IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received February 16, 2018, accepted March 20, 2018, date of publication March 29, 2018, date of current version April 25, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2820719

scMPTCP: SDN Cooperated Multipath Transfer for Satellite Network With Load Awareness

ZHUO JIANG[1,2], QIAN WU[1,3], HEWU LI[1,3], AND JIANPING WU[1,3], (Fellow, IEEE)

[1]Tsinghua National Laboratory for Information Science and Technology, Tsinghua University, Beijing 100084, China
[2]Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China
[3]Institute for Network Sciences and Cyberspace, Tsinghua University, Beijing 100084, China

Corresponding author: Hewu Li (lihewu@cernet.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 91738202 and in part by the Beijing Municipal Science and Technology Commission under Grant Z171100005217001.

ABSTRACT Satellite networks are multilayered, and the number of satellites in a single constellation is also increasing. These characteristics make satellite network more suitable for multipath transmission like multipath transmission control protocol (MPTCP). With the use of MPTCP, the bandwidth of different satellite channels can be aggregated, and the mobility performance of users can be improved. Furthermore, software defined networking (SDN) is introduced to the MPTCP to solve the shared bottleneck problem. However, the performance of existing scheme is still far from optimal. The main problems include: (1) static number setting of MPTCP subflows on a per host basis and (2) unaware of the traffic load during the subflow route selection. The shared bottleneck problem is more serious in the satellite network with lattice-like topology. To solve the above problems, we propose an sdn cooperated MPTCP (scMPTCP) architecture and its related algorithms. We extend TCP options to piggyback the relevant control information to flexibly support the communication between the subflows of transport layer and SDN controller. Then, we propose a load and shared bottleneck aware subflow route selection algorithm and adjust algorithm. These two algorithms select routes for new subflows based on the available bandwidth of each route and avoid the bottleneck of other subflows, and also can adapt to the changes of network load. We implement the scMPTCP and its algorithms. The evaluation results show that compared with searching over the non-overlapping paths or shortest paths schemes, our scheme can achieve much higher total system throughput. Moreover, by adjusting the subflows which share bottlenecks, the total aggregated throughput of that connection is also improved greatly.

INDEX TERMS Multipath transfer, multipath transmission control protocol (MPTCP), satellite networking, software defined networking (SDN).

I. INTRODUCTION

Satellite network has the advantage of large coverage and destroy resistant to terrestrial disaster. Recently with the reduced cost of making and launching satellites, building a satellite network with global coverage capability has become the focus of many large companies including SpaceX [1], Oneweb [2], O3b [3] (invested by SES [4], Google and etc.) and so on. This new trend of satellite networking has two main characteristics: The first one is that their proposed schemes usually contain much more satellites than previous satellite constellation. For example, the SpaceX constellation contains hundreds of satellites while Iridium constellation contains only tens of satellites. The second one is that satellite constellations are multilayered. For example, Oneweb constellation makes use of LEOs, while O3b constellation makes use MEOs.

The above characteristics make satellite network more suitable for multipath transmission like Multipath transmission control protocol (MPTCP) [6]. MPTCP is a popular and standardized multipath transfer protocol that works at transport layer. It makes use of TCP option extensions to support simultaneous transmission over multiple end-to-end paths. And it is compatible to TCP applications. As the satellite constellations contain more and more satellites and they are already multilayered at the same time, more and more users will be simultaneously covered by multiple satellites. This will boost the adoption of MPTCP in satellite network. With the use of MPTCP, the bandwidth of different satellite channels can be aggregated, and the mobility performance of users can be improved.

However, without the cooperation between end host and network, the route selection of different subflows in the same

MPTCP connection may result in inefficient overall throughput. For example, OSPF-ECMP is a typical load balancing scheme. but when it is used together with MPTCP, the main drawback is that it just makes use of transport layer five tuples to schedule flows, which may lead to the result that different subflows of the same connection go through the same bottleneck. As satellite network has lattice-like topology, this kind of shared bottleneck problem will become more common and serious.

Several schemes [8]–[13] have been proposed to combine MPTCP and software defined networking (SDN) to solve the shared bottleneck problem and achieve effective subflow route selection, but their performance is still far from optimal. The main problems are as following:

1) Static number setting of subflows on a per host basis. The two common subflow selection schemes used by MPTCP are fullmesh and ndiffports [5]. Both schemes support the establishment of multiple subflows. But the number of subflows is preconfigured, and cannot change dynamically according to the different network condition, e.g. the occurrence of shared bottleneck. Some other schemes require long live TCP connection between end host and switch or controller, which will consume the resource of controller and switch and are more likely to cause security problems.

2) Unaware of the traffic load during the subflow route selection. Several works [8], [9] solve the shared bottleneck problem by searching for non-overlapping paths without considering the traffic load on each path. Sometimes overlapping but none congested paths may bring large performance improvement, but this situation is left unconsidered. Moreover, when network load changes, existing subflow route selection schemes cannot adapt and avoid shared bottleneck accordingly.

To solve the above problems, and especially for the satellite network with lattice-like topology, we propose scMPTCP (sdn cooperated MPTCP) architecture and its related algorithms. The main contributions are as following:

Firstly, we extend SDN to end host to flexibly support the communication and control between the subflows of transport layer and SDN controller. The key of our scMPTCP architecture is extending TCP options to piggyback the relevant information. Thus, in scMPTCP, SDN controller can dynamically control the subflow numbers of each MPTCP connection.

Secondly, we propose two load and shared bottleneck aware algorithms: mptcp-lba is the subflow route selection algorithm used in MPTCP connection establishment stage, and mptcp-lba-adjust is the subflow route adjustment algorithm used when share bottleneck is detected. Both the above two algorithms select routes for new subflows based on the available bandwidth of each route and avoid the bottleneck of other subflows. And they consider routes by taking the satellite lattice-like topology into consideration other than just shortest path routes.

Finally, we implement scMPTCP and these two algorithms in MPTCP Linux kernel [7] and ryu SDN controller, and then evaluate scMPTCP over satellite lattice topology with Mininet. Evaluation results show that compared with searching over the non-overlapping paths or shortest path scheme, our scheme can achieve much higher total system throughput. Moreover, by adjusting the subflows which share bottlenecks, the total aggregated throughput of that connection is also improved greatly.

II. RELATED WORKS

A. SDN AND MPTCP

In [8] and [9], SDN is used to allocate disjoint paths to different subflows. Reference [10] allocate shortest paths, k shortest paths, k edge disjoint paths to MPTCP subflows under hop limits. For these works, although several subflows can be established for the same flow, the number of subflows cannot be dynamically adjusted after establishment. Moreover, when selecting paths for different subflows, the traffic information on each paths is not considered. Some paths which may provide large bandwidth increase may not be selected.

When given the bandwidth demand of each user, [11] seeks to allocate satellite bandwidth to each while balancing system load. Reference [11] try to meet use's bandwidth demand by making use of residual bandwidth on each link. Their traffic model is not suitable for MPTCP as MPTCP will always saturate the bottleneck link.

To dynamically adjust MPTCP subflow number, [12] set up long live connection between end host and controller. Reference [13] set up long live connection between end host and switch, and the switch transfer the controller message to host. Both scheme adds extra burden to switch or controller. They bring extra security issue to switches or controllers. And they can only modify the subflow number by host, they cannot modify subflow number for each connection on one host.

Reference [14] combines segment routing and MPTCP for datacenter network to save flow table on open flow switch. But taking up segment routing requires further modification of openflow switch. Moreover, when network state changes, it will not adapt accordingly.

B. SDN OVER SATELLITE NETWORK

With the key components of centralized control and separation of control and data plane, a lot of beneficial scenarios have been found for SDN usage in terrestrial network. Originally, SDN is mainly used for wired network and the communication between switch and controller. A typical use case is the WAN traffic engineering designed by Google [15]. Schulz-Zander *et al.* [16] extends SDN to wireless side to support flow level traffic statistics at wireless side and seamless migration of user terminals among adjacent APs. meSDN [17] extends SDN to end host so that 802.11 uplink wireless resources can be managed by the controller. And [18] explores the useful cases of security by extending SDN to end host. Both [17] and [18] require long live connection between end host and controller, and consume lots of controller computing resources.

As for using SDN in satellite network, [19] proposed a SDN based architecture for multi-layer satellite network. In that architecture, control plane in GEO is in charge of controlling the data plane in other LEO/MEO, at the same time the control plane is responsible for executing commands from management plane on the ground. SAT-FLOW [20] optimizes the idle timeout parameter of SDN flow table by considering application qos (quality of service) level and end host handover. SERvICE [21] makes use of SDN to do qos routing and bandwidth allocation. And the benefits of SDN and NFV architecture over satellite and terrestrial integrated network are explored by [22] and [23]. Our work in this paper differs from these paper in that we are focused on the cooperation between SDN and end host for MPTCP optimization.

C. SUBFLOW SELECTION SCHEMES WHICH WORK ON END HOST ONLY

When the network characteristics of different subflows have large differences, the buffer blocking problem [24] may arise. And the total aggregated throughput is reduced due to the existence of subflow which has low performance. Reference [25] makes use of a multipath throughput model which considers both network characteristics and receiver buffer to do subflow selection. Subflow with smaller RTT (Round Trip Time) is selected with higher priority. Reference [26] selects subflow with the consideration of video playout deadline and priority.

For small file transfer like web pages, even though the sender may not be blocked by receiver buffer limit, by allocating data over low performance subflow may bring a large percentage of idle time of the high performance subflow when tail data is allocated to low performance subflow. Reference [27] makes use of large amount of offline data which contains the information of single path and multipath performance, then it uses machine learning to decide in current network configuration and application file size, the suitable transmission scheme (single path or multipath) to use.

Since all these works are end host based subflow selection, they have no information of network congestion state and routes, and cannot adjust subflow routes to further improve end host performance.

III. DEEP ANALYSIS OF MPTCP OVER SATELLITE NETWORK

The subflow route selection scheme plays an important role in improving MPTCP performance. We analyze two of the common subflow route selection schemes here and show that existing schemes still have much improvement space. The two selected schemes are: disjoint route (mptcp-disjoint) [8] and k shortest paths (mptcp-sp) [10]. These two schemes select the subflow route based on network topology without considering the network load.

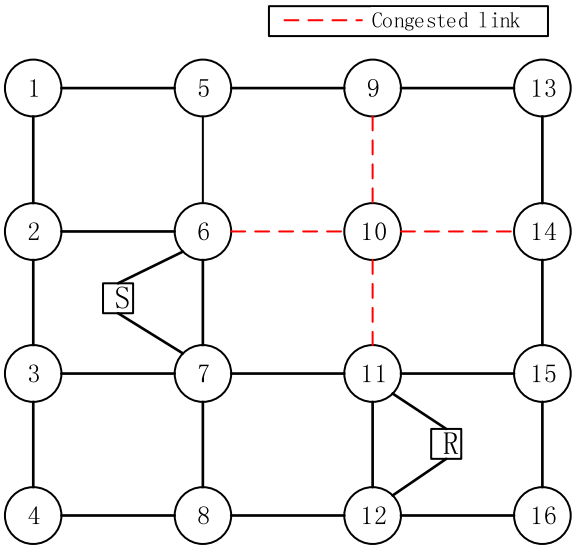

FIGURE 1. A lattice topology of satellite network with four congested links.

A. THE IMPACT OF BACKGROUND TRAFFIC ON SUBFLOW ROUTE SELECTION

Consider the lattice topology shown in Fig.1, two end hosts (represented by square blocks) are communicating over a lattice topology (represented by circles). Satellite 10 is heavily loaded and this results in four congested links which include link (9,10), (6,10), (10,11), (10,14). Other links in the network topology are equally lightly loaded. Both the sender and the receiver have two network interfaces and are connected with two satellites. And MPTCP full mesh subflow management will set up four subflows in total.

To select subflow routes, one of the results for the k shortest paths scheme will (6,10,11), (6,10,11,12), (7,11), (7,11,12). To make the result simple to calculate, here we suppose that when multiple shortest paths are available, the paths which have the most horizontal hops are selected. From the route selection results, we can see that for path (6,10,11) and path (6,10,11,12), they share the same congested links (6,10) and (10,11). Moreover, many other lightly loaded links can be used to further improve bandwidth performance are not selected.

As for the disjoint route selection scheme, during the route selection for new subflow, shortest path from the source to destination are searched in a new network graph. In the new network graph, the links which have been allocated to other subflows are removed. And the selected routes for each subflow can be (6,10,11), (6,5,9,13,14,15,11), (7,11), (7,8,12). Since this scheme searches for link disjoint routes for each subflow, all four subflows do not share any bottleneck. However, for path (6,10,11), it contains one heavy load link, this can be avoided if link load is considered when selecting routes.

B. THE NEED TO REROUTE AND COMBINE SUBFLOWS

Since the network load is frequently changing, the congestion link is also constantly changing. We consider the same

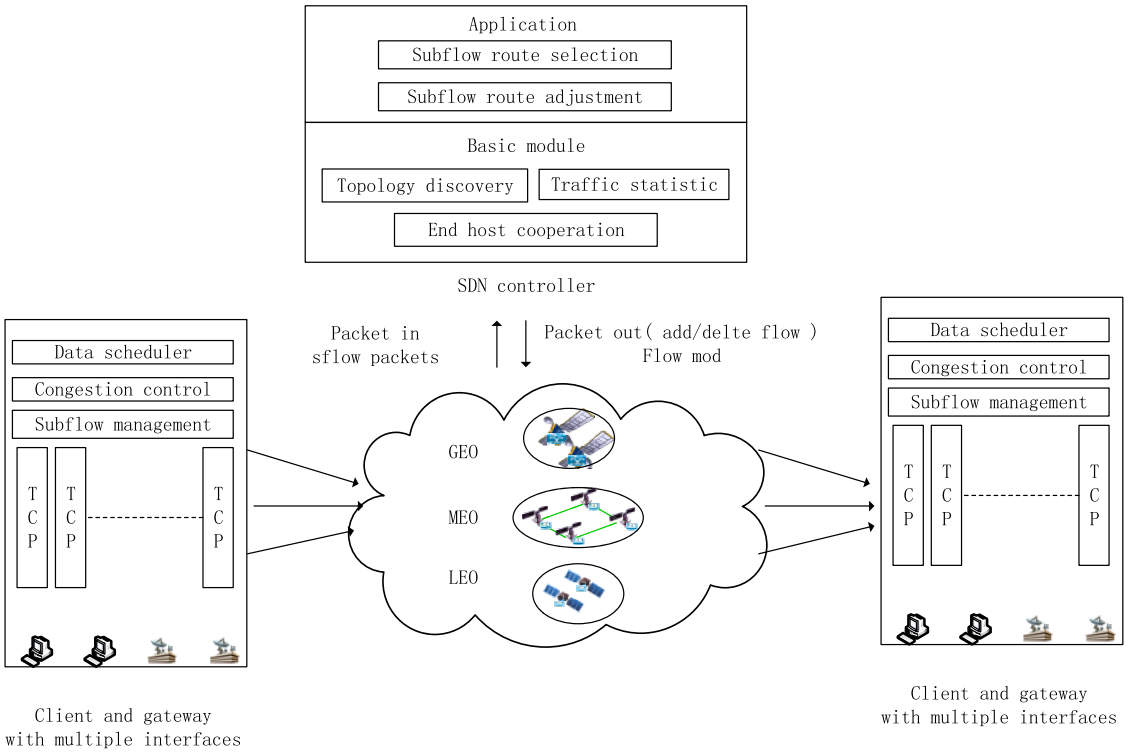

FIGURE 2. scMPTCP architecture.

topology as Fig 1, but with different link load. For example, at first link (7,11) and (11,12) are lightly loaded, and then they change to congested state.

We suppose during the light load time, the network select route (7,11) and (7,11,12) for two subflows. And this kind of route selection performs relatively well when link (7,11) and (11,12) are lightly loaded and the host network card is the bottleneck. However, when the load of link (7,11) and (11,12) changes from lightly loaded to congestion, the routes of these two subflows need to be adjusted to get a better bandwidth aggregation performance.

IV. scMPTCP ARCHITECTURE DESIGN

A. OVERVIEW OF scMPTCP ARCHITECTURE

The scMPTCP architecture is shown in Fig.2. The architecture mainly contains two parts: the controller modules and end host modules. And these two parts are cooperated with each other.

In the controller, apart from some basic functions, an end host cooperation module is added as the basic module. The end host cooperation module provides interface for controller application to send control information to end host. And this module cooperates with subflow management module at the end host to decide the number of subflow of a MPTCP connection at end host. When the packet in message corresponding to the packet sent from receiver to sender is received, the control information is piggybacked in newly defined TCP option in the packet out message sent by controller and will arrive at the end host.

Two new SDN applications are added to the controller. One is subflow route selection, the other is subflow route adjustment. Subflow route selection works during subflow establishment process, the relationship of different subflows which belong to the same connection are detected by looking into the MPTCP related options. To make efficient selection of network route, network load and network path other than shortest path but under certain hop limits are considered. While subflow route adjustment module works by monitoring network bottleneck in a predefined interval and adjusting subflow route selection accordingly to avoid shared bottleneck and increase aggregate throughput.

At the end host side, the subflow management module of MPTCP is modified. When the new path control information arrives at end host, the end host will increase or decrease subflow number correspondingly.

B. OPTIMIZATION MODEL FOR THE PROBLEM OF SUBFLOW ROUTE SELECTION

$$\max\left(\sum_{i=1}^{N}\log\left(\sum_{j=1}^{K_i}x_{ij}\right)\right)$$

$$\begin{cases}\sum\limits_{e\in p_{ij}x_{ij}} \le c_e \ (1)\\ x_{ij}\le x_{ij}y_{ij}\ (2)\\ \sum\limits_{j=1}^{K_i} y_{ij}\le k\ (3)\\ y_{ij}\in\{0,1\}, x_{ij}\ge 0\ (4)\end{cases}$$

The optimization problem is given in above formula. The goal is, for each user i, select at most k paths from the path union which contains K_i paths in total, so that the total system utility is maximized. The utility function is log based which can balance system efficiency and fairness between each user. The log based utility optimization is first proposed by Kelly and expanded to multipath transmission by Huaizhong Han. Based on Han's model, we add new path number constraints.

The variable x_{ij} is the bandwidth of user i over path j. And y_{ij} indicates whether path j of user i is selected. So y_{ij} is a integer variable which can choose the value 0 or 1. Constraint (1) is the link capacity constraint. Constraint (2) leads x_{ij} to be 0 when y_{ij} is 0. Constraint (3) is the maximum path number constraint for each user.

Since the variable y_{ij} is integer variable, this optimization problem belongs to MINLP (Mixed Integer Non Linear Programming). And MINLP is shown to be NP hard in [28]. So we design heuristic algorithm to solve the optimization problem.

C. CONTROLLER MODULES OF scMPTCP

1) SUBFLOW ROUTE SELECTION IN MPTCP CONNECTION ESTABLISHMENT STAGE

Subflow route selection module selects network route for each subflow during the three-way handshake of subflow connection set up process. This module makes use of the information from topology discovery module and traffic statistic module.

The basic functions include: 1) Calculate the link weight for network topology; 2) Get the bottleneck of previous established subflow. Based on the result of 1) and 2), the mptcp-lba algorithm selects subflow route based on link weight under subflow total number limit and avoid sharing bottleneck with previous subflows. If more than one path is available, then notify the connection to initiate more subflows.

a: CALCULATE THE LINK WEIGHT FOR NETWORK TOPOLOGY

To estimate the available bandwidth for a new flow going through a specific network interface, we use the following formula. The parameter n is the number of active flows on that network interface.

$$\text{available\_bandwidth} = \max\{\text{left\_bandwidth}, \frac{\text{total\_bandwidth}}{n+1}\}$$

This formula combines left bandwidth and average bandwidth together, as simply use either one of them can not estimate the available bandwidth accurately. Take the topology shown in Figs. 3 and 4 as an example, host pairs (1,2), (3,4), (5,6), (7,8) set up four MPTCP flows over the topology. The bandwidth of subflow which passes router 1,2 of flow (1,2) is 6 in the upper topology, while total_bandwidth/4=2.5 and lef_bandwidth=6. The bandwidth of the same subflow in the lower topology is 2.5, while the total_bandwidth/4=2.5 and left_bandwidth=1.

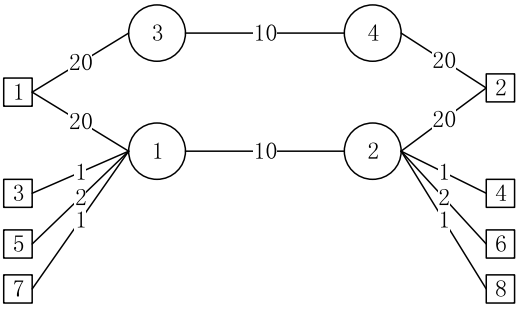

FIGURE 3. Example topology for link weight calculation (with large vacant bottleneck bandwidth).

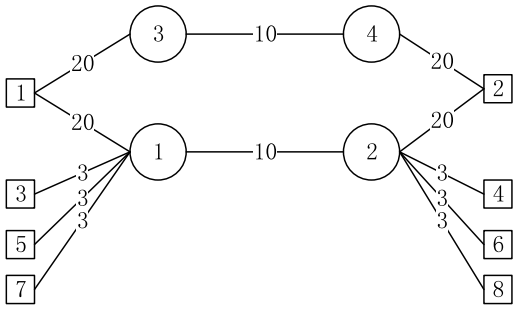

FIGURE 4. Example topology for link weight calculation (with small vacant bottleneck bandwidth).

b: CALCULATE THE BOTTLENECK LINK OF PREVIOUS SUBFLOWS

Link utilization is used to identify the bottleneck of each subflow. The link utilization is collected for every certain period of time. If it is above a predefined threshold, then this link is marked as bottleneck.

Every time a path is allocated for a subflow, the path information and the subflow relationship are stored in memory. When a new subflow initialization message is detected by the controller, the paths of previous established subflows can be found by looking up the stored data. And the links which are bottleneck and belong to the paths of previous subflows are avoided for further subflow route calculation.

c: MPTCP-LBA: LOAD AND BOTTLENECK AWARE SUBFLOW ROUTE SELECTION ALGORITHM

The load and bottleneck aware subflow route selection algorithm (mptcp-lba) we proposed is shown in table 1. The algorithm mainly contains two steps. In the first step, the routes from source to destination are found by breadth first searching. The breadth first search stops when all the nodes in the network have been visited or the hop count of new routes will be greater than the hop count of shortest path plus H (H is selected to be three as adding three extra hops can help bypass one congestion link or congestion satellite in lattice-like topology). After the breadth first search process, the bandwidth of subflow over each path is also calculated.

In the second step, the bottleneck links of previous subflows are found and the path with the highest available bandwidth and does not contain the bottleneck link is selected.

TABLE 1. mptcp-lba: load and bottleneck aware subflow route selection algorithm.

```
1.  Subflow setup packet in received
2.  Get network traffic statistics
3.  Construct graph based on weight and network
    topology
4.  Get available network paths from source to
    destination
5.  Sort all paths based on its estimated bandwidth
6.  For path of previous established subflow
7.    For link in path
8.      If link is bottleneck
9.        bottleneck_list.append(link)
10. For path in sorted all available paths
11.   For link in path
12.     If no link in bottleneck_list:
13.       all_path.append(path)
14. If len(all_path)==0
15.   Instruct host to terminate current connection
16. Elif len(all_path)==1:
17.   Pop the only path of all path and allocate it for
    current subflow
18. Else:
19.   Pop the first path of all path and allocate it for
    current subflow
20.   Instruct the host to set up len(all_path) -1 more
    subflows
```

After the selection of subflow in second step, the graph is updated by updating the link weight and removing the bottleneck links of the subflow. Then the first step is repeated until the number of subflow reach the limit or no new routes are available.

The subflow routes are calculated and cached when the first source destination IP packet pair arrives at the controller. And if no route is available for current subflow, the controller instructs the end host to terminate current subflow. If more than one path is available, the controller instructs the end host to set up more subflows based on the source destination IP pair but with different source destination port number. And controller will allocate paths for these subflow by looking up its cached paths.

2) SUBFLOW ROUTE ADJUSTMENT WHEN LOAD CHANGES

Since network traffic is dynamically changing, the previously light load paths may later become heavy congested. It could lead to that subflows which previously do not share any bottleneck may later share some bottlenecks. So the network load should be monitored and the path of subflows that share the bottleneck should be adjusted accordingly.

The subflow route adjustment algorithm is shown in Table 2. Different subflow which shares the same bottleneck may have different hop count, the larger the hop count is, the more network bandwidth it will take. So the subflow with larger hop count should be adjusted at first. During each network load statistic interval, we identify the bottleneck of network based on the loss count and link utilization of each link, then we iterate over all the subflows' paths cached by the controller to find if any subflow shares bottleneck. The new route calculation for the shared bottleneck subflow is the same to mptcp-lba. If new route cannot be found, the controller should instruct the end host to stop the subflow correspondingly.

TABLE 2. mptcp-lba-adjust: subflow route adjustment algorithm.

```
1.  Get network traffic statistics
2.  Get network bottleneck link according to link loss rate
    and utilization
3.  For connection in all_connection
4.    For subflow in all subflow
5.      If bottleneck_link in subflow
6.        Bottleneck_dict[bottleneck_link].append(subflow)
7.  For (bottleneck_link,subflow_list) in bottleneck_dict:
8.    If len(subflow_list)>1
9.      Sort subflow_list by subflow path hop length
10.     Reallocate route for subflow in subflow_list and
    only keep the last subflow's route unchanged
```

TABLE 3. Subflow management procedure at end host (acts according to the information sent from SDN controller).

```
Send procedure
1.  Master subflow established
2.  MP_ADD_ADDR received
3.  Send SYN packet over all source destination IP pair

Receive procedure
1.  SYN packet received
2.  If path control option received
3.    Send back path control option over master subflow

1.  SYN/ACK packet received
2.  If RST is marked in the SYN/ACK packet
3.    Stop that subflow
4.  Else
5.    update subflow count
6.    Read the new MPTCP option and increase or
    decrease subflow number accordingly
```

3) END HOST COOPERATION

The main function of end host cooperation is to cooperate with end host to do subflow management including: instructing end host to increase or decrease the number of subflows and detecting subflows which belong to the same MPTCP connection.

a: INSTRUCT END HOST TO INCREASE OR DECREASE THE NUMBER OF SUBFLOWS

To instruct end host to increase or decrease the number of subflows, two new MPTCP options are defined. The first

option is used to pass path information to controller. This option contains path identification and the maximum subflow number per connection allowed by each host and is added to subflow connection set up packet. The path identification will be used by controller to map to the subflow TCP five tuple. And this path identification will be used by controller to notify end host to stop subflow or increase subflow number based on the source destination information of the path identification.

The second option contains the information of flag, number, and path identification. Flag is used to identify whether to increase or decrease subflow number based on path identification information. And number is the corresponding increase or decrease value. When controller receive the packet in of TCP packet with SYN ACK marked, it adds the option to corresponding packet out and instruct the end host to respond accordingly.

b: DETECT SUBFLOWS WHICH BELONG TO THE SAME MPTCP CONNECTION

To detect the relationship of subflows which belong to the same MPTCP connection, we use the method introduced by [8] which uses MPTCP options carried in SYN packets to solve the problem. The first subflow carries in three-way handshake packets with MP_CAPABLE option and exchanges both sender's key and receiver's key. The sender's key and receiver's key then are used to generate tokens which identify a unique subflow at both the sender and receiver side. Later subflow with MP_JOIN option with the same token in it will be grouped into the same MPTCP connection.

4) TOPOLOGY DISCOVERY

The topology discovery module is the basis of subflow route selection. Typically, network topology is represented as a graph with nodes (switches and hosts) and edges (the link between switches or hosts). The satellites and connections between satellites can be discovered with LLDP and the connections between end hosts and satellites are discovered by detecting ARP packets.

5) TRAFFIC STATISTIC

The traffic statistic module is responsible for periodically recording the transmission speed of each flow on each port of satellite and getting the total capacity of each port. To do traffic stating, typically two ways can be chosen: making use of sflow [29] or making use of openflow flowstats. We choose the first one, as using openflow flowstats message requires install a flow record for each flow, this takes up a lot of flow table resources. Moreover, by making use sflow, we can make controller simpler and lighter as the code of sflow and controller are independent and run in different process.

D. END HOST MODULES OF scMPTCP

Subflow management module is responsible for setting up and closing subflows. This module works in cooperation with controller. By making use of the default packet in and packet out mechanism of SDN, we omit the need to set up long live TCP control channel between end host and the controller or the switch.

Subflow management procedure at end host is shown is Table 3. At the sender side, one parameter can be configured for subflow management: maximum number of subflows for one MPTCP connection. After the setup of master subflow, each source-destination IP pair set up one subflow at first as usual. In the connection setup packet, new option which contains path identification and maximum subflow number allowed will be added.

When the sender receives new path control information (the path control information is contained in SYN/ACK packet), it will read path control information from it. If the path control information instructs the sender to add new subflow, the sender will initiate new subflow based on the path identification received. If the path control information instructs the sender to stop existing subflow, the sender will send FIN packet over that subflow.

When the receiver receives new path control information (the path control information is contained in SYN packet), the path control information will be send back to the sender over the master subflow.

The data scheduler module is responsible for splitting the traffic of the same application over multiple TCP subflows. The congestion control module is responsible for sharing network resources fairly and effectively. For simplicity and comparison, we adopt MPTCP default data scheduler min-RTT and default coupled congestion control LIA.

V. PERFORMANCE EVALUATION

A. SYSTEM IMPLEMENTATION

The applications and basic modules of controller are implemented in ryu 4.17. Ryu and floodlight are two major controllers used in simulation, we choose ryu as its learning curve is lower than floodlight and can satisfy our simulation needs. The controller makes use of LLDP to discover switches and their connection relationship and learns user's location by proactively listening to ARP packets. During each packet in event, the subflow route selection algorithm is called. And the controller monitors the link status by contacting with sFlow server each predefined interval, if more than one subflow is found to share bottleneck, then the subflow route adjustment algorithm is called. The flowmod packets send to the switches to modify flow table is at the granularity of TCP flow. So the first two packets which contain SYN and SYN/ACK will be send to the controller. If needed, the path control information will be piggybacked to SYN/ACK packet and send back to the sender.

The multipath transmission at the end host is based on MPTCP 4.9.61. The modification to MPTCP is two parts: The first part is when sending SYN packets, after all the options are written, adds path control option which contains the path identification of this subflow and the maximum number of subflows the end host allows. The second part is when receiving packets, checks if the packets contain path

control information. If path control information is detected, increase or decrease subflow accordingly.

Mininet 2.2.1 is used to construct network topology and communicates with controller. All the software listed above runs in a VMware virtual machine. The CPU and memory resources allocated for the virtual machine is 6 core 2.2GHz CPU and 7672 MB.

B. EVALUATION TOPOLOGY AND TRAFFIC GENERATION

The evaluation topology is 4∗4 lattice topology, which is the same as in Fig.1. The link delay between two satellite routes is set to 20ms and the bandwidth is set to 25Mbps as in [30]. The link delay and bandwidth between normal end host and satellite is set to 5ms and 5Mbps.

Two different traffic patterns are constructed to demonstrate the effectiveness of our algorithms. The first traffic pattern is static unbalanced traffic pattern. End hosts with satellite 2,6,10 are randomly picked to send to end hosts with satellites 4,8,12. Both the sender host and the receiver host can have either one network interface or two network interfaces. When they have two network interfaces, they are connected with two adjacent satellites. The total flow numbers are varied over each experiment. Each flow is corresponded to a source destination host pair.

The second one is dynamic unbalanced traffic pattern. During time interval [0s,60s], 16 flows are send between satellite 6,7 and another 16 flows are send between satellite 10,11. From 10s to the end of simulation, sender A which connects with satellite 2,6 send to a receiver B which connects with satellite 12. The interface capacity of this sender and receiver is 20Mbps. From 40s to the end of simulation, 20 flows are set up between satellite 7,8.

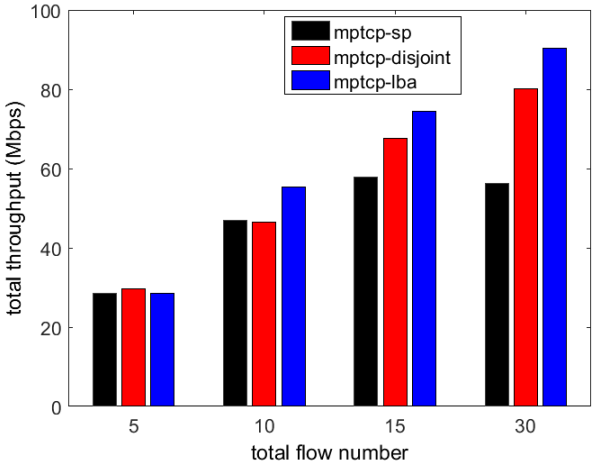

FIGURE 5. Total system throughput variation with different total flow number.

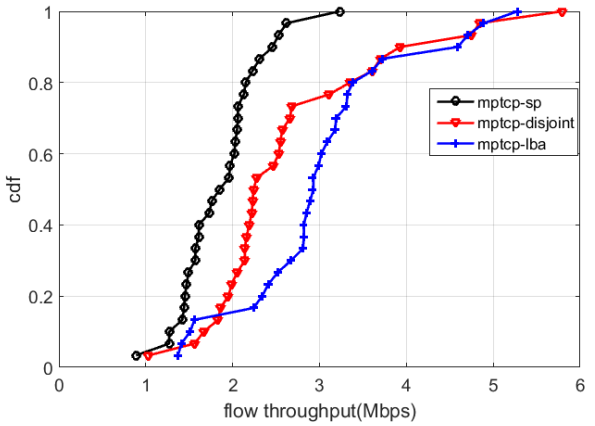

FIGURE 6. Cdf distribution of flow throughput (total flow number=30).

C. EVALUATION RESULTS

1) STATIC UNBALANCED TRAFFIC PATTERN

We compare the total system throughput among three algorithms: mptcp-shortest path(mptcp-sp), mptcp with disjoint multiple paths(mptcp-disjoint) and our load and bottleneck aware subflow route selection scheme(mptcp-lba). The simulation result of total system bandwidth for each scheme is shown is Fig.5. It can be seen that with the increase of total flow number, all these three scheme's system throughput increases correspondingly. This is because when total flow number is low, the host network card is the bottleneck of end to end path. With the increase of total flow number, the utilization of network link also increases and this results in the increase of total system bandwidth. However, when total flow number is high, the increase of mptcp-sp scheme is low as the flow number has already reach the limit of bottleneck link bandwidth.

It can also be seen that when total flow number is 15 or 30, mptcp-lba has a relatively higher total system bandwidth than mptcp-disjoint. And mptcp-disjoint has a relatively higher total system throughput than mptcp-sp. This is because in our unbalanced traffic pattern, the traffic is concentrated over the left down 3∗3 satellites in the 4∗4 topology, at the same time the other 7 satellites in the outside are in a low utilization state, especially for link (14,15) and (15,16). Since mptcp-sp only chooses the shortest paths for each mptcp subflow without considering the bottleneck, link (14,15) and (15,16) will not be selected even when the left down 3∗3 satellites are highly loaded. And for mptcp-disjoint scheme, only a small number of subflows will make use link (14,15) and (15,16). And for our mptcp-lba, since it is load and bottleneck aware, and we allow it to choose paths other than the shortest path with no more than 3 hops, so more subflow will select (14,15) and (15,16) with mptcp-lba than mptcp-disjoint or mptcp-sp. And thus a higher system bandwidth for mptcp-lba.

The cdf distribution of each flow's bandwidth when the total flow number is 30 is shown in Fig.6. It can be seen that mptcp-lba has more percentage of higher bandwidth flows than mptcp-disjoint, and mptcp-disjoint has more percentage of higher bandwidth flows than mptcp-sp. And this also proves that mptcp-lba has the highest total system bandwidth in these three.

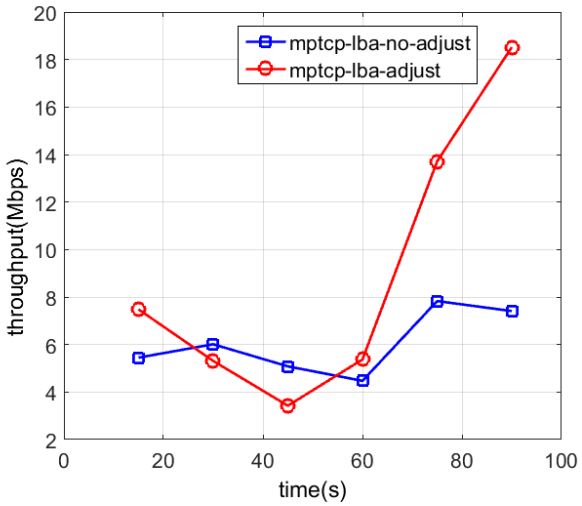

FIGURE 7. Sender A's total throughput.

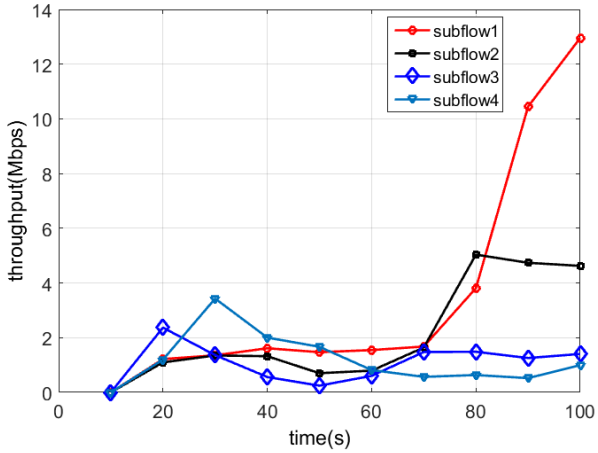

FIGURE 8. Throughput of sender A's four subflows over time.

2) DYNAMIC UNBALANCED TRAFFIC PATTERN

Fig.7. gives the throughput of sender A of mptcp-lba which adjusts dynamically to network load changes and mptcp-lba which does no adjust dynamically. And Fig.8 gives the throughput of each subflow of sender A in dynamical unbalanced traffic pattern when the shared bottleneck subflow's routes are adjusted dynamically.

We can see that at about 60s, the throughput of mptcp-lba-adjust scheme increases rapidly. And the throughput of mptcp-lba-no-adjust still stays at a relatively low level. It is because in dynamic unbalanced traffic pattern, since paths whose hop count are within the limit of shortest path hop count plus 3. The traffic between satellite 6 and satellite 7 will be spread over link (2,3), (6,7), (10,11), (14,15). And these four links become the bottleneck. When these four links become bottleneck, sender A establishes four subflows, among these subflows, two subflows share link (11,12). Later, when network traffic is from satellite 7 to satellite 8, links (3,4), (7,8) and (11,12) become the bottleneck. At the same time, link (15,16) has much vacant capacity. Mptcp-lba-adjust scheme adjusts subflow1 to be send over link (15,16) and this results in the increase of aggregated throughput compared with mptcp-lba-no-adjust.

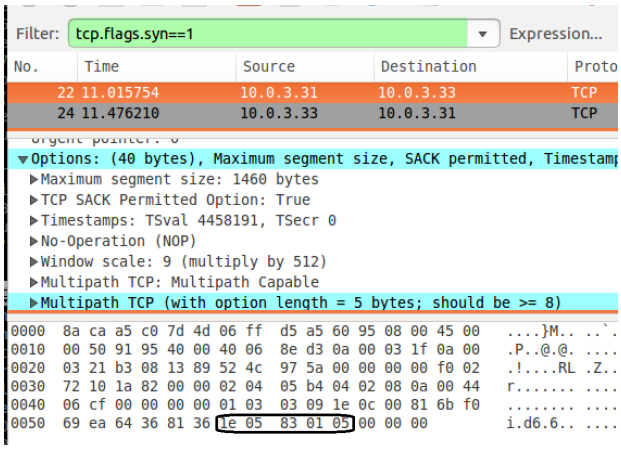

FIGURE 9. The newly added MPTCP option conveys control information for path management to end host.

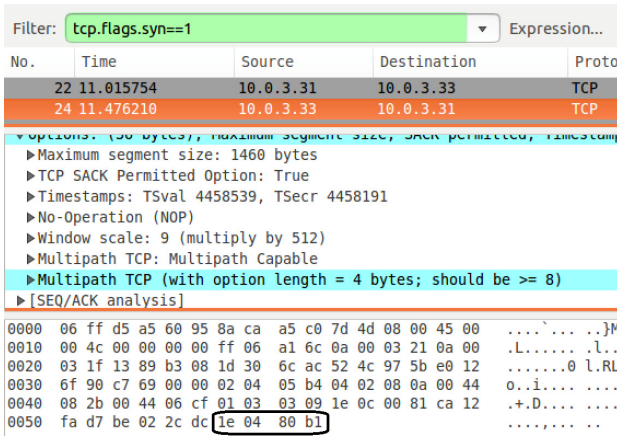

FIGURE 10. The newly added MPTCP option conveys the path identification and maximum subflow number to controller.

Fig.9 and Fig.10 shows MPTCP connection set up packet. In Fig.9, a new MPTCP option is added to the SYN packet sent to the receiver, the last two eight bits indicate the path identification of this subflow is one and the maximum subflow number allowed by the host for this connection is five.

In Fig.10, a new MPTCP option is also added to the SYN/ACK packet. In this packet, the controller packed the control information about path management in the last eight bits of this option. The first bit is one, and indicates that path number needs to be increased by one. And the second to fourth bit are three, meaning the number of subflows to be increased is three. Then another four bits are one, meaning that source address, destination address and destination port of path identification one will be used to established new subflows.

VI. CONCLUSION

In the paper, we propose scMPTCP architecture and its related algorithms. By comparing our algorithms with existing schemes, the efficacy of scMPTCP architecture and its related algorithms is validated.

Currently, the user application we consider is elastic traffic type like file transfer over TCP. And the performance metric

is only the aggregated throughput. In the future, more applications like real time video transmission or DASH (Dynamic Adaptive Streaming over HTTP) which have more stringent network requirements will be considered. And more performance metrics include transfer delay and jitter will be used to evaluate. Correspondingly the subflow route selection algorithms need to be further improved to better support these applications.

REFERENCES

[1] *SpaceX*. Accessed: Apr. 2018. [Online]. Available: https://en.wikipedia.org/wiki/SpaceX
[2] *OneWeb*. Accessed: Apr. 2018. [Online]. Available: https://en.wikipedia.org/wiki/OneWeb
[3] *O3b_Networks*. Accessed: Apr. 2018. [Online]. Available: https://en.wikipedia.org/wiki/O3b_Networks
[4] *SES_S.A.* Accessed: Apr. 2018. [Online]. Available: https://en.wikipedia.org/wiki/SES_S.A.
[5] *Configure MPTCP*. Accessed: Apr. 2018. [Online]. Available: http://multipath-tcp.org/pmwiki.php/Users/ConfigureMPTCP
[6] A. Ford *et al.*, *Architectural Guidelines for Multipath TCP Development*, document RFC6182, IETF, Fremont, CA, USA, 2011.
[7] C. Paasch and S. Barre. *Multipath TCP in the Linux Kernel*. Accessed: Apr. 2018. [Online]. Available: https://www.multipath-tcp.org
[8] M. Sandri *et al.*, "On the benefits of using multipath tcp and openflow in shared bottlenecks," in *Proc. IEEE 29th Int. Conf. Adv. Inf. Netw. Appl. (AINA)*, Mar. 2015, pp. 9–16.
[9] P. Du, X. Li, Y. Lu, and M. Gerla, "Multipath TCP over LEO satellite networks," *Proc. IEEE Wireless Commun. Mobile Comput. Conf.*, Aug. 2015, pp. 1–6.
[10] S. Zannettou, M. Sirivianos, and F. Papadopoulos, "Exploiting path diversity in datacenters using MPTCP-aware SDN," in *Proc. IEEE Comput. Commun.*, Jun. 2016, pp. 539–546.
[11] P. Du, F. Pang, T. Braun, M. Gerla, C. Hoffmann, and J. H. Kim, "Traffic optimization in software defined naval network for satellite communications," in *Proc. MILCOM*, Oct. 2017, 459–464.
[12] J. Duan, Z. Wang, and C. Wu, "Responsive multipath TCP in SDN-based datacenters," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2015, pp. 5296–5301.
[13] A. Hussein, I. H. Elhajj, A. Chehab, and A. Kayssi, "SDN for MPTCP: An enhanced architecture for large data transfers in datacenters," in *Proc. IEEE Int. Conf. Commun.*, May 2017, pp. 1–7.
[14] J. Pang, G. Xu, and X. Fu, "SDN-based data center networking with collaboration of multipath TCP and segment routing," *IEEE Access*, vol. 5, pp. 9764–9773, 2017.
[15] S. Jain *et al.*, "B4: Experience with a globally-deployed software defined WAN," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 43, no. 4, pp. 3–14, 2013.
[16] J. Schulz-Zander *et al.*, "Programmatic orchestration of WiFi networks," in *Proc. Usenix Conf. Usenix Tech. Conf.*, 2014, pp. 347–358.
[17] J. Lee *et al.*, "meSDN: Mobile extension of SDN," in *Proc. 15th Int. Workshop Mobile Cloud Comput. Services*, 2014, pp. 7–14.
[18] M. Moser and F. Jaramillo, "Extending software defined networking to end user devices," in *Proc. Int. Conf. Comput. Commun. Netw.*, 2015, pp. 1–8.
[19] J. Bao, Z. Gong, C. Wu, Z. Feng, W. Yu, and B. Zhao, "OpenSAN: A software-defined satellite network architecture," in *Proc. ACM Conf. SIGCOMM ACM*, 2014, pp. 347–348.
[20] T. Li, H. Zhou, H. Luo, I. You, and Q. Xu, "SAT-FLOW: Multi-strategy flow table management for software defined satellite networks," *IEEE Access*, vol. 5, pp. 14952–14965, 2017.
[21] T. Li, H. Zhou, H. Luo, and S. Yu, "SERvICE: A software defined framework for integrated space-terrestrial satellite communication," *IEEE Trans. Mobile Comput.*, vol. 17, no. 3, pp. 703–716, Mar. 2018.
[22] R. Ferrús *et al.*, "SDN/NFV-enabled satellite communications networks: Opportunities, scenarios and challenges," *Phys. Commun.*, vol. 18, pp. 95–112, Mar. 2016.
[23] L. Bertaux *et al.*, "Software defined networking and virtualization for broadband satellite networks," *IEEE Commun. Mag.*, vol. 53, no. 3, pp. 54–60, Mar. 2015.
[24] C. Raiciu *et al.*, "How hard can it be? Designing and implementing a deployable multipath TCP ," in *Proc. 9th USENIX Conf. Netw. Syst. Design Implement. USENIX Assoc.*, 2012, p. 29.
[25] W. Yang, H. Li, F. Li, Q. Wu, and J. Wu, "RPS: Range-based path selection method for concurrent multipath transfer," in *Proc. 6th Int. Wireless Commun. Mobile Comput. Conf.*, 2010, pp. 944–948.
[26] Y. Cao *et al.*, "CMT-CC: Cross-layer cognitive CMT for efficient multimedia distribution over multi-homed wireless networks," *Wireless Pers. Commun.*, vol. 82, no. 3, pp. 1643–1663, 2015.
[27] S. Deng, "Intelligent network selection and energy reduction for mobile devices," Ph.D. dissertation, Massachusetts Inst. Technol., Cambridge, MA, USA, 2015.
[28] P. Bonami, M. Kilinç, and J. Linderoth, "Algorithms and software for convex mixed integer nonlinear programs," in *Mixed Integer Nonlinear Programming*. New York, NY, USA: Springer, 2012, pp. 1–39.
[29] K. Giotis, C. Argyropoulos, G. Androulidakis, D. Kalogeras, and V. Maglaris, "Combining OpenFlow and sFlow for an effective and scalable anomaly detection and mitigation mechanism on SDN environments," *Comput. Netw.*, vol. 62, no. 5, pp. 122–136, 2014.
[30] T. Taleb, "Explicit load balancing technique for NGEO satellite IP networks with on-board processing capabilities," *IEEE/ACM Trans. Netw.* vol. 17, no. 1, pp. 281–293, Jan. 2009.

ZHUO JIANG received the B.S degree from the Beijing University of Posts and Telecommunications, Beijing, China, in 2012. He is currently pursuing Ph.D. degree with the Department of Computer Science and Technology, Tsinghua University, Beijing. His main research interests include multipath transmission, mobile and wireless network, and cross-layer optimization.

QIAN WU received the M.S. and Ph.D. degrees in computer science from Tsinghua University in 2002 and 2006, respectively. She is currently an Associate Professor with the Institute for Network Sciences and Cyberspace, Tsinghua University. Her research interests include next-generation Internet architecture and protocols, space and earth integrated network, mobile and wireless network, multipath transfer, and mobile multicast.

HEWU LI received the M.S. and Ph.D. degrees in computer science from Tsinghua University in 2001 and 2004, respectively. He is currently an Associate Professor with the Institute for Network Sciences and Cyberspace, Tsinghua University, and the Director of Wireless and Mobile Network Research Laboratory, Network Research Center. His research interests include mobile wireless network architecture, space and earth integrated network, broadband wireless access technology, and mobility architecture in next-generation network.

JIANPING WU (F'11) is currently a member of the Chinese Academy of Engineering and a Professor with the Department of Computer Science, Tsinghua University, where he is serving as the Chairman of the Department of Computer Science and the Dean of the Institute for Network Sciences and Cyberspace. He is also serving as the Director of the Network Center and the Technical Board of China Education and Research Network (CERNET), and the National Engineering Laboratory for Next Generation Internet, a member of the Advisory Committee of National Information Infrastructure for Secretariat of State Council of China, and the Vice President of Internet Society of China. He was also the Chairman of Asia Pacific Advanced Network (2007-2011). He has devoted himself to the research of computer network including technology research, network engineering and cultivating talents for many years. He has led his team to do an in-depth study of network design, network engineering, network core equipment development, and network architecture.