Simulation Modelling Practice and Theory 74 (2017) 28–45

Contents lists available at ScienceDirect

Simulation Modelling Practice and Theory

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

A trusted access method in software-defined network

Jing Liu[a], Yingxu Lai[a,*], Zipeng Diao[a], Yinong Chen[b]

[a] Beijing University of Technology, Beijing, China 100124
[b] School of Computing, Informatics, and Decision Systems Engineering, Arizona State University, AZ 85287, USA

ARTICLE INFO

Article history:
Received 25 November 2016
Revised 26 January 2017
Accepted 5 February 2017
Available online 16 February 2017

Keywords:
Software-defined network
Network access
Trusted computing

ABSTRACT

In software-defined networks (SDN), most controllers do not have an established control function for endpoint users and access terminals to access network, which may lead to many attacks. In order to address the problem of security check on access terminals, a secure trusted access method in SDN is designed and implemented in this paper. The method includes an access architecture design and a security access authentication protocol. The access architecture combines the characteristics of the trusted access technology and SDN architecture, and enhances the access security of SDN. The security access authentication protocol specifies the specific structure and implementation of data exchange in the access process. The architecture and protocol implemented in this paper can complete the credibility judgment of the access device and user's identification. Furthermore, it provides different trusted users with different network access permissions. Experiments show that the proposed access method is more secure than the access method that is based on IP address, MAC address and user identity authentication only, thus can effectively guarantee the access security of SDN.

1. Introduction

Software-Defined Networks (SDN) is originated from Stanford University's Ethane project, which was named OpenFlow in 2006 by Professor Nick Mckeown. With the continuous development to mature, OpenFlow is officially named as SDN in Global Environment for Network Innovations (GENI) project. SDN architecture decouples the control plane and data plane [1] in order to implement the centralized control and management of the entire network. These basic features of SDN architecture increase the flexibility of network deployment and enhance the programmability of the network. Therefore, SDN architecture can meet the needs of a fast and efficient network.

The development of computing and communication industry increasingly requires network flexibility and programmability provided by SDN. However, security is becoming a key factor restricting the development of SDN. For example, most current SDN controllers do not have established user access control functions. This means there can be a problem of identity authentication when a user and a device access the network. The defect can lead to forgery attacks of identity for the purpose of illegally obtaining information, gaining control over the entire network to cause network congestion and even paralysis. In order to solve the problem of security checkup for access terminals, we proposed a method for trusted access in this paper for access authentication. We designed and implemented a trusted access method of the software-defined network, which may acted as the first shield of network access.

* Corresponding author.
E-mail addresses: jingliu@bjut.edu.cn (J. Liu), laiyingxu@bjut.edu.cn (Y. Lai), diaozipeng@emails.bjut.edu.cn (Z. Diao).

http://dx.doi.org/10.1016/j.simpat.2017.02.001
1569-190X/

The rest of the paper is organized as follows: Section 2 reviews security access technologies and Trusted Network Connect related studies in the traditional network, as well as the access security related studies in SDN. The architecture of SDN trusted access is presented in Section 3. The detailed description and Security analysis of the authentication protocol are elaborated in Section 4. Section 5 describes the implementation of trusted access method of SDN. Section 6 presents the experiments that show the credibility and effectiveness of the proposed architecture. Finally, conclusion and outlook of this paper are drawn in Section 7.

2. Related work

Security access of terminals mainly focuses on network users' authentication, which accurately identifies users and the characteristic parameters of the network they used, and verifies the legitimacy of the device and user identity. It restricts or grants the access permission of network by the network host and user's identity, which achieves the goal of access security.

In the traditional network, researchers are constantly introducing new technologies and methods to solve the problem of terminal security access. The concepts such as active defense and trusted access have been proposed accordingly. Currently, the common access authentication technologies include Point-to-Point Protocol over Ethernet (PPPoE), 802.1x, Mac Address Bypass, web authentication technology, etc. [2]. The typical application of PPPoE lies in Asymmetric Digital Subscriber Line (ADSL) access, which is widely used by Internet Service Provider (ISP) for network access of individual customers due to its maturity as a technology. However, this authentication method is not suitable for multi-access networks, such as LAN access authentication [3]. 802.1x protocol is based on the access control and authentication protocol of client/server mode that can restrict unauthorized users and devices to access network through the access port. The protocol implements the separation of authentication and services and ensures the efficiency of network transmission. However, 802.1x itself does not have the related setting for maintaining connection. Consequently, it is necessary to determine the user's status by periodically sending authentication requests. This approach increases the burden on authentication system. In the data center network (flexible SDN network), frequent network changing or moving accessed devices will increase the cost, lower the user's experience, and delay the service. Web authentication is usually combined with DHCP server, which allows user to be authenticated via web pages. So there are no constraints of across layers and multicast protocols. Since the authentication is based on the highest layer of network, Web authentication is convenient with high degree of freedom. However, it also makes it difficult to detect users' abnormal situations caused by the underlying network.

Trusted Network Connect (TNC) theories were proposed in 2003. The main idea is based on the fact that trusted computing technology [4] can complete the integrity measurement of user's platform when the user accesses network. By judging whether conforming to the security policies, it can verify the credibility of the platform. The achievement of these authentications requires binding of the Trusted Platform Module (TPM) [5]. To some extent, TNC is an extension of the network environment from TPM launched by Trusted Computing Group (TCG). Compared with traditional network access technology, TNC increases the credible verification of the platform, which is a proactive network security protection technology. TNC is an open and common architecture that can work with existing network technologies and devices, such as combining 802.1x and PPP to achieve the function of access control. In the past decade, many scholars have carried out thorough studies and discussions of various aspects about the application of TNC in the traditional network. Based on Extensible Authentication Protocol and 802.1x access control architecture, Lin [6] proposed an improved trusted network access solution with fine-grained trusted certification for each accessed terminal to achieve quantitative assessment of the state of credible and access control. Luo et al. [7] proposed a method of security quantitative analysis and security enhancement mechanism. The method was based on semi-Markov process, aiming to TNC protocol. Using Intel IXP2400 network processor, it built a TNC prototype system. Yan et al. [8] proposed a security authentication protocol based on TNC structure and the idea of terminal integrity measurement of TNC. There are studies that combined terminal integrity measurement technology and Public Key Infrastructure (PKI) to ensure the credibility of the terminal platform. Liu [9] with intention of implementation issues about trusted access for WiMAX presented wireless credible access protocol based on EAP-TTLS. Many companies have product supporting TNC architecture, such as Symantec, Juniper Network, Still Secure, Wave Systems, and Extreme Networks. In the traditional network, to ensure the terminal security access via TNC technology has achieved substantial research results, inflicting a significant impact on solutions to network security.

In recent years, as the most promising emerging technology, studies on security mechanisms of SDN have become one of the important research directions. Many researchers have begun to explore the access security issues of SDN. FortNOX [10] architecture, designed by Porras et al., is a security kernel aimed for open source controller NOX with the increase of role-based authentication module. It is used to sign each flow rules and specify the appropriate privileged category to candidate flow rules. The literature proposed the FortNOX prototype system, which is an effective extension for increasing the security performance of the NOX controller. However, NOX was implemented in C language, which brings many difficulties in the development. As a consequence, it has been replaced by a more mature architecture controller. Sasaki et al. [11] combined the user authentication registry and user roles and proposed the role-based access control (RBAC) in the OpenFlow network, which has some positive effects on globally monitoring the entire internal network. However, on the definition of the users, the article in coarse-grained manner is not able to define multi-dimensional access control function. Mattos and Duarte [12] proposed AuthFlow, an authentication and access control mechanism based on host certification. It offered AAA authentication services in conjunction with the controller. It intercepted RAP messages between the host and the RADIUS authentication server, and reported authentication status to OpenFlow controller. The controller could allow or deny the traf-

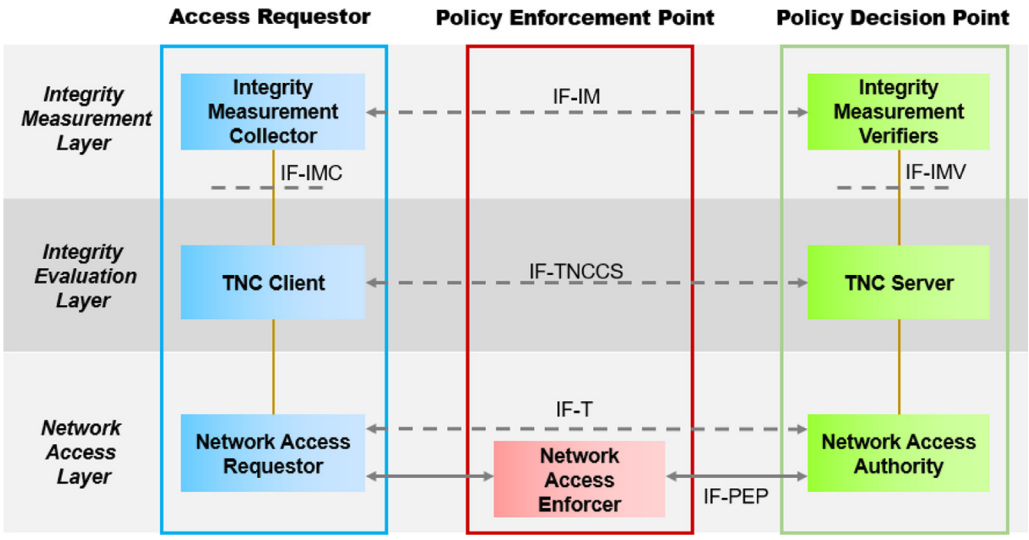

Fig. 1. Trusted network connect architecture.

fic flow based on the information. This access control was achieved via certifications issued by the host. Certificate-based authentication methods, however, it was complex with a high cost. Kinshita et al. [13] pointed out the limitations of conventional access control method. In OpenFlow network, it used VLAN tagging and firewall to realize wireless LAN users access mechanism with full realization of the centralized logic control of controllers in access control. Nevertheless, its network policy definition was complex, causing a higher requirement for network managers' of global network knowledge. Matias et al. [14] proposed FlowNAC, a flow-based access control, which used an improved user authentication 802.1x protocol. According to the target, it also adopted the appropriate granularity to dynamically identify data flow, with the centralized management features of SDN, as well as the global control of flow. However, it increased the authentication process for each access flow, and lacked time and other multi-angle control.

Under the existing SDN architecture, security access only authenticates access user identity, but not the integrity verification between the access terminal operating system and servers. Thus, it lacked proof of their credibility, and there are still security vulnerabilities. For an accessed user or servers, even with a legitimate authentication, it does not guarantee the credibility of its platform [15]. We believe that the idea of trusted access can be introduced into SDN to judge the credibility of accessed devices and to identify user's identity, which can effectively prevent illegitimate users from accessing network. Therefore, combined with the separation of control and forwarding planes in SDN architecture, we proposed a trusted access method in SDN architecture based on the architecture of TNC. The method consists of a set of structures composition and a set of access processes. We also proposed a fast access authentication method and the idea of hierarchical access to the same user via trusted value for the flexibility of SDN network. It is believed that it can effectively avoid the existing shortcomings of access authentication in SDN. In order to meet the security requirements of the access method, we have designed a new security access authentication protocol with the formal description and logical reasoning proved by making use of security analysis based on BAN logic. Analyzing from protocol analysis tools can ensure the security of the access method.

3. The architecture design of SDN trusted access

3.1. TNC architecture

Trusted Network Connect (TNC) includes the open terminal integrity architecture and a set of standards to ensure security interactive operation. Trusted Network Connect architecture is shown in Fig. 1.

TNC architecture has three main entities: Access Requestor (AR), Policy Enforcement Point (PEP) and Policy Decision Point (PDP). Access Requestor refers to a logical entity requesting access to the protected network. Policy Decision Point checks the credibility of the Access Requestor and decides whether the AR can access network. It has its own network access policy. Policy Enforcement Point is trusted certification according to the specific information of AR, and generates decisions.

Access Requestor (AR) consists of three main components:

Network Access Requestor (NAR): The main function is to send a network request, and participate in identity authentication.

Integrity Measurement Collector (IMC): It measures integrity information of AR entities from different security perspectives.

TNC Client (TNCC): TNCC and its own IMC interactive information collect integrity measurement value. Meanwhile, it sends the platform and integrity report back to the NAR.

Policy Enforcement Point's main components:

Network Access Enforcer (NAE): It is responsible for controlling access to the protected network.

Policy Decision Point (PDP) mainly consists of three components:

Network Access Authority (NAA): NAA determines whether an AR entity can access network, and inquires the TNC Server whether the integrity state of AR meet the security policies of the NAA.

TNC Server (TNCS): TNCS exchanges information with TNCC of entity AC, collects its own IMV decisions, forms the final decision and passes it to the NAA.

Integrity Measurement Verifier (IMV): According to the measurement result of the AR, it will verify the integrity of the AR.

TNC architecture consists of three layers: Network Access Layer (NAL), integrity assessment layer (IEL) and integrity measurement layer (IML).

Network Access Layer: it authenticates entities identities, and supports the connectivity of the traditional network.

Integrity Assessment Layer: The main task is to assess the integrity of the entity AR from perspectives of different access policies.

Integrity Measurement Layer: it mainly collects integrity measurement value associated with the entity and checks integrity.

In the TNC architecture, the main flow of the integrity check is as follows:

Before performing verification, TNCC first initializes each IMC, respectively, while TNCS initializes IMV. Next, NAR sends an access request to the PEP, then PEP sends a request to the PDP, which makes decision according to the AR identity and the state of platform integrity with correspondence to its own security policy. The decision-making results are enabled or disabled. Finally, PDP will send the results to the PEP, PEP implements this decision after receiving the results. Then the network connection process ends.

3.2. A trusted access architecture of SDN

TNC, an active defense technology, implements authentication of user identity and platform before the terminal access network. This can improve the security and credibility of the entire network. At the same time, as a general-purpose architecture based on open standards, TNC is applicable to a variety of different platforms. Therefore, the introduction of TNC architecture in the SDN can achieve the terminal's trusted access, and provide better protection for the security of SDN. However, TNC has some limitations. On one hand, TNC applies the trusted computing technology to the network access control through the technical means, but currently this access lack support of credible theory. On the other hand, multiple entities in TNC architecture need a lot of information interaction, but the TNC framework itself just simply describes how to transmit the message instead of giving the corresponding security protocol. Therefore, it is lack of support of security protocol.

In view of the above problems, the researchers have given some solutions in the traditional network. However, the SDN reconfigures the working mechanism of the network through the separation of control and data plane. In the SDN architecture adopting OpenFlow protocol, the working mechanism is a controller (or cluster), which collects the topology, traffic, and other information of the entire network, and calculates the path of traffic forwarding, and sends the forwarding flow table to the switch device through the OpenFlow protocol. The switch implements the forwarding action according to the flow table. For example, in the traditional network applying TNC framework, generally the policy enforcement point will control traffic by the way similar to 802.1X protocol. Physical ports in the switch is divided into two logical ports: controlled and uncontrolled, which can achieve separation between business and authentication. However, the switch in the SDN, implementing data forwarding does not have the right to control the port and deciding right of forwarding. This means that the 802.1X protocol needs to be improved to apply to the SDN. In the authentication process, if the user is authentic but the platform is untrusted, the 802.1X protocol cannot independently solve the function of limited access. As a result, the time of the terminal device will increase and the user experience will be reduced. Therefore, this paper considers that it is more suitable to achieve more credible access capabilities by combination of web authentication and TNC architecture in SDN.

In the future extended research, using the characteristics of controller dispatching strategy, the scheme designed in this paper can divide the terminals with credible user identities but untrusted by the platform into a VLAN, in which the patch distribution server is provided. Users can make security update about the terminal operating system, which can be accessed to the network after completion of update. It can dynamically adjust the corresponding patch distribution server according to the situation of the terminal. In addition, due to the reconstruction of SDN working mechanism, the application of TNC architecture also need to design a new secure access authentication protocol. Therefore, the existing solutions cannot be directly applied in the SDN. Re-research and feasibility studies are needed to apply TNC into SDN.

In consideration of the SDN architecture that makes the control layer focus on the controller, and the switch focus on forwarding, we proposed a trusted access architecture of SDN. From the perspective of TNC architecture, trusted access architecture of SDN is shown in Fig. 2. From SDN architecture perspective, we designed the structure shown in Fig. 3.

The trusted access architecture of SDN can classify all equipments involved in network access into Access Requestor, Policy Enforcement Point, and Policy Decision Point.

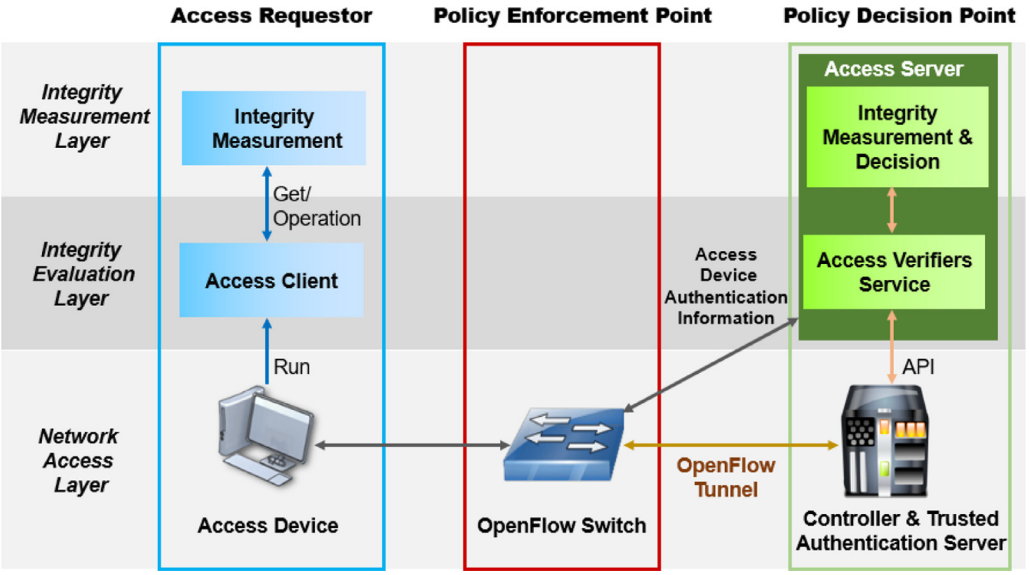

Fig. 2. A trusted access architecture of SDN (the perspective of TNC architecture).

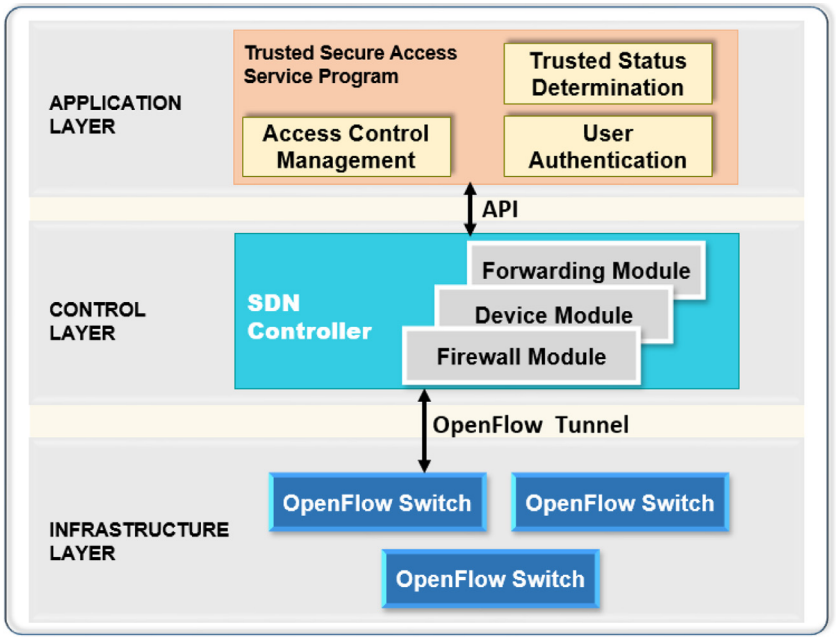

Fig. 3. A trusted access architecture of SDN (SDN architecture perspective).

Access device: Access Requestor

We have added the access client in the access device. The access client will be responsible for providing network access authentication service. Meanwhile, in order to achieve trusted access function, the client can obtain the integrity measurement value of access devices collected from other measurement application. The client can also measure application related to network access, and the measurement value is saved into an idle PCR (Platform Configuration Register). After obtaining the results, the measurement value will be passed to the server for verification.

Access devices have connections at data layer with OpenFlow switches. Therefore, AR and PEP can communicate, and AR can initiate a request to access to the PEP.

OpenFlow (SDN) Switches: Policy Enforcement Point

With combination of features that OpenFlow switches focus on forwarding according to flow table, we set OpenFlow switch as the PEP, executing decisions from the control layer, and application layer. In the beginning of user access, OpenFlow switch will help users to transmit trusted assessment data and identity authentication to the controller. After the

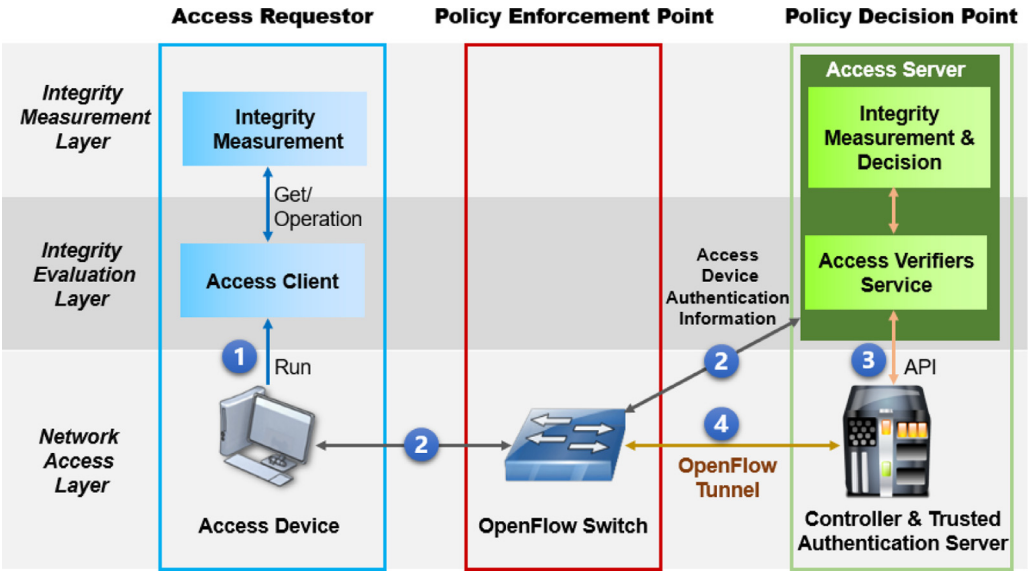

Fig. 4. SDN trusted access procedure.

controller sent a decision, OpenFlow switch will forward normal communication data or restrict communication according to the flow table. OpenFlow switches and controllers have a connection with the management layer, and have the data layer connection with access server side. Therefore, a variety of requirements for information exchange can be achieved between the PEP and PDP.

Controller: Policy Decision Point

The controller manages the network connection, and is the control center for user access request. The customer trusted state judgment and access authentication applications could be implemented by our application in the SDN application layer. This application, as an access server, sends the results to the controller after it completes judgment of the user request. The controller can then issue the flow entries to OpenFlow switch through its own secure channel to complete the control over access users.

3.3. SDN trusted access procedure

This paper presents a process of accessing devices through trusted method to access SDN, as shown in the Fig. 4.

Initial stage of network enabled: Except devices in the whitelist, controllers issue flow entries to OpenFlow switches, blocking network traffic except for certificated ones.

Process of device accessing network:

(1) Access device (physical devices or cloud virtual machines) are ready to access SDN. Users login the access client, which obtains the PCR value of the system as well as the user authentication information. Then, the necessary data for access will be formatted in accordance with the access protocol, during which the corresponding data structures will be generated.
(2) Access client sends formatted access authentication data to access server to complete the request process of network access authentication via OpenFlow switch and data link.
(3) After the access server (Including Access Verifiers and Integrity Measurement & Decision) receives the authentication request, it will conduct an evaluation of integrity measurement of equipment and user identity authentication according to the information, and make decision for the access. Next, the access server transmit access decision to the controller through the API.
(4) After the controller receives the access decisions, it will generate the corresponding flow table entries by device connection and send them to the OpenFlow switches via OpenFlow channel. OpenFlow switch will release or block network access of the access device based on flow table entries.

4. Authentication protocol FOR trusted access

In order to achieve secure and trusted network access in SDN, we designed a new trusted access authentication protocol. The protocol provides the needed data structure, sequence, and encryption during the information exchange stage.

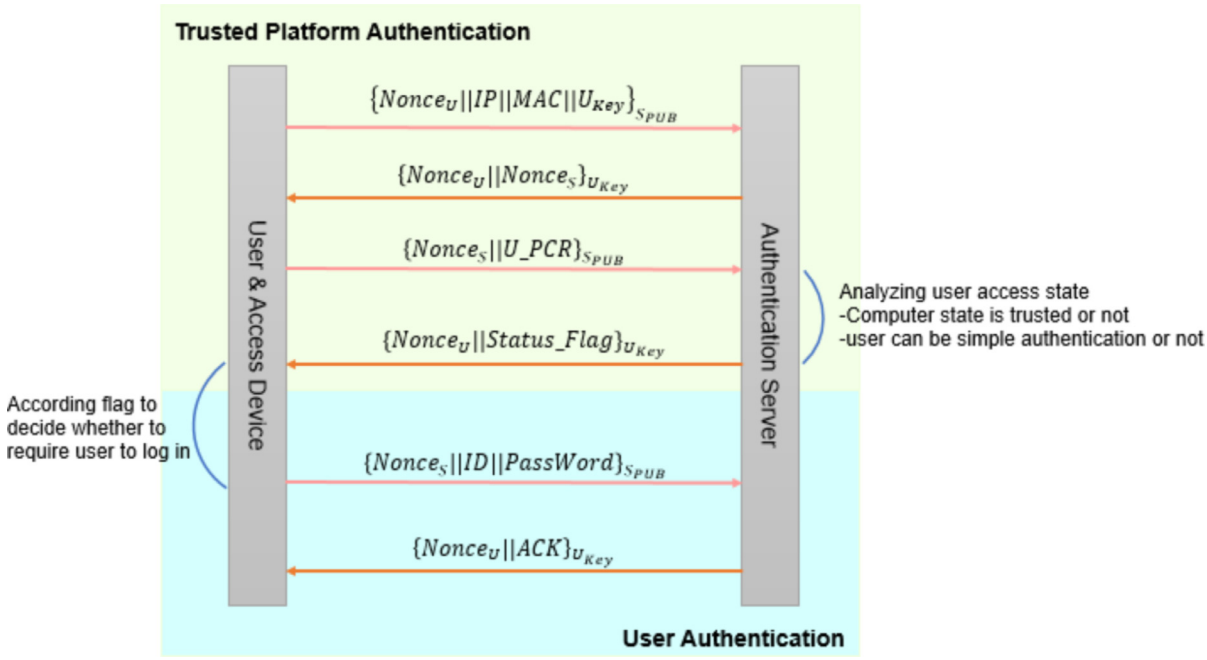

Fig. 5. The authentication process of trusted access protocol.

4.1. Related symbol description

(1) U: request initiator
(2) S: request receiver
(3) U_{Key}: encryption key lodged by the request initiator
(4) S_{PUB}: public key published by the request receiver
(5) $Nonce_U$: Authentication random number generated by the request initiator
(6) $Nonce_S$: Authentication random number generated by the request receiver
(7) ACK: Acknowledgement message
(8) U_PCR: trusted measurement value of the request initiator
(9) ID: access username that the request initiator requires authenticating
(10) PassWord: password of access username that the request initiator requires authenticating
(11) X: data contained in the packet that both sides exchange (plaintext)
(12) Y: format of the data that both sides exchange (such as data structure during transmission)

4.2. Workflow of trusted access authentication protocol

The overall workflow is divided into two stages: the initial platform trusted authentication and user identity verification. The trusted authentication of the platform before the user identity verification can handle the security risk caused by normal users' access to the network by unsafe devices.

Only the trusted authentication devices meet the basic conditions for network access. Otherwise, the users need to access network after removing the unusual environment of that device. The specific authentication process is shown in Fig. 5.

Step 1 Access Device (request initiator) initiates an access request to the authentication server. In the process, it specifies a random number owned by the request initiator to the authentication server. At the same time, the request carries device information (IP address and MAC address) so that the authentication server can correspond to the requesting device. In order to ensure the security of data transmission process, the message contains an encryption key of the access device. The authentication server will use the key for encryption as soon as it replies message to the access device. All of the above information uses public key encryption that authentication server previously disclosed. The formal representation is shown in (1).

$$U \rightarrow S: \quad \left\{Nonce_U||IP||MAC||U_{Key}\right\}_{S_{PUB}} \tag{1}$$

Step 2 Authentication Server (request receiver) verifies the user's access status (such as whether access) and determines whether it is allowed to access the network. If it allows the device to access the authentication on the server side,

it will reply to the access device about the random number verification code of the authentication server for this access request. For security considerations, the authentication server needs to send the random number provided by accessed device when it replies. The above information will use the key appointed by accessed device in step 1 for encryption. The formal representation is shown in (2).

$$S \rightarrow U: \ \{Nonce_U || Nonce_S\}_{U_{Key}} \tag{2}$$

Step 3 When the access device receives a reply from the authentication server, it will first authenticate whether the random number returned by the authentication server is the same as the previously specified number. If they are different, the login will be terminated. If they are the same, it will enter into the transmission and validation step of the trusted information. The access device will call the trusted measurement PCR value stored in the trusted platform configuration register, and sent a packet to the authentication server. When sending data, it will carry the random number from the authentication server side proposed by the authentication server to verify the connection status. All of the above information uses public key encryption that the authentication server previously disclosed. The formal representation is shown in (3).

$$U \rightarrow S: \ \{Nonce_S || U\_PCR\}_{S_{PUB}} \tag{3}$$

Step 3.1 When the authentication server receives the data sent by the access device, it will firstly authenticate whether the random number of the authentication server is the same as the previous number, if they are different, it will disconnect. If they are the same, it will begin to query whether this access device meets the conditions of quick access.

The authentication server will record the recently accessed device in a data structure. The data structures organized into list can record the recently accessed device and their status information, which are named as “the device list of recently connection”. The entry is valid before the expiration time, and invalid to be cleared after the expiration date arrives.

Fast access condition refers to the IP and MAC address of the device in the presence of “the device list of recently connection”, and the present PCR value is the same with that of the last time. Fast access is available if the conditions are met. Fast access means to skip the stage of user authentication and directly permit network access (i.e. the network access procedure ends here, the process of 4–6 is no longer performed). Through the use of SDN features and reliability of the trusted computing technology, quick access can save user access time, reduce network overhead, and enhance the user experience without decreasing security.

If the access device does not meet the conditions of fast access, the trusted status of the user will be judged (evaluated), and general access authentication process will be performed. Trusted judgment results are divided into three categories: fully trusted, partially trusted and untrusted. Fully trusted and partially trusted are referred to as “trusted”.

Untrusted: the key parameters do not match the record.

Partially trusted: the key parameters are in consistent with the record while non-critical parameters are not.

Fully trusted: both key and non-critical parameters are in consistent with the record.

Step 4 After the authentication server completed the judgment, it would feed back results to the access device along with the random number initially specified by the access device to verify this access. The above information will use the key specified by the access device for encryption. The formal representation is shown in (4).

$$S \rightarrow U: \ \{Nonce_U || Status\_Flag\}_{U_{Key}} \tag{4}$$

Step 4.1 After the access device received the reply from the authentication server, it will first judge the random number sent from the authentication server. If the random number is different, the access will be rejected; when the random number is the same, then the judgment result of the authentication server begins to be processed.

The judgment results are divided into: fully trusted, partially trusted, untrusted, and state to meet the conditions for fast access.

When the judgment result is trusted (it contains fully trusted and partially trusted), it will prompt the user trusted authentication results and demand the access device to enter the user authentication phase. When the judgment result is untrusted, it will interrupt the login process and the access device cannot access the network. When the result is in consistent with the conditions for fast access, it will remind users of related information and that the network has been connected.

Step 5 (Optional) After entering into the user authentication phase, the client of access device prompts the user to identify their own username and password. When the user completes the input of authentication information, this information will be sent to the authentication server. For verification, the random number specified by authentication server will also be sent to it. All of the above information uses public key encryption that authentication server previously disclosed. The formal representation is shown in (5).

$$U \rightarrow S: \ \{Nonce_S \parallel ID \parallel PassWord\}_{S_{PUB}} \tag{5}$$

Step 6 (Optional) Authentication server authenticates the username and password provided by the access device, and gives the final result of the judgment. If the result is to allow access, the access state of the device, IP address, MAC address, PCR value and the login username will be written to the device list of recent connections, and expired time of the entry will be specified. At the same time, authentication server opens access device's access permission to the network. Access permissions are opened based on the trusted state information of the device and the setting of the administrator. In principle, fully trusted devices have full network access permission. While partially trusted device can only access public resources (such as Windows Server Update Services, the company forums, etc.). The message replied by the authentication server contains the determined access results and the random number initially prescribed by the access device. The above information will use the key specified by the accessed device for encryption. The formal representation is shown in (6).

$$S \rightarrow U: \ \{Nonce_U || ACK\}_{U_{Key}} \tag{6}$$

4.3. Protocol security analysis

In order to verify the security of this protocol, BAN predicate logic [16] is used to analyze its security. BAN logic does not take into account the implementation of specific cryptographic algorithms in the protocol, and it is effective under the circumstance that subjects involved in the protocol are assumed to be honest. The logic has the characteristics of clear concept, easy application and high abstract degree, which can reveal the security flaws and redundancy in security protocols. However, it also has some limitations. For example, when analyzing security protocols, BAN logic cannot correctly recognize the environment the protocol is facing and fail to make correct judgment, which leads to incorrect or incomplete initial assumptions. Therefore, the improved class BAN logic occurs. The original BAN logic analysis is simpler and more practical. Therefore, this paper makes use of Dolev–Yao (DY) [17] attack model and AVISPA [18] security protocol analysis tool to test the security of the protocol, which can make up the analysis deficiency of BAN logic and validate the security of this protocol.

According to the actual needs of trusted access methods of SDN, we designed the mechanisms to meet the following security objectives:

Goal 1. $S| \equiv U| \sim X$ Authentication server believes the received authentication data X is issued by the access device.
Goal 2. $S \triangleleft S| \sim X$ Only the authentication server receives and understands the authentication data X issued by the access device.
Goal 3. $S| \equiv U| \equiv Nonce_U$ The authentication server believes the random number of the access device, and access device believe server's random number.
Goal 4. $U| \equiv S| \sim X$ The access device believes authentication server have replied the request it sent.
Goal 5. $U| \equiv S| \equiv Nonce_S$ The access device believes authentication server and the random number of the authentication server.
Goal 6. $S| \equiv U\_PCR$, $S| \equiv ID$,$S| \equiv PassWord$ The authentication server trusts user authentication data.

According to the actual application situation of trusted access methods of SDN, we set the following initialization assumptions about the logic BAN:

A1. $U| \equiv \xrightarrow{S_{PUB}} S$ The user believes the authentication server has a server public key.
A2. $U| \equiv U \overset{Y}{\leftrightarrows} S$ Only the access device knows how to communicate with the authentication server.
A3. $U| \sim <X>_Y$ Communications between the access device and the authentication server are in line with the communications specifications.
A4. $S| \equiv U| \Rightarrow U| \sim X$ Authentication server believes users has jurisdiction over sent messages X.
A5. $U| \equiv S| \Rightarrow S| \sim X$ The user believes authentication server has jurisdiction over the sent messages X.
A6. X is equivalent to the plain text part of {} in the formal description.

Based on the initial assumptions of BAN logic and the formal description of trusted access authentication protocol, set security objections are presumed.

(1) Proof of Goal 1: $S| \equiv U| \sim X$
Because $U| \equiv U \overset{Y}{\leftrightarrows} S$, and $U| \sim <X>_Y$.
Combination of message meaning rules can infer $S| \equiv U| \sim X$.
(2) Proof of Goal 2: $S \triangleleft S| \sim X$
Because $U| \equiv \xrightarrow{S_{PUB}} S$, and U→S $\{X\}_{S_{PUB}}$, and $S| \equiv U| \sim X$.
Combination of seeing rules can infer $S \triangleleft S| \sim X$.
(3) Proof of Goal 3: $S| \equiv U| \equiv Nonce_U$
Because $S| \equiv U| \Rightarrow U| \sim X$, and $S| \equiv U| \sim (X)$, and {X} is equivalent to $\{Nonce_U||IP||MAC||U_{Key}\}$, the plain text of U→S.
Combination of belief rules can infer $S| \equiv U| \equiv Nonce_U$.

(4) Proof of Goal 4: $U| \equiv S| \sim X$

Because S→U contains $Nonce_U$, which belongs to $Q \overset{Y}{\leftrightarrows} P$.

According to the rule $(P| \equiv Q \overset{Y}{\leftrightarrows} P, P \triangleleft <X>_Y)| - (P| \equiv Q| \sim X)$ in message meaning rules, we can infer $U| \equiv S| \sim X$.

(5) Proof of Goal 5: $U| \equiv S| \equiv Nonce_S$

Because X can be equivalent to $Nonce_U||Nonce_S$, and $U| \equiv S| \sim X$, and $U| \equiv S| \Rightarrow S| \sim X$.

According to nonce verification rules $(P| \equiv \#(X), P| \equiv Q| \sim X)| - (P| \equiv Q| \equiv X)$, we can infer $U| \equiv S| \equiv \{Nonce_U||Nonce_S\}$.

Then according to the jurisdiction rules $(P| \equiv Q| \Rightarrow X, P| \equiv Q| \equiv X)| - (P| \equiv X)$, we can infer $U| \equiv \{Nonce_U||Nonce_S\}$.

Finally, according to belief rules, we can get $U| \equiv S| \equiv Nonce_S$.

(6) Proof of Goal 6: $S| \equiv U\_PCR$, $S| \equiv ID$and$S| \equiv PassWord$

X refers to $Nonce_U||ID||PassWord$.

Because $S| \equiv U| \sim X$, and $S| \equiv U| \Rightarrow U| \sim X$.

According to nonce verification rules and jurisdiction rules, we can infer $S| \equiv \{X\}$.

That is $S| \equiv \{Nonce_U||ID||PassWord\}$.

Then according to belief rules, we can prove $S| \equiv U\_PCR$, $S| \equiv ID$and$S| \equiv PassWord$.

4.4. Protocol security testing

After the BAN logic analysis, the protocol is tested by using DY (Dolev–Yao) model. The tool used in the test is the AVISPA protocol analysis tool. The description language is HLPSL.

DY model can simulate attacks of packet capture, tampering replay and so on. So the trusted access authentication protocol can use DY model for testing. Combined with the scenarios and content used in the protocol, we set the following security goals:

(1) The PCR value and the username and password used in the user access are secret in the transmission process.
(2) $Nonce_U$ and $Nonce_S$, which are generated by both parties, are secret in the transmission process.
(3) Both communication keys U_{Key} are secret in the transmission process.

In the HLPSL language, the above requirements can be summarized by the following two objectives:

Secret(E,id,S): Declares that the subject S secretly shares the information E, and the secret is defined as a constant id, which is used in the target definition.

Request(B,A,id,E): This is a strong authentication with the meaning of B claiming that B has indeed received information E from A, which will be named a constant id. The id will be used in the target definitions.

The role model is described using alice_bob, which states all the behavior content and status changes of the user and the authentication server.

User model:

```
role alice (A, B : agent,Kb : public_key, Succ : hash_func,SND, RCV : channel(dy))
played_by A
def=
local  State : nat,
       Na,Nb : text,
       IP,MAC: text,
       ID,PASS:text,
       Kab:symmetric_key,
       Flag,ack:text,
       PCR:text
const alice_bob_kab, alice_bob_na, bob_alice_nb, alice_bob_pcr, alice_bob_id, alice_bob_pass: protocol_id
init State := 0
transition
0. State = 0/\RCV(start) =|>
   State':= 2/\Na':= new()/\Kab':=new()/\SND(A.{Na',IP,MAC,Kab'}_Kb) /\secret(Kab',kab,{A,B})
2. State = 2/\ RCV({Succ(Na).Nb'}_Kab) =|>
   State':= 4/\ PCR':=new()/\ SND({Succ(Nb'),PCR'}_Kb) /\ request(B,A,bob_alice_nb,Nb')
4. State=4/\ RCV(Succ(Na).Flag'}_Kab)=|>
   State':=6/\ ID ':=new()/\ PASS':=new()/\ SND({Succ(Nb),ID',PASS'}_Kb)
6. State=6/\RCV(Succ(Na).ack'}_Kab)=|>
   State':=8
end role
```

```
% OFMC
% Version of 2006/02/13
SUMMARY
  SAFE
DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
PROTOCOL
  C:\progra~1\SPAN\testsuite
   \results\auth.if
GOAL
  as_specified
BACKEND
  OFMC
COMMENTS
STATISTICS
  parseTime: 0.00s
  searchTime: 0.03s
  visitedNodes: 1 nodes
  depth:0 plies
```

```
SUMMARY
  SAFE
DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
  TYPED_MODEL
PROTOCOL
  C:\progra~1\SPAN\testsuite
  \results\auth.if
GOAL
  As Specified
BACKEND
  CL-AtSe

STATISTICS

  Analysed : 4 states
  Reachable : 0 states
  Translation: 0.03 seconds
  Computation: 0.01 seconds
```

Fig. 6. The results using OFMC and CL-AtSe of AVISPA test the protocol.

Fig. 7. Implementation order of the access client.

Authentication server behavior (omitting the same part, showing only the transition part):

```
transition
1.State=1/\RCV(A.{Na',IP,MAC,Kab'}_Kb)=|>
  State':=3/\Nb':=new()/\SND({Succ(Na).Nb'}_Kab)
3.State=3/\ RCV({Succ(Nb'),PCR'}_Kb)=|>
  State':=5/\Flag':=new()/\SND({Succ(Na).Flag'}_Kab)
5.State=5/\ RCV({Succ(Nb'),ID',PASS'}_Kb)=|>
  State':=7/\ack':=new()/\ ID ':=new()/\ PASS':=new()/\ SND({Succ(Na).ack'}_Kab)
/\Request (A,B, alice_bob_id,ID') /\Request (A,B, alice_bob_pass,PASS')
```

We use AVISPA protocol analysis tool to test the protocol. The results are shown in Fig. 6. This means the trusted access authentication protocol we designed can meet the expected security goals.

Protocol information such as PCR information, random numbers, session keys and other information will not be leaked in the transmission process. At the same time, when these conditions are correct, it can be judged that the corresponding access device is trusted, and the access permission of the Internet can be given to the access device. The security trusted access method of SDN designed in this paper can meet the network security requirements.

5. Implementation of trusted access method

The security trusted access scheme implemented in this research is mainly composed of two parts: the access client and the access server. The access client runs on the user's access device, and it initiates the access process. The access server is an application, which runs as an authentication server on the controller.

5.1. The access client

The access client runs on devices that require to be accessed. It has two main functions: one is to obtain local information and implement simple trusted state measurement; the other is to complete the authentication work of access network in accordance with the security trusted access method in subscribed ways and data structure to exchange information with the access server side. The function and the implementation order, for which access client are responsible, are shown in the Fig. 7. Access client will pre-include the public key that is required by the network to communicate with the access server to ensure security during data exchange.

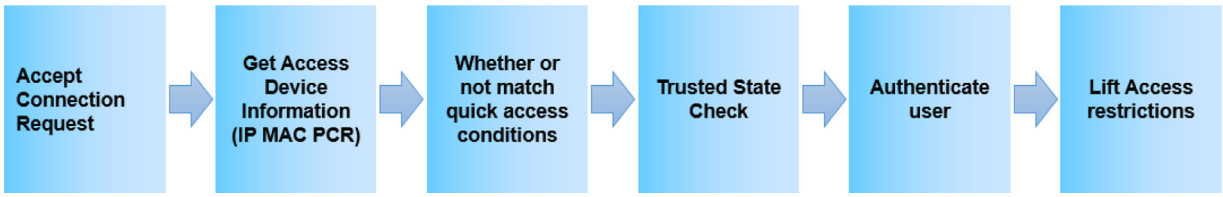

Fig. 8. Implement sequence of access server.

During the access authentication process, the access device needs to submit certain equipment data. At the same time, in order to implement trusted computing technology in the process of login, we need to obtain the information of the current system and prepare data for the follow-up access process and the simulation of trusted measurement. During the login process, the information needed on the access client side mainly includes IP address, MAC address, and other network-related data. In addition, in order to simulate the process of trusted measurement, it also needs to collect the information related to equipment as the source value of the measurement. The host information is obtained by building the system information structure *utsname* and *uname* method.

As the trusted access method requires the support of the trusted measurement results, the trusted measurement function is added to the access client. We use the TPM emulator, assuming that the behavior of the access device it enters the operating system is trusted. Meanwhile, the trusted measurement is only a static measurement, and the metric information is from the results obtained from the system information and the related system files of Linux. The measurement of key trusted information comes from the Linux system files (including the kernel files vmlinuz and initrd). The measurement of the non-critical trusted information comes from the system information collected from the access client during the early phase and the initialization script of Linux system. The metric object is to obtain the digestive value of the corresponding file or get the specific value.

As for connection with the access service, in order to ensure the transmission quality, TCP protocol is adopted. To ensure transmission security, the access client pre-set two different encryption algorithms: AES symmetric encryption and RSA asymmetric encryption. RSA is mainly used to the session key negotiation with the access server for the first time. Then, the AES encrypted data is used for transmission, which can reduce the following-up communication costs. At the same time, since both RSA and AES require raw data, a split / fill function is specially prepared to perform segmentation and padding of the data, as shown in code segment 1.

Code segment 1 data segmentation and padding function of RSA

```
Traversal the original data string by character (when the remaining length > the specified length)
    Record tmp content = string [specified length]
    Change the string [specified length] to add to string terminator: "\ 0"
    The current string is output by the string copy function
    Restore the tmp content to the string [specified length]
    Get start address of the new string = String [specified Length]
Output current string by string copy function (last)
Fill the insufficient length
Return the function
```

5.2. The access server

Consider the SDN application layer. The access server is deployed on a physical server where the controller is located, and the server acts as the authentication server as well. The access server, as part of the application layer in the SDN architecture, can communicate with the access client (all initiated by the client) and handle the user's access request. When there is no user request, it will check the status of the logged-on device, find and remove the passively disconnected access device. Its main operational functions and sequence are shown in Fig. 8.

When the user sends access request and provides relevant platform information and trusted information, the program will first determine whether the user can quickly access to the network. If so, no user authentication is required. If the user does not meet the fast access conditions, the access server will check the trusted information and the reliability of the access device. The determination of the trusted condition is shown in code segment 2.

Code segment 2 Trusted State Determination

```
Flag_trusted state = 0
Traversing the Trusted State Library:
    If the key parameter is untrusted, Flag_trusted state = 0 ends the traversal
    If the key parameter is trusted, Flag_trusted state + 1
    If the non - critical parameter is trusted, Flag_trusted state + 1
    Returns the trusted state (0: not trusted 1: partially trusted 2: fully trusted)
```

If the trusted requirements are met, further user authentication is required. After the authentication succeeds, the access restriction function is invoked to release the network access restriction of the access device. If the trusted status is judged as non-trusted, the connection is immediately interrupted.

The access restriction function designed in this research is that the access server sends commands to the Floodlight firewall module to restrict unauthorized users and devices from accessing network. Access restrictions are mainly divided into the initial start of the access server and authentication process. In the initial startup, the access server will turn on the Floodlight firewall function through the controller API, and then deploy the default management items. In addition to the whitelisting devices, other devices are not allowed to access networks and services except for the authentication server (shown in code segment 3).

Code segment 3 access restriction code

```
Initial rule:
    Allows users in the specified IP address segment to access the authentication server's authentication port
    Prohibits user from specifying the IP address segment to access the network
    Allows other IPs to pass through the switches (indicated by the switch ID)
Push the rule to the Floodlight controller
```

When the user completes the authentication, the access server informs the Floodlight controller of the result, de-restricts access to the access device based on the trusted state, releases device's access permission to the network, records the device's connection information to the device list of recent connections, and will give an expiration time based on administrator settings. When the expiration time is reached, the application will query the controller if the device is online. If it is already offline, the corresponding entries in the table will be deleted.

For access devices with different trusted states, different access rights are given. The current defaults are: the fully trusted devices have full network access, and partially trusted devices have administrator-defined network access permissions (typically less than fully trusted). As is shown in code segment 4.

Code segment 4 setting different network privileges depending on the level of trusted status

```
Access rules
If fully trusted:
    Allows users with specified IP and MAC to access the network (range is large)
If partially trusted:
    Allows users with specified IP and MAC to access the network (range is small)
```

6. Experimental verification and analysis

6.1. Experimental environment

The hardware environment consists of one access device, two servers for access, and a network of OpenFlow switches and controllers. The access device has an i5 dual-core CPU and 2 G memory. Server is with i5 quad-core CPU and 4 G memory.

The operating system of access device is the Ubuntu 14.04 LTS, equipped with a virtual trusted root TPM_ Emulator, trusted software stack TSS and the OpenSSL encryption library required for running. The server's operating system of the installation of controller software is Ubuntu 14.04 LTS, and the controller version is Floodlight 1.0. The server also installs the trusted access server program designed in this paper as the authentication server.

The network topology is shown in Fig. 9. The access device's IP is 10.0.1.10/24, connect to OF switch 1. Two servers connect to the OF Switch 2, setting different IP address segment, to simulate the two servers in different networks.

6.2. Experimental scenario 1

The security trusted access procedure in the SDN designed in this research is adopted on the access device and the authentication server. In experiment scenario 1, the access device has a fully trusted operating environment, where the device environment and the system environment are all trusted. The user of the operating device has a valid username and password. The access device sends ICMP packets without the need of reply to both servers at a rate of 25kB/s. Figs 10 and 11 show the traffic of the three devices. The before-the-green line time period in the Fig. 10 is the process of the access device completing the access authentication. As can be seen, before the green line moment, the access device sends data continuously, while the two servers failed to receive them in Fig. 11. After the green line moment, the access device passed the authentication and was determined as having fully trusted environment, the access device has full access permission to the network. The two servers can receive the data sent by the access device.

6.3. Experimental scenario 2

The access device has a partially trusted environment. The device environment is trusted while the system environment is untrusted. The user of the operating device has the same network access username and password as those in the scenario

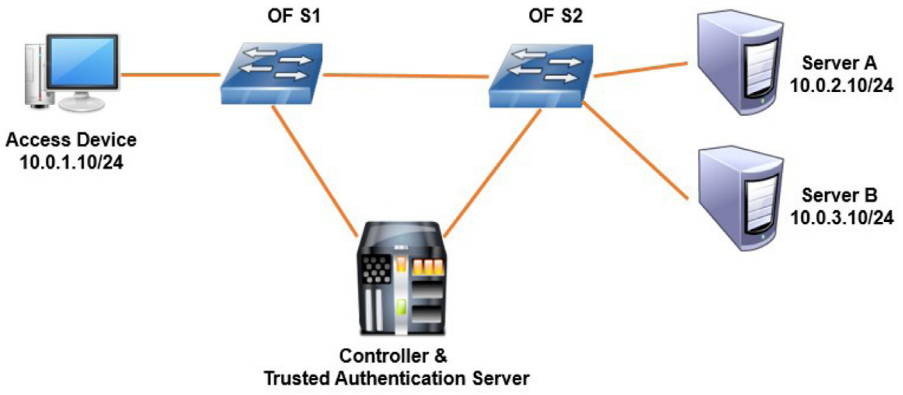

Fig. 9. Network topology of experimental environment.

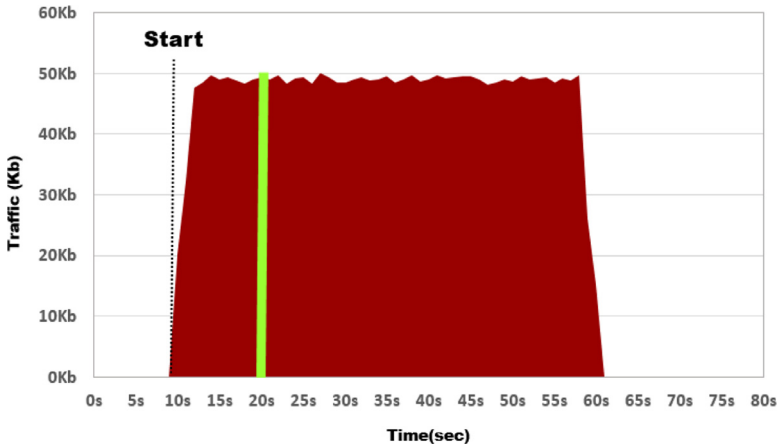

Fig. 10. Traffic information of the access device.

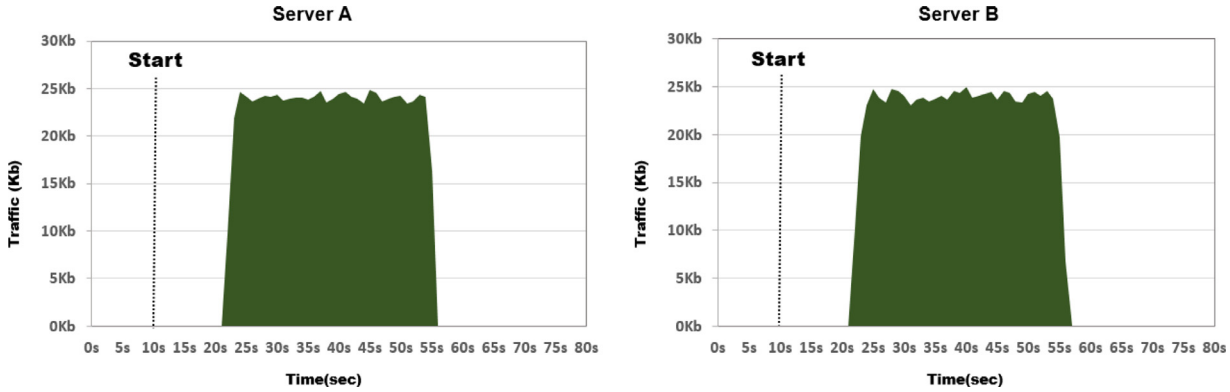

Fig. 11. Traffic information of Server A and Server B.

1. The access device also starts to send data to both servers from the access network. Figs. 12 and 13 show the traffic of the three devices during this experiment. The before-the-green line time period in the Fig. 12 is the process for the access device to complete the access authentication. It can be seen that before the green line, the access device continuously sends data, but neither server receives it. After the green line time, the access device completes the access authentication and is assumed that it has partial trusted environment. Due to the partially trusted environment, the access device has partial access permission to the network (the administrator has set partial permissions that the device can access 10.0.3.0/24). At this point, the server 10.0.3.10 can receive data from the access device, but the access device cannot communicate with the higher security level server 10.0.2.10.

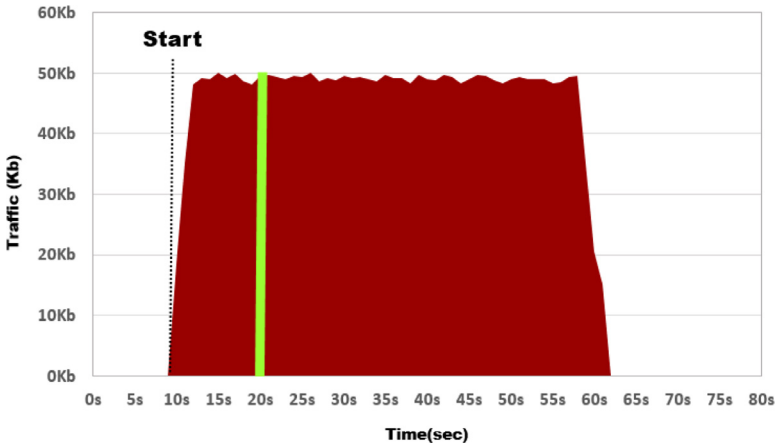

Fig. 12. Traffic information of the access device.

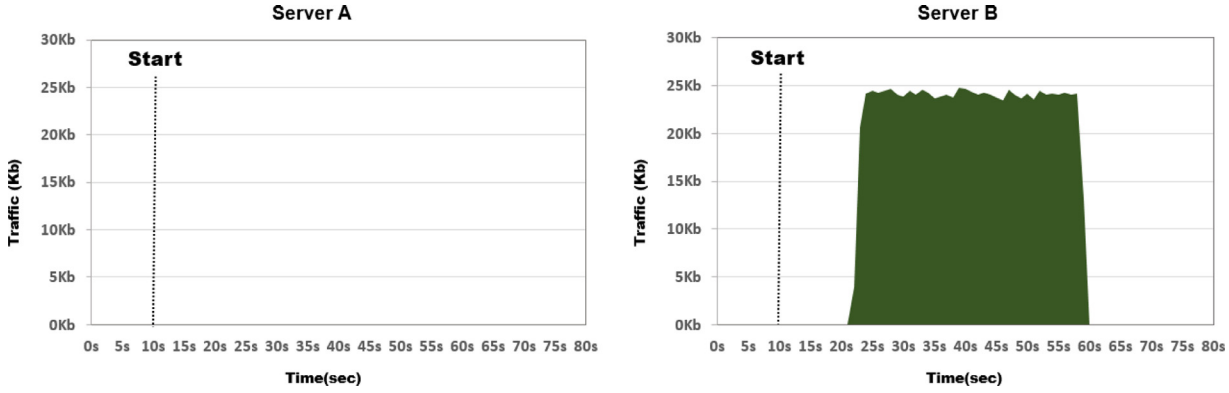

Fig. 13. Traffic information of Server A and Server B.

Through the three different experimental scenarios, it is shown that the trusted access method of SDN designed in this paper can complete the judgment of the credibility of the access device and the user identification. It also provides users of different credible situations with different network access permissions.

6.4. Performance testing

In the performance test, we increase the number of access devices. Under the situation that different number of devices access the network at the same time, we compared to use trusted access method designed in this paper with not to use. The delay time used by this device whose IP is 10.0.1.10/24 to connect server A is shown in the Fig. 14. The load of the server with the controller is shown in the Fig. 15.

It can be seen from the experimental results that the latency of the device access network is increased after adopting the method designed in this paper, but the increased delay time is within the acceptable range. The CPU utilization of the server having the controller does not change significantly. It is proved that the design feasible and effective.

7. Conclusion and outlook

Based on the characteristics of TNC architecture and SDN architecture, we designed and implemented a security trusted access method in software defined network. The method included the structure design of access method and a new security access authentication protocol. We formalized description and logical reasoning that proved the security access authentication protocols. We also had attack testing through the protocol analysis tool to prove that the protocol can meet the expected security goals.

The experimental results showed that this method could provide effective security authentication service and could provide different access permissions according to the different conditions of the access devices. Compared with the access mode based on IP address, MAC address and user identity only, the proposed access method is more secure.

The method designed in this paper still has some limitations. User management in the authentication server is relatively simple. In the future, we will connect with the Radius server or set up a separate database, so we can carry out more

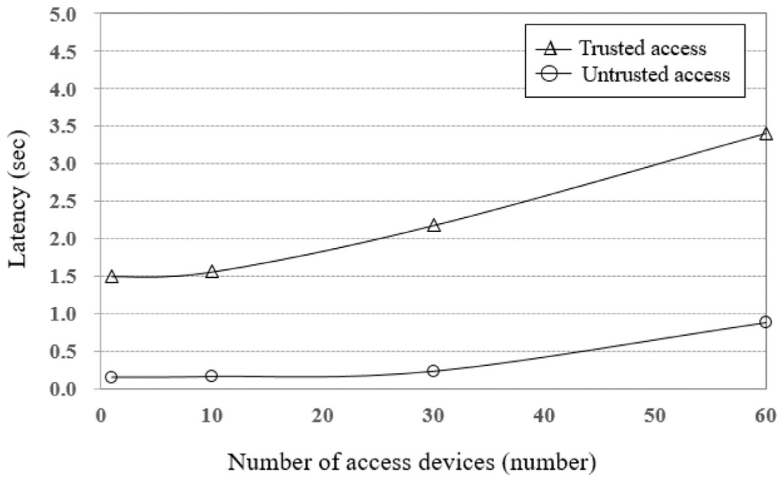

Fig. 14. Variation in latency.

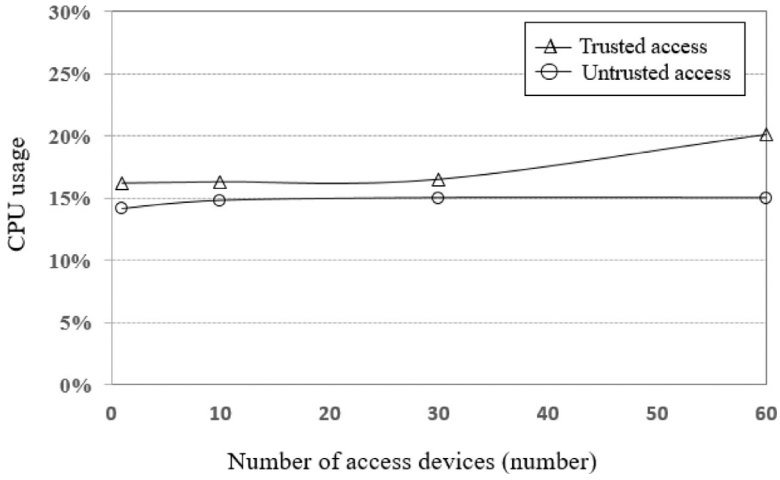

Fig. 15. Variation in CPU usage.

precise user management. In addition, the trustworthiness of the access device is relatively complex to manage in a large-scale network. We intend to scale our method to better handle complex systems.

Acknowledgments

This research partly sponsored by the Funding Project for Beijing Natural Science Foundation (4162006), Open Research Fund of Beijing Key Laboratory of Trusted Computing.

References

[1] S. Scott-Hayward, S. Natarajan, S. Sezer, A survey of security in software defined networks., IEEE Commun. Surv. Tut. 18 (1) (2015) 623–654.
[2] S. Jacobs, Security Management of Next Generation Telecommunications Networks and Services., John Wiley & Sons, 2013.
[3] J. Zhu, Web and Radius-based Access Authentication System for Wireless LAN, Southwest Jiaotong University, 2006.
[4] TCG Group, TCG specification architecture overview., TCG Spec. Rev. 1 (2007) 1–24.
[5] Trusted Computing Group. TPM Specifications v1.2. 2011.
[6] Y. Lin, The Research and Implementation of Network Access Control System., University of Electronic Science and Technology of China, 2014.
[7] A. Luo, C. Lin, Y. Wang, et al., Security quantifying method and enhanced mechanisms of TNC., Chin. J. Comput. 05 (2009) 887–898.
[8] F. Yan, J. Ren, K. Dai, et al., Design and implementation of secure authenticated protocol based on TNC., Comput. Eng. 12 (2007) 160-162+165.
[9] E. Liu, Research on the Identity in Heterogeneous Wireless Integrated Networks., PLA Information Engineering University, 2009.
[10] P. Porras, S. Shin, V. Yegneswaran, et al., A security enforcement kernel for OpenFlow networks, in: Proc. of the 1st Workshop on Hot topics in Software Defined Networks, Helsinki: ACM, 2012, pp. 121–126.
[11] T. Sasaki, Y. Hatano, K. Sonoda, et al., Load distribution of an openflow controller for role-based network access control, in: Proceedings of Network Operations and Management Symposium (APNOMS), 2013 15th Asia-Pacific, Hiroshima, Japan, 2013, pp. 1–6.
[12] D.M.F. Mattos, O.C.M.B. Duarte, AuthFlow: authentication and access control mechanism for software defined networking., Ann. Telecommun. (2014) 1–9.

[13] S. Kinshita, T. Watanabe, J. Yamato, et al., Implementation and evaluation of an OpenFlow based access control system for wireless LAN roaming, in: Proceedings of the 36th Annual IEEE International Computer Software and Applications Conference Workshops Izmir, IEEE Computer Society, 2012, pp. 82–87.
[14] J. Matias, J. Caray, A. Mendiola, et al., FlowNAC: flow-based network access control, in: Proceedings of the 3rd European Workshop on Software-Defined Networks, Budapest: IEEE, 2014, pp. 79–84.
[15] I. Bente, J. Vieweg, J. von Helden, Privacy enhanced trusted network connect, in: International Conference on Trusted Systems, Berlin Heidelberg, Springer, 2009, pp. 129–145.
[16] M. Burrows, M. Abadi, R.M. Needham, A logic of authentication, Proc. R. Soc. London A, The Royal Society 426 (1871) (1989) 233–271.
[17] D. Dolev, A. Yao, On the security of public key protocols, IEEE Trans. Inf. Theory 29 (2) (1983) 198–208.
[18] A. Armando, D. Basin, Y. Boichut, et al., The AVISPA tool for the automated validation of internet security protocols and applications, in: International Conference on Computer Aided Verification, Berlin Heidelberg, Springer, 2005, pp. 281–285.

Jing Liu Ph.D. candidate from Beijing University of Technology now, M.S. degree from Beijing University of Technology in 2006, B.S. degree from Beijing University of Technology in 2002. Her research interest covers information network security and trusted computing.

Yingxu Lai Received Ph.D. from Chinese Academy of Sciences in 2003. Now she is a professor at the College of Computer Science, Beijing University of Technology. Her research interest covers information network security and trusted computing.

Zipeng Diao Pursuing his maser degree of computer science in Beijing University of Technology. Received his B.E. degree from same university in 2016. His research interests include information security, trusted computing and cloud computing.

Yinong Chen received his Ph.D. from the University of Karlsruhe (KIT), Germany in 1993. He did postdoctoral research at Karlsruhe and at LAAS-CNRS in France. From 1994 to 2000, he was a faculty at the Wits University, Johannesburg, South Africa. He joined the faculty at Arizona State University in 2001. His research areas include computer science education, service-oriented computing, autonomous decentralized systems, and embedded systems. He (co-) authored 10 books and more than 200 research papers.