A Traffic Meter Based on a Multicolor Marker for Bandwidth Guarantee and Priority Differentiation in SDN Virtual Networks

Steven S. W. Lee and Kwan-Yee Chan

***Abstract*—Network virtualization is a technology that enables multiple tenants to share a common physical network. A virtual network (VN) needs to have dedicated bandwidth and a priority differentiation capability to provide quality of service (QoS) for its serving flows. In this paper, we propose a traffic meter based on a multicolor marker (MCM) to fulfill both inter-VN and intra-VN QoS requirements. In our system, a VN's guaranteed bandwidth is dedicated to the VN, and the bandwidth is not influenced by the traffic generated from other VNs in the same physical network. In addition, the high-priority traffic of a VN can always achieve its needed bandwidth unless the total amount of high-priority traffic exceeds the VN's guaranteed bandwidth. This feature enables a network to provide independent QoS differentiation for each VN. Our MCM meters can be applied in two operation modes. In the remaining bandwidth nonsharing (RBNS) mode, the guaranteed bandwidth is treated as an upper bound for each VN, even if the network has unused capacity remaining. However, in the remaining bandwidth sharing (RBS) mode, the unused network capacity is shared among VNs so that a VN can use more than its guaranteed bandwidth. To validate our design, we implemented our MCM meter in P4 programmable switches. The experimental results on a testbed reveal that our design can accurately achieve bandwidth isolation and QoS differentiation for network virtualization in P4-based software-defined networking (SDN).**

***Index Terms*—Virtual network, network slicing, bandwidth guarantee, QoS differentiation, P4 switch, color marker.**

I. INTRODUCTION

THERE has been continued demand for the ability to share a physical network with multiple tenants. A virtual local area network (VLAN) is a common technology for providing multiple logical networks in a LAN. A virtual private network (VPN) is another technology for reliably interconnecting multiple branches of an enterprise. With the continuing increase in network capacity, it is increasingly cost-effective to share a physical network with multiple virtual networks (VNs). The strong demands for deploying VNs come from not only campus and enterprise users but also data centers and the telecommunication industry. In 5G communications, network virtualization is called network slicing. Network slicing has been identified in 3GPP specifications as a key technology for next-generation mobile core networks [1]. The standard organization Open Networking Foundation (ONF) has proposed a software-defined networking (SDN)-based architecture for 5G network virtualization [2].

Several issues need to be resolved to support the QoS for each VN in a shared physical network. Bandwidth isolation among VNs is the first issue that needs to be considered. At any time, a VN must have its subscribed bandwidth ready regardless of the bandwidth usage of the other VNs in the same network. In addition to bandwidth isolation between VNs, traffic prioritization is another key feature for QoS differentiation inside a VN.

SDN is considered a promising technology for facilitating the realization of network virtualization. By separating the control plane and the data plane, SDN provides the capabilities for flexible network control and management.

OpenFlow is the major commercially available SDN technology; consequently, most SDN VNs designs are based on OpenFlow. Many OpenFlow switches provide metering functions. A set of flows can be associated with a meter. By enforcing the rate of the meter, the total bandwidth consumed by the set of flows can be controlled.

In addition to metering functions, some advanced OpenFlow switches provide QoS queues (e.g., priority queues and weighted fair queues). They can be used to realize packet prioritization. Similar to a meter function, we could associate a set of flows to a QoS queue of a link. Accordingly, traffic belonging to a set of flows has the priority pf assessing the bandwidth of the link.

The most common QoS queue provides eight classes of priorities for a physical link. Although meters can enforce bandwidth usage control and QoS queues can guarantee packet prioritization, combining meters and QoS queues cannot provide traffic prioritization inside a VN because the QoS queue works only for traffic just before they leave a switch. Consequently, a meter function has to be applied in front of QoS queues. As a result, packets belonging to excess bandwidth are dropped regardless of their priority. Therefore, there is only bandwidth isolation but not QoS differentiation in OpenFlow-based VNs.

P4 is a language for new-generation programmable switches. In this paper, we call the programmable switches that support P4 language P4 switches, and we call a network

Manuscript received November 3, 2018; revised March 14, 2019 and June 4, 2019; accepted June 11, 2019. Date of publication June 14, 2019; date of current version September 9, 2019. The associate editor coordinating the review of this paper and approving it for publication was G. Bianchi. *(Corresponding author: Steven S. W. Lee.)*

The authors are with the Department of Communications Engineering, National Chung Cheng University, Chiayi 621, Taiwan (e-mail: ieeswl@ccu.edu.tw; kwanyee86@gmail.com).

Digital Object Identifier 10.1109/TNSM.2019.2923110

that is constituted by P4 switches a P4 network. Compared to an OpenFlow switch, a P4 switch provides more flexible traffic control. An OpenFlow switch is basically a stateless switch. OpenFlow performs fixed match-action rules for incoming packets. Packets with headers matching a same-flow entry in the flow table receive the same action regardless of when the packets arrive. In contrast, a P4 switch can be programmed as a stateful switch. Multiple registers inside a P4 switch enable the switch to maintain the network state. This feature facilitates more flexible traffic control on SDN networks. In addition, a P4 switch provides multiple color markers (CMs) that can be used to indicate bandwidth usage of flows.

In this work, we propose a meter based on a multicolor marker (MCM) that can guarantee the bandwidth for each VN inside a P4 network. In addition, the traffic inside a VN follows a strict priority that drops low-priority packets as the total capacity usage exceeds the guaranteed bandwidth. An MCM meter includes a set of CMs. Each CM in the system assigns a color (green or red) to packets. Green is used to mark packets that fall into the given rate limitation (i.e., the guaranteed bandwidth), and red is used to mark packets that use bandwidth beyond the limitation. By cascading multiple CMs along with the proposed control logic, the desired QoS differentiation and bandwidth guarantee can be achieved.

Although the CM is a common functionality in many programmable switches, except for the token rate, most parameters, such as the number of tokens inside a bucket, are not readable and are not allowed to be modified dynamically. However, when designing an MCM meter, the number of tokens inside a bucket must be able to be dynamically reconfigured. To overcome this problem, in our MCM meter, each CM is associated with a variable called compensation credit. Compensation credit is used to represent the offset of the number of tokens in a CM. When the compensation credit of a CM is positive (negative), the number of tokens inside the CM is less (greater) than its actual number. The sum of the number of tokens inside a CM bucket and its compensation credit is the real amount of tokens that the CM should have.

The proposed MCM meter can support two operation modes depending on how the unused bandwidth is shared in the network. In the remaining bandwidth nonsharing (RBNS) mode, the upper-bound bandwidth usage of a VN is its guaranteed bandwidth. Any packets using excess bandwidth are unconditionally dropped. Conversely, in the remaining bandwidth sharing (RBS) mode, the unused bandwidth of a VN is shared by VNs that have traffic demands larger than their guaranteed bandwidth. Depending on the business model, an operator can flexibly choose one of the modes for their network.

In summary, our major contributions in this work are as follows.

- We propose the architectural design of an MCM meter for multi-VN networks. The proposed MCM meter can guarantee the bandwidth of each VN. In addition, a strict priority is realized to support the QoS differentiation for the traffic classes inside each VN.
- We introduce a method for implementing the MCM meter in a P4 programmable switch [3].
- Our implementation can be operated in either RBNS or RBS mode, which provides flexibility for network operators in the control of their networks.
- We conduct experiments to validate the proposed MCM meter. The experimental results show that the MCM meter can accurately control bandwidth usage and achieve intra-VN priority differentiation.

The remainder of the paper is organized as follows. In Section II, we present surveys of related work. In Section III, we present the system architecture and the control logics of the proposed MCM-based meter system. We demonstrate the implementation of the system in P4 programmable switches in Section IV. In Section V, the experimental results are analyzed. Finally, we provide our concluding remarks and discuss our future research in Section VI.

II. Related Work

A comprehensive survey on the design of SDN network virtualization can be found in [4]. To embed VNs in a physical substrate network, topology assignment flexibility, bandwidth isolation, and address separation are the major technical issues that need to be resolved first. Bandwidth slicing is the simplest approach for implementing VNs in SDN networks. It directly partitions the link bandwidth to the VNs. Unused bandwidth is not shared for a network using bandwidth slicing.

FlowVisor [5] is the first work to implement bandwidth slicing in OpenFlow networks. In FlowVisor, the topology of each VN is essentially the same as or is the subset of the substrate network. All VNs share the same address space. A VN does not have a completely separate address space.

A more advanced version of VN design provides each user with a custom network topology [6], [7], [8], [9], [10]. Because the topologies of the VNs and the substrate network are allowed to be different, the control functions are more complex than FlowVisor. VeRTIGO [6] is an extended version of FlowVisor. Similar to FlowVisor, it is unable to provide each VN with an independent address space. By introducing an additional VLAN and/or MPLS tags to separate the address space, subsequent systems are able to support full address separation [7], [8], [9], [10].

Bandwidth assigned to a VN is usually a fixed constant value. Based on the measured bandwidth usage, dynamic bandwidth slicing could improve network throughput by frequently adjusting VN bandwidth assignment. EnterpriseVisor [11] is a controller-based system for dynamic bandwidth slicing in OpenFlow networks. The controller periodically examines the bandwidth requirement of each VN to determine how to partition the network capacity. Each VN is assigned a priority, and high-priority VNs are the first to obtain additional bandwidth when there is remaining capacity available. Although EnterpriseVisor introduces priorities to VNs, the priority value is used to determine the preference for the remaining bandwidth usage. Because EnterpriseVisor relies on the controller to collect the statistical data for the bandwidth usage of each VN, it introduces additional delays for bandwidth allocation. In addition, bandwidth usage prediction performed by the controller might be inaccurate, thus resulting in wasted bandwidth.

Although existing VN design approaches allow bandwidth slicing in OpenFlow networks, intra-VN QoS differentiation is not considered. A user that owns a VN still must provide differentiated services for different classes of traffic. Implementing QoS differentiation inside a VN is still not addressed in the literature. Although some OpenFlow switches provide QoS queues, intra-VN priority differentiation is not applicable due to the inflexibility of using those queues.

The hierarchical token bucket (HTB) [12] has been implemented for traffic control in Linux systems. Many software switches/routers, such as the Open vSwitch (OVS) [13] and the Linux IP/MPLS router, support the HTB. The HTB applies a class-based hierarchical system for traffic control. Traffic classes are configured in a tree structure. Each class can be configured with a guaranteed bandwidth for packet transmission. A class can both use its guaranteed bandwidth and borrow tokens from its parent class to acquire additional bandwidth.

There are some HTB-based traffic control schemes. In [14], the HTB is used to provide guaranteed bandwidth for VNs in a network using OVS switches. Each VN is assigned a given priority. The bandwidth allocated to each VN is directly proportional to its priority, and high-priority VNs have more guaranteed bandwidth. In [15], based on the timing requirement, traffic flows are prioritized into real-time, non-real-time, and best-effort classes. According to the remaining token amount inside the HTB buckets, this algorithm dynamically adjusts the bandwidth assignment of classes.

TrafficLight [16] uses HTB to perform hierarchical bandwidth management. It provides a management system to facilitate users to configure the network. By monitoring network traffic, the final target of the TrafficLight project is to realize self-learning and dynamic adaptability on network resources management.

Another bandwidth control system for SDN is proposed in [17]. This system uses the OpenFlow controller to monitor bandwidth usage. Based on the collected bandwidth usage, the controller dynamically configures HTB to reallocate VN bandwidth to improve network resources usage.

A management system for dynamic bandwidth sharing among virtual machines (VM) is presented for cloud data center networks [18]. The system works on top of HTB. Each VM in the data center is provisioned a deserved bandwidth, which is the upper-limit bandwidth that the VM can use. The management system monitors the bandwidth usage of each VM. As the given bandwidth is not fully utilized, the deserved bandwidth is reduced; otherwise, the bandwidth is increased. By adjusting the deserved bandwidth for each VM, the network resources are shared among all of the VMs. Although this paper addresses VM bandwidth management, the idea can be applied to manage VNs.

In the above approaches, bandwidth sharing among VNs is based on controlling the HTB. Traffic prioritization intra-VN is not taken into account. Although the HTB can assign each class a priority value, the priority is used to determine the order of packet scheduling. The guaranteed bandwidth of a class is reserved even if its priority value is low.

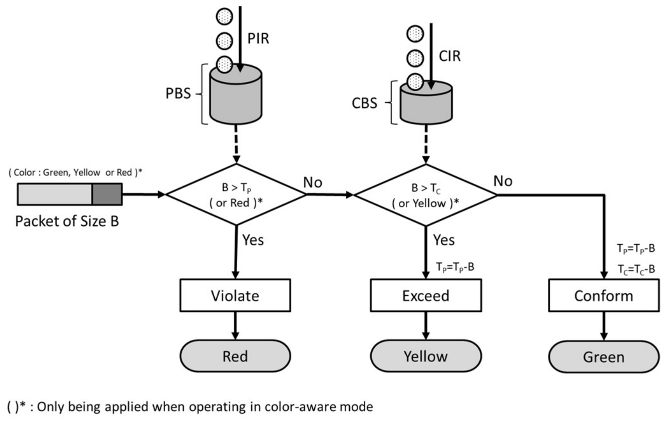


Fig. 1. TrTCM.

Although the HTB could be applied to map a VN into a bandwidth-guaranteed parent class and leave different service flows in prioritized leaf classes, implementation of a strict priority inside a VN is still lacking. In addition, due to the high complexity, the HTB appears only in the software system; there is no HTB embedded inside commercial SDN switches yet. Therefore, in this work, we propose using only the CM to realize traffic control in SDN networks.

The CM has been used to indicate the conforming levels of traffic by assigning a color to each packet. IETF includes a single-rate three-color marker (srTCM) [19] and a two-rate three-color marker (trTCM) [20] in RFCs.

As shown in Fig. 1, a trTCM consists of two buckets: the peak burst size (PBS) bucket and the committed burst size (CBS) bucket. The token filling rates of the PBS and CBS buckets are the peak information rate (PIR) and the committed information rate (CIR), respectively. An incoming packet is colored red if the number of tokens inside the PBS is smaller than the packet size B. If the first condition is not satisfied, the marker further examines if the number of tokens inside the CBS is larger than B. If this number is larger, the packet is colored green; otherwise, the packet is colored yellow.

The above CM is called operating in color-blind mode. Based only on the input packet, a color-blind CM makes its decision to determine the color of the incoming packet. When a marker is operating in color-aware mode, it takes not only the incoming packet but also an input color into account. The operations of a color-aware trTCM can also be found in Fig. 1.

A CM-based QoS control for VNs within an SDN network is proposed in [21]. Each VN is provided guaranteed bandwidth. In addition to the original two token buckets in CM, this design employs two additional token buckets, called global token buckets (GTBs), which are used to collect idle tokens from each VN. With the help of GTBs, the unused bandwidth can be shared among all VNs. Although this approach can realize bandwidth isolation and unused bandwidth sharing, intra-VN packet prioritization is still not supported.

In this work, our target is to provide a guaranteed bandwidth for each VN while maintaining a strict priority for intra-VN traffic policing. Unlike the work proposed in [21], we use only standard CMs in designing our MCM meter. Our solution can be implemented in a new-generation programmable switch (e.g., P4 switch) that supports CM functions.

A typical control plane architecture can be found in [5], [6], [7], [8], [10]. In these networks, there is a main controller and multiple VN controllers. Each user uses their VN controller to control their own VN. The main controller is the actual controller that directly communicates with the physical switches. All commands issued from VN controllers are forwarded to the main controller. We follow the same control architecture to control our VNs. In this paper, we focus on presenting the proposed MCM meter.

III. Architecture and Basic Principles of the MCM Meter

In this section, we present the architectural design of the proposed MCM meter and its control algorithm. In a multi-VN network, each VN has a virtual topology that includes virtual nodes and virtual links. In the system, the bandwidth usage of each virtual link (port) of each VN is controlled by the proposed MCM meter. This meter determines which packets can be transmitted on the link or dropped to prevent excessive bandwidth usage.

For the simplicity of presentation, we focus on the realization of the bandwidth control on a single virtual link in this section. By applying multiple copies of the proposed meter, a switch can control bandwidth usage and perform priority differentiation for all VNs.

To implement the proposed MCM meter, we need only CMs that can provide two colors to indicate if a flow is confined within its committed rate. Any CM meeting the requirement can be used in our system. Because srTCM and trTCM are the most common CMs used in switches, we focus on demonstrating our design using srTCM and trTCM. Because we need only a CM that can mark packets as green and red, both the srTCM and the trTCM in either color-blind mode or color-aware mode can satisfy our requirement. However, this focus does not preclude the use of any other kinds of CMs in our system.

A. Design of RNBS MCM Meter Based on Color-Blind CMs

The architecture of the RNBS MCM meter that uses color-blind CMs is shown in Fig. 2(a). There are n VNs using physical port p. For each VN, there is a corresponding MCM meter. Each MCM meter is responsible only for metering the packets belonging to its responsible VN. Figure 2(b) depicts the functionality of the proposed MCM meter. The number of CMs in the MCM meter for VN i is the same as the number of priority classes in the VN. We assume the considered VN consists of k priority classes. The priority values are listed in descending order such that priority 1 is the highest priority and priority k is the lowest priority. All CMs in an MCM meter of a VN are set to have the same token adding rate. More specifically, for a VN with a guaranteed bandwidth of r Mbps, the token adding rate of each CM inside its responsible MCM meter is set to r Mbps.

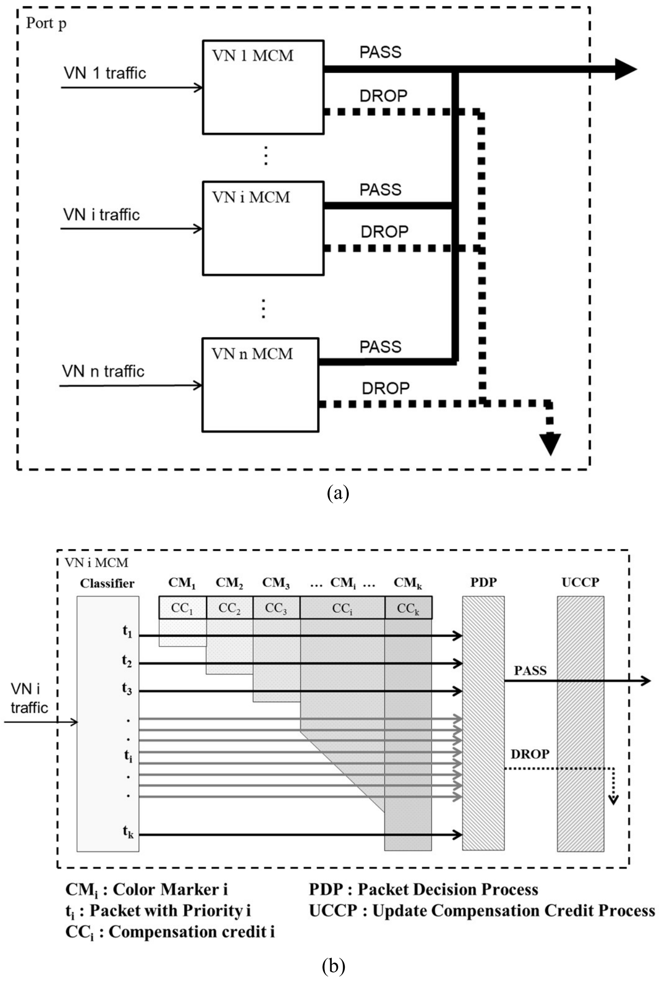


Fig. 2. Architecture of the RBNS MCM meter using color-blind CMs. (a) Functional blocks of applying the MCM meters to control bandwidth usage for physical output port p. (b) Detailed structure of the MCM meter for a VN.

The traffic of VN i is classified based on the priority. As shown in Fig. 2(b), the packet with the highest priority, priority 1, goes through all of the CMs. The packet with priority 2 goes through all of the CMs except CM 1. The packet with the lowest priority, priority k, goes through only CM k. For a packet with priority i, CM i is its primary CM, and those CMs with IDs larger than i are the packet's subsidiary CMs. A packet must go through its primary CM and all of its subsidiary CMs. As a result, CM j is shared by packets with a priority larger than or equal to j. By examining the packet color of CM j, we can determine the bandwidth usage of all traffic with a priority greater than or equal to j.

Before describing the operations of the proposed MCM meter, let us denote by $p(t)$ and $l(t)$ the priority and the size of packet t, respectively. The CM with ID $= p(t)$ is the primary CM of a packet t. $S(t)$ denotes the set of subsidiary CMs of this packet. Packet t is independently colored by each of the CMs in the set $S(t) \cup \{p(t)\}$. We also denote by $c(t, i)$ the color marked by CM i for the packet t. The compensation credit of CM i is denoted as $cc(i)$. Whether a packet t can pass the MCM meter depends on the color $c(t, p(t))$ assigned by its primary CM and the compensation credit $cc(p(t))$ of its primary CM.

The compensation credit is used to adjust the rate of admission of a class of traffic. In most switch systems (including the one used in our study), the tokens inside a CM are unknown to the outside world. In addition, a user is not able to dynamically modify the number of tokens inside the CM's bucket. Consequently, we use the compensation credit to indicate the offset of the number of tokens in a CM. In our design, each CM has its corresponding compensation credit. If the compensation credit is a negative value x, there are a surplus of x tokens inside the token bucket of the CM. Otherwise, if the compensation credit is a positive value, there is a shortage of x tokens inside the token bucket of the CM. The sum of the number of tokens inside a CM bucket and its compensation credit is the real amount of tokens that the CM should have.

Because the primary CM and the subsidiary CMs work independently, there are two cases in which the value of the compensation credit must be adjusted. (1) A packet that is colored green by its subsidiary CM i indicates that $l(t)$ tokens have been taken away from the token bucket of CM i as the packet passes through. However, if the final decision is to drop the packet, those $l(t)$ tokens are mistakenly taken away from CM i. (2) A packet that is colored red by its subsidiary CM i indicates that no token has been taken away from the token bucket of CM i as the packet passes through. However, if the final decision is to transmit the packet on the outgoing link, CM i has $l(t)$ surplus tokens. One approach is to restore $l(t)$ tokens back to CM i in case (1) and to remove $l(t)$ tokens from CM i in case (2). However, since we are not able to restore/remove tokens from the token bucket of CM i, we adjust the value of compensation credit to represent the offset value.

The decision to pass or drop a packet and the updating of the compensation credit are performed in the packet decision process (PDP) and the update compensation credit process (UCCP), respectively. The detailed algorithms to determine the passing or dropping of the packet and the update of the compensation credit are presented in Fig. 3. The compensation credit for each CM is initially set to zero. The whole algorithm can be divided into the following four cases.

```
Algorithm for MCM metering
(1) if $c(t, p(t))$=Red && $cc(p(t)) \geq l(p(t))$ then
    action: pass
    $cc(p(t)) = cc(p(t)) - l(p(t))$
    for each $j \in S(t)$
        if $c(t,j)$=Green then $cc(j)$ remains unchanged
        if $c(t,j)$=Red then $cc(j)=cc(j) - l(p(t))$
(2) if $c(t, p(t))$=Red && $cc(p(t)) < l(p(t))$ then
    action: drop
    $cc(p(t))$ remains unchanged
    for each $j \in S(t)$
        if $c(t,j)$=Green then $cc(j)=cc(j)+ l(p(t))$
        if $c(t,j)$=Red then $cc(j)$ remains unchanged
(3) if $c(t, p(t))$=Green && $cc(p(t)) + l(p(t)) > 0$ then
    action: pass
    $cc(p(t))$ remains unchanged
    for each $j \in S(t)$
        if $c(t,j)$=Green then $cc(j)$ remains unchanged
        if $c(t,j)$=Red then $cc(j)=cc(j) - l(p(t))$
(4) if $c(t, p(t))$=Green && $cc(p(t))+ l(p(t)) \leq 0$ then
    action: drop
    $cc(p(t)) = cc(p(t)) + l(p(t))$
    for each $j \in S(t)$
        if $c(t,j)$=Green then $cc(j)=cc(j)+ l(p(t))$
        if $c(t,j)$=Red then $cc(j)$ remains unchanged
```

Fig. 3. The algorithm for MCM meter control.

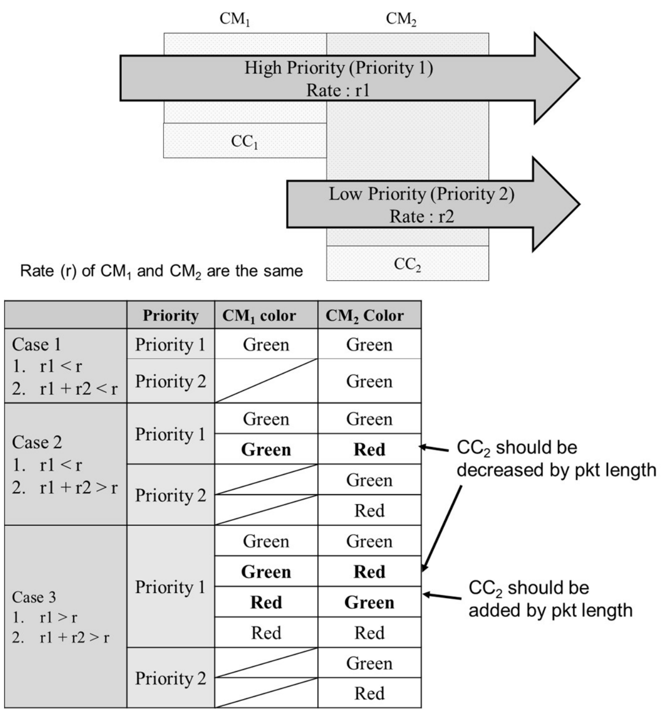
- Case 1: If the color of the primary CM is red but the compensation credit of it is greater than or equal to the packet length, the packet will be transmitted onto the outgoing link. The compensation credit of the primary meter is then reduced by the packet length. For each subsidiary CM, if the color marked by this meter is green, the compensation credit of this subsidiary meter remains unchanged; otherwise, if the color is red, the compensation credit is reduced by the packet length.
- Case 2: If the color of the primary CM is red and the compensation credit is less than the packet length, the packet will be dropped. The compensation credit of the primary meter remains unchanged. For each subsidiary meter, if the marked color is green, the compensation credit of this subsidiary meter is increased by the packet length; otherwise, if the color is red, the compensation credit remains unchanged.
- Case 3: If the color marked by the primary CM is green and the compensation credit of the primary CM is greater than zero after adding the packet length, the packet will be transmitted. The compensation credit of the primary meter remains unchanged. For each subsidiary meter of the packet, if the color of this meter is green, the compensation credit remains unchanged; otherwise, if the color is red, the compensation credit is reduced by the packet length.
- Case 4: If the color of the primary CM is green and the compensation credit of the primary CM is less than or equal to zero after adding the packet length, the packet will be dropped. The compensation credit of the primary meter is increased by the packet length. For each subsidiary meter of the packet, if the color of this meter is green, the compensation credit of this meter is increased by the packet length; otherwise, if the color is red, the compensation credit remains unchanged.

Before concluding this subsection, we use an example to describe how to use the MCM meter to control the bandwidth usage of a VN. In this example, we assume the bandwidth for the VN is r Mbps. There are two types of traffic (priority 1 and priority 2) in this VN. The traffic arriving rates for priority 1 (high priority) and priority 2 (low priority) are $r1$ and $r2$, respectively.

There are two CMs serving this VN. The token adding rates for both CMs are r Mbps. The priority 1 traffic passes through CM 1 and CM 2, and the priority 2 traffic passes through only CM 2. The primary CM for priority 1 packets is CM 1. Because CM 1 only handles priority 1 traffic, the tokens inside CM 1 are dedicated to priority 1 traffic. It results in packets belonging to priority 1 always receiving green marks if the rate $r1$ is not larger than r. For priority 2 traffic, the primary meter is CM 2. Since both priority 1 and priority 2 traffic pass through CM 2, the tokens are shared between both types of traffic. Thus, even if $r2$ is smaller than the token rate r Mbps, if $r1+r2$ is greater than r, some of the packets passing through CM2 are marked red.

The table inside Fig. 4 presents all cases of packet coloring. In case 1, because the sum of $r1$ and $r2$ is less than r, every packet is colored green by CM 1 and CM 2. In the second case, priority 1 packets are assigned green by CM 1. Because the total input rate $r1+r2$ is larger than r, some of priority 1 packets are colored red by CM 2. In case 3, the rate of $r1$ is larger than r. Both colors are possible for a packet to be marked. Because CM 1 and CM 2 are working independently, there is no synchronization for adding tokens to the token buckets of those two meters. Consequently, it is possible that the priority 1 packet is colored red by CM 1 but is colored green by CM 2.

When the priority 1 packet is colored green by CM 1 but is colored red by CM 2, the packet is allowed to be transmitted. However, the tokens are taken away from only CM 1 but not CM 2. In this case, the compensation credit of CM 2 is reduced by the length of the packet. On the other hand, a priority 1 packet receives red in CM 1 but green in CM 2. The packet is dropped based on the decision of its primary CM (i.e., CM 1). However, the tokens have been taken away as the packet is going through CM2. Thus, the compensation credit of CM 2 has to be increased by the packet length.


CM$_1$
CM$_2$
High Priority (Priority 1)
Rate : r1
CC$_1$
Low Priority (Priority 2)
Rate : r2
CC$_2$
Rate (r) of CM$_1$ and CM$_2$ are the same
CC$_2$ should be decreased by pkt length
CC$_2$ should be added by pkt length

	Priority	CM$_1$ color	CM$_2$ Color
Case 1 1. r1 < r 2. r1 + r2 < r	Priority 1	Green	Green
	Priority 2		Green
Case 2 1. r1 < r 2. r1 + r2 > r	Priority 1	Green	Green
		Green	**Red**
	Priority 2		Green
			Red
Case 3 1. r1 > r 2. r1 + r2 > r	Priority 1	Green	Green
		Green	**Red**
		Red	**Green**
		Red	Red
	Priority 2		Green
			Red

Fig. 4. Example of a VN with two types of traffic.

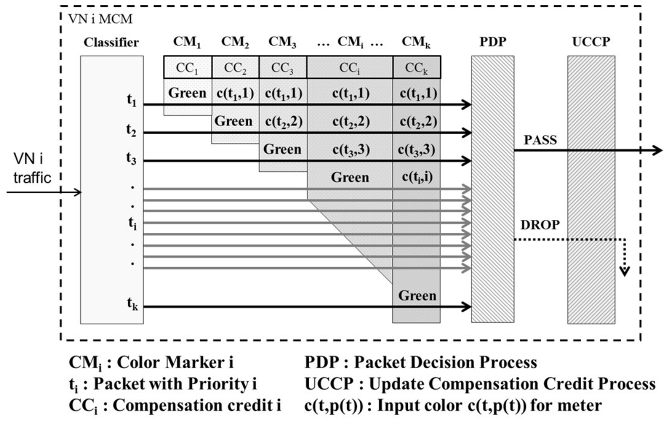

VN i MCM
Classifier
CM$_1$ CM$_2$ CM$_3$ ... CM$_i$ ... CM$_k$
PDP
UCCP
CC$_1$ CC$_2$ CC$_3$ CC$_i$ CC$_k$
Green c(t$_1$,1) c(t$_1$,1) c(t$_1$,1) c(t$_1$,1)
Green c(t$_2$,2) c(t$_2$,2) c(t$_2$,2)
Green c(t$_3$,3) c(t$_3$,3)
Green c(t$_i$,i)
Green
t$_1$
t$_2$
t$_3$
t$_i$
t$_k$
VN i traffic
PASS
DROP
CM$_i$: Color Marker i
t$_i$: Packet with Priority i
CC$_i$: Compensation credit i
PDP : Packet Decision Process
UCCP : Update Compensation Credit Process
c(t,p(t)) : Input color c(t,p(t)) for meter

Fig. 5. Architecture of the RBNS MCM meter using color-aware CMs.

B. Design of a RNBS MCM Meter Based on Color-Aware CMs

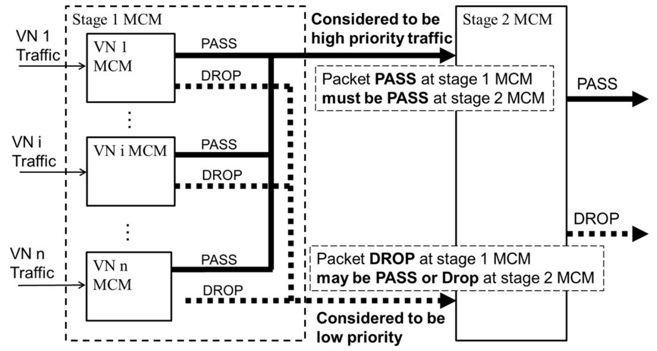
Figure 5 depicts another MCM design based on color-aware CMs for networks operating in RNBS mode. For color-aware CMs, in addition to the input packet, an input color has to be given. A packet of priority i has to pass from CM i to CM k (k is the number of priority), where CM i is the primary CM and those CMs with IDs within $[i+1, k]$ are the subsidiary CMs. The primary CM is the first one that an incoming packet has to pass through. The input color for a primary CM is fixed as green. More specifically, as an incoming packet is classified as priority i, the input color of CM i is set to green. For each subsidiary CM, the input color of the packet is the output color of its primary CM.

Let us consider a priority 2 packet as an example. In this case, CM 2 is the primary meter and CMs $3, 4, \ldots, k$ are the subsidiary CMs. The input color for the primary meter, CM 2, is fixed to be green. We denote by c the output color of CM 2. After being colored by CM2, this packet and the color c are sent to all of its subsidiary meters.

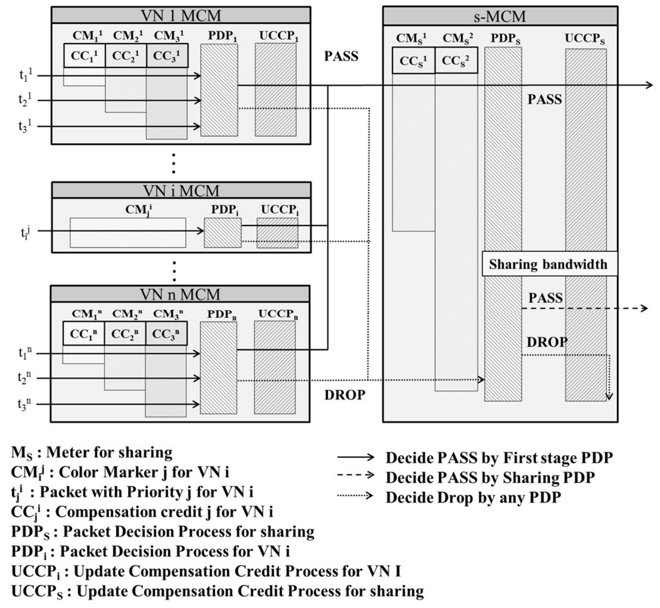
Although the architecture of the MCM meter with color-aware CMs differs from the architecture with color-blind CMs, the decision logic and the update of the compensation credit are interestingly the same for both types of MCM meters. The detailed control algorithm for the MCM meter with color-aware CMs is the same as that shown in Fig. 3.

C. Design of an RBS MCM Meter

In RBNS mode, the maximum bandwidth that a VN can use is its configured bandwidth. Any packet using excess bandwidth is unconditionally dropped. If the configured bandwidth of a VN is not fully used, the unused bandwidth is wasted. In RBS mode, the remaining bandwidth of a VN is shared


Stage 1 MCM
VN 1 Traffic
VN 1 MCM
PASS
DROP
VN i Traffic
VN i MCM
PASS
DROP
VN n Traffic
VN n MCM
PASS
DROP
Considered to be high priority traffic
Packet PASS at stage 1 MCM must be PASS at stage 2 MCM
Stage 2 MCM
PASS
DROP
Packet DROP at stage 1 MCM may be PASS or Drop at stage 2 MCM
Considered to be low priority

(a)


VN 1 MCM
CM_1^1 CM_2^1 CM_3^1 PDP_1 $UCCP_1$
CC_1^1 CC_2^1 CC_3^1
t_1^1
t_2^1
t_3^1
PASS
VN i MCM
CM_j^i PDP_i $UCCP_i$
t_i^j
VN n MCM
CM_1^n CM_2^n CM_3^n PDP_n $UCCP_n$
CC_1^n CC_2^n CC_3^n
t_1^n
t_2^n
t_3^n
DROP
s-MCM
CM_S^1 CM_S^2 PDP_S $UCCP_S$
CC_S^1 CC_S^2
PASS
Sharing bandwidth
PASS
DROP
M_S : Meter for sharing
CM_j^i : Color Marker j for VN i
t_j^i : Packet with Priority j for VN i
CC_j^i : Compensation credit j for VN i
PDP_S : Packet Decision Process for sharing
PDP_i : Packet Decision Process for VN i
$UCCP_i$: Update Compensation Credit Process for VN I
$UCCP_S$: Update Compensation Credit Process for sharing
Decide PASS by First stage PDP
Decide PASS by Sharing PDP
Decide Drop by any PDP

(b)

Fig. 6. RBS MCM meter. (a) Functional blocks of the RBS MCM meter. (b) Detailed structure of the RBS MCM meter.

by VNs that have traffic demands larger than their guaranteed bandwidth.

We use the RBNS MCM meter as a basis to design the RBS MCM meter. The RBS MCM meter consists of two stages. The stage 1 MCM includes n RBNS MCM meters, one for each VN. The token rate for a stage 1 MCM is the bandwidth of its VN. The stage 2 MCM includes only one RBNS MCM meter. The token rate for the stage 2 MCM is the capacity of the outgoing link. We call the meter in the second stage the s-MCM to distinguish it from the other meters. Figure 6 depicts the architecture of a physical port.

Figure 6(a) shows the functional blocks of the RBS MCM meter. Each stage 1 MCM handles the traffic of their own VN. Similar to the MCM meters in the RBNS mode, each packet is determined to be either "pass" or "drop" after passing through a stage 1 MCM meter. However, unlike the operations in RBNS, the packets with decision to "drop" are not eliminated immediately. Each packet and their pass/drop mark are sent to the s-MCM.

Traffic that comes to s-MCM is classified as high priority or low priority. High-priority packets are those with "pass"

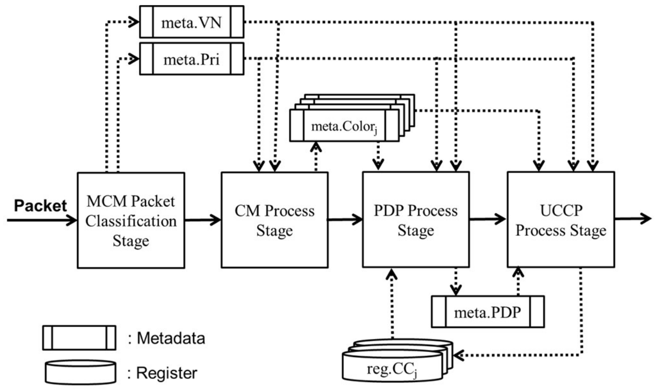


Fig. 7. MCM meter stage.

marks assigned by the stage 1 MCMs, and low-priority packets are those with "drop" marks. Thus, the packets with "pass" marks are unconditionally transmitted to the outgoing link. The packets with "drop" marks are allowed to use link capacity only if there is any unused bandwidth remaining.

Figure 6(b) demonstrates the detailed structure of the RBS MCM meter. There are two CMs inside the s-MCM meter. The control logic for all of the MCMs follow the same algorithm shown in Fig. 3.

IV. Implementation on P4 Switches

P4 is a programming language for programmable switches [3]. A programmer can define his or her own rules for switches to parse incoming packets. In addition to a rich set of actions, a P4 switch also provides basic arithmetic and logic capabilities for computations. There are many registers inside a P4 switch. In addition, a P4 switch can attach metadata to a packet to facilitate forwarding decisions and computations. P4 also provides CMs to help traffic control. Those features enable us to realize the proposed MCM meter inside a P4 network.

We use four stages to implement an MCM meter inside a P4 switch. The language used is P4-14 [22]. As shown in Fig. 7, those stages are named MCM packet classification stage, CM process stage, PDP process stage, and UCCP process stage. Each incoming packet has to pass through all of these stages. To simplify the discussion, we present only the bandwidth control in the program. The logics for routing are not included in the following description. subsequently, we present the implementation of RBNS MCM by using both color-blind CMs and color-aware CMs. RBS MCM can be realized using similar techniques.

A. MCM Packet Classification Stage

The target of this stage is to identify the VN and the priority of an incoming packet. P4 provides a flexible mechanism for traffic parsing. A user can define the rule for traffic classification. For example, we could use VLAN and the protocol port to identify VN and priority. Figure 8 shows the P4 codes of the packet classification. In this example, *classify_table* requires that the VLAN and UDP destination port to be exactly

```
1.  table classify_table {
2.          reads{
3.                  vlan.vid : exact;
4.                  udp.dstPort : exact;
5.          }
6.          actions{
7.                  classify_action;
8.          }
9.  }
10. action classify_action(VN_id, Pri_id) {
11.         modify_field(meta.VN, VN_id);
12.         modify_field(meta.Pri, Pri_id);
13. }
```

Fig. 8. P4 codes of the packet classification process.

```
1.  table CM_table {
2.          reads{
3.                  meta.VN : exact;
4.                  meta.Pri : exact;
5.          }
6.          actions{
7.                  VN1_Pri1_CM_action;
8.                  VN1_Pri2_CM_action;
9.                  VN1_Pri3_CM_action;
10.         }
11. }
12. action VN1_Pri1_CM_action() {
13.         execute_meter(VN1_CM, 1, meta.Color1);
14.         execute_meter(VN1_CM, 2, meta.Color2);
15.         execute_meter(VN1_CM, 3, meta.Color3);
16. }
17. action VN1_Pri2_CM_action() {
18.         execute_meter(VN1_CM, 2, meta.Color2);
19.         execute_meter(VN1_CM, 3, meta.Color3);
20. }
21. action VN1_Pri3_CM_action() {
22.         execute_meter(VN1_CM, 3, meta.Color3);
23. }
```

Fig. 9. P4 codes of the CM process with color-blind mode.

matched in the flow table. By configuring the flow table that maps the VLAN and the UDP port to the *VN_id* and the *Pri_id*, the matching results obtain the VN ID and packet priority of the incoming packet. We further use *modify_field* in line 11 and line 12 to save both IDs into metadata *meta.VN* and metadata *meta.Pri*.

B. CM Process Stage

The CMs provided in P4 are trTCMs. Because we need only two colors for our MCM meter, for each VN, we allocate an MCM meter that sets the PIR to the guaranteed bandwidth and the CIR to 0 for all CMs inside the MCM meter. As a result, the trTCM will assign red to packets that violate the rate limitation of the PIR, and the packets are colored yellow otherwise. A packet colored yellow is considered to be green in order to meet the algorithm shown in Fig. 3.

For a VN supporting *k* priorities, there are *k* CMs. Those CMs are dedicated to their VN. We present the P4 codes of the CM process for the color-blind mode for VN 1 in Fig. 9. Similar codes can be derived when color-aware trTCMs are applied.

CM_table in the program states that the VN ID (i.e., *meta.VN*) and packet priority (i.e., *meta.Pri*) are matched in the table. The matching result determines which action to be taken for further processing. The action indicates which trTCM meters to be called.

For example, for a priority 2 packet, the mapping of its *meta.VN* and *meta.Pri* indicates *VN1_Pri2_CM_action* at line 17 of Fig. 9 to be executed. Inside the action, it calls *execute_meter. execute_meter* is the P4 system function that activates trTCM to mark the packet with color. The parameter *VN1_CM* is an array of these three meters. The indexes 1, 2, and 3 in *VN1_CM* correspond to CM 1, CM 2, and CM 3. The third parameter $meta.Color_j$ is not only an input but also an output variable for the function *execute_meter* and is used to provide the initial color for the trTCM. After performing this function, the output color is also saved in $meta.Color_j$.

In P4, each trTCM is in fact color-aware. However, because green is the lowest priority, an input color for trTCM of green does not influence the decision for packet coloring. In other words, this trTCM with fixed input green color is in fact a color-blind trTCM. We use this technique to implement our color-blind MCM. The input color, $meta.Color_j$, for each trTCM in lines 13, 14, and 15 of Fig. 9 is set to green.

C. PDP Process and UCCP Process Stage

The operations of the PDP process stage and UCCP process stage are used to realize the algorithm shown in Fig. 3. The former is responsible for determining whether the packet can be passed to the outgoing link or should be dropped; the latter is responsible for updating the compensation credits. Because the number of tokens in a bucket cannot be modified in a P4 switch, we use independent register $reg.CC_j$ for each CM *j* to save the compensation credit. In P4, when a switch receives a packet, the packet length is automatically saved in a metadata called *standard_metadata.packet_length*. Accordingly, the packet length *l*(*p*(*t*)), coloring result *c*(*t*,*j*), and compensation credit *cc*(*j*) in Fig. 3 are mapped into *standard_metadata.packet_length*, $meta.Color_j$ and register $reg.CC_j$ in our P4 program, respectively.

The PDP process employs s*tandard_metadata.packet_length*, $reg.CC_j$, and $meta.Color_j$ to handle a packet with priority *j*. The PDP process saves the computation results in a metadata called *meta.Decision*, which indicates the final action for passing or dropping the packet.

The UCCP process updates compensation credits. The inputs of UCCP include s*tandard_metadata.packet_length, meta.Decision*, and the set of colors determined by those CMs that the packet has gone through in CM process. For example, the coloring results for the packet with priority 1 are $meta.Color_1$, $meta.Color_2$, and $meta.Color_3$. Based on the coloring results, the UCCP process updates $reg.CC_j$ according to the algorithm shown in Fig. 3.

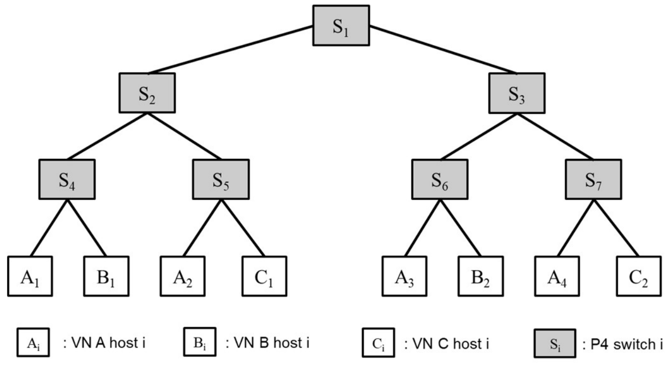


Fig. 10. Experimental network topology.

V. Experimental Results

We have conducted experiments to validate the proposed MCM meter for both RBNS and RBS modes. Figure 10 shows the experimental network. The tree topology includes seven P4 switches (S1, S2, . . ., S7) and eight hosts. The P4 switches are PCs running the bmv2 emulator [3]. The network is shared by the three VNs, A, B, and C. For simplicity, the logical topology of those three VNs is the same as the physical topology. We denote X_i as the *i*-th host of VN *X*. Each P4 switch used in S1, S2, and S7 includes an i7-8700 CPU with clock rate of 3.7 GHz. The CPUs used in S4 and S6 are i7-7700 CPUs with clock rates of 4.2 GHz. S3 uses an i7-6700 CPU with a clock rate of 4 GHz, and S5 uses an i7-4790 CPU with a clock rate of 3.6 GHz. The link capacity of the network is set to 90 Mbps, which is the maximum rate that the bmv2 emulator can support.

The guaranteed bandwidths for VNs A, B, and C are 40 Mbps, 30 Mbps, and 20 Mbps, respectively. There are four communication pairs, A1 to A3, A2 to A4, B1 to B2, and C1 to C2. In VN A, there are three classes of traffic. The priority order of those classes from high to low is p1, p2, and p3.

In the experiments, each communication pair sets up multiple UDP connections. The traffic is generated using Iperf [23]. The receiving data are logged by Wireshark [24]. VN A is our major observing VN. We examine if VN A can use bandwidth independently from the interference of VN B and VN C. Because the traffic of VN B and VN C is considered background traffic, VNs B and C send only the highest priority traffic in the experiments. However, VN A generates three classes of traffic. We observe the receiving rates for those three classes of traffic under various experimental settings.

Because the traffic flows come from the left part of the network and go toward the right part of the network, the tree structure of the topology makes the links of S1 and S2 become bottlenecks. For VN A, at any time in the experiments, both communication pairs (A1, A3) and (A2, A4) send the same traffic. More specifically, host A1 and host A2 are sending traffic toward their receivers A3 and A4 with the same priority and the same data rate. Thus, if we indicate that the input rate of VN A is *x* Mbps, then *x*/2 Mbps come from (A1, A3), and the other half comes from (A2, A4).

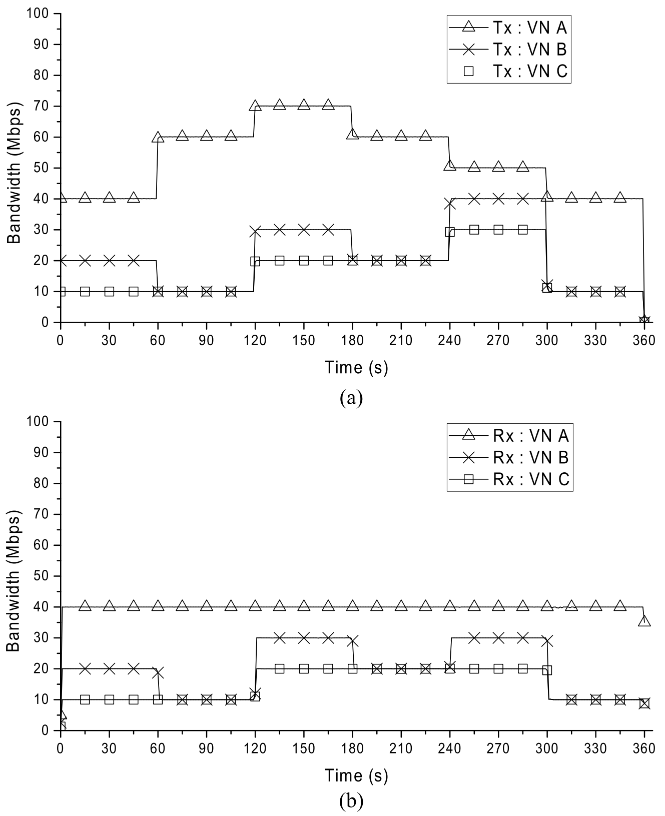


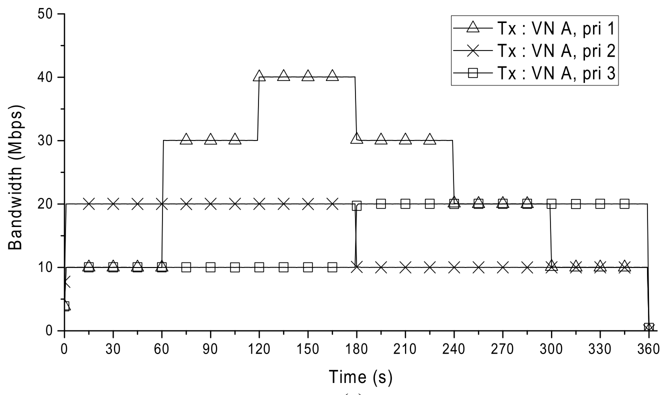
Fig. 11. Bandwidth isolation among VNs in the RBNS mode. (a) Total input rates of VN A, VN B, and VN C. (b) Total receiving rates of VN A, VN B, and VN C.

The experiments are divided into two parts. In the first part, we examine the experimental results when the MCM meters operate in RNBS mode. In the second part, the operation mode of the MCM meters is changed to become the RBS mode.

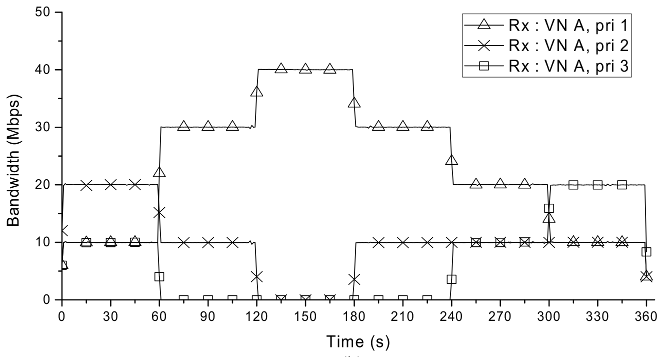
A. Experimental Results for MCM Meters Operating in RBNS Mode

In RBNS mode, each VN can use only their own bandwidth. For each port, we install an MCM meter for each VN to control their packets. Because bandwidth isolation and priority differentiation are the key issues, we perform two sets of experiments to examine them in the following two subsections. We further perform an experiment to compare priority differentiation for network with and without applying the proposed MCM meters. The results of the comparisons are displayed in the third subsection.

1) Bandwidth Isolation Among VNs: We first examine whether inter-VN bandwidth isolation is achieved by the proposed MCM meters. Figure 11(a) shows the total input rates of the three VNs. Because the upper-limit bandwidths of VN A, VN B, and VN C are 40 Mbps, 30 Mbps, and 20 Mbps, respectively, the total injection rates exceed the guaranteed bandwidth of VN A during the time period (60, 300) and over the guaranteed bandwidths of VN B and VN C during the time period (240, 300). From Fig. 11(b), we can observe that the MCM meters successfully force all VNs to obey

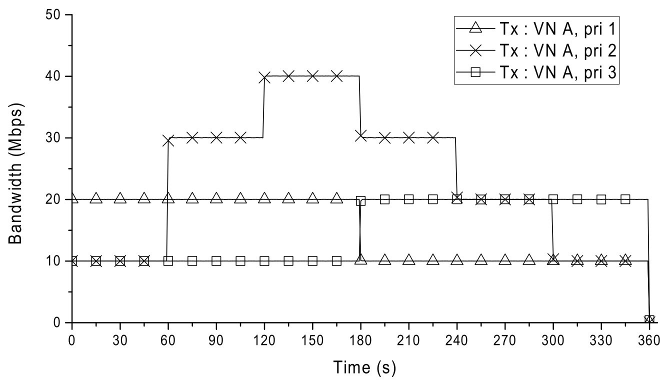


(a)

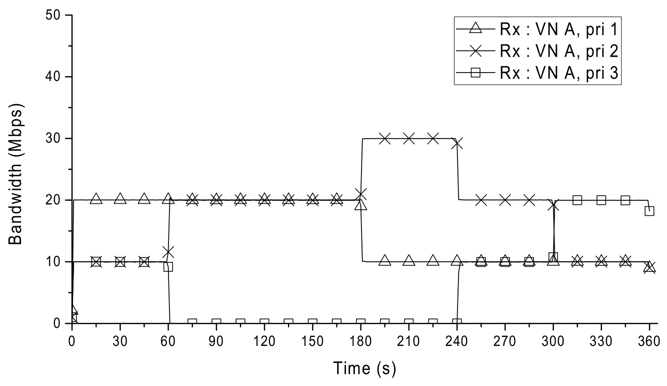


(b)

Fig. 12. Intra-VN priority differentiation for case 1 traffic patterns in the RBNS mode. (a) Input rate of VN A. (b) Receiving rate of VN A.



(a)



(b)

Fig. 13. Intra-VN priority differentiation for case 2 traffic patterns in RBNS mode. (a) Input rate of VN A. (b) Receiving rate of VN A.

their guaranteed bandwidths by dropping the packets using excessive bandwidth.

2) Priority Differentiation Inside a VN: The second set of experiments is designed to evaluate intra-VN priority differentiation. We examine the receiving rates of those three traffic classes inside VN A. The bandwidth limitation of VN A is 40 Mbps. To examine the interactions between traffic of different priorities, we consider the following three cases.

Case 1: The input rates change every 60 s. The rates of p2 and p3 traffic are fixed from 0-180 s. Their rates exchange at 180 s and then remain fixed from 180-360 s.

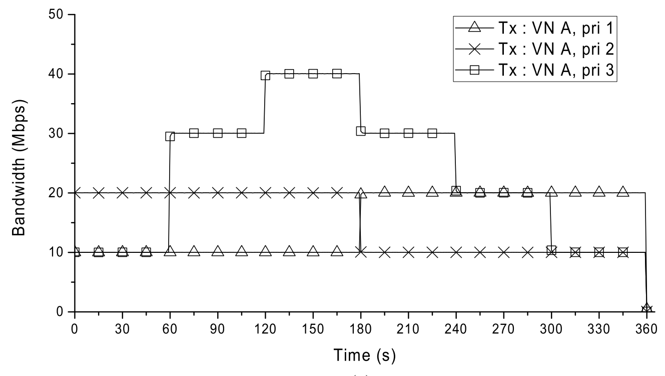
Case 2: The input rates change every 60 s. The rates of p1 and class p3 traffic are fixed from 0-180 s. Their rates exchange at 180 s and then remain fixed from 180-360 s.

Case 3: The input rates change every 60 s. The rates of p1 and p2 traffic are fixed from 0-180 s. Their rates exchange at 180 s and then remain fixed from 180-360 s.

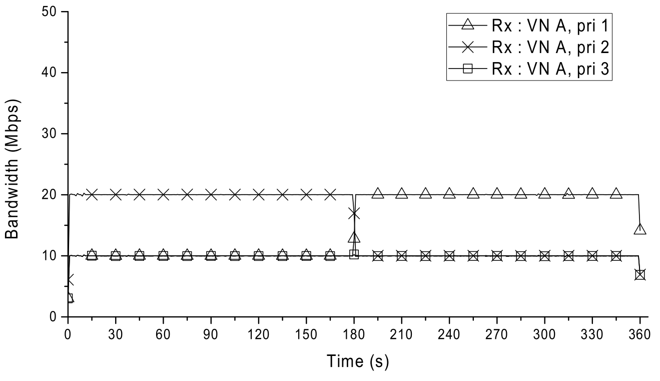
In the experiment shown in Fig. 12, we fix the input rates of the p2 and p3 traffic to 20 Mbps and 10 Mbps, respectively, during period (0, 180) and exchange their rates in period (180, 360). We gradually increase the input rates of the p1 traffic from 10 Mbps to its peak at 40 Mbps and then reduce the rates back to 10 Mbps. During the period (0, 60), the total input rate is 40 Mbps, which does not exceed the bandwidth limitation of VN A. We observe that all of the traffic can be received. In period (60, 120), the total input traffic becomes 60 Mbps. The highest priority p1 obtains all of its required 30 Mbps to support its traffic. The remaining capacity 10 Mbps is taken by p2 traffic. The traffic belonging to the lowest priority p3 is blocked because there is no capacity remaining. As the p1 traffic continues to increase to 40 Mbps at 120 s, the network allows only this highest priority traffic to enter the network. The receiving rates of both p2 and p3 are reduced to zero. The input patterns during period (180, 240) are similar to those during period (60, 120) (except for the exchange of the input rates for p2 and p3). Although p2 and p3 exchange their input rates, Fig. 12(b) shows that the receiving rates are the same in periods (60, 120) and (180, 240). The second priority p2 can still receive the remaining 10 Mbps and all of the p3 traffic is blocked. The results in periods (240, 300) and (300, 360) also confirm that the proposed MCM meters can accurately realize priority differentiation.

We further use the input patterns that follow the case 2 scenario and show the results in Fig. 13. The input patterns of Fig. 13 are the same as those of Fig. 12 (except for the exchange of the traffic classes). When the rate of p2 traffic increases at 60 s, only the p3 traffic has a decreased receiving rate. The total receiving rate for p1 is maintained at 20 Mbps. At 180 s, the p2 traffic can increase by 10 Mbps due to the reduction of 10 Mbps from p1. The results of this experiment reveal that the receiving rate of the p1 traffic remains unchanged regardless of how the p2 traffic and p3 traffic change their input patterns.

In case 3, the p3 traffic changes its input rates during the experiment. Because the priority of class 3 is lowest, its rate



(a)



(b)

Fig. 14. Intra-VN priority differentiation for case 3 traffic patterns in RBNS mode. (a) Input rate of VN A. (b) Receiving rate of VN A.

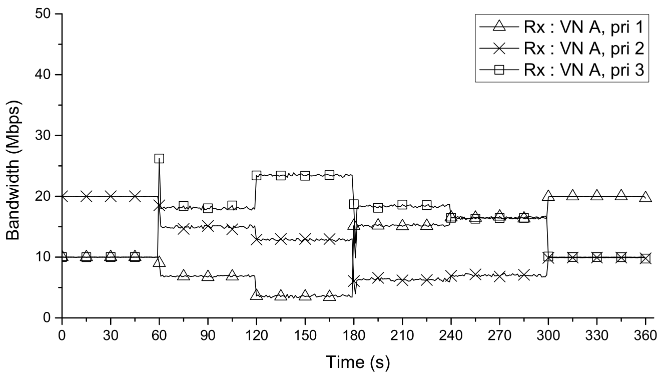


Fig. 15. Receiving rate of VN A without intra-VN priority differentiation when case 3 traffic patterns are applied.

change does not affect the receipt of the p1 traffic and the p2 traffic. In this set of experiments, the total input rate of the p1 and p2 traffic is 30 Mbps. As a result, as shown in Fig. 14(b), the receiving rate of p3 traffic becomes a constant 10 Mbps at any time during the experiment.

3) Comparisons of Priority Differentiation for Networks With and Without Applying MCM Meters: To make comparisons between a network with and without intra-VN priority differentiation, we further establish an experimental network using HP5900 OpenFlow switches. The network includes the same three VNs as those in the P4 network. The guaranteed bandwidths for VNs A, B, and C are the same as those used in the P4 network. Because the network constituted by OpenFlow switches can realize only bandwidth isolation, this network is not able to provide priority differentiation inside a VN. This experiment adopts the case- 3 traffic pattern shown in Fig. 14(a). Figure 15 displays the receiving rates of traffic with priority 1, 2, and 3 inside VN A. During 0-60 s, the total input of the three types of traffic is not larger than VN A's bandwidth limitation (i.e., 40 Mbps). Each type of traffic can obtain its required bandwidth. However, as the total input rates become greater than 40 Mbps during the period (60, 300), different types of traffic have different packet loss rates. We observe that the ratios of the receiving rates for those three types of traffic are basically proportional to their input rates. The high-priority traffic is no longer guaranteed to obtain the required bandwidth. For example, the input rate of the lowest priority p3 traffic reaches its 40 Mbps peak in period (120, 180), it makes the output rates of p1 and p2 become less than their input rates. The results show that high-priority traffic could be dropped for a network without intra-VN priority differentiation.

B. Experimental Results for MCM Meters Operating in the RBS Mode

In the final set of experiments, we apply the proposed MCM meter to a network operating in the RBS mode, which allows the remaining capacity to be shared among VNs. We first examine whether the remaining capacity can be seized by VNs that have bandwidth demands larger than their guaranteed bandwidth. Figure 16(a) displays the traffic of the three VNs. The guaranteed bandwidths for VN A, B, and C are 40 Mbps, 30 Mbps, and 20 Mbps, respectively. In this experiment, VN A always enters 90 Mbps of traffic into the network, while VN B and VN C gradually reduce their input rates. The receiving rates of those VNs are shown in Fig. 16 (b).

During the period (0, 60), no VNs generate traffic less than their guaranteed bandwidth. Consequently, the receiving rate of each VN is bounded by their guaranteed bandwidth because there is no remaining capacity available. During the period (60, 120), the input rate of VN B is 15 Mbps less than its guaranteed 30-Mbps bandwidth. Thus, VN B contributes an extra 15 Mbps for bandwidth sharing. VN A and VN C compete for this 15 Mbps, and the extra bandwidth means that both VNs can deliver traffic exceeding their bandwidth limitations. In period (120, 180), VN B stops generating traffic. VN A and VN C share the 90-Mbps link capacity. After 180 s, VN C gradually reduces its bandwidth demands. As a result, VN A captures all the remaining capacity in the network. Finally, during the period (300, 360), only VN A is generating traffic, which enables VN A to obtain all of the 90-Mbps link capacity.

Figure 16(c) decomposes the 90-Mbps input traffic demand of VN A into the p1, p2, and p3 traffic classes, where the exact demands for those three compositions are 20 Mbps, 30 Mbps and 40 Mbps, respectively. We further examine the intra-VN priority differentiation inside VN A and show the results in Fig. 16(d). The results of the RNBS are also included for easier comparisons. Because the demand of the p1 traffic falls within the guaranteed bandwidth, the required 20 Mbps is

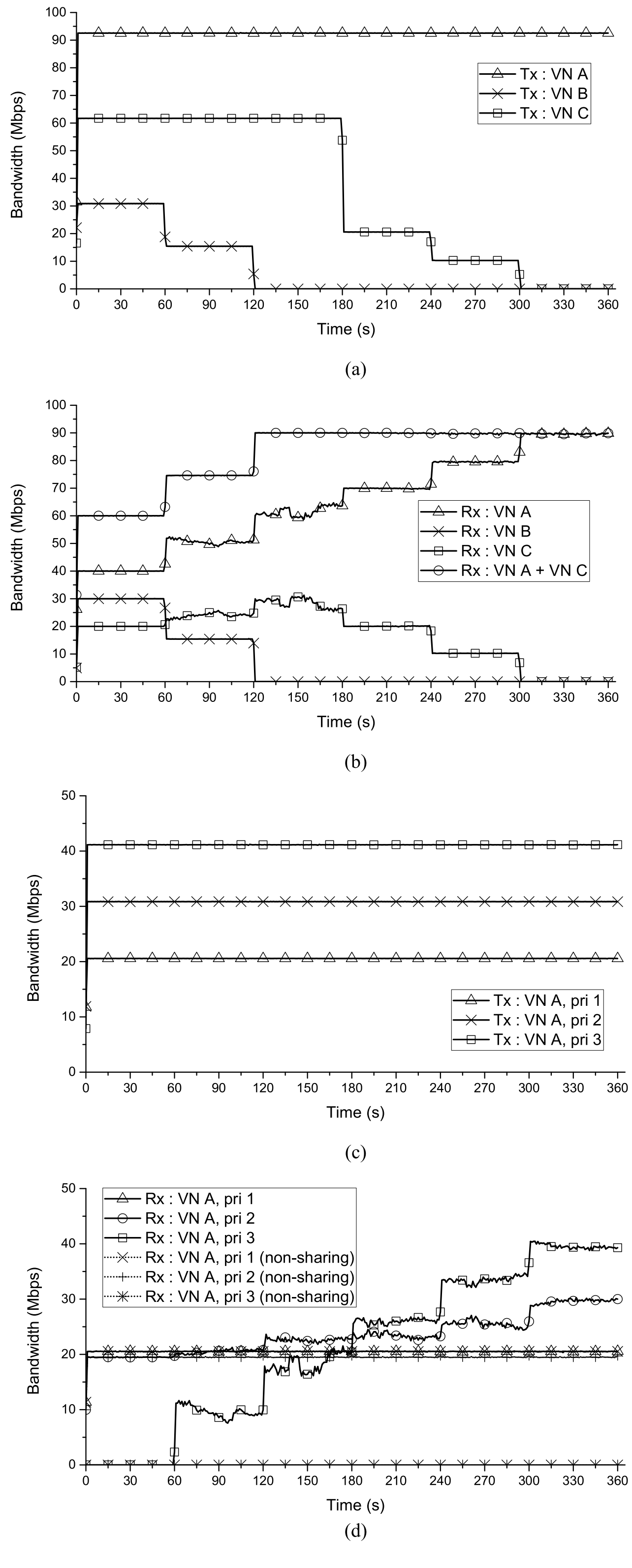


Fig. 16. Bandwidth isolation and remaining capacity sharing among VNs in the RBS mode. (a) Input rates of VN A, VN B, and VN C. (b) Receiving rates of VN A, VN B, and VN C. (c) Input rates of traffic classes p1, p2, and p3 of VN A. (d) Intra-VN priority differentiation of VN A.

always provided by the network. In addition, we can expect that the p2 traffic can obtain at least 20 Mbps after extracting 20 Mbps of the p1 traffic from the guaranteed 40-Mbps bandwidth. In RBNS mode, the bandwidths that the p1, p2, and p3 traffic can obtain are a constant 20 Mbps, 20 Mbps, and 0 Mbps, respectively. However, the p2 and p3 traffic can receive extra bandwidth when the system is operating in RBS mode. Consequently, when the network has positive remaining bandwidth after 60 s, the received data rates for both p2 and p3 in RBS mode are larger than those in RNBS mode.

VI. Conclusion

Network virtualization has received considerable attention in academic and industrial research due to its capability to enable network resource sharing among multiple tenants. Although some approaches based on direct bandwidth partitioning in SDN have been published in the literature, implementation of QoS differentiation inside a VN is still lacking. In this paper, we present an MCM-based meter for priority differentiation and bandwidth-guaranteed multi-VN networks. According to the design, a VN is free from the interference of the traffic generation and bandwidth usage of the other VNs in the same network. Our design also guarantees that the bandwidth usage follows the strict priority rule inside a VN for classes with different priorities. Our MCM meter can support both sharing and nonsharing modes for unused network bandwidth. In RBNS mode, each VN can use only its guaranteed bandwidth. In RBS mode, each VN has its guaranteed bandwidth, and the unused bandwidth can be further shared by all VNs that have demands greater than their guaranteed bandwidth.

We have established a testbed using P4 programmable switches to realize the proposed MCM meters. The experimental results demonstrate that the proposed MCM meter can successfully guarantee the bandwidth of each VN and the traffic inside VNs follows their priority order to use the bandwidth.

Our current implementation is performed on PCs with bmv2 emulators. Realizing our design in a real programmable switch is one of our planned future works. Since our implementation is based on standard P4, we believe our design can be realized in P4 switches supporting the standard P4 language.

For the current version of our RBS meter, the unused bandwidth is shared by VNs that have demands greater than their guaranteed bandwidth. The split of the unused bandwidth is proportional to their excessive demands but not the VN and class priority. Another research direction for the future is to design an MCM meter that can be configured to include not only excessive bandwidth demands but also user-defined parameters such as the VN ID and traffic priority in controlling unused network bandwidth sharing.

Acknowledgment

The authors would like to thank the anonymous reviewers for their helpful and constructive comments, which greatly contributed to improving the quality of this paper.

REFERENCES

[1] *3GPP TS 22.261 Version 15.5.0 Release 15*. Accessed: Jul. 2018. [Online]. Available: https://www.etsi.org/deliver/etsi_ts/122200_122299/122261/15.05.00_60/ts_122261v150500p.pdf

[2] "Applying SDN architecture to 5G slicing," Org. Internationale de la Francophonie, Paris, France, OIF Rep. TR-526, Apr. 2016.

[3] *P4 Bmv2 Switch*. Accessed: May 7, 2019. [Online]. Available: https://github.com/p4lang/behavioral-model

[4] A. Blenk, A. Basta, M. Reisslein, and W. Kellerer, "Survey on network virtualization hypervisors for software defined networking," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 1, pp. 655–685, 1st Quart., 2016.

[5] R. Sherwood *et al.*, "FlowVisor: A network virtualization layer," OpenFlow Consortium, Rep. TR-2009-1, Oct. 2009. [Online]. Available: https://www.gta.ufrj.br/ensino/cpe717-2011/openflow-tr-2009-1-flowvisor.pdf

[6] R. D. Corin, M. Gerola, R. Riggio, F. De Pellegrini, and E. Salvadori, "VeRTIGO: Network virtualization and beyond," in *Proc. EWSDN*, Darmstadt, Germany, Oct. 2012, pp. 24–29.

[7] E. Salvadori, R. D. Corin, A. Broglio, and M. Gerola, "Generalizing virtual network topologies in OpenFlow-based networks," in *Proc. IEEE GLOBECOM*, Houston, TX, USA, 2011, pp. 1–6.

[8] S. S. W. Lee, K.-Y. Li, K.-Y. Chan, Y.-C. Chung, and G.-H. Lai, "Design of bandwidth guaranteed OpenFlow virtual networks using robust optimization," in *Proc. IEEE GLOBECOM*, Austin, TX, USA, 2014, pp. 1916–1922.

[9] D. Drutskoy, E. Keller, and J. Rexford, "Scalable network virtualization in software-defined networks," *IEEE Internet Comput.*, vol. 17, no. 2, pp. 20–27, Mar./Apr. 2013.

[10] A. Al-Shabibi *et al.*, "OpenVirteX: Make your virtual SDNs programmable," in *Proc. HotSDN*, Chicago, IL, USA, 2014, pp. 25–30.

[11] J.-L. Chen, Y.-W. Ma, H.-Y. Kuo, and W.-C. Hung, "Enterprise visor: A software-defined enterprise network resource management engine," in *Proc. IEEE/SICE Int. Symp. Syst. Integr. (SII)*, Tokyo, Japan, 2014, pp. 381–384.

[12] M. Brown. *Traffic Control HOWTO, Version 1.0.2*. Accessed: Oct. 28, 2006. [Online]. Available: https://www.tldp.org/HOWTO/pdf/Traffic-Control-HOWTO.pdf

[13] *Open vSwitch, Version 2.11.1*. Accessed: Apr. 19, 2019. [Online]. Available: http://openvswitch.org/

[14] Q. Fu, L. Qing, A. Yingzhu, and F. Yamei, "A priority based virtual network bandwidth guarantee method in software defined network," in *Proc. 6th IEEE Int. Conf. Softw. Eng. Service Sci. (ICSESS)*, Beijing, China, 2015, pp. 153–156.

[15] C.-H. Lee and Y.-T. Kim "QoS-aware hierarchical token bucket (QHTB) queuing disciplines for QoS-guaranteed DiffServ provisioning with optimized bandwidth utilization and priority-based preemption," in *Proc. Int. Conf. Inf. Netw. (ICOIN)*, 2013, pp. 351–358.

[16] T. Bhattacharjee, V. Gopal, L. N. Nganggoua, and C. Raghunanth. *TrafficLight: Network Traffic Monitoring and Allocation*. Accessed: May 28, 2018. [Online]. Available: http://people.cs.vt.edu/∼tirtha23/CSGrad/TrafficLight.pdf

[17] J. Wang and P. Shi, "Research on bandwidth control technology based on SDN," in *Proc. IMCEC*, Xi'an, China, May 2018, pp. 705–708.

[18] C. B. Hauser and S. R. Palanivel, "Dynamic network scheduler for cloud data centres with SDN," in *Proc. UCC*, Austin, TX, USA, 2017, pp. 29–38.

[19] J. Heinanen and R. Guerin, "A single rate three color marker," IETF, Fremont, CA, USA, RFC 2697, Sep. 1999.

[20] J. Heinanen and R. Guerin, "A two rate three color marker," IETF, Fremont, CA, USA, RFC 2698, Sep. 1999.

[21] J. Shi and S.-H. Chung, "A traffic-aware quality-of-service control mechanism for software-defined networking-based virtualized networks," *Int. J. Distrib. Sensor Netw.*, vol. 13, no. 3, 2017. [Online]. Available: https://journals.sagepub.com/doi/full/10.1177/1550147717697984

[22] *P4–14 Language Specification, Version 1.0.5*. Accessed: Nov. 26, 2018. [Online]. Available: https://p4.org/p4-spec/p4–14/v1.0.5/tex/p4.pdf

[23] *Iperf, Version 2.0.5*. Accessed: Jun. 2, 2018. [Online]. Available: https://iperf.fr

[24] *Wireshark, Version 2.6.8*. Accessed: Apr. 8, 2019. [Online]. Available: https://www.wireshark.org/

Steven S. W. Lee received the Ph.D. degree in electrical engineering from National Chung Cheng University, Taiwan, in 1999. From 1999 to 2008, he was with Industrial Technology Research Institute, Taiwan, where he was the Leader of Intelligent Optical Networking Project and a Section Manager of Optical Communications Department. He was also a Research Associate Professor with National Chiao Tung University from 2004 to 2007. In 2008, he joined National Chung Cheng University, where he is currently a Professor with the Department of Communications Engineering. He has published over 80 papers in international journals and conferences and has coauthored 20 international patents in the areas of broadband communications. His research interests include optical and broadband networking, network optimization, green network, and software-defined networking.

Kwan-Yee Chan received the B.S. degree in computer science and information engineering from Nanhua University, Taiwan, in 2012. She is currently pursuing the Ph.D. degree with the Department of Communications Engineering, National Chung Cheng University, Taiwan. Her research interests include survivable network design and software-defined networking. She was a recipient of the Best Student Paper Award at IEEE CloudNet 2015. In 2017, she and her team members won the first prize on the SDN/NFV design contest hosted by Industrial Development Bureau, Ministry of Economic Affairs, Taiwan.