Received December 25, 2019, accepted January 5, 2020, date of publication January 13, 2020, date of current version January 28, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2966399

FIMPA: A Fixed Identity Mapping Prediction Algorithm in Edge Computing Environment

SHUO ZHANG[1], YAPING LIU[1], SHUDONG LI[1], ZHIYUAN TAN[2], (Member, IEEE), XIAOMENG ZHAO[3], AND JUNJIE ZHOU[3]
[1]Cyberspace Institute of Advanced Technology, Guangzhou University, Guangzhou 510000, China
[2]School of Computing, Edinburgh Napier University, Edinburgh EH10 5DT, U.K.
[3]Guangdong Provincial Key Laboratory of High Performance Computing, Guangdong Science and Technology Infrastructure Center, Guangzhou 510000, China

Corresponding author: Zhiyuan Tan (z.tan@napier.ac.uk)

This work was supported in part by the National Key Research and Development Program of China under Grant 2018YFB1003602, in part by the Key Area Research and Development Program of Guangdong Province under Grant 2019B010137005 and Grant 2019B010137004, and in part by the National Natural Science Foundation of China under Grant U1636215, Grant 61871140, Grant 61872100, Grant 61572153, and Grant 61672020.

ABSTRACT Edge computing is a research hotspot that extends cloud computing to the edge of the network. Due to the recent developments in computation, storage and network technology for end devices, edge networks have become more powerful, making it possible to integrate locator/identity separation protocol (LISP) into these networks. Accordingly, in this paper, we introduce LISP into edge routers at the edge network, focusing primarily on the delay problem of mapping resolution and cache updating in LISP with the help of edge computing. To solve this delay problem, we first analyze the communication process of the locator/identity separation network and consider using the prediction method to underpin this research. In order to achieve a good prediction result, we propose and implement a Fixed Identity Mapping Prediction Algorithm (FIMPA) based on collaborative filtering, and further verify the effectiveness of the proposed algorithm through experiments on real-world data.

INDEX TERMS Edge computing, LISP, mapping resolution, prediction, recommendation.

I. INTRODUCTION

In recent years, an increasing number of computing technologies (such as cloud computing [1], [2], cluster computing, IPTV [3], etc.) have emerged and been widely applied in various fields. However, all of them are designed so that the majority of functions are processed in the core (datacenter), while the terminal is relatively thin and sometimes even has no functions to process. Due to developments in the computation ability and storage technology of terminal devices, some tasks or functions may be offloaded to terminals from the core, which is the approach adopted by edge computing [4]–[6]. Edge computing, which has become a new research hotpot and an expansion of cloud computing, involves pushing computing, data, storage, and networking services away from centralized nodes to the logical extremes of a network.

Edge computing is an efficient method for optimizing cloud computing by performing data processing at the edge of the network, near the source of the data. This approach has many advantages [4]: (a) it reduces the communication bandwidth required between edge nodes and the datacenter by performing computation and storage at or near the original location of the data; (b) it may limit or remove a major bottleneck and a potential point of failure in the cloud computing environment; (c) it improves data security, as data is encrypted before being moved to the network core; (d) it achieves good scalability due to the virtualization in most edge nodes. Fog computing [7], [8], similar to edge computing, was first proposed by Cisco. It refers to extending cloud computing to the edge of an enterprise's network, facilitating the operation of computation, storage, and networking between end devices and cloud data centers.

Researchers at Cisco proposed and implemented the LISP protocol in its routes deployed in some networks with the aim of resolving issues related to mobility, multi-homing and the IP semantic overload problem. The LISP protocol splits existing IP addresses into entity identity (EID) and router identifier/locator (RLOC); moreover, it describes the locator/identity separation protocol on the network perspective, meaning that it can prevent the end hosts' network protocol

The associate editor coordinating the review of this manuscript and approving it for publication was Ying Li.

architecture from changing while only changes the working mode of network devices. In a LISP network, EID is independent of network topology, and is used to identify an end host when communicating with other hosts. EID can be allocated reasonably according to the deployment requirements of the mapping system.

Moreover, since the computation, storage, and networking resources are offloaded from the datacenter to the edge network, it is also feasible for the edge network to solve the traditional problems associated with LISP, such as delay of mapping resolution, limited storage of mapping entries, and so on. Therefore, it represents a feasible way to integrate edge computing and LISP. Our aim is to solve some issues in the integration process.

With the help of edge computing, we focus on the delay problem of mapping resolution; this problem arises when an identity cannot be resolved locally, which triggers a mapping resolution request to the mapping system and waits for the response.

This paper first analyzes the communication process of the Locator/Identity separation network, focusing on the delay problem of mapping resolution in LISP and the update mechanism of mapping the cache in edge routers. Subsequently, we propose a Fixed Identity Mapping Prediction Algorithm (FIMPA) based on collaborative filtering. Finally, we verify the effectiveness of our proposed algorithm through experimentation. The experimental results illustrate that the FIMPA algorithm can significantly improve the hit rate and reduce the delay.

The contribution of our work mainly lies in that solving some issues (such as delay problem of mapping resolution and cache updating) in LISP with the help of edge computing.

The remainder of this paper is organized as follows. In section II, we introduce some background concepts, including the communication process of the LISP network and the delay problem. We provide our motivation in Section III, while the FIMPA algorithm is proposed in Section IV. In Section V, we conduct some experiments and evaluations. Finally, we present the conclusion and discuss future work in Section VI.

II. BACKGROUND

A. EDGE COMPUTING

In recent years, there has been a large body of research focusing on cloud computing. Cloud computing is an on-demand computing model that enables on-demand access to a shared pool of configurable networked resources (e.g., CPUs, storage, VMs, networks, applications, servers) that can be rapidly provisioned with minimal management effort. Service providers offer clouds with predefined quality of service (QoS) terms to interested clients through the Internet on a subscription basis, providing them with a set of easy-to-use, scalable, and inexpensive services. In cloud computing, almost all functions are processed in the core (datacenter), while the client is relatively thin and may even have no processing functionality, making it a server-centric computing paradigm.

Although cloud computing has several advantages including easy maintenance, centralized management, and high server utilization, its limitations have been exposed in the mobile Internet era: for example, the terminals should have higher processing capability, there are some security problems with the terminals, etc. Accordingly, transparent computing [16], [17] proposes a promising solution for the mobile Internet. The core idea is that all data and software – including operation systems, apps, and user information – are stored on servers, with data computing being performed on terminals. This approach has a number of advantages: it reduces terminals' complexity and cost, improves user experience, and offers high-level security and compatibility for cross-platform applications [17]. This is also suitable for access control [18].

Edge computing has become a research hotspot due to the rapid development of the IoT (Internet of Things) over the past three years, meaning that applications, data and networking services are being pushed away from centralized nodes in datacenters to the end devices. This approach takes advantage of the terminal's powerful computation and storage capabilities by offloading some tasks or functions to clients from the core; doing so has a number of advantages, including reducing the communications bandwidth between edge nodes and the datacenter, eliminating or removing bottlenecks and potential failures point in the cloud environment, improving data security, and achieving good scalability. Additional research into edge computing can be found in [19]–[21], which considers using blockchain [22].

B. CACHE UPDATE MECHANISM

Regarding the update mechanism of the mapping cache in edge routers, the related research both at home and abroad can be broadly divided into two categories: namely, cache replacement algorithms and cache prediction/prefetching algorithms.

Cache replacement algorithms are primarily focused on how to replace the mapping entries in the cache under conditions of limited space. Research in this area typically focuses on either temporal locality or spatial locality.

LRU (Least Recently Used) [23] is a traditional algorithm, which assumes that recently visited objects are the most likely to be visited again in the future. Thus, it always removes the oldest object that has not been accessed from the cache. Although LRU is simple, it is also the most popular algorithm of its kind; however, due to involving the object time factor, its effectiveness is not high. The 2Q (Two queues) algorithm [25] is an improved approach based on the LRU-2 algorithm that divides the cached pages into 'cold' and 'warm', then maintains two FIFO queues to cache them. When a page is first accessed, it will be inserted into the cold page queue; if the accessed page is already in the cold queue, it may be considered a warm page and put into the warm page queue. Both of these algorithms are based on temporal locality. However, they are not suitable for all situations.

Moreover, the LFU (Least Frequently Used) algorithm [26] considers the access frequency of recent objects, taking full advantage of the historical scheduling information in the cache and removing the last least-visited objects from the cache. The LFU algorithm takes full advantage of the frequency characteristics of user access to preferred resources; however, it cannot distinguish between objects that are frequently accessed in the early stage or the later stage. In addition, it may keep "expired" objects to occupy the cache space, causing a serious cache pollution problem. The LFU-Aging algorithm is an improved optimization algorithm for LFU that solves one of its key problems, namely that the access frequency of resources in the cache is constantly increasing and never decreases. LFU-Aging [27] considers both the access frequency and the survival time of the resource in the cache, and therefore proposes that the value of access frequency is inversely proportional to the survival time. In the end, the access frequency of the long-lost resources in the cache will become smaller and smaller until it is eliminated. These two algorithms are based on spatial locality; however, there are some cases in which these algorithms cannot work.

Based on the spatial locality of resources, cache prediction/prefetching algorithms can predict which resource are to be accessed in the future using the current access, push the prediction content to the local cache, and replace the mapping item in the cache using the cache replacement algorithm. As the prefetching algorithm uses a prediction model to describe the mapping request, its prediction hit rate is closely related to that of the prediction model. At present, frequently used prefetching models include those based on data mining [28], multitask [29], Web semantics [30], the Markov model [24], probabilistic model [31], and so on. The prefetching model based on data mining predicts the users' potential next page by mining a large amount of potential information contained in the users' browsing history, while the model based on Web semantics extracts the feature key and analyzes user behavior to predict the next request. Moreover, the model based on the Markov model uses the transition probability matrix to describe the users' request behavior and thereby predicts the users' next request. However, the quality of these algorithms depends on the accuracy of the prediction algorithm. The feature key can be obtained using the method in [32].

III. MOTIVATION

To solve the delay problem, we first analyze the communication process of the Locator/Identity Separation network. Subsequently, we discuss the delay problem in LISP.

A. COMMUNICATION PROCESS OF THE LISP NETWORK

Much like common networks, end hosts in a LISP network can also be divided into two types: namely, fixed hosts and mobile hosts. In order to quickly distinguish between these two host types, the LISP protocol uses a specific EID segment to represent mobile nodes, while the remaining address segments are assigned to fixed nodes.

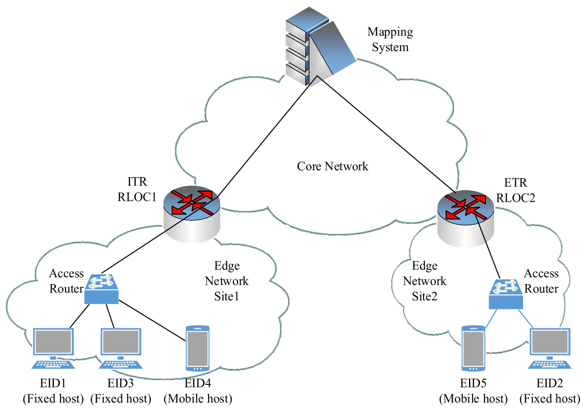


FIGURE 1. An instance of Location/Identity separation network.

Considering the fixed hosts managed by an access router, their EIDs can be allocated regularly and aggregated into one EID-prefix, which can be used for EID mapping resolution. The mapping system stores the EID-to-RLOC (for mobile hosts) or EID-prefix-to-RLOC (for fixed hosts) mapping entries. The RLOC is the edge router's IP address used for forwarding packets in the core network, which is usually related to the network topology. The RLOC can be obtained by querying the EID-to-RLOC mapping database in a router's local cache or mapping system, as shown in Fig.1.

In addition, LISP introduces the concept of Ingress Tunnel Router (ITR) and Egress Tunnel Router (ETR), which are used to encapsulate/decapsulate packets for identifier communication. ITR is the first-hop access router of the sender end host, while ETR is the last-hop access router of the receiver end host. ITR receives a LISP packet sent by one end host, which contains the EID of both communication parties in the form of a source and destination address in a LISP header of the packet. ITR uses the destination EID as the keyword to query the EID-to-RLOC mapping, either locally or remotely, in order to obtain the RLOC of the access router served for the destination end host. After that, ITR constructs a new packet and forwards it to the core network, encapsulating this LISP packet with an outer header in the process. For the outer header, the destination address is the RLOC of the access router at the destination, and the source address is the RLOC of the ITR. ETR receives this new packet encapsulated by ITR, strips the outer header, and forwards the inner LISP packet to the destination end host indicated by the destination EID.

If there is no mapping entry EID-to-RLOC of the destination EID in the ITR local buffer, it will wait for a long time to receive the mapping response from the mapping system after initiating a mapping query request to the mapping system; this increases the delay in the mapping resolution and affects the communication quality and performance. The key issue here is that of how to resolve the delay problem of mapping resolution. Some research has proposed a data-driven method to model the Internet route [10], adopted evolutionary game theory in the Internet of Vehicles [11], or used transformation-based process in IoT [12]. We consider deploying our work in a smart campus [13] and privacy issue [14] in the future.

B. THE DELAY PROBLEM IN LISP

In the study of LISP architecture, we mainly use the local mapping cache and identity mapping prediction/prefetching technology to solve the identity mapping resolution delay problem. The local mapping caching technology requires the edge router to cache the mapping entry of the identity obtained by mapping resolution in its local cache. Because this technology uses the temporal locality principle of mapping requests, it can thus reduce sending the mapping resolution request to the mapping system in order to reduce the mapping resolution delay.

However, given the increasing update frequency of network resources, the cache performance brought about by the upgrade cannot meet the performance needs. Accordingly, to further reduce the mapping resolution delay, mapping prediction/prefetching technology is introduced. This technology is used to forecast the future access according to the currently available historical information. However, successfully integrating the caching and forecasting technology to further reduce the mapping resolution delay remains a challenge.

As shown in Fig.1, host EID1 in an edge network Site1 wants to communicate with host EID2 in another edge network Site2. Their communication process can be described via the following steps:

- Host EID1 sends a LISP packet (packet1) to the edge router ITR of Site1, setting its own EID1 and the host EID2 as the source and destination identifier of packet1, respectively. According to the local buffered EID-to-RLOC mapping entries, the ITR of Site1 acquires the corresponding address RLOC of Host EID2 (for example, RLOC2 in Fig.1). If there is no local mapping entry for EID2, the ITR will send a mapping resolution request to the mapping system and store a new mapping entry in its local buffer after receiving the mapping response.
- The ITR encapsulates packet1 with an outer packet header to construct a new packet (packet2), which uses RLOC1 (RLOC of ITR) as the source address, and chooses RLOC2 (RLOC of ETR) as the destination address. Subsequently, the ITR forwards packet2 to the core network.
- This new packet (packet2) is routed to the ETR of Site2 through the core network, according to the destination address RLOC2.
- The ETR decapsulates this new packet (packet2) and sends the LISP packet (packet1) to Host EID2, according to the destination identifier EID2.

If no mapping entry EID-to-RLOC of EID2 is found in the local mapping buffer of ITR, the ITR will initiate a mapping query request to the mapping system; it may then have to wait for a long time to obtain the mapping response from the mapping system. This may consequently increase the mapping resolution delay and thus affect the communication quality and performance. If we could accurately predict the destination identifiers with which a user will communicate in the future, and if the mapping system supported pushing the mapping entries of certain identifiers to edge routers, this could effectively reduce the mapping resolution delay. Furthermore, we could also integrate caching and forecasting technology together to reduce this delay even further.

It should be noted here that Internet users in different edge networks may have different access interests. Depending on the social characteristics of the persons involved [33], a particular group of persons may exhibit particularly frequent patterns of access to certain network resources. For example, persons on a campus network may frequently visit Google Scholar or Baidu Academic to search for scientific papers. Similarly, if the access resources of the persons in two edge networks are similar, it is reasonable to push the resources frequently accessed by persons in one edge network to persons in another edge network. In the future, we will consider using the key management means in sensor networks [34], [35] and blockchain technology [36] to design an effective mapping management system, and will consider using deep learning [37]–[39] to deal with unknown issues.

IV. THE FIXED IDENTITY MAPPING PREDICTION ALGORITHM

In a locator/identity separation network, the edge router xTR (denoting either ITR or ETR) in an edge network receives a packet from its inner interface (that is connected to devices in the edge network). It will first query the local cache for the related EID-to-RLOC (for mobile EID) or EID-prefix-to-RLOC (for fixed EID) mapping entry for the destination identifier of the packet. If the local cache cannot find this related mapping entry, a mapping resolution request will be issued to the mapping system. Edge routers can record all resources accessed by persons in this edge network, since all the network's import and export flows pass through them.

In terms of recently recommended techniques and algorithms, collaborative methods based on collaborative filtering [40] have been widely recognized and promoted by researchers of late. The core idea behind these methods is to use group wisdom for prediction and recommendation, determine the relevance of users or items by using information, e.g. users' hobbies, and then making predictions and recommendations based on the correlation. Collaborative filtering is divided into user-based collaborative filtering [9], item-based collaborative filtering and matrix decomposition-based collaborative filtering.

In a locator/identity separation network, the users in an edge network have access preferences regarding network resources, causing some edge networks to have a correlation. We can use the idea of collaborative filtering to predict which resources in an edge network will be accessed next time and

TABLE 1. An array of request frequency data from edge routers.

xTR \ EID-prefix	D_1	D_2	D_3	D_4	…	D_n
S_1	H_{11}	H_{12}	H_{13}	H_{14}	…	H_{n1}
S_2	H_{21}	H_{22}	H_{23}	H_{24}	…	H_{n2}
S_3	H_{31}	H_{32}	H_{33}	H_{34}	…	H_{n3}
…	…	…	…	…	…	…
S_m	H_{m1}	H_{m2}	H_{m3}	H_{m4}	…	H_{mn}

thus push the corresponding identity mapping to edge routers in advance.

We use the collaborative filtering idea on the fixed end hosts case, and call this the Fixed Identity Mapping Prediction Algorithm (FIMPA). The idea behind FIMPA can be described as follows:

- collect the packets received by an edge router over a given period of time;
- using statistics of the packets, get the request frequency of different EID-prefixes over this period of time, then upload the request frequency information to the mapping system;
- the mapping system calculates the prediction model based on the collaborative filtering method, and makes a fixed identification prediction for the edge router based on the front prediction model;
- the mapping system actively pushes the correlation mapping entries to the edge router;
- the edge router updates its mapping entries according to the cache replacement policy.

A. SOME CRITICAL CONSIDERATIONS IN THE FIMPA ALGORITHM

1) THE FREQUENCY STATISTICS OF THE FIXED EID-PREFIX FROM THE HISTORY PACKETS

By using the request frequency statistics of the EID-prefixes recorded in an edge router over a given period of time, we can determine the data correlation, which is used as the basis for forecasting/prediction. It is therefore important to collect the packets received by an edge router over the specified period of time.

For each packet in the fixed hosts case, the edge router maps its destination identifier to the corresponding EID-prefix, and searches to determine whether there is an EID-prefix-to-RLOC mapping entry in the local cache. We can thus record the corresponding request frequency of the EID-prefixes accessed by the edge router over the specified period of time. The edge router then sends the EID-prefix request frequency data to the mapping system for centralized processing.

The mapping system builds an array according to the EID-prefix request frequency information from all edge routers in the edge networks in order to access different EID-prefixes. This array is presented in Table 1.

In this array, $\{\mathbf{S_1}, \mathbf{S_2}, \ldots, \mathbf{S_m}\}$ represents the set of $\mathbf{m}$ edge routers in the LISP network, while $\{\mathbf{D_1}, \mathbf{D_2}, \ldots, \mathbf{D_n}\}$ represents the set of $\mathbf{n}$ EID-prefixes in LISP network. The request frequency data of an EID-prefix accessed by an edge router is represented by a matrix $\mathbf{m^*n}$, where $\mathbf{H_{ij}}$ represents that the edge router $\mathbf{S_i}$ has accessed $\mathbf{H_{ij}}$ times on the EID-prefix $\mathbf{D_j}$.

2) MAKING PREDICTIONS BASED ON THE REQUEST FREQUENCY INFORMATION

After the mapping system collects the EID-prefix frequency information of all edge routers in the LISP network, it can carry out the prediction of EID-prefixes for the fixed end hosts based on collaborative filtering. More specifically, we use matrix decomposition-based collaborative filtering to handle all EID-prefix request frequency information.

The relationship between an edge router and an EID-prefix can be implied by assessing the number of times that the edge router accesses this EID-prefix. In other words, the higher the number of times that the edge router accesses this EID-prefix, the greater the likelihood that the edge router will access this EID-prefix next time.

In general, it is not possible for an edge router to access the whole set of EID-prefixes; thus, it can be seen that the request frequency of all EID-prefixes in one edge router is not all non-zero, meaning that the EID-prefix request frequency matrix $\mathbf{R_{m*n}}$ is sparse.

3) DEFINITIONS IN THE FIMPA ALGORITHM

The problem to be solved here is the calculation of the missing values of the request frequency matrix $\mathbf{R_{m*n}}$, after which prediction is carried out based on the missing values. We use a matrix decomposition method to calculate the missing values in the request frequency matrix. The idea here is that the request frequency matrix can be decomposed into the product of two small matrixes, with their product approximating the request frequency matrix. The request frequency matrix is sparse and contains a lot of zeros; however, the product of the small matrix is dense and complements the missing element. The key to FIMPA lies in the solution of the two small matrices.

The request frequency matrix is denoted by $\mathbf{R_{m\times n}}$; moreover, $\mathbf{R_{m\times n}}$ is approximated by the product of two small matrices $\mathbf{U_{m\times k}}$ and $\mathbf{V_{n\times k}}$: $\mathbf{R_{m\times n}} \approx \mathbf{U_{m\times k}V^T_{n\times k}}$, where $\mathbf{k}$ is much smaller than **m** and **n**. $\mathbf{U_{m\times k}}$ is the characteristic matrix of an edge router xTR, $\mathbf{V_{n\times k}}$ is the characteristic matrix of an EID-prefix, and **k** is the number of recessive factors. A target is the error generated by reconstructing **R** through **U** and **V**, which can be directly quantified. This can be done with other method, such as fast classification [41].

We use the Frobenius norm $||\mathbf{R} - \mathbf{UV^T}||_{\mathbf{2F}}$ to quantify the reconstruction error: that is, the score between each element and the reconstruction matrix. In fact, only non-zero frequency reconstruction error is required, meaning that the target or loss function can be written as follows:

$$\sum_{(i,j\in R)} (r_{ij} - u_i^T v_j)^2$$

For the reconstruction error, **R** is the set of (edge router, EID-prefix) corresponding to the non-zero request frequency, u_i^T is the implied eigenvector representing the edge router **i**, v_j is the implied eigenvector representing the EID-prefix **j**, and r_{ij} represents the request frequency of the edge router **i** on the EID-prefix **j**. Moreover, the inner product of the vector sum $u_i^T v_j$ is the approximation of the request frequency of the edge router **i** on the EID-prefix **j**.

In order to prevent over fitting, a regularization term is introduced, such that the loss function can ultimately be written as follows:

$$C = \sum_{(i,j)\in H} [(r_{ij} - (u_i)v_j)^2 + \lambda(||u_i||^2 + ||v_j||^2)] \tag{1}$$

Here, $\boldsymbol{\lambda}$ is the coefficient of the regularization term in (1).

The matrix **U** and **V** can be solved by minimizing the loss function. To calculate the value of the matrix **U** and **V**, we introduce the ALS alternating least squares method.

B. THE INTRODUCTION OF THE FIMPA ALGORITHM

An overview of the algorithm is presented in Algorithm 1. Some details of the implementation are as follows:

Step 1: randomly generate a $\mathbf{U^{(0)}}$, which can be set as the global mean.

Step 2: fix $\mathbf{U^{(0)}}$, and solve $\mathbf{V^{(0)}}$.

At this point, the loss function can be expressed via the following equation:

$$C = \sum_{(i,j)\in H} [(r_{ij} - (u_i^{(0)})v_j)^2 + \lambda(||u_i^{(0)}||^2 + ||v_j||^2)] \tag{2}$$

Fixing **j** (**j** = 1,2, …,**n**), the derivative of **C** is:

$$\frac{\partial C}{\partial v_j} = 2\sum_{i=1}^{m} [((u_i^{(0)})^T u_i^{(0)}) + \lambda)v_j - r_{ij}u_i^{(0)}] \tag{3}$$

Let $\frac{\partial C}{\partial v_j}$ =0; thus, we get

$$\sum_{i=1}^{m} [((u_i^{(0)})^T u_i^{(0)} + \lambda)v_j] = \sum_{i=1}^{m} r_{ij}u_i^{(0)} \tag{4}$$

Equation (4) is equivalent to

$$(UU^T + \lambda E)v_j = Ur_j^T \tag{5}$$

We define $\mathbf{M_1} = UU^T + \lambda E$, $\mathbf{M_2} = Ur_j^T$, so that (5) can be converted to

$$v_j = M_1^{-1}M_2 \tag{6}$$

$\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_n}$ are calculated in accordance with (6); thus, we obtain $\mathbf{V^{(0)}}$, which consists of $\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_n}$.

Step 3: fix $\mathbf{V^{(0)}}$, then solve $\mathbf{U}^{(1)}$.

The loss function at this time is:

$$C = \sum_{(i,j)\in H} [(r_{ij} - (u_i)v_j^{(0)})^2 + \lambda(||u_i||^2 + ||v_j^{(0)}||^2)] \tag{7}$$

Similarly, we calculate the value of $\mathbf{u_i}$:

$$\frac{\partial C}{\partial u_i} = 2\sum_{j=1}^{n} [((v_j^{(0)})^T v_j^{(0)}) + \lambda)u_i - r_{ij}v_j^{(0)}] \tag{8}$$

Let $\frac{\partial C}{\partial u_i} = 0$; thus, we get

$$\sum_{j=1}^{n} [((v_j^{(0)})^T v_j^{(0)} + \lambda)u_i] = \sum_{j=1}^{n} r_{ij}v_j^{(0)} \tag{9}$$

Equation (9) is equivalent to

$$(VV^T + \lambda E)u_i = Vr_i^T \tag{10}$$

We define $\mathbf{M_3} = VV^T + \lambda E$, $\mathbf{M_4} = Vr_i^T$, so that (10) can be converted to

$$u_i = M_3^{-1}M_4 \tag{11}$$

$\mathbf{u_1}, \mathbf{u_2}, \ldots, \mathbf{u_m}$ are calculated in accordance with (11); thus, we obtain $\mathbf{U^{(1)}}$, which consists of $\mathbf{u_1}, \mathbf{u_2}, \ldots, \mathbf{u_m}$.

The FIMPA algorithm loops the execution of Step 2 and 3, and stops after iterating **N** times. Following the execution, we obtain the optimal solution **U** and **V**. The sum of the optimal solution **U** and **V** complements the missing value for the request frequency matrix **R**. For the edge router x, we select the corresponding row of x from the matrix $\mathbf{UV^T}$ and sort the elements in the row from large to small after removing the original values. The larger the value, the greater the likelihood that edge router x will make an access request to this EID-prefix in the future.

We then take the Top-N EID-prefixes as the prediction for this edge router x.

The mapping system establishes a time-push mapping table timer. When the setting time is reached, the mapping system generates EID-prefix-to-RLOC mapping entries according to the FIMPA algorithm and actively pushes them to the edge routers. When the edge routers receive these EID-prefix-to-RLOC mapping entries, they will update the mapping cache according to their cache replacement strategy (e.g. LRU and LFU). It can be improved using semi-supervised learning [38] in the future.

The FIMPA algorithm decomposes the request frequency matrix into the product of two small matrices: the characteristic matrix of an edge router and the characteristic matrix of EID-prefixes. It then carries out the prediction based on the

Algorithm 1 FIMPA Algorithm

Input: request frequency matrix **R**, iteration number **T**, number of features **k**, an edge router **x**
Output: the prediction result set **S** for the edge router **x**
Data: **U** stands for the characteristic matrix of an edge router, **V** for the characteristic matrix of EID-prefixes, **map** for the key-value set of (EID-prefix **i**, prediction degree **p**)
The pseudo code of the FIMPA algorithm is as follows:

```
1. U←initM(k)
2. for i = 1 to T do
       V←calV(U)
       U←calU(V)
   end for
3. for each EID-prefix i in getUnvisited(R, u) do
       p←U[x]*V[i]
       map.add(i, p)
   end for
4. S←getResult(map)
```

complete value of the product of these two matrices. Algorithm 1 presents the pseudo code of the FIMPA algorithm; here, **initM(k)** initializes the feature matrix according to the number of features, **calV(U)** calculates **U** using **V**, **calU(V)** calculates **V** using **U**, **getUnvisited(R, u)** returns a collection of EID-prefixes that edge router **u** has not accessed, and **getResult(map)** obtains the prediction by sorting the results.

The correction of FIMPA algorithm can be guaranteed by the method of the matrix decomposition-based collaborative filtering, which is proved to be a feasible solution for the recommended problem.

V. EXPERIMENTS AND EVALUATION

In this paper, we implement the FIMPA algorithm with LRU and LFU as mapping replacement algorithms, then conduct comparisons with the existing LRU and LFU algorithms.

In this section, we evaluate the FIMPA algorithm using two indicators: namely, the cache hit rate and the hit rate convergence time. The cache hit rate is the ratio of the number of requests hits in the local cache to the total number of access requests in the time period.

The experimental data used in this simulation is real network traffic data collected from the Internet, namely the NLANR Auckland-VIII dataset, which represents the identity mapping request traffic. The data format of the dataset is ERF. In total, the dataset contains more than 6 million packets collected over 60 minutes. In order to evaluate the cache performance and use this dataset with the FIMPA algorithm, we need to map the source and destination IP addresses of the packets to the corresponding prefixes. We use the BGP prefix as the EID-prefix and download the BGP core routing table from Route Views, which maps the IP addresses to prefixes. In this paper, we used Java to implement the FIMPA algorithm and ran the simulation program on a desktop PC,

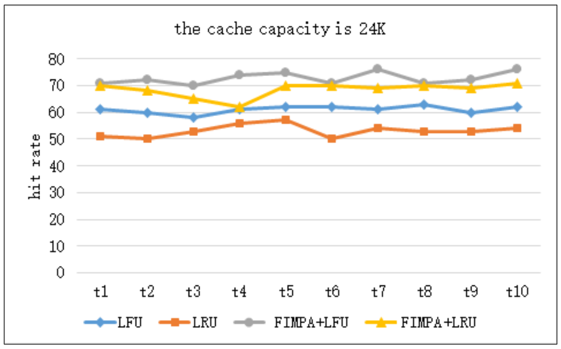


FIGURE 2. Hit rate results with 24K cache.

which has an Intel i5-7200U CPU, 8GB memory and the Windows 7 operating system.

According to the flow analysis statistics, the simulation sets the statistical period to 11 minutes, and further selects 60 seconds as the prediction time period (hereafter 'time period' for short). The FIMPA algorithm makes a prediction based on historical packets every time period. The packet in the first time period is the initial input data. The packets in the same time period are divided into different subsets according to the address prefix mapped by the source IP address. The different subsets represent the messages received by different edge routers. Subsequently, the packets in each subset are processed as follows: map the destination IP address to the address prefix; count the number of packets belonging to different prefixes; and simulate the frequency with which the edge router accesses the end host.

The simulator pushes all predicted results to the mapping cache of the edge router. In the initial case, the mapping cache table is empty and all predicted mappings are saved. For the second time period, the destination IP address of each packet is mapped to the BGP prefix. If the prefix is already in the mapping cache table, the simulator will increase the number of hits by one and process the next message; if no mapping is found in the mapping cache table, the simulator will record the mapping if the table is not full, or alternatively perform a mapping update using a mapping replacement algorithm if the table is full. After processing the second time period, the simulator continues to forecast and push the program forward using the second time period as the historical data. The subsequent 9 time periods of packet processing are similar.

A. EXPERIMENT FOR THE CACHE HIT RATE

In this paper, the size of the mapping cache table is set to 24K, 28K, and 36K respectively. We compare four algorithms (LFU, LRU, FIMPA+LFU, FIMPA+LRU) on the mapping cache table hit rate in the second to the eleventh periods, named t1 to t10, as shown in Fig. 2, Fig. 3, and Fig. 4.

As shown in Fig. 2, when the size of the mapping cache table is 24K, the mapping cache table hit rate when FIMPA+LFU is used as the mapping strategy is 5% higher than when LFU alone is used; moreover, this result is 6% when comparing FIMPA+LRU to LRU.

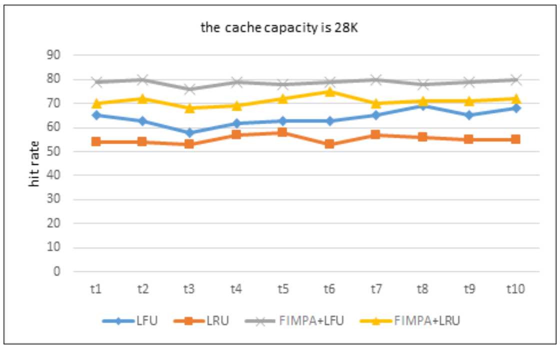


FIGURE 3. Hit rate results with 28K cache.

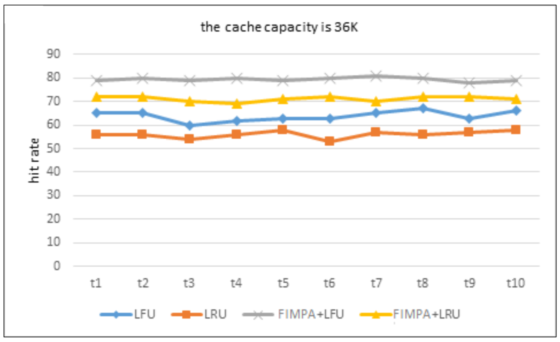


FIGURE 4. Hit rate results with 36K cache.

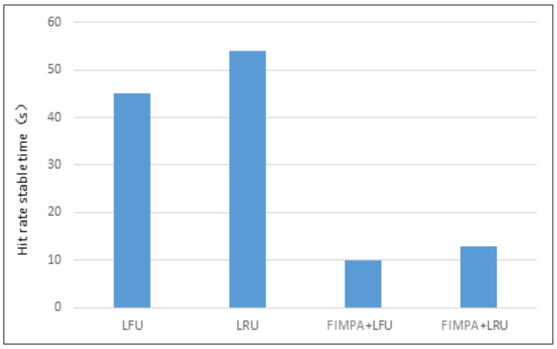


FIGURE 5. Hit rate convergence time.

Furthermore, when the size of the mapping cache table is set to 28K or 36K, the mapping cache table hit rate when FIMPA+LFU or FIMPA+LRU is used is 7% higher than when LFU or LRU alone are used.

B. EXPERIMENT FOR THE HIT RATECONVERGENCE TIME

In the second period, the mapping cache table is initially empty, and the hit rate of the mapping cache table gradually stabilizes from zero when new mappings are added into cache mapping table. The time duration from the initial state to the steady state is the convergence time. Figure 5 presents the comparison results of the convergence time for these four algorithms (LFU, LRU, FIMPA+LFU, FIMPA+LRU) while the cache map size is set to 24K. It can be seen from the figure that the convergence times for LFU and LRU are 45 seconds and 54 seconds, respectively. Due to the prediction processes and active push in FIMPA+LFU and FIMPA+LRU, the hit rate of the mapping cache table stabilizes quickly for these algorithms, resulting in a convergence time of only 10 seconds and 13 seconds respectively. In short, these results show that the convergence time achieved by FIMPA+LFU and FIMPA+LRU is a full 3 times faster than LFU and LRU.

In summary, for these two indicators, the FIMPA algorithm achieves better performance than the traditional algorithm. In particular, FIMPA algorithm achieves a hit rate convergence time that is 3 times faster than the comparison algorithms.

VI. CONCLUSION AND FUTURE WORK

By combining the FIMPA prediction algorithm with the replacement strategies (LRU and LFU), the hit rate of the cache mapping table can be significantly improved, especially in the initial state. When the cache mapping table is empty, the hit rate rapidly achieves higher stability in a short convergence time. However, FIMPA gains this improvement in hit rate by sacrificing the algorithmic performance, meaning that the time complexity and spatial complexity of FIMPA are relatively high and increase exponentially with the number of edge routers. To address some security issues in our work, we will consider using technologies such as blockchain [42], [43], the Tor network [44], covert communications [45], sensor networks [46], [47] and SDN networks [48] in our future work.

This paper aimed to solve the delay problem for the fixed end hosts; we will consider solutions for the mobile identity case in the future work. Possible solutions may include using a tree storage structure for mobile identity mappings or incorporating blockchain technology [49], SVM algorithm [50], among others.

ACKNOWLEDGMENT

The authors would like to thank Xunjun Zhang very much for his research work in the LISP delay problem, as well as Xianglilan Zhang for her help and useful suggestions in the revision of this paper. Also would like to thank all the anonymous reviewers for their insightful comments and suggestions, which will help to improve this paper.

REFERENCES

[1] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility," *Future Gener. Comput. Syst.*, vol. 25, no. 6, pp. 599–616, Jun. 2009.

[2] M. Armbrust, A. Fox, and R. Griffith, "Above the clouds: A berkeley view of cloud computing," Univ. California, Berkeley, CA, USA, Tech. Rep. UCB/EECS-2009-28, Feb. 2009.

[3] Y. Xiao, X. Du, J. Zhang, F. Hu, and S. Guizani, "Internet Protocol Television (IPTV): The Killer Application for the Next-Generation Internet," *IEEE Commun. Mag.*, vol. 45, no. 11, pp. 126–134, Nov. 2007.

[4] *Edge Computing*. Accessed: 2019. [Online]. Available: https://en.wikipedia.org/wiki/Edge_computing

[5] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet Things J.*, vol. 3, no. 5, pp. 637–646, Oct. 2016.

[6] Y. Yin, L. Chen, Y. Xu, and J. Wan, "QoS prediction for service recommendation with deep feature learning in edge computing environment," *Mobile Netw. Appl.*, to be published, doi: 10.1007/s11036-019-01241-7.

[7] *Fog Computing*. Accessed: 2019. [Online]. Available: https://en.wikipedia.org/wiki/Fog_computing

[8] I. Stojmenovic, S. Wen, X. Huang, and H. Luan, "An overview of fog computing and its security issues," *Concurrency Comput., Pract. Exper.*, vol. 28, no. 10, pp. 2991–3005, Jul. 2016.

[9] X. Tan and Y. Kim, "User acceptance of SaaS-based collaboration tools: A case of Google Docs," *J. Ent Inf. Manage.*, vol. 28, no. 3, pp. 423–442, Apr. 2015.

[10] Z. Tian, S. Su, W. Shi, X. Du, M. Guizani, and X. Yu, "A data-driven method for future Internet route decision modeling," *Future Gener. Comput. Syst.*, vol. 95, pp. 212–220, Jun. 2019.

[11] Z. Tian, X. Gao, S. Su, J. Qiu, X. Du, and M. Guizani, "Evaluating reputation management schemes of Internet of vehicles based on evolutionary game theory," *IEEE Trans. Veh. Technol.*, vol. 68, no. 6, pp. 5971–5980, Jun. 2019.

[12] H. Gao, Y. Duan, L. Shao, and X. Sun, "Transformation-based processing of typed resources for multimedia sources in the IoT environment," *Wireless Netw.*, early access, Nov. 2019, doi: 10.1007/s11276-019-02200-6.

[13] Z. Tian, Y. Cui, L. An, S. Su, X. Yin, L. Yin, and X. Cui, "A real-time correlation of host-level events in cyber range service for smart campus," *IEEE Access*, vol. 6, pp. 35355–35364, 2018.

[14] K. Yan, W. Shen, Q. Jin, and H. Lu, "Emerging privacy issues and solutions in cyber-enabled sharing services: From multiple perspectives," *IEEE Access*, vol. 7, pp. 26031–26059, 2019.

[15] J. Yu, Z. Kuang, B. Zhang, W. Zhang, D. Lin, and J. Fan, "Leveraging content sensitiveness and user trustworthiness to recommend fine-grained privacy settings for social image sharing," *IEEE Trans. Inf. Forensics Security*, vol. 13, no. 5, pp. 1317–1332, May 2018.

[16] Y. Zhang and Y. Zhou, "Transparent computing: A new paradigm for pervasive computing," in *Proc. Int. Conf. Ubiquitous Intell. Comput.*, Hubei, China, 2006, pp. 1–11.

[17] Y. Zhang, K. Guo, J. Ren, Y. Zhou, J. Wang, and J. Chen, "Transparent computing: A promising network computing paradigm," *Comput. Sci. Eng.*, vol. 19, no. 1, pp. 7–20, Jan. 2017.

[18] Y. Xu, Q. Zeng, G. Wang, C. Zhang, J. Ren, and Y. Zhang, "An efficient privacy-enhanced attribute-based access control mechanism," *Concurrency Comput., Pract. Exper.*, to be published, doi: 10.1002/cpe.5556.

[19] Z. Tian, W. Shi, Y. Wang, C. Zhu, X. Du, S. Su, Y. Sun, and N. Guizani, "Real-time lateral movement detection based on evidence reasoning network for edge computing environment," *IEEE Trans. Ind. Inf.*, vol. 15, no. 7, pp. 4285–4294, Jul. 2019.

[20] Y. Xu, G. Wang, J. Ren, and Y. Zhang, "An adaptive and configurable protection framework against android privilege escalation threats," *Future Gener. Comput. Syst.*, vol. 92, pp. 210–224, Mar. 2019.

[21] Y. Xu, G. Wang, J. Yang, J. Ren, Y. Zhang, and C. Zhang, "Towards secure network computing services for lightweight clients using blockchain," *Wireless Commun. Mobile Comput.*, vol. 2018, Nov. 2018, Art. no. 2051693.

[22] Y. Xu, J. Ren, Y. Zhang, C. Zhang, B. Shen, and Y. Zhang, "Blockchain empowered arbitrable data auditing scheme for network storage as a service," *IEEE Trans. Serv. Comput.*, to be published, doi: 10.1109/tsc.2019.2953033.

[23] A. Dan and D. Towsley, "An approximate analysis of the LRU and FIFO buffer replacement schemes," *SIGMETRICS Perform. Eval. Rev.*, vol. 18, no. 1, pp. 143–152, Apr. 1990.

[24] I. Zukerman, D. W. Albrecht, and A. E. Nicholson, "Predicting users' requests on the WWW," in *Proc. Int. Conf. User Modeling*, New York, NY, USA, 1999, pp. 275–284.

[25] Johnson, Theodore, Shasha, "2Q: A low overhead high performance buffer management replacement algorithm," in *Proc. VLDB*, Santiago, Chile, 1994, pp. 439–450.

[26] G. Karakostas, O. St, and D. N. Serpanos, "Practical LFU implementation for Web caching," Princeton Univ., Princeton, NJ, USA, Tech. Rep. TR-622-00, Jun. 2000.

[27] B. Feng, H. Zhou, and G. Li, "Least popularly used: A cache replacement policy for information-centric networking," *J. Internet Technol.*, vol. 17, no. 1, pp. 1–10, 2016.

[28] A. Nanopoulos, D. Katsaros, and Y. Manolopoulos, "A data mining algorithm for generalized Web prefetching," *IEEE Trans. Knowl. Data Eng.*, vol. 15, no. 5, pp. 1155–1169, Sep. 2003.

[29] J. Yu, C. Hong, Y. Rui, and D. Tao, "Multitask autoencoder model for recovering human poses," *IEEE Trans. Ind. Electron.*, vol. 65, no. 6, pp. 5060–5068, Jun. 2018.

[30] C.-Z. Xu and T. Ibrahim, "A keyword-based semantic prefetching approach in Internet news services," *IEEE Trans. Knowl. Data Eng.*, vol. 16, no. 5, pp. 601–611, May 2004.

[31] H. Gao, W. Huang, and X. Yang, "Applying probabilistic model checking to path planning in an intelligent transportation system using mobility trajectories and their statistical data," in *Proc. AUTOSOFT*, Jun. 2019, pp. 547–559.

[32] J. Yu, M. Tan, H. Zhang, D. Tao, and Y. Rui, "Hierarchical deep click feature prediction for fine-grained image recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, to be published, doi: 10.1109/tpami.2019.2932058.

[33] Y. Yin, J. Xia, Y. Li, Y. Xu, W. Xu, and L. Yu, "Group-wise itinerary planning in temporary mobile social network," *IEEE Access*, vol. 7, pp. 83682–83693, 2019.

[34] M.-L. Messai and H. Seba, "A survey of key management schemes in multi-phase wireless sensor networks," *Comput. Netw.*, vol. 105, pp. 60–74, Aug. 2016.

[35] X. Du, Y. Xiao, M. Guizani, and H.-H. Chen, "An effective key management scheme for heterogeneous sensor networks," *Ad Hoc Netw.*, vol. 5, no. 1, pp. 24–34, Jan. 2007.

[36] A. A. Omar, M. Z. A. Bhuiyan, A. Basu, S. Kiyomoto, and M. S. Rahman, "Privacy-friendly platform for healthcare data in cloud based on blockchain environment," *Future Gener. Comput. Syst.*, vol. 95, pp. 511–521, Jun. 2019.

[37] X. Chen, C. Li, D. Wang, S. Wen, J. Zhang, S. Nepal, Y. Xiang, and K. Ren, "Android HIV: A study of repackaging malware for evading machine-learning detection," *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 987–1001, 2020.

[38] K. Yan, C. Zhong, Z. Ji, and J. Huang, "Semi-supervised learning for early detection and diagnosis of various air handling unit faults," *Energy Buildings*, vol. 181, pp. 75–83, Dec. 2018.

[39] X. Yan, B. Cui, Y. Xu, P. Shi, and Z. Wang, "A method of information protection for collaborative deep learning under GAN model attack," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, to be published, doi: 10.1109/tcbb.2019.2940583.

[40] H. Gao, Y. Xu, Y. Yin, W. Zhang, R. Li, and X. Wang, "Context-aware QoS prediction with neural collaborative filtering for Internet-of-Things services," *IEEE Internet Things J.*, to be published, doi: 10.1109/jiot.2019.2956827.

[41] K. Yan, Z. Ji, H. Lu, J. Huang, W. Shen, and Y. Xue, "Fast and accurate classification of time series data using extended ELM: Application in fault diagnosis of air handling units," *IEEE Trans. Syst. Man Cybern, Syst.*, vol. 49, no. 7, pp. 1349–1356, Jul. 2019.

[42] Z. Tian, M. Li, M. Qiu, Y. Sun, and S. Su, "Block-DEF: A secure digital evidence framework using blockchain," *Inf. Sci.*, vol. 491, pp. 151–165, Jul. 2019.

[43] Y. Xu, J. Ren, G. Wang, C. Zhang, J. Yang, and Y. Zhang, "A blockchain-based nonrepudiation network computing service scheme for industrial IoT," *IEEE Trans. Ind. Inf.*, vol. 15, no. 6, pp. 3632–3641, Jun. 2019.

[44] Q. Tan, Y. Gao, J. Shi, X. Wang, B. Fang, and Z. Tian, "Toward a comprehensive insight into the eclipse attacks of Tor hidden services," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 1584–1593, Apr. 2019.

[45] S. Yan, Y. Cong, S. V. Hanly, and X. Zhou, "Gaussian signalling for covert communications," *IEEE Trans. Wireless Commun.*, vol. 18, no. 7, pp. 3542–3553, Jul. 2019.

[46] X. Du, M. Guizani, Y. Xiao, and H.-H. Chen, "Transactions papers a routing-driven elliptic curve cryptography based key management scheme for heterogeneous sensor networks," *IEEE Trans. Wireless Commun.*, vol. 8, no. 3, pp. 1223–1229, Mar. 2009.

[47] X. Du and H. H. Chen, "Security in wireless sensor network," *IEEE Wireless Commun. Mag.*, vol. 15, no. 4, pp. 60–66, Aug. 2008.

[48] L. Zhu, X. Tang, M. Shen, X. Du, and M. Guizani, "Privacy-preserving DDoS attack detection using cross-domain traffic in software defined networks," *IEEE J. Select. Areas Commun.*, vol. 36, no. 3, pp. 628–643, Mar. 2018.

[49] G.-J. Ra, D. See, M. Z. A. Bhuiyan, and I.-Y. Lee, "A study on anonymous protocol in a permission blockchain with ensure privacy for a member," in *Proc. 12th Int. Conf. Secur., Privacy Anonymity Comput., Commun. Storage*, Atlanta, CA, USA, Jul. 2019, pp. 456–464.

[50] K. Yan, Z. Ji, and W. Shen, "Online fault detection methods for chillers combining extended Kalman filter and recursive one-class SVM," *Neurocomputing*, vol. 228, pp. 205–212, Mar. 2017.

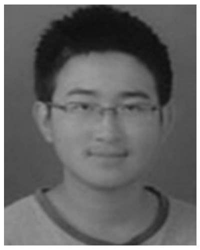
SHUO ZHANG received the B.S. and Ph.D. degrees in computer science from the National University of Defense Technology, Changsha, China, in 2008 and 2014, respectively. He is currently an Associate Professor with the Cyberspace Institute of Advanced Technology, Guangzhou University, Guangzhou, China. His research interests include computer networks, cloud computing, and distributed systems.

YAPING LIU received the B.S., M.S., and Ph.D. degrees from the College of Computer, National University of Defense Technology, China, in 1994, 1997, and 2006, respectively. She joined the College of Computer, National University of Defense Technology, as a Faculty Member, in 1997, and an Associate Professor promotion and a Full Professor promotion in 2004 and 2013, respectively. She is currently a Professor with the Cyberspace Institute of Advanced Technology, Guangzhou University, Guangzhou, China. Her current research interests include network architecture, interdomain routing, network virtualization, and network security.

SHUDONG LI received the M.S. degree in applied mathematics from Tongji University, China, in June 2005, the Ph.D. degree from the Beijing University of Posts and Telecommunications China, in July 2012. From 2013 to 2018, he held a postdoctoral position with the National University of Defense Technology, China. He is currently an Associate Professor with the Cyberspace Institute of Advanced Technology, Guangzhou University, China. His current researches include big data and its security, malware identification, information security and cryptography, and the robustness of complex networks.

ZHIYUAN TAN (Member, IEEE) received the Ph.D. degree from the University of Technology Sydney, Australia, in 2014. He was a Postdoctoral Researcher with the University of Twente, from 2014 and 2016. He is currently a Lecturer with the School of Computing, Edinburgh Napier University, U.K. His research focuses on cyber security, machine learning, data analytics, virtualization, and cyber-physical systems. He is EAI Member and BCS Member. His research contribution on cyber security is internationally recognized. He has received various Research awards, including the Best paper awards from the SecurIT'12 and SmartIoT 2019, the National Research Award 2017 from the Research Council of the Sultanate of Oman, the IEEE Outstanding Service Award and an Honourable Mention in SICSA Supervisor of the Year 2019 award, over the past years.

XIAOMENG ZHAO received the B.S. and master's degrees in optical engineering science from the National University of Defense Technology, Changsha, China, in 2003 and 2008, respectively. He is currently a Senior Engineer with the Guangdong Provincial Key Laboratory of High Performance Computing, Guangdong Science and Technology Infrastructure Center, Guangzhou, China. His research interests include optical communication, computer networks, and cloud computing.


JUNJIE ZHOU received the master's degree in optical engineering from Jinan University, Guangzhou, China, in 2017. He is currently an Information Engineer with Guangdong Science and Technology Infrastructure Center, Guangzhou, China. His recent research interests include management of science and technology, transfer learning, pattern recognition, and information security.