Big Data Analysis-Based Secure Cluster Management for Optimized Control Plane in Software-Defined Networks

Big Data Analysis-based Secure Cluster Management for Optimized Control Plane in Software-Defined Networks

Jun Wu, Mianxiong Dong, Kaoru Ota, Jianhua Li, and Zhitao Guan, *Member*, *IEEE*

***Abstract*—In software-defined networks (SDN), the abstracted control plane is its symbolic characteristic, whose core component is the software-based controller. The control plane is logically centralized, but the controllers can be physically distributed and composed of multiple nodes. To meet the service management requirements of large-scale network scenarios, the control plane is usually implemented in the form of distributed controller clusters. Cluster management technology monitors all types of events and must maintain a consistent global network status, which usually leads to big data in SDNs. Simultaneously, the cluster security is an open issue because of the programmable and dynamic features of SDNs. To address the above challenges, this paper proposes a big data analysis-based secure cluster management architecture for the optimized control plane. A security authentication scheme is proposed for cluster management. Moreover, we propose an ant colony optimization approach that enables big data analysis scheme and the implementation system that optimizes the control plane. Simulations and comparisons show the feasibility and efficiency of the proposed scheme. The proposed scheme is significant in improving the security and efficiency SDN control plane.**

***Index Terms*—Software-defined networks, big data, swarm computing, security, cluster management.**

I. Introduction

With the rapid development of modern society and innovations in technology, computer networks and information technology in general have become an important part of the infrastructure and now play a significant supporting role in business and everyday life. After decades of development, network technology has made remarkable progress in terms of coverage, node capacity, transmission speed, and operational cost. The creation of the basic communication architecture based on the core TCP/IP protocol represented a fundamental advancement [1][2]. As software and hardware technologies have developed and begun to cooperates with each other, an increasing number of network services have gradually appeared. Such services include wireless networks, mobile Internet, virtualization and cloud computing, which is aimed at meeting the rapidly increasing demands of network traffic, in addition to supporting different types of network equipment and diverse network applications [3][4]. However, existing network technology is faced with increasingly serious technical challenges due to size, performance and security issues, all of which play large roles in enterprise data centers and are important to network operators. After years of development, the traditional network architecture is bloated and ossified. Therefore, it is difficult for traditional networks to address application scenarios such as big data and cloud computing, which are characterized by massive data transmission and flexible resource demand. The result is frequent bottlenecks [5][6][7].

The symbolic characteristic of SDN is the abstracted control plan, at the core of which lies the software-based controller [8][9][10][11]. The control plane is logically centralized, but the controllers can be physically distributed and composed of several nodes [12][13]. By centralizing the logic for network control and providing call interface to the application layer, SDN makes the network become allocable resources, elevates network capacity, and improve the flexibility of managing network resources [14]. The dynamic and flexible control aspects of SDN provide large benefits, but they also raise some limitations. First, programmable features introduce additional vulnerabilities into SDN, and malicious actions easier. Second, management and monitoring requirements of SDN are higher and they demand a flexible and controllable traffic management mechanism when used for data transfer purposes. Third, cluster management technology must monitor all types of events and maintain a consistent global network status, leading to big data in SDNs, which are helpful for optimizing SDN control. To address abovementioned challenges, it is necessary to realize a secure and efficient cluster management mechanism based on big data, which is the main contribution of this paper.

In SDN, the control plane is responsible for network status monitoring, link discovery, policy development, and the generation as well as management of the forwarding table. In

This work was supported in part by the National Natural Science Foundation of China under Grant 61431008, 61571300 and partially supported by the JSPS KAKENHI Grant Number JP16K00117, JP15K15976, KDDI Foundation.

Jun Wu and Jianhua Li are with the School of Cyber Security, Shanghai Key Laboratory of Integrated Administration Technologies for Information Security, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: junwuhn@sjtu.edu.cn, lijh888@sjtu.edu.cn).

Mianxiong Dong and Kaoru Ota are with the Department of Information and Electric Engineering, Muroran Institute of Technology, Muroran 050-8585, Japan(e-mail: mx.dong@ieee.org, ota@csse.muroran-it.ac.jp).

Zhitao Guan is with School of Control and Computer Engineering, North China Electric Power University (NCEPU), Beijing 102206, China (e-mail: guan@ncepu.edu.cn).

Corresponding author: Mianxiong Dong

addition, the control plane interacts with the forwarding plane through the southbound interface and services management program through the northbound interface to realize advanced network management functions. Network state monitoring involves two aspects of network state monitoring. The first is real-time traffic monitoring in the network, which enables timely detection of network congestion, providing a basis for the adjustment of the network strategy. The second is that because a controller usually controls more than one piece of network equipment, it needs to detect abnormal situations, such as abnormal node connections, quickly enough to adjust the network topology structure. In the OpenFlow protocol, a flow forwarding table is produced by the controller but stored at the switch. Therefore, a single controller cannot know all the routing information in real time, and the existing link discovery protocol must be modified to adapt to the SDN environment. The core function of an SDN is policy formulation. Since the underlying network information is opaque to the application plane, the network strategy specified by the controller is directly related to the network service quality. Service policy formulation includes generating, deleting or updating flow tables to optimize the routing information, for example, increasing or deleting switch nodes based on network traffic. Initially, SDN was prototype platform deployed in the campus network environments, where a single controller node is able to meet the needs at that time. However, after the scale of the network expands to a certain extent, because the computing ability of a single node is limited, it is impossible to for a single node to manage an unlimited amount of network equipment; consequently, a cluster controller composed of multiple controller is needed [15]. If a number of controllers cooperate with each other, it would be possible to work around single node failures and other types of accidental network failures, thus enhancing network stability. For these reasons, research on cluster control technology has become a hotspot during the commercialization phase of SDN [16]. In addition, the control plane plays an important role in SDN security. In the SDN architecture, a controller can obtain a global view of the network, thus, the controller can both control the network centrally and enhance network security. This capability has sparked some improvements and new solutions for protecting the network security.

One of the main bottlenecks of SDN applications is the extensibility of the control plane. A centralized architecture makes global information available to the controller and provides flexible control capability based on programmable interfaces. However, because the number of nodes and traffic flow in networks tend to increase, SDN has limitations that include restrained processing power, long control latency and low control plane reliability. In fact, cluster management technology must monitor all types of events to maintain a consistent global network status, which leads to big data in SDN. Moreover, when moving from static to dynamic services, there are corresponding increases in resource requirements; application server clusters must move from a single cluster in a static environment into multiple cluster nodes in a dynamic environment. The functionality provided to each user varies dynamically with the user's business requirements. Therefore, dynamic services can generate large amounts of data. These big data can support the optimized cluster in SDN. In fact, there are different equipment interfaces in a controller that can map to different network types. Then, these different types of networks can be controlled and configured through the abstract interface in the controller. In other words, a common cluster management scheme is proposed that can manage multiple different types of networks. There are two challenges in creating big data analysis based clusters. On one hand, an efficient big data analysis algorithm is a must. In large-scale SDN scenarios, the control plane is usually implemented in the form of distributed controller cluster. It is necessary to propose an efficient and feasible algorithm for the controller cluster based on the analysis of the dynamic and massive data. On the other hand, various security problems occur when a controller cluster is created[17]. To address the above challenges, this paper proposes a big data analysis based secure cluster management architecture for an optimized control plane.

The remainder of this paper is organized as follows. Preliminaries are discussed in Sect. II. The cluster management requirements are described in Sect. III. In Sect. IV, basic architecture is provided. In Sect. V, the details of the proposed big data analysis based secure cluster management are presented. Sect. VI presents the implementation system. Sect. VII discusses the simulations and evaluations. Finally, Sect. VIII concludes this paper.

II. Related Works

A. SDN and Cluster

Because of the limited computing power of a single controller node, when computing operations is high, the controller node consumes considerable battery power and may become unavailable. To resolve this problem, the following approaches can be considered. First, a larger SDN network must to be divided into several smaller network domains. Each controller node is responsible for managing a network domain, the adjacent domain controllers need to able to communicate with each other to update the global network state and execute the global strategy. Second, cluster controllers consisting of multiple controllers are needed. Meanwhile, when controllers cooperate with each other, it is possible avoid single controller node failures, communication failures and other kinds of accidental network failures, which enhances the network stability. It is highly important to divide the network, and to define the communication rules between the controllers. These problems have become frequent topics in the research on SDN cluster control technology [18][19].

Consider an extreme case as follows, if each controller node has a one-to-one correlation with each device in the network, the SDN can be viewed as a network in the traditional architecture. As shown in Fig. 1, multiple parameters must be balanced during the division of the network domain, including the controller load capacity, communication delay between controllers and devices, communication delay between controllers, scalability, fault tolerance rate, and load balancing.

B. *Big Data Framework*

Big data refers to data collection within a certain time range, managed and processed by conventional software tools [20][21]. The purpose of big data analysis is to achieve better decision making, insight into the mass forces and process optimization, support a high growth, and the diversification of information assets. The main features of big data include volume, velocity, variety, value and veracity. Several big data tools are available, such as Hadoop [22][23], MapReduce [24], and the Dryad of Microsoft Neptune of Ask.com [25]. Of the above tools, the most popular is Hadoop. Additionally, MapReduce is a popular distributed big data analysis tool designed by Google. The aforementioned big data tools can be applied to enhance the efficiency and scalability of service recommendations in big data applications.

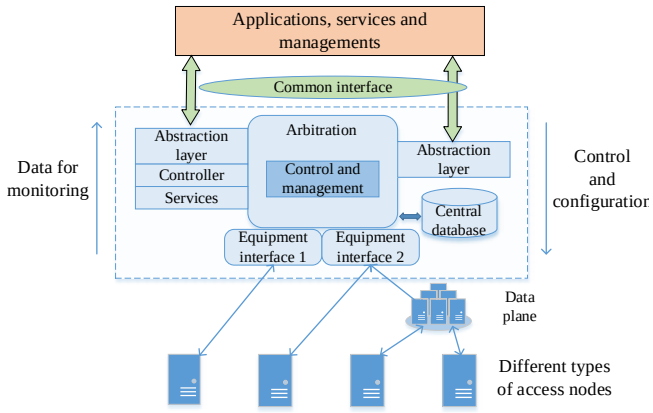


Fig. 1. Controller on different types of networks in SDN.

C. *Ant Colony Algorithm*

Ant colony algorithm is usually used in data mining and analysis [26][27]. Additionally, it has been adapted for some cluster schemes in communication and network systems[28][29]. The basic idea behind the ant colony algorithm includes the following features. First, the pheromone intensity and the taboo table of the ants will be pre-initialized. Second, each ant will choose a node destination based on to certain probability rules and the rules in the taboo table, until the final formation of a legitimate path. Third, the path length generated by each ant will be calculated, which consists of the sum of the lengths of the edges in the path. Fourth, the pheromone level on each edge is updated. Each edge elicits ants to release the pheromone; then the pheromone released by the ants according to the length of the paths generated by the ants is calculated. Fifth, when all the ants have updated the pheromone, the current shortest path is recorded, the taboo table is initialized and a pheromone is added to each ant. Control returns to the second step. The above five steps loop until the termination condition of the algorithm is satisfied.

III. Requirements Analysis of Cluster Management

Based on the hierarchical relationships among the controller nodes, SDN clusters can be divided into two types: vertical SDN clusters and horizontal SDN clusters. A vertical SDN cluster has a subordinate relationship between the controllers. This hierarchical structure can be reflected in the realization of the controller functions. In this type of cluster, network tasks are assigned to different controllers based on the amount and frequency of required information. Generally, more frequent tasks will be handed to the switch that is topologically closer to the controller. On one hand, this approach reduces the delay caused by transmission between layers. On the other hand, higher level controllers correspond to more switches, which can avoid excessive loads in the high-level controller. Tasks that require global network information such as routing requests, will be executed by the root controller. There are several existing vertical cluster schemes. Kandoo is a 2-layer SDN model [30], in which there is neither traffic between every controller nor information about the global network state in the bottom layer. Logically focused, the controllers in the upper layer are responsible for maintaining the global network state. Only local programs can run in the controllers in the bottom-layer controllers, which require information from only a single or a few switches. Because the majority of frequent requests are conducted using this approach, it becomes easier for the controllers in the upper layer to proceed. DIFANE changes the passive control mode for the control plane to accept queries for routing of switches, which defines two kinds of switches of the control plane [31]. One involves ordinary switches, which possess only the ability for data inquire and transfer; the other involves authoritative switches, which contain the higher-level information for ordinary switches to make inquiries. The control plane refreshes the transfer strategy and offers it to the authoritative switches each time packages transfer, ensuring there are sufficient resources in the transfer plane. In this way, packages proceed in the control plane without experiencing a delay between the transfer plane and the control plane, and the calculation of the controllers is reduced, which obviously enlarges the maximum amount of equipment. DevoFlow is recommended based on OpenFlow; it reduces the traffic between controllers and switches in two processes: the foundation of forwarding streams and the collection of network information [32].

Unlike a vertical SDN cluster, there are no subordinate relationships between controllers in a horizontal SDN cluster. Instead, each controller knows a part of the network state and implementing of global functions requires integrating the information of multiple controllers using a specific protocol. The entire network is divided into different control domains that do not normally intersect. Each controller manages some physically adjacent switches. On one hand, the horizontal structure divides the control domain geographically, which reduces communication delays between switches and controllers. On the other hand, the independence of each control domain makes it convenient to perform local management strategies. There are several types of horizontal clusters for SDN. DISCO is specially designed for DFS, in

which each controller supervises its own network field and offers end-to-end network services by corresponding with other controllers [33]. The connection between controllers based on a swift channel protocol can collect and share information adaptively. HyperFlow is a distributed control plane based on event triggers [34]. Besides the advantages of centralized control, HyperFlow enhances the extensibility of SDN. HyperFlow actively synchronizes network information of controllers to ensure that every controller has access to the global network state to address complicated requirements. At the same time, HyperFlow significantly reduces the delays between switches and controllers. Onix is a global control platform that runs distributed server. Controllers obtain messages from switches, estimate the network state, constitute the network protocols and issue commends to switches. The key idea in Onix is to maintain global network state by NIB (Network Information Base), which has a layered topology [35]. Its extensibility is determined by the dispensing strategy of the network state and the communication benchmark of distributed controllers.

By centralizing the logic of network control and providing a call interface to the application layer, an SDN turns the network into an allocable resource and elevates the network capacity while making the management of network resources more flexible. Some specific application scenarios have higher requirements of network management and monitoring. This demands a flexible and controllable traffic management mechanism when using an SDN for data transfer.

To address abovementioned deficiencies, it is necessary to realize a traffic management mechanism that can perceive the identity of an application and allow the application to specify the network service requirements. These application aware clustered traffic management modules should have the following technical features:

1) The application should set specific required network parameters directly, including bandwidth, priority, network permissions, and so on.

2) The network control layer should be able to identify traffic-generating application to develop an optimized forwarding strategy.

3) The application should apply statistics to the network data, test the quality of service, and adjust the network parameters.

IV. Basic Architecture

The security cluster is composed primarily of the following modules: cluster secure authentication, big data analysis-based clustering, and a clustering job pool. At the first job execution, considering the local task and network bandwidth, the master node of each cluster is assigned to the job, and the node queue is added to the cluster job pool. The data from different clusters is collected and stored in distributed storage system using Hadoop architecture. Hadoop Distributed File System (HDFS) is widely used in big data systems. In the proposed scheme, the collected big data is stored on the basis of HDFS.

The basic architecture of the proposed security cluster mechanism is shown in Fig. 2. Firstly, cluster authentication is performed when an SDN controller joins a cluster; this ensures only legitimate SDN controllers are included. Second, big data analysis can provide a clustering policy for the SDN controller. In this study, the ant colony algorithm is adapted to perform the big data analysis. The initial status calculation module uses TaskTrackers to record the running status. Here, TaskTrackers is a collection of elements of TaskTrackerStatus. The data replica query is realized based on the communications among the NameNode, which supply backup of the input data for the currently scheduled job. The cluster job assignment module performs policy implementation based on the results of big data analysis and returns the cluster response to the SDN controller.

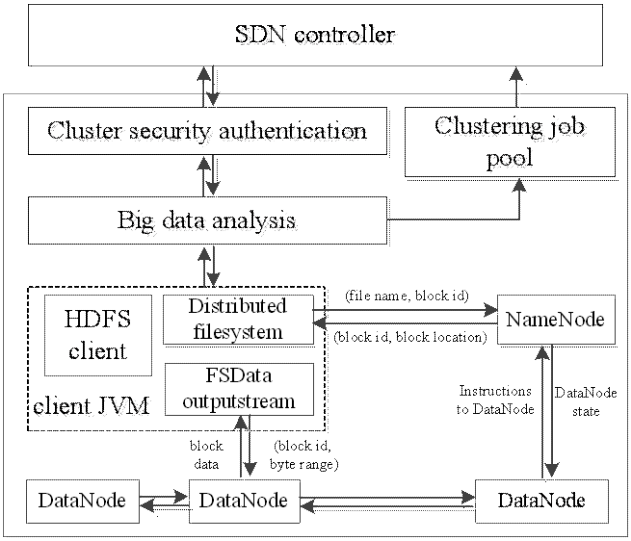


Fig. 2. Basic architecture of the proposed security cluster mechanism.

V. Proposed Security Cluster Management

A. Secure Authentication for Cluster Control

To secure the cluster control for control plane in SDN, secure authentication is imperative when a controller joins the cluster in SDNs. As an existing security token, username/password is feasible to propose related authentication protocols. However, this security token alone does not enough security protection for the authentication. In other words, additional protections such as confidentiality, integrity, and nonreputation cannot be provided. To adapt the username/password token for this authentication, a hash function and a message authentication code (MAC) are introduced to improve security during clustering.

There are two phases in the secure authentication for clustering in control plane: “credential generation” and “credential authentication”. The notations and their meanings as used in the proposed authentication protocol are listed in Table I. In the proposed authentication protocol, a hash-based MAC is used as the MAC technology. The lightweight computation in hash reduces the complexity of the authentication protocol. In fact, the hash based MAC is a proven cryptology algorithm. In this paper, the MAC value is computed based on $MAC(Key, M)=H(Key+M+Key)$.

The principle by which an SDN controller node generates a specific signature is shown in Table 2. First, to generate the credential, a fresh nonce N_C is generated by the controller *Con*. Next, *Con* computes a digest value of its own password as well as the fresh nonce using the hash function. Note that both the controller and the master node of the cluster can store the password. Clearly, the credential can achieve nonrepudiation, because only the controller itself can generate the credential. Meanwhile, the fresh nonce ensures the freshness. Then, a security token *ST* is generated by *Con* based on the username/password. In addition, to improve the defense capability against replay attacks, created time and nonce are added. Then, the key of MAC from *C*'s Password and *salt* is generated by *Con*. In the proposed authentication protocol, *ClusteringRequest* is used to denote the original access request message. Next, a confidentiality token for *ClusteringRequest* is generated by *Con*. Subsequently, the *ST* and the credential are sent to the master node in the cluster by the controller.

TABLE I
NOTATIONS USED BY CLUSTER AUTHENTICATION PROTOCOL

Notation	Meaning
Con	Controller node in the SDN
M	Mater node which is in charge of the cluster of the control plane of SDN
N_A	A nonce generated by entity *A*, which is a random value to defend replay attacks
Key	The key of hash based MAC algorithm
T	The created time of a message when the controller join a cluster
Salt	128-b randomized valued generating the key for symmetric cryptography
Hash(*E*)	Computes the digest of message *E* based on hash algorithm to calculate
MAC(*E, Key*)	Uses *Key* to compute MAC for message *E*

TABLE II
CREDENTIAL GENERATION

Credential Generation:
1. A fresh nonce N_C is generated *by Con*
2. *Con* calculates *PasswordDigest*=*H*(*T*+R_C+*Password*)
3. *Con* generate a security token *ST*=(*ControllerID, PasswordDigest, T*, N_C, *salt*)
4. *Key*=*Hash*(*Password*+*salt*) is calculated by *Con*
5. *Con* calculates the credential of *ClusteringRequest* as *MAC*(*Key, ClusteringRequest*)

TABLE III
CREDENTIAL AUTHENTICATION

Credential Authentication	
Con	*M*
ST, *MAC*(*ClusteringRequest, Key*)	
1. send clustering request →	
	2. *M* derives *PasswordDigest, T*,N_C, *salt* from *ST*
	3. *M* computes *PasswordDigest' =H*(*T*+R_C+*Password*)
	4. *M* authenticates *PasswordDigest*
	5. *M* compute *Key* based on credential generation
	6.*M* computes MAC value of *ClusteringResponse*
← *MAC*(*ClusteringResponse, Key*)	
7. *M* send clustering response	

After receiving the *ST* and the credential from the controller *Con*, the password of *Con* is retrieved by the master node *M* of the cluster from the local memory. Next, *PasswordDigest*' is computed by *M*, and compared to the original *PasswordDigest*. Then, *M* verifies whether the identity of the controller is valid. We use the *ClusteringResponse* message to denote the original clustering response message. Moreover, to obtain the MAC of *ClusteringResponse*, *Key* is calculated. Then, the master node of the cluster computes the credential of *ClusteringResponse* and sends it to the client node. Table III shows the detailed principle of credential authentication.

B. Ant System based Modeling for Cluster Management

The ant colony algorithm is a feasible and proven approach for data analysis. However, realizing the cluster of the SDN control plane remains an open issue. In this section, the ant colony algorithm is adapted for the cluster of SDN control plane. The ant system specifically includes the following three operations.

1) Initialization: The pheromone on each edge is initialized to a small value τ_0; and there is a taboo table of each ant which records the nodes that have passed through. The taboo table's length is initialized to 1 and the node that the ant is currently on is recorded. The amount of pheromone on each edge released by the ants is initialized to 0.

2) Construct ants' paths: The ants determine their next location to arrive based on a certain probability; this process is called state transition. The probability calculation is shown in the Eq. (1).

$$P_{ij}(t)=\begin{cases}\dfrac{\left[\tau_{ij}(t)\right]^{\alpha}\left[\eta_{ij}\right]^{\beta}}{\sum_{u\in allowed}\left[\tau_{tu}(t)\right]^{\alpha}\left[\eta_{tu}\right]^{\beta}} & if\ j\in allowed\\ 0 & otherwise\end{cases} \tag{1}$$

Eq. (1) shows the probability of choosing city *j* from location *i* at time *t*, which is called state transition rule. α means the weight of pheromone in probability calculation, which is termed pheromone weight parameter in this paper. The greater the pheromone value is, the greater the effect of pheromone during the process of ant transfer. β is the weight of the heuristic factor (which is usually expressed as the reciprocal of d_{ij} in the traveling salesman problem) in the probability calculation, and it is called heuristic factor weight parameter in this paper. The larger its value is, the larger the effect of heuristic factor is during the process of ant transfer. Eq. (1) shows that the ants will not choose locations in the taboo table, which ensures the legitimacy of the solution.

3) The operations on the pheromones: In the ant colony model, when all the ants have found a legitimate path, the pheromone on that path is updated, as shown in Eq. (2).

$$\tau_{ij}(t+1)=\rho\cdot\tau_{ij}(t)+\Delta\tau_{ij}(t,t+1) \tag{2}$$

where τ_{ij} means the pheromone on the edge *ij* at time *t*. ρ is the pheromone maintenance factor, and $1-\rho$ is the pheromone

volatile factor. $\Delta\tau_{ij}(t,t+1)$ is the sum of the pheromones released by all the ants on the edge *ij* from time *t* to *t*+1, as shown as Eq. (2).

$$\Delta\tau_{ij}(t,t+1)=\sum_{k=1}^{m}\Delta\tau_{ij}^{k}(t,t+1) \tag{3}$$

The *m* in the formula is the number of ants, and $\Delta\tau_{ij}^{k}(t,t+1)$ denotes the amount of pheromones released by ant *k* on edge *ij* from time *t* to *t*+1.

$$\Delta\tau_{ij}^{k}(t,t+1)=\begin{cases}\dfrac{Q}{L_k} & \text{Path formed by the ant } k \text{ containing edge } ij\\ 0 & \textit{Otherwise}\end{cases} \tag{4}$$

To construct the initial SDN clustering classification under the influence of pheromone and heuristic factors, the ants follow Eq. (4) to select attribute entries from all the selectable attribute entries as a conjunctive item of the rule's prerequisite. For an attribute entry, if the other attribute entry for the same attribute does not exist in the rule's previous part, the attribute entry is optional. Eq. (4) shows that during the process of constructing the initial classification rules front, for all the optional attribute entries, the ant selecting an attribute item into the rule's previous part does not consider which attribute the attribute belongs to but only which pheromone and inspired information are associated with it. This approach breaks the boundaries between attribute entries that belong to different attributes.

$$p_{ij}=\frac{\eta_{ij}\cdot\tau_{ij}(t)}{\sum_{i=1}^{a}x_i\cdot\sum_{j=1}^{b_i}\left(\eta_{ij}\cdot\tau_{ij}(t)\right)} \tag{5}$$

In Eq. (5), p_{ij} is the selection probability value of the *j*-th attribute of attribute entry *i*. The higher the value is, the greater the probability is, where the corresponding attribute entry is added into the rule. η_{ij} is the heuristic information that corresponds to the *j*-th attribute entry in attribute *i*, and it is determined by Eq. (5). The heuristic information of each attribute entry is proportional to the ability of that attribute entry to classify samples. $\tau_{ij}(t)$ is the amount of pheromone associated with the property of attribute *i* at the *t*-th loop iteration. Attribute entries that exist in high-quality rules will have the opportunity to obtain more pheromones; therefore that attribute entry is more likely than the other optional attribute entries to be placed into the rule's previous part by the ant in the next loop iteration. Here, *a* is the number of attributes in the training data set. x_i plays the role like a tabu list, ensuring that no two attribute entries in the rule's previous part belong to the same attribute. When attribute *i* has not been chosen by an ant, *x* will be set to 1. When the attribute i already has an attribute entry that exists in the rule's previous part, x_i is set to 0. b_i is the number of attribute entries contained in attribute *i*.

During the process of the ant constructs the initial rule's previous part, the selected attribute entry is not determined by the attribute selection probability but by the Roulette Selection Mechanism [36]. The Roulette Selection Mechanism has the advantage that attribute entries with a larger selection probability are more likely to be selected. However, attribute entries with a smaller selection probability also have the opportunity to be selected, which reduces the risk that the algorithm will converge to some local optimal solutions.

$$\eta_{ij}=\frac{\log_2 k-H\left(W\mid A_i=V_{ij}\right)}{\sum_{i=1}^{a}\mathrm{x}_i\sum_{j=1}^{b_j}\log_2 k-H\left(W\mid A_i=V_{ij}\right)} \tag{6}$$

In Eq. (6), $H\left(W\mid A_i=V_{ij}\right)$ refers to the information entropy corresponding to the attribute item . The information entropy is determined by Eq. (7), and k is the number of classes in the data set.

$$H\left(W\mid A_i=V_{ij}\right)=-\sum_{w=1}^{k}\left(P\left(w\mid A_i=V_{ij}\right)\cdot\log_2 P\left(w\mid A_i=V_{ij}\right)\right) \tag{7}$$

C. Ant Colony Optimization Algorithm based Cluster

To realize the SDN clustering in the control plane, data objects (e.g., SDN traffic) are regarded as ants with various attributes, and the clustering master node is regarded as the food source. The detailed clustering process is to imitates the foraging process of ants. Ants are distributed over a number of food sources in this model, and "move" jumpily among the sources. Meanwhile, a food source's characteristics can change dynamically as the ants change. For example, if a controller node moves from one cluster to another cluster, the absence of cluster controller nodes will cause the ants to change, which changes the characteristics of the food source.

Through the interaction between the ant colony and the food source, the distribution of ants in different food sources can be achieved, and the dissimilarities among the ants within the same food source is minimized. Suppose there are N data objects to be analyzed. Each data object has m attributes. The data object is defined as $X=\left\{X\mid X_i=(x_{i1},x_{i2},\cdots,x_{im}),i=1,2,\cdots,N\right\}$. The degree of dissimilarity between different objects is measured by the Euclidean distance between them. The smaller the distance is, the smaller the degree of dissimilarity is.

According to the definition of the ant colony algorithm, ants move randomly. The probability that the ant X_i will move to the food source where ant X_j is locate based on the Eq. (8).

$$p_{ij}=\frac{\tau_{ij}^{\alpha}(t)\eta_{ij}^{\beta}(t)}{\sum_{s\subset S}\tau_{sj}^{\alpha}(t)\eta_{sj}^{\beta}(t)} \tag{8}$$

When all the ants have been "moved" (which is equivalent to the idea that the data objects contained in each class have changed), the cluster center of each class must be recalculated (which is equivalent to the idea that the classes have changed). Therefore, the dissimilarity degree of the internal data in each class also needs to be recalculated. The dissimilarity degree of the internal data in the same class is calculated based on Eq. (9).

$$D_j = \sum_{k=1}^{J} \sqrt{\sum_{i=1}^{m} (x_{ki} - x_{ji})^2} \tag{9}$$

And for a class, the center point is determined according to Eq. (10).

$$C_j = \frac{1}{J} \sum_{k=1}^{J} X_k \tag{10}$$

If the sum of the dissimilarity degree of all classes is less than ε, the clustering terminates and results of the analysis are returned. Otherwise, the cluster needs to be re-analyzed. The procedures above are repeated until the termination condition is met.

VI. Design of Implementation System

A. Overview of Implementation System

According to the above functional technical features, an abstract module interaction diagram is shown in Fig. 3.

The main goal of the modules is to realize the interactions between the application plane and the SDN control plane. In other words, it aims to provide the application with the ability to interact with the SDN controller enabling the application to submit network resource requirements to the controller. Then, the control plane issues a specific network policy for the corresponding traffic based on the requirements of the application layer and sends it to the network devices.

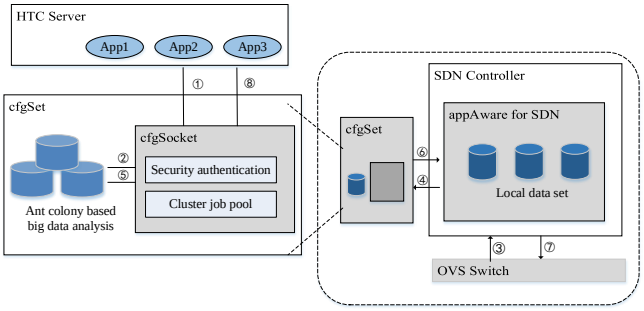


Fig. 3. Implementation system of security cluster for control plane of SDN.

B. Capabilities Entities of the Implementation System

The specific meaning of each process is as follows: 1) The application sends network demand parameters to cfgSocket, the TCP socket program on the controller server. 2) cfgSocket parses the data and stored it into the Configuration database cfgDB. 3) Because there is no matching flow table entry, the OpenFlow switch forwards the packets received from the application to the controller. 4) The controller queries the network demand parameters from the policy setting module cfgSet based on the packet information. 5) The database returns the network requirement parameter. 6) The cfgSet module returns the formatted demand parameters. 7) The controller sets the forwarding policy based on the demand parameter and sends the flow table entry to the switch. 8) The application passes the cfgSocket query.

Based on module interaction requirements, during realization, the application-aware cluster traffic management modules can be divided into two entities: the of network parameter configuration entity, cfgSet, and the application-aware entity, appAware. Here, appAware is an application that runs on the SDN controller platform, while cfgSet is a stand-alone program running on the same server as the overall SDN controller.

HTC is a cluster computing mode that improves the utilization rate of cluster resources by dispatching batch tasks in a cluster of computer. The nodes involved in the HTC cluster can be distributed around the world. More importantly, the tasks that HTC schedule may have different network requirements. To provide stable and reliable web services, beyond improving network bandwidth, HTC must provide a flexible fine-grained supervisor mode. Existing traffic management and optimizing methods are generally implemented only at the application layer or network layer; this results in a situation where the network layer cannot provide accurately customized services for applications in terms of bandwidth, time delay, throughput, and so on. For instance, by considering the topological relations when scheduling tasks and optimizing the task scheduling algorithms, SLURM reduces the data transmission requirements. However, SLURM can provide only limited functionality because it cannot interact with the network layer. Because the network-based traffic management tools cannot distinguish between different applications running on the same node, they cannot provide optimized network policies for specific tasks.

- **Network Parameter Configuration Entity**

The main function of cfgSet is to interact with the application plane to obtain its requirements for network resources, including bandwidth, priority and other requirements. Then cfgSet formats these requirements and stores them in the database.

Moreover, cfgSet must be able to receive query requests from appAware and return the corresponding application's network parameters based on the key value to appAware, which is the basis for developing a forwarding strategy.

- **Entity of Application-Aware**

The main function of appAware is to parse the packets passed into the controller to determine the application to which they belong. Then, it queries cfgSet to obtain the network parameters as the basis for developing a forwarding strategy.

These two modules typically run in the same server to reduce the time delay for the internal communication between the modules while avoiding additional processing steps such as NAT conversion.

- **Implementation Module**

In this paper, the SDN controller used in the research process is ONOS, which runs in the Karaf container. ONOS conforms to the OSGi technical standard, provides functionally isolated modules, and interacts through the interface calls. AppAware is an application module that is implemented in ONOS, while cfgSet is a stand-alone java program.

- **Configuring the Network Parameters**

Network parameters represent to the specific application requirements such as network protocols, bandwidth, connectivity, and other aspects. HTC scheduler usually has its own data structure that represents the resource requirements for the HTC task. As an example, HTCondor stores this information in a list called ClassAd. Each HTCondor task and HTCondor machine has a unique ClassAd member. All the ClassAd members of a task are defined when the task is committed. We achieve the representation of the parameters of the network requirement by extending the existing ClassAd mechanism.

- **Implementation of Application-Aware Function**

The focus of realizing the application-aware function is to associate a data stream with an application.

In general, two mechanisms are used to achieve this goal: Deep Packet Inspection and IP/Port Mapping. Deep packet Inspection analyzes the packet contents and performs pattern matching with an application. This article uses IP/port mapping to map packets and applications. To initialize this mechanism, public IP addresses must be assigned to tasks using a bridge pattern. Open vSwitch provides one way of accomplishing this task.

The virtual network card of OvS connects with the physical network card, causing the IP address and the addressing request of the physical network card to transfer to the virtual network card. A virtual network adapter is generated when each scheduling task is established. The container running a task and OvS are connected through an exclusive virtual network card to create a private network namespace. Meanwhile, each container is given a single IP address used in data transmission. In this way, an data stream is associated with an the application through the IP address and port number.

The first time the data plane receives an application packet, it sends the PACKET_IN message to the controller because no corresponding item exists in the flow table. The controller calls the function in the PacketService interface to analyze the package. Then, the controller obtains the corresponding network configuration by parsing the source IP address and port number of the packet header, and queries cfgDB using key value.

- **Implementation of Traffic Management Function**

Tasks running in the same HTCondor node share a certain amount of bandwidth, that is, the bandwidth allocated by the system physical network card. The HTCondor node performs allocations according to the bandwidth requirements declared by the task. If the idle bandwidth is insufficient, the new task will be stopped. According to the OvS technology standard, the application's bandwidth demand could not be allocated during operation, because there is an upper limit for the bandwidth that can be allocated for the virtual port by the physical network card. That is, the bandwidth allocated for the task by the traffic management module must be valid during the task's running cycle.

C. Implementation of Communications

In an SDN, the core idea of the cluster control is to maintain a complete view between the dynamic management of network devices and the distributed controller nodes. It is important to be able to control the communications by the controller within intra cluster. In fact, each controller has an exclusive NodeID and each node in a cluster has an exclusive DeviceID. Thus, there is a many-to-many mapping relation among the controllers and other nodes in the cluster. To maintain the logical consistency, a node in the cluster is managed by just one controller. In the controller cluster, each controller manage the status information of a networking zone. In addition, the status information can be broadcasted to all the controllers.

The northbound interface of ONOS provides the ability to obtain a global network view so that the application can perform various advanced communication and management functions. The primary consideration for maintaining the global view is to maintain network state consistency between controller nodes. At any time, each node needs to have the ability to provide accurate and complete network status. To enable the controller nodes to obtain the global network view quickly and to ensure the integrity and accuracy of the view, ONOS uses the following technical means:

1) MASTER identifier: For a node in the cluster, the identity of the MASTER node is a key message, because only the MASTER node can operate on it. Therefore, the collection of MASTER mappings for all devices is an important part of the global network view. To ensure the consistency of the MASTER mapping relationship, each device has a MASTER identifier indicating the serial number of the current MASTER node.

2) Topology event sequence: During a given MASTER mapping validity period, the MASTER node receives a series of topology update events. To flag these events in the global network state, we set a monotonically increasing topology event sequence similar to the MASTER identifier.

3) Logical timestamp: In the case where both the MASTER identifier and the topology event sequence are determined, we can obtain a unique global network view. This view makes it possible to connect the above two identifiers as a timestamp of the global network state.

4) Network topology state machine: When a topology update event occurs, the MASTER node first generates a logical timestamp and then updates the local network state machine. Then, the event information and the logical timestamps are broadcasted to all the nodes in the cluster. The receiving node first determines whether the local state machine is out of date based on the timestamp, if so, it updates its own state machine if it is out of date.

VII. Simulations And Evaluations

A. Simulation Setup

In this implementation of an SDN using cluster controller to process jobs, the cluster controller consists of 16 Intel(R) Xeon(R) E5620 CPU (2.40 GHz), 16 G memory, 500 G disk and a bandwidth capacity of 1000 Mbps. The operation system is Linux ubuntu 3.2.0-29-generic. The main function of cfgSet is to interact with the application plane to obtain its requirements for network resources, including bandwidth,

priority and other requirements, and store them in the database.

TABLE IV
THE NETWORK ATTRIBUTES OF TASK

Attributes	Meaning
IPProtocol	Prior IP protocol (IPv4/IPv6)
RequestBandwidth	Bandwidth task required
NetworkAccouting	the network traffic statistic or not
InboundConnectivity	Whether it needs inward connection
OutboundConnectivity	Whether it needs external connection
MAC(*E, Key*)	Uses *Key* to compute MAC for message *E*

To test the performance and availability of the cluster traffic management modules implemented above. This study expands the ONOS network. In this network, three HTCondor tasks share the 1,000 Mbps physical bandwidth of the node and download files from three remote HTTP servers. The network attributes for task are shown in Table IV.

Three users, A, B and C submit tasks to the HTCondor node. The bandwidth required for A's task is 100Mbps, and the file is downloaded from the server A. The bandwidth required for B's task is 300Mbps, and the file is downloaded from server B. The bandwidth required for C's task is 600Mbps, and the file is downloaded from server C. Because A, B and C are given equal priority, they can perform a task in each time slot. The bandwidth allocated to the three tasks is equivalent to the node's physical bandwidth, that is, there is no idle bandwidth c. Therefore, no new tasks would be executed. Only after the task is completed, and idle bandwidth appears; new tasks can turn into the execution state.

When the traffic management module is enabled or disabled the bandwidth occupied by each task changes over time.

B. *Simulation of Traffic Rate*

Figure 4 shows that the bandwidth used by each task is limited to the bandwidth level they declared in the configuration files when the traffic management module is enabled, and these would not be redistributed even if idle bandwidth appeared during operation. Figure 4 shows that the three tasks are assigned equal bandwidth when the traffic management module is not enabled.By comparing Fig. 4 and Fig. 5, we can conclude that the traffic management modul's bandwidth managements for the different tasks realizes the design goals.

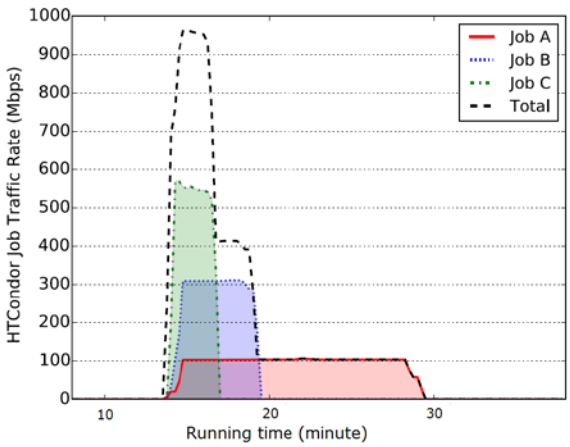


Fig. 4. The bandwidth occupied with traffic management.

Next, we change the script used to submit the tasks in the experiment to make three users A,B and C cycle their task submission, and the bandwidth usage is shown in Fig. 6.

A cluster management experiment for data transmission of distributed network devices is performed in multiple ONOS control domains.

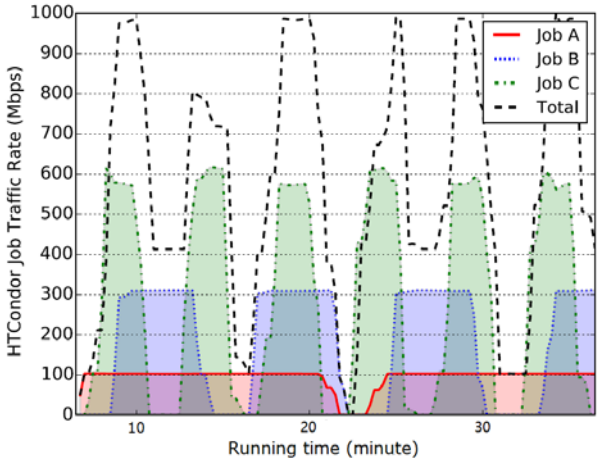


Fig. 5. The bandwidth occupied with cluster management and cycle to submit tasks.

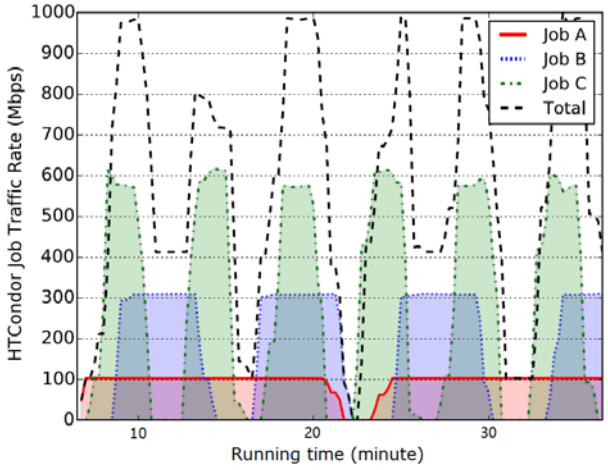


Fig. 6. The bandwidth occupied with cluster management and cycle to submit tasks.

In the cluster traffic management experiment, the server and the client respectively run in the OvS control domains that shown in the figure. The bandwidth required for job 1 is set to 250Mbps, and the total bandwidth of OvS2 is set to 100Mbps. No traffic management is applied to the job 2 and job 3.The bandwidth occupied by the three tasks measured by the experiment is shown in Fig. 6-8.

As shown in Fig. 7, the bandwidth occupied by job 1 is approximately 250Mbps, which accords with its bandwidth configuration. The total bandwidth occupied by job 2 and job 3, that is, the physical bandwidth of OvS2 is 100Mbps, reaching its upper physical bandwidth limit.

Figure 7 shows the bandwidth respectively occupied by job 2 and job 3.

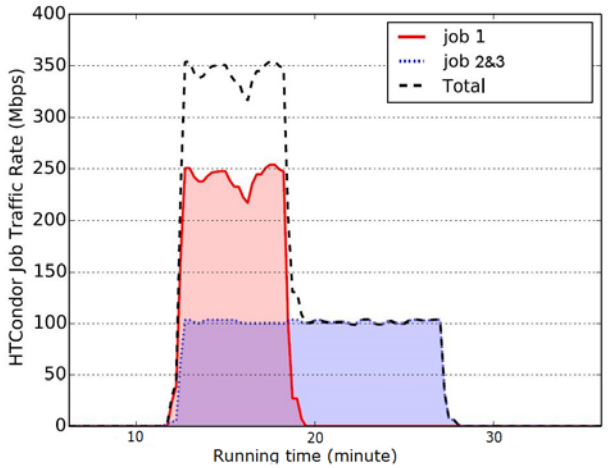


Fig. 7. The bandwidth occupied in different jobs.

C. Simulation of Time Overhead and Energy Consumption

To evaluate the communication overhead of the proposed secure cluster scheme, Mininet is used to simulate an SDN that includes 10,000 network nodes and 500 OpenFlow switches. The available bandwidth is 100 Mbps. The cluster scheme in [37] is introduced for comparison purposes. Figure 9 shows that the communication overhead of the proposed cluster scheme is lower than that of the scheme in [37]. This is because the proposed ant colony optimization enables big data analysis and provide an optimized cluster controller for SDN.

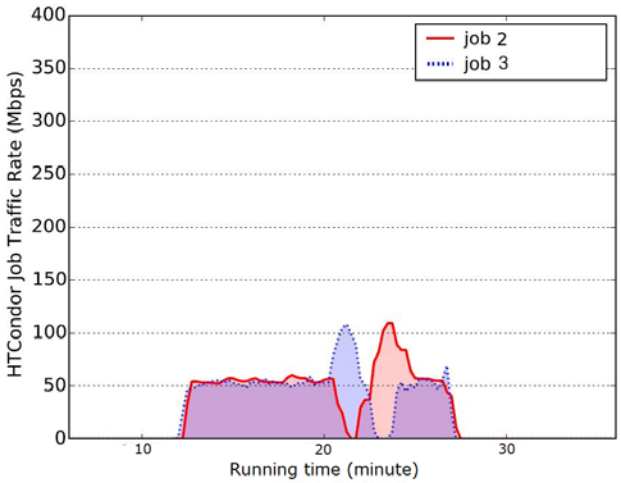


Fig. 8. The bandwidth respectively occupied by job 2 and job 3.

In addition, as shown in Fig. 10, we vary the number of sensor nodes to extensively check the energy consumption of our proposed algorithm. The power consumption of an SDN node in activated mode and sleeping mode is set to P_a=12:0mW and P_s=270μW, respectively, which have been verified and adopted in [38]. The network size varies from 200 to 500. Additionally, the comparison of the energy consumption is performed between the proposed cluster scheme and the scheme in [37]. As shown in Fig. 9, the energy consumption is less than that of the cluster scheme in [37], especially for large scale-networks. This because the ant colony optimization results are better when more data can be got from

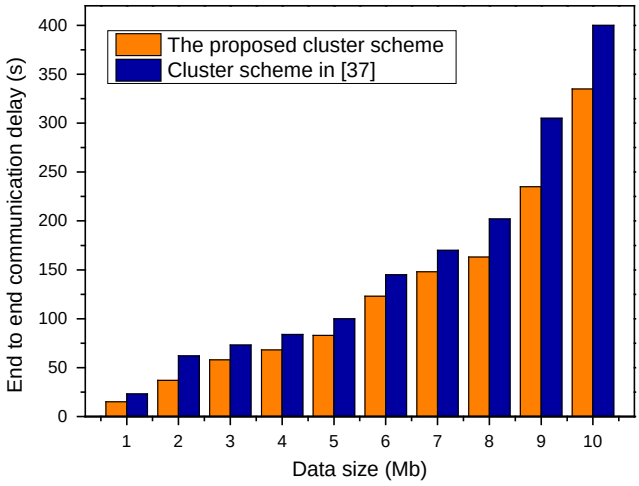


Fig. 9. Communication overhead.

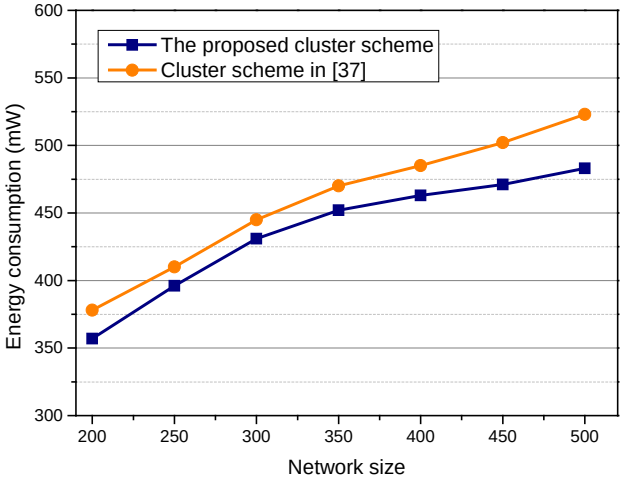


Fig. 10. Energy consumption.

the networks.

VIII. Conclusions

In large-scale SDN, multiple controllers in the control plane must be able to collaborate to manage the entire network. Control plane extensibility is an important issue. Clustering is a feasible and proven approach to achieve efficient SDN management, in which the cluster monitors all types of events and maintain a consistent global network status. This usually involves big data in SDN. At the same time, the legality of the data sources should be ensured, thus the big data for cluster must be trustworthy. To address the above challenges, this paper proposed a big data analysis-based secure cluster management architecture for optimized control plane. A secure authentication scheme was proposed to ensure the legality of the data sources. Next, ant colony optimization was used to enable a big data analysis scheme and an implementation system was proposed to optimize the control plane. This work is significant in improving the performance and efficiency of applications running in SDN. In future work, a distributed security data storage scheme for the SDN controller cluster will

be proposed.

ACKNOWLEDGEMENT

This work was supported in part by the National Natural Science Foundation of China under Grant 61431008, 61571300, and partially supported by the JSPS KAKENHI Grant Number JP16K00117, JP15K15976, KDDI Foundation.

REFERENCES

[1] A. Blenk, A. Basta, M. Reisslein, W. Kellerer,“Survey on Network Virtualization Hypervisors for Software Defined Networking,” IEEE Communications Surveys & Tutorials, vol. 18, no. 1, pp. 655-685, 2016.

[2] S. Scott-Hayward, S. Natarajan, S. Sezer, “A Survey of Security in Software Defined Networks,” IEEE Communications Surveys & Tutorials, vol. 18, no. 1, pp. 623654, 2016.

[3] G. Li, M. Dong, K. Ota, J. Wu, J. Li, and T. Ye, "Deep Packet Inspection based Application-Aware Traffic Control for Software Defined Networks," IEEE Global Communications Conference (GLOBECOM 2016), pp. 1-6, 2016.

[4] W. Han, M. Dong, K. Ota, J. Wu, J. Li, G. Li, "SD-OPTS: Software-Defined On-Path Time Synchronization for Information-Centric Smart Grid," IEEE Global Communications Conference (IEEE GLOBECOM 2017), pp. 1-6, 2017.

[5] L. Cui, F. R. Yu, and Q. Yan, "When Big Data Meets Software-Defined Networking: SDN for Big Data and Big Data for SDN," IEEE Network, vol. 30, no. 1, pp. 58-65, 2016.

[6] L. Kuang, L. T. Yang, X. Wang, P. Wang, and Y. Zhao, "A Tensor-based Big Data Model for QoS Improvement in Software Defined Networks," IEEE Network, vol. 30, no. 1, pp. 30-35, 2016.

[7] Z. Jiao, H. Ding, M. Dang, R. Tian, and B. Zhang, "Predictive Big Data Collection in Vehicular Networks: A Software Defined Networking Based Approach," in Proc. of IEEE Global Communications Conference (GLOBECOM), pp. 1-6, 2016.

[8] H. Li, M. Dong, and K. Ota, "Control Plane Optimization in Software-Defined Vehicular Ad Hoc Networks," IEEE Transactions on Vehicular Technology, vol. 65, no. 10, pp. 7895-7904, 2016.

[9] Y. Fu, Jun Bi, Z. Chen, K. Gao, B. Zhang, G. Chen, and J. Wu, "A Hybrid Hierarchical Control Plane for Flow-Based Large-Scale Software-Defined Networks," IEEE Transactions on Network and Service Management, vol. 12, no. 2, pp. 117-131, 2015.

[10] M. Ambrosin, M. Conti, F. D. Gaspari, R. Poovendran, "LineSwitch: Tackling Control Plane Saturation Attacks in Software-Defined Networking," IEEE/ACM Transactions on Networking, vol. 25, no. 2, pp. 1206-1219, 2017.

[11] S. Luo, M. Dong, K. Ota, J. Wu, Jianhua Li, "A Security Assessment Mechanism for Software-Defined Networking based Mobile Networks," Sensors, vol. 15, no. 12, pp. 31843-31858, 2015.

[12] S. Song, H. Park, B. Y. Choi, T. Choi, H. Zhu, "Control Path Management Framework for Enhancing Software-Defined Network (SDN) Reliability," IEEE Transactions on Network and Service Management, vol. 14, no. 2, pp. 302-316, 2017.

[13] Y. Fu, J. Bi, Z. Chen, Kai Gao, B. Zhang, G. Chen, and J. Wu, "A Hybrid Hierarchical Control Plane for Flow-Based Large-Scale Software-Defined Networks," IEEE Transactions on Network and Service Management, vol. 12, no. 2, pp. 117-131, 2015.

[14] S. Zhao, D. Medhi, "Application-Aware Network Design for Hadoop MapReduce Optimization Using Software-Defined Networking," IEEE Transactions on Network and Service Management, vol. 14, no. 4, pp. 804-816, 2017.

[15] X. Lyu, H. Tian, W. Ni, R. P. Liu, and P. Zhang, "Adaptive Centralized Clustering Framework for Software-Defined Ultra-Dense Wireless Networks," IEEE Transactions on Vehicular Technology, vol. PP, no. 99, pp. 1-5, 2017.

[16] A. S. Muqaddas, A. Bianco, P. Giaccone, and G. Maier, "Inter-Controller Traffic in ONOS Clusters for SDN Networks," in Proc. of 2016 IEEE International Conference on Communications (ICC 2016), pp. 1-6, 2016.

[17] R. Macedo, R. de Castro, A. Santos, Y. Ghamri-Doudane, and M. Nogueira, "Self-Organized SDN Controller Cluster Conformations against DDoS Attacks Effects," in Proc. of IEEE Global Communications Conference (GLOBECOM), pp. 1-6, 2016.

[18] R. Xie, Z. Umair, and X. Jia, "A Wireless Solution for SDN (Software Defined Networking) in Data Center Networks," in Proc. of IEEE Global Communications Conference (GLOBECOM 2016), pp. 1-6, 2016.

[19] J. Al-Jaroodi, N. Mohamed, H. Jiang, and D. Swanson, "Middleware infrastructure for parallel and distributed programming models in heterogeneous systems," IEEE Transactions on Parallel and Distributed Systems, vol. 14, no. 11, pp. 1100-1111, 2016.

[20] B. Lin, W. Guo, N. Xiong, G. Chen, A. V. Vasilakos, and H. Zhang, "A Pretreatment Workflow Scheduling Approach for Big Data Applications in Multicloud Environments," IEEE Transactions on Network and Service Management, vol. 13, no. 3, pp. 581-594, 2016.

[21] P. Fiadino, P. Casas, A. D Alconzo, M. Schiavone, and A. Baer, "Grasping Popular Applications in Cellular Networks With Big Data Analytics Platforms," IEEE Transactions on Network and Service Management, vol. 3, no. 3, pp. 681-685, 2016.

[22] C. Y. Lin, J. Y. Liao, "An SDN App for Hadoop Clusters," in Proc. of IEEE 7th International Conference on Cloud Computing Technology and Science (CloudCom 2015), pp. 458-461, 2015.

[23] M. Khan, Y. Jin, M. Li, Y. Xiang, and C. Jiang, “Hadoop Performance Modeling for Job Estimation and Resource Provisioning,” IEEE Transactions on Parallel and Distributed Systems, vol. 27, no. 2, pp. 441-454, 2016,.

[24] J. Dean and S. Ghemawat, “MapReduce: Simplified Data Processing on Large Clusters,” Communications of the ACM, vol. 51, no. 1, pp. 107-113, 2005.

[25] M. Isard, M. Budiu, Y. Yu, A. Birrell, and D. Fetterly, “Dryad: Distributed Data-Parallel Programs from Sequential Building Blocks,” in Proc. of European Conf. Computer Systems, pp. 59-72, 2007.

[26] I. Michelakos, N. Mallios, E. Papageorgiou, and M. Vassilakopoulos, "Ant Colony Optimization and Data Mining: Techniques and Trends," in Proc. of 2010 International Conference on P2P, Parallel, Grid, Cloud and Internet Computing, pp. 284-289, 2010.

[27] F. Haneef, G. R. Kushwaha, and A. K. Dubey, "Analysis with Data Mining and Ant Colony Algorithm for Implementing of Object Pool Optimization," in Proc. of 2011 International Conference on Communication Systems and Network Technologies, pp. 313-317, 2011.

[28] B. Li, C. Zhao, H. Zhang, Z. Zhou, A. Nallanathan, "Efficient and Robust Cluster Identification for Ultra-Wideband Propagations Inspired by Biological Ant Colony Clustering," IEEE Transactions on Communications, vol. 63, no. 1, pp. 286-300, 2015.

[29] J. Wang, J. Cao, B. Li, S. Lee, and R. S. Sherratt, "Bio-Inspired Ant Colony Optimization based Clustering Algorithm with Mobile sinks for Applications in Consumer Home Automation Networks," IEEE Transactions on Consumer Electronics, vol. 61, no. 4, pp. 438-444, 2015.

[30] S. H. Yeganeh and Y. Ganjali, "Kandoo: A Framework for Efficient and Scalable Offloading of Control Appcations," in Proc. of the first ACM Workshop on Hot Topics in Software Defined Networks, pp. 19-24, 2012.

[31] M. Yu, J. Rexford, M. J. Freedman, and J. Wang, "Scalable flow-based networking with DIFANE," ACM SIGCOMM Computer Communication Review, vol. 41, no. 4, pp. 351-362, 2011.

[32] A. R. Curtis, J. C. Mogul, J. Tourrilhes, P. Yalagandula, P. Sharma, and S. Banerjee, "DevoFlow: Scaling Flow Management for High-Performance Networks," ACM SIGCOMM Computer Communication Review, vol. 41, no. 4, pp. 254-265, 2011.

[33] K. Phemius, M. Bouet, and J. Leguay, "DISCO: Distributed Multi-Domain SDN Controllers," in Proc. of IEEE Network Operations and Management Symposium (NOMS 2014), pp. 1-4, 2014.

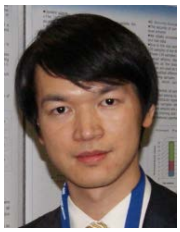
[34] A. Tootoonchian and Y. Ganjali, "HyperFlow: A Distributed Control Plane for OpenFlow," in Proc. of 2010 Internet Network Management Conference on Research on Enterprise Networking, pp. 1-6, 2010.

[35] T. Koponen, M. Casado, N. Gude, J. Stribling, L. Poutievski, M. Zhu, R. Ramanathan, Y. Iwata, H. Inoue, T. Hama, and S. Shenker, "Onix: A Distributed Control Platform for Large-Scale Production Networks," in Proc. of the 9th USENIX Conference on Operating Systems Design and Implementation (OSDI 2010), pp. 351-364, 2010.

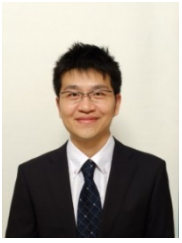
[36] E. D. Lumer, B. Faieta, "Diversity and Adaptation in Population of Clustering Ants," in Proc. of the 3rd International Conference on Simulation of Adaptive Behavior: From Animals to Animats (SAB 94), pp. 501-508, 1994.

[37] M. Shimizu, Y. Watashiba, S. Date, and S. Shimojo, "Adaptive Network Resource Reallocation for Hot-Spot Avoidance on SDN-Based Cluster System," in Proc. IEEE International Conference on Cloud Computing Technology and Science (CloudCom 2016), pp. 608-613, 2016.

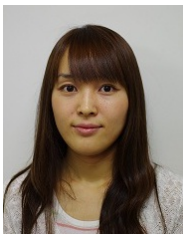
[38] D. Zeng, P. Li, S. Guo, T. Miyazaki, J. Hu, and Yong Xiang, "Energy Minimization in Multi-Task Software-Defined Sensor Networks," IEEE Transactions on Computers, vol. 64, no. 11, pp. 3128-3139, 2015.



Jun Wu (M'12) received the Ph.D. degree in information and telecommunication studies from Waseda University, Japan, in 2011. He was a Post-Doctoral Researcher with the Research Institute for Secure Systems, National Institute of Advanced Industrial Science and Technology (AIST), Japan, from 2011 to 2012. He was a Researcher with the Global Information and Telecommunication Institute, Waseda University, Japan, from 2011 to 2013. He is currently an associate professor of School of Cyber Security, Shanghai Jiao Tong University, China. He is also the vice director of National Engineering Laboratory for Information Content Analysis Technology, Shanghai Jiao Tong University, China. He is the chair of IEEE P21451-1-5 Standard Working Group. He has hosted and participated in a lot of research projects including National Natural Science Foundation of China (NFSC), National 863 Plan and 973 Plan of China, Japan Society of the Promotion of Science Projects (JSPS), etc. His research interests include the advanced computing, communications and security techniques of software-defined networks (SDN), information-centric networks (ICN) smart grids, Internet of Things (IoT), etc. He has been a Guest Editor of the IEEE Sensors Journal. He is an Associate Editor of the IEEE Access. He is a member of IEEE.



Mianxiong Dong (M'13) received B.S., M.S. and Ph.D. in Computer Science and Engineering from The University of Aizu, Japan. He is currently an Associate Professor in the Department of Information and Electronic Engineering at the Muroran Institute of Technology, Japan. He was a JSPS Research Fellow with School of Computer Science and Engineering, The University of Aizu, Japan and was a visiting scholar with BBCR group at University of Waterloo, Canada supported by JSPS Excellent Young Researcher Overseas Visit Program from April 2010 to August 2011. Dr. Dong was selected as a Foreigner Research Fellow (a total of 3 recipients all over Japan) by NEC C&C Foundation in 2011. His research interests include Wireless Networks, Cloud Computing, and Cyber-physical Systems. He has received best paper awards from IEEE HPCC 2008, IEEE ICESS 2008, ICA3PP 2014, GPC 2015, IEEE DASC 2015, IEEE VTC 2016-Fall, FCST 2017 and 2017 IET Communications Premium Award. Dr. Dong serves as an Editor for IEEE Transactions on Green Communications and Networking (TGCN), IEEE Communications Surveys and Tutorials, IEEE Network, IEEE Wireless Communications Letters, IEEE Cloud Computing, IEEE Access, as well as a leading guest editor for ACM Transactions on Multimedia Computing, Communications and Applications (TOMM), IEEE Transactions on Emerging Topics in Computing (TETC), IEEE Transactions on Computational Social Systems (TCSS). He has been serving as the Vice Chair of IEEE Communications Society Asia/Pacific Region Meetings and Conference Committee, Leading Symposium Chair of IEEE ICC 2019, Student Travel Grants Chair of IEEE GLOBECOM 2019, and Symposium Chair of IEEE GLOBECOM 2016, 2017. He is the recipient of IEEE TCSC Early Career Award 2016, IEEE SCSTC Outstanding Young Researcher Award 2017 and The 12th IEEE ComSoc Asia-Pacific Young Researcher Award 2017.



Kaoru Ota (M'12) was born in Aizu-Wakamatsu, Japan. She received M.S. degree in Computer Science from Oklahoma State University, USA in 2008, B.S. and Ph.D. degrees in Computer Science and Engineering from The University of Aizu, Japan in 2006, 2012, respectively. She is currently an Assistant Professor with Department of Information and Electronic Engineering, Muroran Institute of Technology, Japan. From March 2010 to March 2011, she was a visiting scholar at University of Waterloo, Canada. Also she was a Japan Society of the Promotion of Science (JSPS) research fellow with Kato-Nishiyama Lab at Graduate School of Information Sciences at Tohoku University, Japan from April 2012 to April 2013. Her research interests include Wireless Networks, Cloud Computing, and Cyber-physical Systems. Dr. Ota has received best paper awards from ICA3PP 2014, GPC 2015, IEEE DASC 2015, IEEE VTC 2016-Fall, FCST 2017 and IET Communications 2017. She is an editor of IEEE Transactions on Vehicular Technology (TVT), IEEE Communications Letters, Peer-to-Peer Networking and Applications (Springer), Ad Hoc \& Sensor Wireless Networks, International Journal of Embedded Systems (Inderscience) and Smart Technologies for Emergency Response \& Disaster Management (IGI Global), as well as a guest editor of ACM Transactions on Multimedia Computing, Communications and Applications (leading), IEEE Communications Magazine, IEEE Network, etc. Also she was a guest editor of IEEE Wireless Communications (2015), IEICE Transactions on Information and Systems (2014), and Ad Hoc \& Sensor Wireless Networks (Old City Publishing) (2014). She was a research scientist with A3 Foresight Program (2011-2016) funded by Japan Society for the Promotion of Sciences (JSPS), NSFC of China, and NRF of Korea. She is the recipient of IEEE TCSC Early Career Award 2017.

Jianhua Li is a professor/Ph.D. supervisor and the dean of School of Cyber Security, Shanghai Jiao Tong University, Shanghai, China. He is also the director of National Engineering Laboratory for Information Content Analysis Technology, the director of Engineering Research Center for Network Information Security Management and Service of Chinese Ministry of Education, and the director of Shanghai Key Laboratory of Integrated Administration Technologies for Information Security, China. He is the vice president of Association of Cyber Security Association of China. He got his BS, MS and Ph.D. degrees from Shanghai Jiao Tong University, in 1986, 1991 and 1998, respectively. He was the chief expert in the information security committee

experts of National High Technology Research and Development Program of China (863 Program) of China. He was the leader of more than 30 state/province projects of China, and published more than 300 papers. He published 6 books and has about 20 patents. He made 3 standards and has 5 software copyrights. He got the Second Prize of National Technology Progress Award of China in 2005. His research interests include information security, signal process, computer network communication, etc.

Zhitao Guan is currently an Associate Professor at the School of Control and Computer Engineering, North China Electric Power University (NCEPU). He received his BEng degree and PhD in Computer Application from Beijing Institute of Technology (BIT), China, in 2002 and 2008, respectively. His current research focuses on smart grid security, wireless security and cloud security. Dr. Guan has authored over 20 peer-reviewed journal and conference papers in these areas. He is a Member of the IEEE.