of [8,120]. In order to maintain the consistency, the SHA-1 function is used in the schemes of [21,22]. It can be seen from the test results that the solution time increases exponentially with the increase of D. After D becomes greater than 12, the solution time for the proposed scheme is obviously higher than that of [21,22], and it increases much faster. In this paper, the authentication request contains many factors and the responding side cannot determine the solution directly based on the difficulty coefficient. Therefore, the authentication time is longer. A longer authentication time indicates the higher validity and stronger anti-attack ability of the proposed scheme. In addition, the test results are shown in Fig. 13.

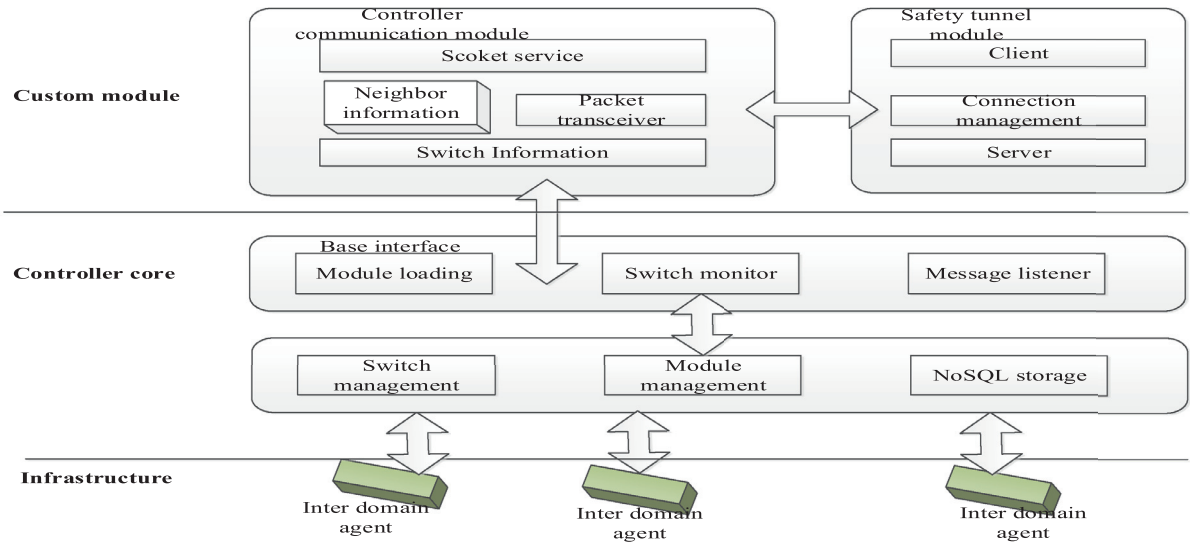

Fig. 14. The functional architecture.

4.3.2. Message security

During the communication process of the controller, the message transmission is carried out by the secure tunnel, which depends on the transmission mechanism of DTLS. Its security mainly includes to ensure the confidentiality during the transmission of data encryption, and the data integrity by the message authentication code. The secure tunnel can utilize the packet return time to deal with the packet loss problems due to connectionless transmission. To prevent being out of order and enhance transmission reliability, the serial number is increased in the packet. When an error is detected, the secure tunnels can notify the other side, where the security can be improved.

4.4. Implementation and testing

In this paper, the controller secure communication function is designed and implemented on the open source controller Floodlight. In the network simulation environment, Mininet is used for testing.

4.4.1. Function design

Floodlight is written in Java. In the Floodlight, all functions are executed in the form of standalone modules. When the program starts, the controller application in singleton mode is implemented in each module, which can be easily loaded by the module configuration file. In the secure communication mechanism of this paper, the communication mode of the controller is implement based on the basic function of Floodlight and the establishment of the secure tunnel is realized. According to the requirements, the controller communication module and the secure tunnel module need to be designed on the basis of the controller basic function. Moreover, the implementation of the custom module depends on the core service and the interface by the Floodlight. Fig. 14 demonstrates the functional architecture.

4.4.1.1. Controller communication module. The controller communication module provides the data packet transmission and the reception function for controllers. Since the packets between the controllers are transmitted by message forwarding of the border switch, the IOFMessageListener interface should be implemented to receive the OpenFlow messages sent by the switch. In the secure communication mechanism, the message of the neighbor discovery should be broadcasted through the switch. Therefore, the controller communication module is required to send messages to the switch. In addition, the IOFSwitchListener interface can provide the connection information of the switch.

Fig. 15 demonstrates the UML in the controller communication module. It should be noted that the class name of the controller communication module is C2CManager, which implements three official interfaces and IC2CManagerService custom interface. In particular, the three official interfaces include IFloodlightModule, IOFMessageListener, IOFSwitchListener. Meanwhile, its properties depend on two official interfaces (IOFSwitchService, IFloodlightProviderService) and Neighbor custom class. The description of each class or interface is shown in Table 2.

4.4.1.2. Secure tunnel implementation. According to the Bouncy Castle Open Source Password Kit, the secure tunnel is implemented for the server and client in a multi-threading manner respectively. Moreover, the multiple clients are allowed to initiate connections, and multiple connections are also allowed to be received. The IC2CManagerService by the controller communication module is implemented to send and receive the data packets of the client and the server. Fig. 16 illustrates the UML of the secure tunnel with the description of each class or interface is shown in Table 3.

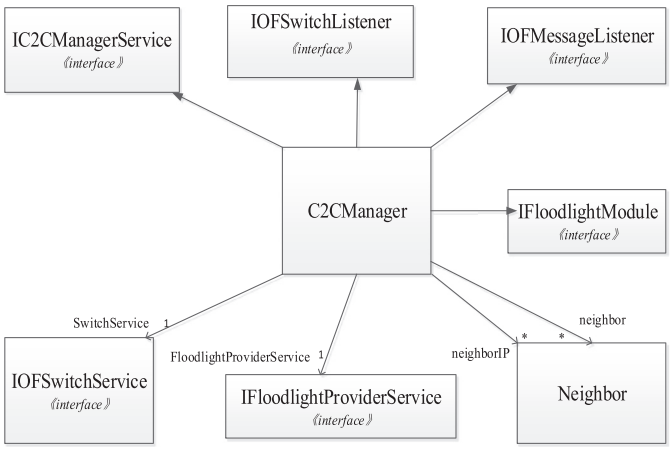

Fig. 15. The UML of the controller communication module.

Table 2
The class description of the controller communication module.

Class name	Function
IFloodlightModule	Implement this interface to instantiate modules
IOFMessageListener	Implement the received method in the interface to receive the OpenFlow message
IOFSwitchListener	Implement the added switch and removed switch methods of the interface to keep the connection between the switch and the controller in real time
IC2CManagerService	Custom interface, including receiving Packet and sending Packet; provide C2CManager-based Socket communication to other modules
IOFSwitchService	Register switch listener and get the switch object to send OpenFlow messages to the switch
IFloodlightProviderService	Register the OpenFlow message listener
Neighbor	Custom neighbor class, including the AS number of neighbors, physical address, IP address, connection location (switch, port number), and other information

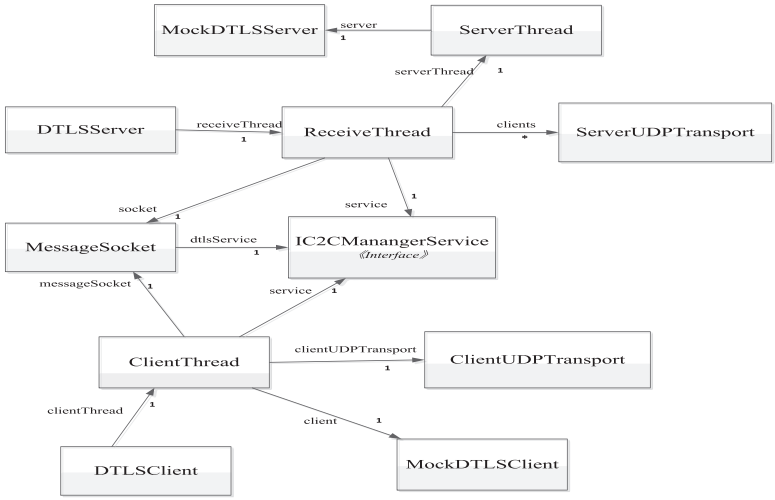

Fig. 16. The UML of the secure tunnel.

4.4.2. Testing

Firstly, the implementation of the secure communication mechanism of the controller is tested in order to validate the feasibility of the scheme and the correctness of the design. In particular, there are two Floodlight controllers in the test environment, where each of them is connected to three switches. In addition, there exists a pair of boundary switches connected between the two control domains. Both of controllers support the secure communication mechanism. After the network is constructed, Controller1 and Controller2 should send the neighbor discovery message. Moreover, one controller will act as the client and request the establishment of the secure tunnel after a neighbor is discovered. The network topology is shown in Fig. 17.

Table 3
Description of safety tunnel.

Class name	Function
DTLSServer	Server class
ReceiveThread	Packet receive thread class
ServerThread	Server thread class
MockDTLSServer	Server prototype class
ServerUDPTransport	Server packet transfer class
MessageSocket	IC2CManagerService based on Socket
DTLSClient	Client class
ClientThread	Client thread class
MockDTLSClient	Client prototype class
ClientUDPTransport	Client data packet transfer class

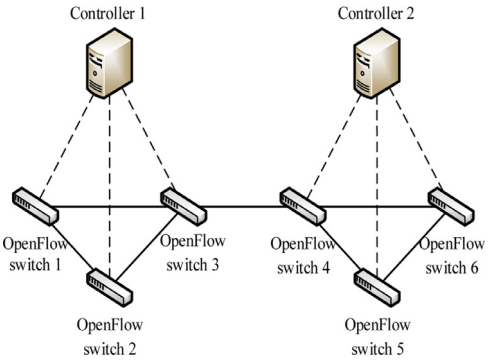

Fig. 17. Test environment.

Fig. 18. Message capture controller.

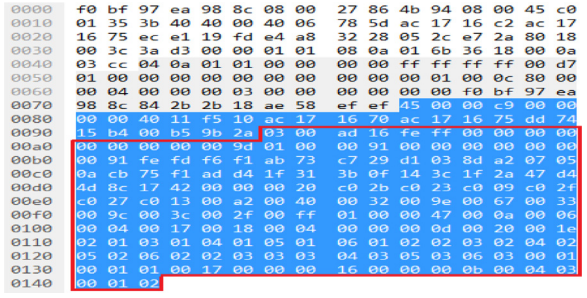

Fig. 19. Secure tunnel messages.

Data packets, secure tunnel messages, and neighbor discovery messages are captured between controllers by Wireshark with the results shown in Figs. 18, 19 and 20 respectively.

It can be observed from Fig. 18 that the highlighted area represents the data packet type 0xEFEF of controller messages. In Fig. 19, the first byte of the message 0×03 represents the secure tunnel messages. The outlined part in Fig. 20 indicates the message content of the controller. In addition, the first byte is 0×01, which implies that the message type is neighbor

```
0000  f0 bf 97 ea 98 8c 08 00  27 86 4b 94 08 00 45 c0
0010  00 92 29 17 40 00 40 06  8b 29 ac 17 16 c2 ac 17
0020  16 75 ec dd 19 fd 5f e4  16 fe 2c f9 18 fb 80 18
0030  00 3c 37 a6 00 00 01 01  08 0a 01 6b 23 68 00 09
0040  fc 5a 04 0a 00 5e 00 00  00 00 ff ff ff ff 00 34
0050  01 00 00 00 00 00 00 00  00 00 00 01 00 0c 80 00
0060  00 04 00 00 00 02 00 00  00 00 00 00 ff ff ff ff
0070  ff ff f0 bf 97 ea 98 8c  ef ef 45 00 00 26 00 00
0080  00 00 40 11 b8 3b ac 17  16 75 ff ff ff ff 15 b3
0090  15 b3 00 12 4c 16 01 00  0a 00 00 0c a5 01 15 b4
```

Fig. 20. Neighbors found messages.

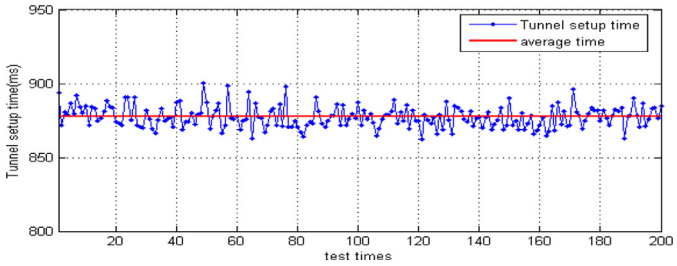

Fig. 21. Establishment of secure tunnel time.

discovery message. Moreover, two bytes are $0 \times 000a$, which indicates the message length of 10 bytes. The fourth bytes to the seventh are $0 \times 00000ca5$, which implies that the AS number is 3237. The last third bytes are 0×01, which shows that the secure tunnel protocol is supported. Finally, the last two bytes specify the $0 \times 15b4$ port number to be 5556.

Ultimately, the setup time of the controller secure tunnel is tested. In order to test the impact of the secure communication mechanism on the transmission performance, a multiple connection is established from one controller to another controller. Fig. 21 shows the delay of the connection with 200 times of tunnel establishments.

It can be seen from the figure that tunnel establishment time is ranging from 860 ms to 900 ms (averaged as 877 ms) based on the test of the secure tunnel establishment. Therefore, the secure communication mechanism of the controller brings some overhead characteristic to the network transmission performance in this paper. However, it does not greatly reduce the real-time data transmission.

5. Conclusions

In this paper, a novel distributed controller is proposed and implemented based on the multi-granularity secure communication mechanism, which can improve the security of SDN in a single control domain and multiple control domains simultaneously. In a single control domain, a more complete operating mechanism of the multi-granularity controller is designed with various security functions in the security architecture, where the intra-domain security of SDN is improved. In the multiple control domains, a secure communication mechanism is designed for the inter-domain security of SDN based on boundary switch. Meanwhile, the security services in the intra-domain and inter-domain is integrated and granulated in this paper to establish the multi-granularity security service model according to the security architecture. Subsequently, flexible and customized security services for network managers can be provided. Moreover, the prototype of this mechanism is constructed and tested. Based on the comparison between the authentication time and establishment time for the secure tunnel, the experiment implies that the SDN domain can be effectively kept secure by the secure controller communication mechanism. In addition, there is a low impact on network performance degradation caused by the security functions.

In future work, it would be interesting to further design message types of security controller, which accommodates to a variety of information transmission among controllers.

Acknowledgments

The authors would like to thank the Chongqing Research Program of Basic Research and Frontier Technology (No. cstc2016jcyjA0590). This work is partially funded by the National Natural Science Foundation of China (No. 61672004, 61602073, 41571401) and the Scientific and Technological Research Program of Chongqing Municipal Education Commission (KJ1500439).

References

[1] McKeown N. Keynote talk: software-defined networking. In: Proceedings of IEEE INFOCOM'09, April; 2009. p. 30–2.
[2] Mckeown N, Anderson T, Balakrishnan H, Parulkar G, Peterson L, Rexford J, et al. OpenFlow: enabling innovation in campus networks. ACM SIGCOMM Comput Commun Rev 2010;38(2):69–74.
[3] Caesar M, Caldwell D, Feamster N, Rexford J, Shaikh A, Jacobus VDM. Design and implementation of a routing control platform. In: Conference on symposium on networked systems design & implementation; 2005. p. 15–28.
[4] Greenberg A, Hjalmtysson G, Maltz DA, Myers A, Rexford J, Xie G, et al. A clean slate 4D approach to network control and management. ACM SIGCOMM Comput Commun Rev 2005;35(5):41–54.
[5] Casado M, Garfinkel T, Akella A, Freedman MJ, Boneh D, Mckeown N, et al. Sane: a protection architecture for enterprise networks. In: Conference on USENIX security symposium; 2006. p. 10.
[6] Casado M, Freedman MJ, Pettit J, Luo J, Mckeown N, Shenker S. Ethane: taking control of the enterprise. ACM SIGCOMM Comput Commun Rev 2007;37(4):1–12.
[7] Open Networking Foundation. Software-defined networking: the new norm for networks. [2012-4-13]. https://www.opennetworking.org/images/stories/-downloads/sdnresources/white-papers/wpsdn-newnorm.pdf.
[8] Yan Z, Prehofer C. Autonomic trust management for a component-based software system. IEEE Trans Dependable Secure Comput 2011;8(6):810–23.
[9] Porras P, Shin S, Yegneswaran V, Fong M, Tyson M, Gu G. A security enforcement kernel for OpenFlow networks. In: Proceedings of the first workshop on hot topics in software defined networks. ACM; 2012. p. 121–6.
[10] Qiu X, Zhao L, Gao T. VSA and SDS: two security architectures in SDN. J Chin Comput Syst 2013;34(10):2298–303.
[11] Lin P, Bi J, Hu H. BTSDN: BGP-based transition for the existing networks to SDN. In: Sixth international conference on ubiquitous and future networks (ICUFN). 2014; 2014. p. 419–24.
[12] Duan W, Xiao L, Li D, Zhou Y, Liu R, Ruan L, et al. OFBGP: a scalable, highly available BGP architecture for SDN. In: Mobile ad hoc and sensor systems (MASS), 2014 IEEE 11th international conference on. IEEE; 2014. p. 557–62.
[13] Thai P, De Oliveira JC. Decoupling policy from routing with software defined interdomain management: interdomain routing for sdn-based networks. In: Computer communications and networks (ICCCN), 2013 22nd international conference on. IEEE; 2013. p. 1–6.
[14] Feamster N, Rexford J, Shenker S, Levin D, Clark R, Bailey J. SDX: a software defined internet exchange. In: Open networking summit; 2013. p. 1.
[15] Lin P, Bi J, Wang Y. WEBridge: west transition for the existing networks to SDN. Sixth international conference on networks, 2014; 2014. 8:1926/Comp.
[16] Phemius K, Bouet M, Leguay J. DISCO: distributed multi-domain SDN controllers. In: Network operations and management symposium (NOMS), 2014 IEEE. IEEE; 2013. p. 1–4.
[17] Zadeh LA. Fuzzy sets and information granulatiorL advances in fuzzy set theory and applications. Amsterdam: North-Holland Publishing; 1979. p. 433–48.
[18] Hobbs JR. Granularity. In: Readings in qualitative reasoning about physical systems; 1990. p. 542–5.
[19] Wang G, Zhang Q, Hu J. An overview of granular computing. CAAI Trans Intell Syst 2007;2(6):8–26.
[20] Tang Q, Jeckmans A. Efficient client puzzle schemes to mitigate DoS attacks. In: Computational intelligence and security (CIS), 2010 international conference on. IEEE; 2010. p. 293–7.
[21] Raluca C, Monica B. TLS protocol: secure protocol with client puzzles. In: Electronics and telecommunications (ISETC), 2010 9th international symposium on. IEEE; 2010. p. 149–52.
[22] Dong Q, Gao L, Li X. A new client-puzzle based DoS-resistant scheme of IEEE 802.11i wireless authentication protocol. In: International conference on biomedical engineering and informatics; 2010. p. 2712–16.

Fengjun Shang is currently a Professor of Chongqing University of Posts and Telecommunications. His research interests include sensor network, future internet, IOT, network optimization and Cloud Computing.

Yan Li is a postgraduate student of Chongqing University of Posts and Telecommunications. Her research interests include software-defined sensor network and network coding.

Qiang Fu holds his M.S. from Chongqing University of Posts and Telecommunications, China. His research interests include software defined network.

Wenkai Wang holds his M.S. from Chongqing University of Posts and Telecommunications, China. His research interests include data storage.

Jiangfan Feng received his Ph.D. degree from Nanjing Normal University, in 2007. His research interests include spatial information integration and multi-media geographical information system.

Li He is currently an Associate Professor of Chongqing University of Posts and Telecommunications. Her research interests include cloud computing.