SDN-enabled Session Continuity for Wireless Networks

Wei-Wen Chen, Li-Hsing Yen, Chia-Lin Chuo, Ting-Hsuan Heish and Chien-Chao Tseng
Department of Computer Science
National Chiao Tung University
Hsinchu, Taiwan, R.O.C.
wwchen.cs97g@nctu.edu.tw, lhyen@cs.nctu.edu.tw, fox91119@gmail.com, kansokusha@gmail.com, cctseng@cs.nctu.edu.tw

***Abstract*—All active sessions of an ordinary host will be broken if the host changes its IP address as a result of migrating to a new subnet. Traditional solutions toward this problem either need modifying mobile hosts or create tunnels that cause inefficient triangle routing. SDN-based mobility schemes, on the other hand, focus on handover latency reduction or fast packet redirection after handover. There is no SDN-based approach that handles autonomous IP address changes by mobile hosts. As a remedy, we propose SDN-enabled Session Continuity (SDN-SC), as an SDN-based mobility management mechanism that retains session for hosts roaming across subnets in an SDN network. Particularly, SDN-SC suppresses possible address re-configurations with techniques such as gateway spoofing and DHCP lease renewal for mobile hosts away from home. We studied the performance of SDN-SC and compared it with that of MIPv4 and PMIPv6. The results show that SDN-SC outperforms both MIPv4 and PMIPv6 in terms of handover latency.**

Keywords—Session Continuity, Mobility Management, Software Defined Network (SDN), Handover Delay

I. INTRODUCTION

A large network in the Internet is typically divided into multiple subnets to facilitate network management and traffic engineering. In such a multi-subnet network, IP address plays dual roles. IP address as a *routing locator* is used in the network layer to locate nodes in the network. IP address as a *node identifier* is used in the transport layer to identify a participant or an endpoint of a networking application.

Particularly, IP address as a node identifier is embedded in and thus tightly coupled with session identifiers. A session identifier uniquely identifies a session established between two end hosts in the Internet. A session identifier should be valid throughout the lifetime of the session. However, when a roaming host changes its attachment to the Internet, it might change its IP address as well for IP address being used as a routing locator. Such a change effectively breaks all on-going sessions associated with the previous IP address. Therefore, we need a *session continuity* scheme that preserves all active sessions for roaming hosts despite possible changes of IP addresses.

The key to session continuity is to decouple the dual roles of IP address. IP address as a node identifier should be always respected for session continuity. On the other hand, IP address as a routing locator can be changed when mobile host changes locations. MIPv4 [1] is a mobility management scheme that takes home IP address as node identifier. When a mobile host enters a new routing domain, it needs to allocate a care-of address in that domain as a routing locator. The problem with MIPv4 is that all packets destined to the mobile host's home address should be tunneled to its care-of address. This creates inefficient triangle routings. Furthermore, the allocation of care-of address might involve executing dynamic host configuration protocol (DHCP) and duplicate address detection (DAD) procedures, which leads to long handover delays. MIPv6 [2] needs neither tunneling nor the execution of DHCP and DAD. However, it (as well as MIPv4) demands protocol modifications on mobile devices, which may not be viable. On the other hand, PMIPv6 [3] is a network-based mobility management scheme that needs modifying only network devices. However, it is still associated with drawbacks such as tunnel usage overhead and triangle routing paths.

In this paper, we are interested in adopting Software Defined Network (SDN) [4] technology for routing. SDN is an emerging networking architecture that decouples control plane from data plane. In SDN, a centralized controller has the visibility over both the networking topology and the exact attached points of all end devices. Therefore, it is capable to track the location and arrange traffic flows for each mobile host. More importantly, SDN no longer relies on IP address as the sole routing directive, making it a promising solution to support host mobility yet maintain session continuity.

We propose SDN-enabled Session Continuity (SDN-SC) as a network-based mobility management mechanism that retains sessions for hosts roaming across subnets. The design of SDN-SC ensures the following two properties. First, mobile hosts need not be aware of the proposed scheme. It is SDN-SC that takes care of location tracking and packet delivery. Second, mobile hosts retain their IP addresses even when they are away from their home subnets. These two properties are essential to session continuity but pose the main design challenges because mobile hosts may autonomously detect the need to change their IP addresses. SDN-SC suppresses possible address re-configurations with techniques such as gateway spoofing and DHCP lease renewal for mobile hosts away from home. Unlike conventional network-based schemes, SDN-SC uses direct end-to-end routes instead of tunnels to deliver packets to mobile hosts undergone handovers. This feature avoids potential inefficient triangle routings.

TABLE I. COMPARISON AMONG MOBILITY MANAGEMENT MECHANISMS

Name	MIPv4	MIPv6	PMIPv6	SDN-SC
Type	Host-based	Host-based	Network-based	Network-based
Handover Delay	Long	Long	Short	Short
Host Modification	Yes	Yes	No	No
Anchor Point Forwarding	Yes	No	Yes	No
Tunneling	Yes	No	Yes	No

A comparison among SDN-SC and several existing approaches is presented in Table I.

The rest of this paper is organized as follows. Section II presents background information and related works. Section III details the proposed scheme. Numerical results are reported in Section IV. Finally, Section V concludes the paper.

II. BACKGROUND AND RELATED WORK

SDN has been commonly associated with the OpenFlow (OF) [5] protocol, which is a south-bound protocol used between the controller and all switches being managed. The controller knows the topology of the whole network and the exact attached points of all end devices. For this reason, a controller can easily manage the whole network and track user's location.

Packet delivery in SDN is based on flows, and flows are basically created via *reactive flow creation*. When an OF-enabled switch receives a packet with a matching flow, the switch handles (probably forwards) that packet based on the instruction in the flow table. If there is no match for this packet in the flow table, the switch sends this packet to the controller with a packet-in message. The controller creates a flow for this packet, inserts appropriate flow entries to all switches along the route, and sends back a packet-out message to let the switch continue its forwarding.

Figure 1 shows the network architecture of SDN-SC. There is an SDN controller that manages all OF devices in the target network. These devices include a border gateway (BGW) that connects the network to the rest of the Internet, several subnet gateways (SGWs), one for each subnet, and other OF switches. The BGW and all SGWs are connected by wired or wireless links and collectively form a backbone network. Within each subnet, mobile hosts are assumed to use IEEE 802.11 links to connect to IEEE 802.11 access points (APs), which are then connected to the SGW via Ethernet. APs are used as native layer-two devices that simply bridge the Ethernet and 802.11 links.

Sen and Sivalingam [6] proposed an SDN framework for seamless mobility management. The framework deploys many light APs in the network. When an IEEE 802.11 device attaches to a light AP, the AP processes only Beacon and Probe frames. Other frames such as authentication and association are handled directly by SDN controller. All light APs are configured with the same Service Set Identifier (SSID) and Basic Service Set Identifier (BSSID) to enable seamless handover. In fact, such a configuration creates an illusion to a mobile host that there is only one AP in the whole network so it does not invoke AP discovery and re-authentication procedures. The controller designates one light AP to serve a mobile host, and manages the state of the AP to enable such a service.

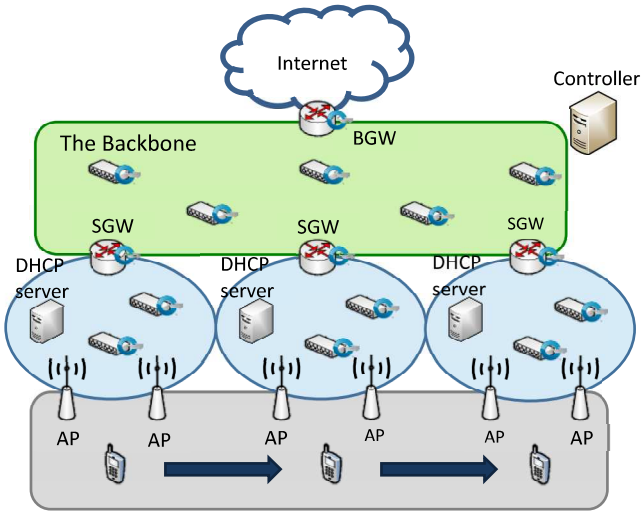

Fig. 1. SDN-SC netwrok architecture

This design significantly shortens handover delay, but it also poses considerable overhead on the controller. The design also relies on the availability of light APs. To avoid possible packet loss during a handover, this method needs creating a tunnel between the home AP and the new AP. The tunnel may cause the triangle routing problem.

Lin et al. [7] proposed Mobility SDN (M-SDN) to reduce traffic pause time due to handovers. M-SDN installs N-casting rules in relevant switches to prepare post-handover traffic flows for all potential handover targets. In this way, a mobile host can immediately receive redirected packets right after a layer-two handover. M-SDN also supports handovers across SDN domains. There is a location server that detects inter-SDN domain handover and informs controllers in all the potential target domains to prepare post-handover traffic flows.

Although M-SDN as well as other SDN approaches [8, 9] effectively reduces the latency of packet redirection after a handover, these approaches ignore the possibility that mobile hosts may autonomously change their IP addresses for several reasons. Besides increasing overall handover delay, such an address change will effectively break all active sessions. For this reason, suppressing the occurrence of address change is as important as packet redirection for a seamless handover that ensures session continuity.

III. THE PROPOSED MECHANISM

SDN-SC provides four major functions: *packet delivery*, *location tracking*, *packet redirection* and *suppression of IP reconfiguration*. *Packet delivery* is done with aggregated routes. *Location tracking* tracks the location of each mobile host so as to create specific flows for each host. *Packet redirection* modifies routes for all active inter-subnet flows in a mobile host following a layer-two handover of that host. *Suppression of IP reconfiguration* prevents possible IP address changes activated by mobile hosts so as to achieve session continuity. This section details these functions.

A. Aggregated Routes

The problem with basic reactive flow creation in SDN is scalability. Every time a new session comes in, the controller creates a new flow and installs needed flow rules to all associated devices. When the number of sessions becomes huge, there will be considerable flow rules in the backbone network which places significant burden on the controller as well as switches.

Our solution to this problem is to proactively create aggregated routes. We partition the whole address space into blocks and assign one block to each subnet. Flows for packet delivery within a subnet are still reactively created because intra-subnet packet delivery is based on MAC addresses for which no aggregated routes can be created. If the packet-in message corresponds to a broadcast ARP request looking for the MAC address of the host's default gateway, the controller instructs the SGW that forwards the packet-in message to return the MAC address that corresponds to the SGW's ingress port (on which the ARP request was received) to the host. This effectively assigns the SGW to be the host's default gateway.

For inter-subnet packet delivery, the controller proactively installs flow rules that implement traditional IP prefix routing (routing based on the prefix of the destination's IP address) to all SGWs in the backbone network. When hosts are in their home subsets, this implementation delivers packets destined to these hosts from any other SGW to the SGW of the destination's home. This approach significantly reduces the number of flow rules in the backbone network. However, we should additionally deal with packet delivery for mobile hosts that are out of their home networks. This is exactly what the location tracking function does.

B. Location Tracking

Any mobile host that joins the network should first attach to an IEEE 802.11 AP. Each AP is connected to some port of an SGW via Ethernet. Therefore, to locate a mobile host which attaches to some AP, the controller needs to know the Datapath Identifier (DPID) of the SGW and the port number of the SGW port to which the AP is connected. These two data, together with the MAC address of the mobile host, form a *location tuple* that allows the controller to uniquely locate and identify a roaming host. The problem is how to keep track of one location tuple for each mobile host.

After attaching to an AP, a new mobile host attempts configuring its address settings by broadcasting a DHCP discover message. This and subsequent messages do not cause the SGW to inform the controller with a packet-in message. Therefore, the initial configuration will not immediately create a location tuple in the controller. It is subsequent packets that triggers a location update.

After the initial configuration, the mobile host may issue various types of packets. Most intra-subnet unicast packets can be forwarded either by the AP or the SGW (if the packets match some flow in the SGW). Inter-subnet packets that match existing flows (for IP prefix routing) can also be handled without difficulty. On the other hand, packets that do not match any flow in the SGW will trigger a packet-in message sending to the controller. The message will be attached with the host's MAC address and ingress port number (among others). On receiving the packet-in message, the controller will perform a *location update*. If the MAC address of the mobile host is new to the controller, the controller will insert a location tuple for this newly attached host. Otherwise, SDN-SC will update an existing tuple with the same MAC address. The controller will also generate a Movement Detected event to other function modules of SDN-SC to take appropriate actions.

Therefore, the key to a correct and timely location update is to trigger a packet-in message right after a mobile host attaches to an AP. After the initial configuration, the first packet issued by the mobile host will usually be a broadcast ARP request that looks for the MAC address of the host's default gateway. For mobile host that has undergone a handover, the first packet will usually be a unicast ARP request or router solicitation for previous gateway detection (discussed later). In both cases, no matching flow can be found in the SGW and thus will trigger a location update.

C. Packet Redirection

Besides location update, SDN-SC also needs to perform *packet redirection*, i.e., modifying routes for all active inter-subnet flows after a layer-two handover of a mobile host. To this end, SDN-SC maintains a *session record* for each host. A session record keeps track of all active inter-subnet sessions. Recall that every new session triggers a pack-in message to the controller because the new session does not match any existing flow. Upon such events, SDN-SC updates the associated session record. When SDN-SC receives a Movement Detected event, it calculates new routes for all active sessions and installs relevant rules. These rules are specific to the mobile host, i.e., exceptions to the default aggregated routes.

When an OF switch detects that the condition of some flow rule has not been matched for an extended period of time, the switch deletes the corresponding rule and informs the controller of that deletion by Flow_Removed OF message. On receiving that message, SDN-SC removes the associated session from the session record.

D. Suppressing IP Re-configuration

As we do not modify the behaviors of mobile host, it is challenging to retain the mobile host's IP address when the host is away from home. There are many occasions for an ordinary host to autonomously configure a new IP address and related network-layer settings for its IEEE 802.11 interface. We discuss these occasions and possible treatments as follows.

Initial Configuration. A host should configure its IP settings before its packets can be delivered throughout the network. We assume that the IP settings are dynamically configured through DHCP protocol. Initial configuration is essential and causes no problem to session continuity.

Change of Service Set Identifier (*SSID*). When a mobile host switches from one AP to another with a different SSID, the mobile host may be forced to change its IP settings via DHCP even if these two APs belong to the same subnet [10]. To avoid possible change of SSIDs, we configure all APs with the same SSID.

Failure of Previous Gateway Detection. Although all APs are configured with the same SSID, they have different BSSIDs

(i.e., MAC addresses). Therefore, mobile hosts can detect the change of APs. For that reason, the first action performed by a mobile host after a layer-two handover is usually[1] *previous gateway detection* which checks the reachability of the serving gateway in the current subnet. For IPv4, previous gateway detection is realized by a unicast ARP request that looks for the MAC address of the configured default gateway. For IPv6, previous gateway detection is realized by a router solicitation message with the expectation of a router advertisement message in return. If the two APs belong to different subnets, the detection surely fails because neither the unicast ARP request nor the router solicitation message can cross subnets. The failure of the previous gateway detection will cause a layer-three handover, i.e., change of IP settings.

To prevent possible failure of the previous gateway detection, SDN-SC implements *gateway spoofing*. After being informed of a Movement Detected event and knowing that the packet-in message is a unicast ARP request, SDN-SC sends back a packet-out message to the SGW to instruct the SGW to return an ARP response to the mobile host. This action makes the mobile host believe that its default gateway is still reachable and thus prevents the host from initiating a new address configuration.

Failure of DHCP Lease Time Update. All IP settings configured through a DHCP are associated with respective lease durations. Before the expiration of the current lease (usually at halfway through the lease period), the host should request renewal and extension of the lease from the DHCP server. This is done through a unicast DHCP request message. If the host receives an acknowledgement (DHCP Ack) before the current lease expires, the host can keep its settings. Otherwise, the host should re-configure its IP settings and may possibly change its IP address.

To make the host believe that the DHCP server grants its request, SDN-SC sends back DHCP Ack to the host. Besides, SDN-SC forwards the host's request to the intended DHCP server to extend the lease. Furthermore, because DHCP messages are delivered via UDP which may get lost, SDN-SC establishes a TCP connection for the delivery of DHCP-related packets between the controller and each SGW. There is still a small probability that the request for lease update fails because TCP connections end at SGWs rather than hosts and DHCP servers.

E. Implementation Details

The software architecture of SDN-SC is shown in Fig. 2. We implemented SDN-SC as an App running on Ryu controller. The App cooperates with an official Topology App which retrieves network topology data. We augmented the Topology App with the proposed location tracking function. The modified Topology App will notify the SDN-SC App of a Movement Detected event when it detects a host movement.

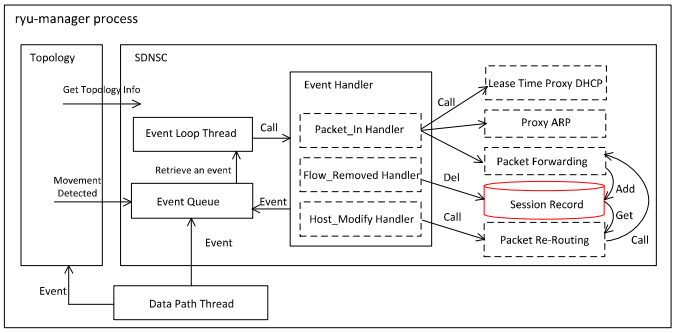

Fig. 2. Software Architecture of SDN-SC

The SDN-SC App has its own event queue that stores events from the data path, the Topology App, and itself. Running as an event loop thread, the SDN-SC App repeatedly picks one event from the event queue and calls one of the following three event handlers: *Packet_In*, *Flow_Removed* and *Host_Modify*. The event handler may then call other function modules.

The *Packet_In* handler processes packet-in messages and calls an appropriate function module. If the packet-in message corresponds to DHCP-related packets, the handler calls Lease Time Proxy DHCP module, which prevents potential failures of DHCP lease renewal. If the message corresponds to ARP packets, the handler calls Proxy ARP module, which implements gateway spoofing. All other packet-in messages are processed by the Packet Forwarding module, which, for each new session, creates a flow and adds an entry to the session record.

The *Flow_Removed* handler processes Flow_Removed messages. It removes obsolete flow rules and deletes associated entry from the corresponding session record.

The *Host_Modify* handler takes care of Movement Detected events and invokes Packet Re-Routing module. The module implements packet redirection.

IV. PERFORMANCE EVALUATION

We conducted experiments to compare SDN-SC with MIPv4 and PMIPv6. We are primarily concerned with the latency caused by each scheme.

A. MIPv4 vs. SDN-SC

We used OpenNet [11] to simulate the experimental environment as shown in Figures 3 and 4. In all simulations, a mobile node (MN) used a TCP connection to send 8 Mb data to a corresponding node (CN). During the transmission, the MN experienced one to four inter-subnet handovers. We varied the number of handovers to investigate several time metrics. The results were obtained based on TCP dump data collected at the CN with Wireshark. We measured the performance of MIPv4 and SDN-SC in terms of handover delay, effective transmission time and total file transmission time.

[1] This is implementation dependent.

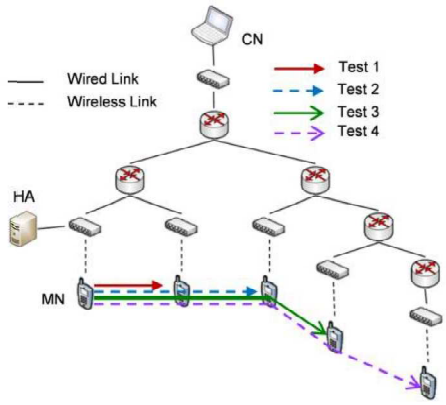

Fig. 3. Experimental Environment - MIPv4

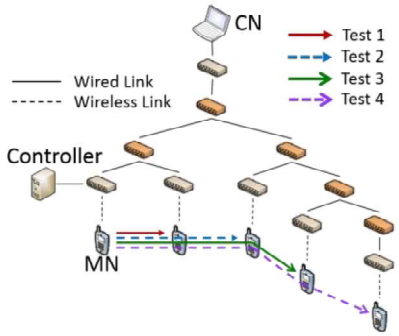

Fig. 4. Experimental Environment – SDN-SC

Handover delay (HD) is the length of the duration measured at the CN that starts from the receipt of the last packet before handover to the receipt of the first packet after handover. *Effective transmission time* (ETT) measures the total time actually spent in packet transmission. *Total file transmission time* (TFTT) is the time between the receipts of the first and the last packets at the CN. Figure 5 shows the components that comprise the TFTT for a session between the CN and the MN that undergoes two handovers. The components include the ETT before the first handover (ETT1), the first HD (HD1), the ETT after but before the second handover (ETT2), the second HD (HD2) and the ETT after the second handover (ETT3). Figures 6 to 8 show the results of HD, ETT, and TFTT, respectively. Observe that SDN-SC outperforms MIPv4 in all settings.

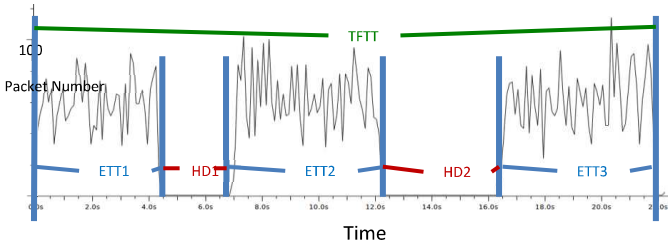

Fig. 5. The components of TFTT

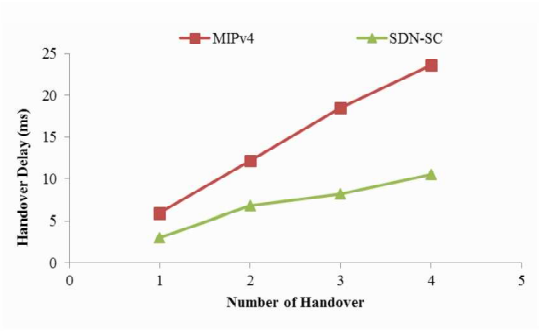

Fig. 6. Handover Delay: MIPv4 vs.SDN-SC

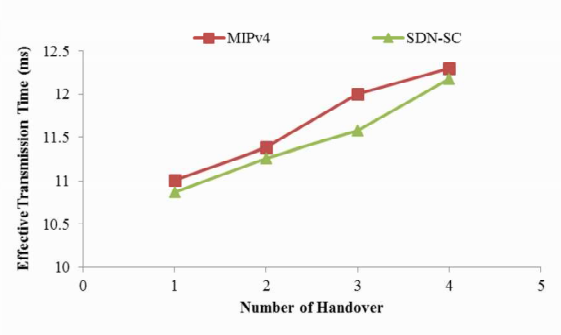

Fig. 7. Effective Transmission Time: MIPv4 vs. SDN-SC

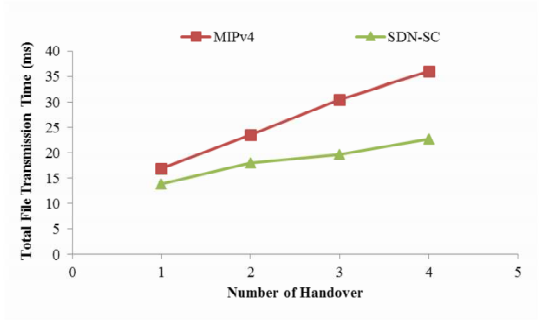

Fig. 8. Total File Transmission Time: MIPv4 vs. SDN-SC

B. *PMIPv6 vs. SDN-SC*

As we found no implementation of PMIPv6, we analyzed first all the tasks to be done for handovers in PMIPv6 and SDN-SC, respectively. The results are shown in Table II. The main difference between PMIPv6 and SDN-SC is that PMIPv6 needs no flow rule installation and SDN-SC needs no explicit location update message.

TABLE II. HANDOVER DELAY COMPOSITION: PMIPV6 VS. SDN-SC

Phase	PMIPv6	SDN-SC
L2H	V	V
L3H Detection	V (Router solicitation)	V (Unicast ARP)
Location Update	V (PBU, PBA)	X (Done by L3H Detection)
Flow Rule Installation	X	V
First Packet Delivery	V (Tunnel & Anchor point forward)	V (Flow rule match & Direct route)

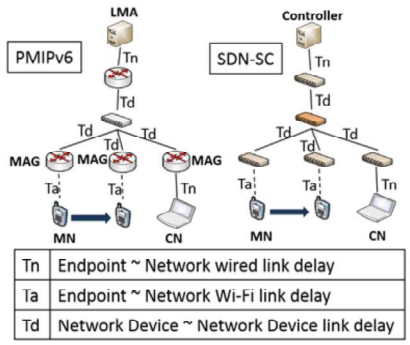

Fig. 9. Experimental Environment: PMIPv6 vs. SDN-SC

TABLE III. LINK DELAY: PMIPV6 VS. SDN-SC

Phase	PMIPv6	SDN-SC
L2H	Same	Same
L3H Detection	2Ta	2Ta+4Td+2Tn
Location Update	4Td+2Tn	X
Flow Rule Installation	X	V
First Packet Delivery	Ta+4Td+3Tn	Ta+2Td+Tn

Figure 9 shows the experimental environments for PMIPv6 and SDN-SC, respectively. We ignored device's processing time and focused on link delay. The link delay of each link is shown in Fig. 9. With this setting, Table III shows the delays incurred by the tasks of PMIPv6 and SDN-SC, respectively. For the sum of delays caused by layer-two handover (L2H), layer-three handover (L3H) detection and location update phases, the result of PMIPv6 is identical to that of SDN-SC.[2] Therefore, the relative performance of these two schemes only depends on data transmission time, i.e., the time to the delivery of the first packet after location update (for SDN-SC, the time for flow installation also counts.)

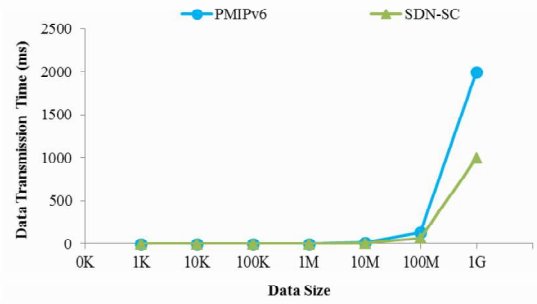

Fig. 10. Data Transmission Time: PMIPv6 vs. SDN-SC

We used OpenNet to simulate the network environment shown in Fig. 9. In simulations, MN created a TCP connection to send data to CN. We varied the amount of data sent and used qperf [12] to measure data transmission time after location update. Although we do not have PMIPv6 implementation, we created a tunnel to simulate packet deliver in PMIPv6 and thus measured the data transmission time. The data transmission time of PMIPv6 and SDN-SC is shown in Fig. 10. As expected, SDN-SC has a shorter data transmission time than PMIPv6. The difference is more significant with a larger data size.

V. CONCLUSIONS

We have proposed SDN-enabled Session Continuity (SDN-SC) to retain sessions for hosts roaming across subnets in an SDN network. SDN-SC is a network-based mobility management scheme that ensures session continuity and provides direct routes. Unlike previous SDN-enabled mobility approaches which only focus on packet redirection, SDN-SC particularly suppresses autonomous IP re-configurations. When compared with existing schemes, MIPv4 and PMIPv6, numerical results indicate that SDN-SC yielded the lowest latency.

Future extensions of this work include actual implementation of PMIPv6 for performance comparison and investigating the impact of the *Flow Rule Installation Time* on performance. We shall study additional performance metrics like packet loss rate. We also consider reducing the number of flow rules by using non-direct routes. Finally, M-SDN may be integrated with SDN-SC to predict handover targets and pre-install flow rules.

Acknowledgements

This work was supported in part by Ministry of Science and Technology under the contract numbers MOST 105-2622-8-009-008 and MOST 104-2221-E-009-021-MY3.

References

[1] C. Perkins, "IP Mobility Support for IPv4, Revised" IETF RFC 5944, Nov. 2010.

[2] C. Perkins, "IP Mobility Support in IPv6" IETF RFC 6275, July 2011.

[3] S. Gundavelli et al., "Proxy Mobile IPv6", IETF RFC 5213, Aug. 2008.

[4] Open Networking Foundation, "Software-Defined Networking: The New Norm for Networks" ONF White Paper, 2012.

[5] N. McKeown et al., "OpenFlow: enabling innovation in campus networks," *ACM SIGCOMM Computer Communication Review*, 38(2): 69-74, Apr. 2008.

[6] A. Sen and K. M. Sivalingam, "An SDN framework for seamless mobility in enterprise WLANs," *IEEE 26th Annual Int'l Symp. on Personal, Indoor, and Mobile Radio Communications* (PIMRC) , Aug.-Sept. 2015.

[7] C. Chen et al., "Mobility management for low-latency handover in SDN-based enterprise networks," *IEEE Wireless Communications and Networking Conference* (WCNC), Apr. 2016.

[8] C. Guimaraes et al., "Empowering software defined wireless networks through media independent handover management," *IEEE Global Communications Conference* (GLOBECOM), Dec. 2013.

[9] Y. Wang and J. Bi, "A solution for IP mobility support in software defined networks," *23rd Int'l Conf. on Computer Communication and Networks* (ICCCN), Aug. 2014.

[10] L.-H. Yen et al., "Experimental study of mismatching ESS-subnet handoffs on IP over IEEE 802.11 WLANs," *8th Int'l Conf. on Wireless and Optical Communications Networks* (WOCN), May 2011.

[11] M.-C. Chan et al., "OpenNet: a simulator for software-defined wireless local area network," *IEEE Wireless Communications and Networking Conference* (WCNC), Apr. 2014.

[12] qperf, URL: http://linux.die.net/man/1/qperf

[2] Here we ignore the delay incurred by AAA authentication in PMIPv6.