A Proposal of SDN Based Mobility Management in Multiple Domain Networks

Misumi Hata*, Satoru Izumi†, Toru Abe*†, and Takuo Suganuma*†
*Graduate School of Information Sciences, Tohoku University
†Cyberscience Center, Tohoku University
2-1-1 Katahira, Aoba-ku, Sendai 980-8577, JAPAN
E-mail:{m-hata, izumi}@ci.cc.tohoku.ac.jp, {beto, suganuma}@tohoku.ac.jp

***Abstract*—In this paper, we propose a mobility management architecture based on Software Defined Network (SDN) for multiple domain networks. In this architecture, each SDN controller cooperates with each other to optimize communication route covering multiple domain networks for seamless IP services even when the mobile devices move to other domain. This paper presents the basic design of the architecture and protocol to cover wide area for mobility management. We also show the initial experiments to confirm the behavior of our proposed architecture.**

I. Introduction

Cloud services are widely prevalent while mobile devices and wireless technologies are developing rapidly. Thus, peaple can enjoy various IP services anytime, anywhere, even during displacement. Some IP services such as file transfer system and video communication require seamless IP communications. Therefore, mobility management for realizing continuous IP services has attracted attention.

Mobile IP is a standardized mobility management technology and is widely used [1][2][3]. This is a communication protocol to enable a mobile device to keep continuous IP services when its user moved to other domain networks by maintaining permanent IP address. However, Mobile IP has some difficulty in optimizing communication routes. Communicating via unsuitable routes after handover leads to communication delay. Some papers show approaches to apply Software Defined Network (SDN) to mobility management in order to solve the problem of Mobile IP [4][5]. However, the existing approaches focus on intra-domain route optimization. Hence, effective mobility management that supports inter-domain handover has not been realized.

In this research, we propose SDN based mobility management architecture for multiple domain networks. In this architecture, each SDN controller manages its own domain network and they cooperate with each other to optimize inter-domain communication routes when mobile devices move to other domain networks while maintaining seamless IP services .

This paper presents the basic design of the architecture and a protocol that we propose for inter-domain mobility management. We will also show the results of initial experiments to confirm the behavior of our proposed architecture.

II. Relate Works

A. Mobile IP

Mobile IPv4 [1] is a protocol to maintain communication between a Mobile Node (MN) and a Correspondent Node (CN), which the MN is communicating, after the MN moves to other network domain. Mobile IP uses Home Address (HoA) and Care-of-Address (CoA). Mobile IP introduces a Home Agent (HA) to manage mobility binding which is a mapping information between MN's HoA and current CoA. The HA forwards packets from the CN to the MN based on the mobility binding.

However, there is a possibility of triangle routing problem due to the location of the HA. The communication route after movement of MN may be lengthy, which may cause communication delay. Mobile IPv6 [2] has a route optimization function. However, only the MNs with support for route optimization can use this function.

Proxy Mobile IPv6 (PMIPv6) [3] is a protocol to support network based mobility management. In this protocol, network devices process mobility management. Thus, this approach does not require MN to be involved in signaling message exchanges. PMIPv6 introduces a Local Mobility Anchor (LMA) to forward packet to the MN and a Mobility Access Gateway (MAG) to detect the MN and establish connection to the LMA. The LMA forwards all packets from the MN to the MAG to communicate to the CN. However, there is also a possibility of triangle routing problem which comes from the location of the LMA.

B. Mobility Management based on SDN

SDN and OpenFlow have attracted attention as a way to solve the route optimization problem because they enable a flexible network management by having centralized control over a network. Hence, there are approaches to apply SDN and OpenFlow to mobility management. In paper [6], triangle routing is allowed for inter-domain handover.

In the approach in [5], all SDN controllers inform the movement of a MN to each other when an inter-domain handover occured. Therefore, amount of traffic increases at frequent handover or networks with many domains.

The approach in [4] assigns IDs to each MN and updates mapping of ID and IP address of the MN at handover.

© Copyright IEICE – The 18th Asia-Pacific Network Operations and Management Symposium (APNOMS) 2016

However, it has no function to optimize communication route among network domains after handover.

These approaches focus on inter-domain mobility management and route optimization. In recent network environment, users move freely to various domain networks while using IP services. To realize mobility management among multiple domain networks, the SDN controllers need to inform its own domain information to each other. However, it is hard to share domain information among numerous domains. Therefore, we need effective mobility management in multiple domain networks.

III. A Proposal of SDN Based Mobility Management Architecture

A. Overview

To deal with the problem we presented in the previous section, we propose a SDN based mobility management architecture to optimize routes after inter-domain handovers. As an architecture to manage inter-domain SDN network, WE-Bridge is proposed in [7]. We extend this architecture for mobility management.

When a MN makes an inter-domain move, the domain to which the MN belongs shares information of the MN with the domain to which a CN belongs and the other domains involving in this move.

Our architecture optimizes inter-domain routes between MNs and CNs. To optimize the inter-domain routes, the involving domains cooperate and recognize the current position of the MN and the CN.

Thus, in our proposed architecture, domains share information of a MN and calculate new optimized routes between the MN and a CN after the MN moves. We choose the least needed domains including following domains to share the information.

1) The domain to which the MN belonged before move
2) The domain to which the MN belongs after move
3) The domain to which the CN belongs

B. Design

The architecture of the proposal is designed as shown in Figure 1. In this figure, a source domain (D_s) is the domain to which a MN belonged before a movement, a destination domain (D_d) is the domain to which the MN belongs after the movement, and D_c is the domain to which a CN belongs. $D_{p\#}$ is the domain the MN is communicating with the CN via after the movement. Each domain have SDN switches that are connected to a SDN controller.

The architecture has two layers of network: a controller network and a switch network. In the controller network, the SDN controllers exchange their information. The switch network supports IPv6 network and MNs needs no extra features except IPv6 support.

When a MN moves, a "node connecting information" is shared between domains. Node connecting information is a set of information that indicates the prior position and the current position of the MN. Node connecting information consists of following information.

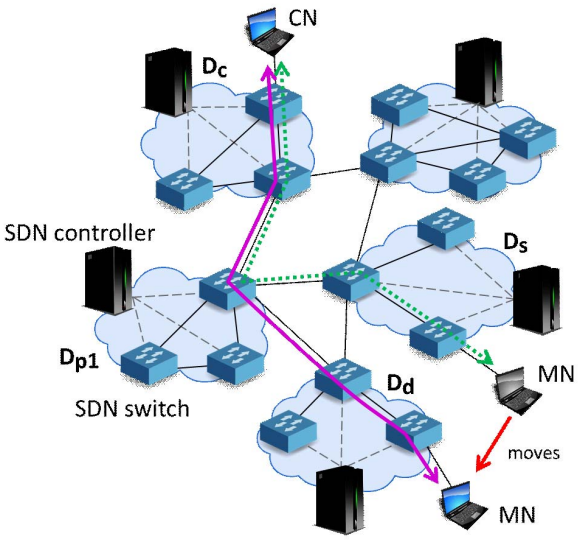

Fig. 1: Overview of SDN based mobility management architecture

- MAC address of MN
- IP address which the MN was using in the prior domain
- IP address whice the MN is using currently
- network prefix length of each IP address

This architecture conducts inter-domain handover in the order we show below. As an example of an inter-domain handover, we assume that the MN in D_s is communicating with a CN in D_c and the MN moves to D_d as shown in Figure 2. The architecture behaves as below. In this example, the node connecting information includes the following contents.

- MAC address of MN : 13:24:35:46:57:68
- IP address which the MN was using in the prior domain : fc00:2580:0300::ac33/64
- IP address which the MN is using currently : 2001:db0::200b:446a/64

1) When a inter-domain handover occurs, the controller in D_d detects the attach of MN.
2) The controller in D_d searches for D_s and the controller in D_s sends the IP address that MN was using to the controller in D_d as shown in Figure 2.
3) The controller in D_d generates a node connection information and announces the node connection information to controllers in D_s and in D_c. Figure 3 shows its behavior.
4) The controllers in D_d calculates communication route between the MN and the CN. Inter-domain routes are calculated by the controller in D_d using inter-domain topology map. Intra-domain routes are calculated by each controller managing their own domain as shown in Figure 4.
5) The controllers install flow entries to their managing switches according to the calculated communication route. We rewrite the destination or source IP address at the switch which the CN is connected to as shown in

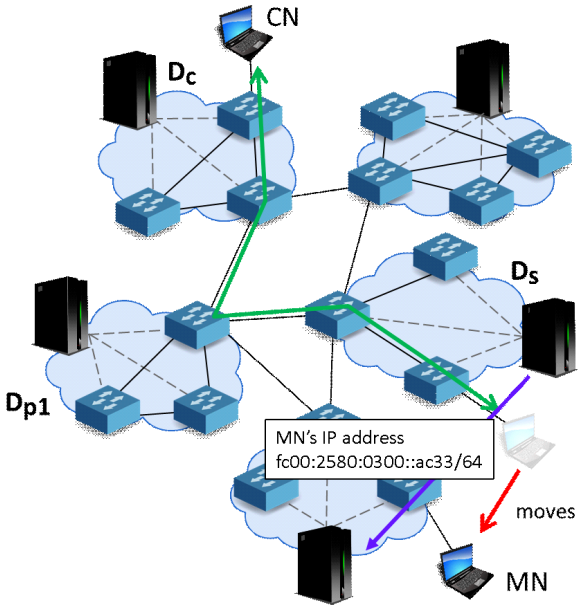

Fig. 2: Transmitting IP address after movement of MN

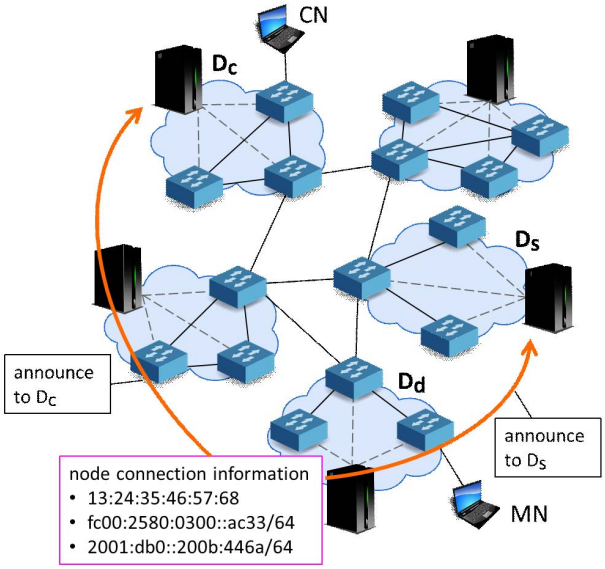

Fig. 3: Announcement of node connection information

Figure 4.

By following the procedure mentioned above, MN and CN can continue communicating by optimized routes. Moreover, the CN can communicate with the MN using MN's prior IP address even after the MN moved to other domain and got a new IP address.

IV. Experimentation

A. Methods

In this section, we would show the procedures and results of initial experiment. We conducted this initial experiment to

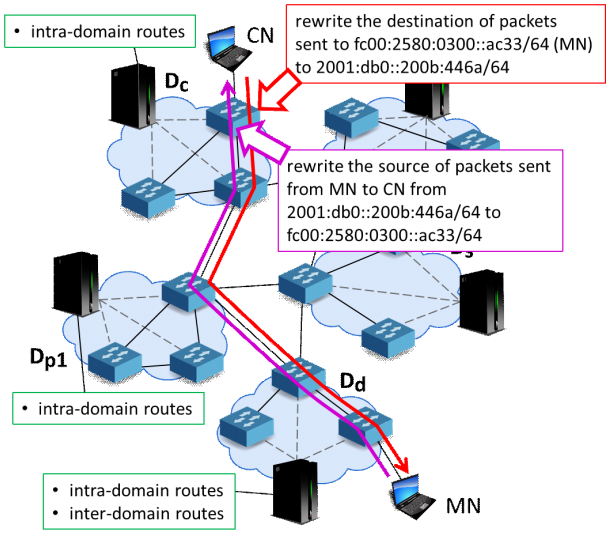

Fig. 4: Calculation of optimized routes

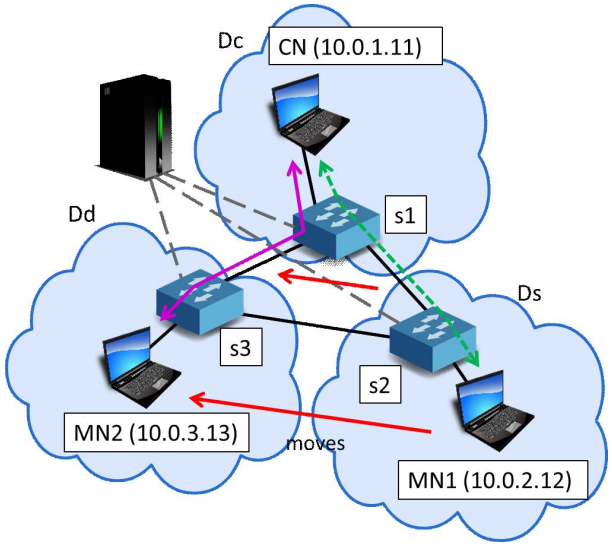

Fig. 5: Experimental scenario

confirm that a MN and a CN are able to continue communicating after the MN moved and its IP address changed.

In Table I, we show the experimental environment and tools. We constructed the experimental environment on a virtual machine.

The controllers communicate with switches using OpenFlow protocol. In Figure 5, we show the experimental network configuration. We built a virtual network in Mininet and set

TABLE I: Experimental environment

OS in virtual machine	Ubuntu 14.04
virtual network	Mininet 2.2.1
SDN controller	OpenDaylight Hydrogen
SDN switch	Open vSwitch 2.0.2
communication protocol between controller and switch	OpenFlow 1.0

Time	Source	Destination	Protocol
21:48:01.599614	10.0.1.11	10.0.2.12	ICMP
21:48:01.599652	10.0.2.12	10.0.1.11	ICMP
21:48:02.599607	10.0.1.11	10.0.2.12	ICMP
21:48:02.599633	10.0.2.12	10.0.1.11	ICMP
21:48:03.599607	10.0.1.11	10.0.2.12	ICMP
21:48:03.599635	10.0.2.12	10.0.1.11	ICMP
21:48:04.599606	10.0.1.11	10.0.2.12	ICMP
21:48:04.599639	10.0.2.12	10.0.1.11	ICMP
21:48:05.599614	10.0.1.11	10.0.2.12	ICMP
21:48:05.599653	10.0.2.12	10.0.1.11	ICMP
21:48:06.599614	10.0.1.11	10.0.2.12	ICMP
21:48:06.599641	10.0.2.12	10.0.1.11	ICMP
21:48:07.599603	10.0.1.11	10.0.2.12	ICMP
21:48:07.599639	10.0.2.12	10.0.1.11	ICMP
21:48:08.599606	10.0.1.11	10.0.2.12	ICMP
21:48:08.599770	10.0.2.12	10.0.1.11	ICMP
21:48:09.599615	10.0.1.11	10.0.2.12	ICMP
21:48:09.599699	10.0.2.12	10.0.1.11	ICMP
21:48:10.599608	10.0.1.11	10.0.2.12	ICMP
21:48:10.599649	10.0.2.12	10.0.1.11	ICMP
21:48:11.599627	10.0.1.11	10.0.2.12	ICMP
21:48:11.599678	10.0.2.12	10.0.1.11	ICMP
21:48:12.599608	10.0.1.11	10.0.2.12	ICMP
21:48:12.599648	10.0.2.12	10.0.1.11	ICMP
21:48:13.599605	10.0.1.11	10.0.2.12	ICMP
21:48:13.599647	10.0.2.12	10.0.1.11	ICMP
21:48:14.599605	10.0.1.11	10.0.2.12	ICMP

Fig. 6: Experimental result: packets captured at CN

Time	Source	Destination	Protocol
21:48:01.599644	10.0.2.12	10.0.1.11	ICMP
21:48:02.599619	10.0.1.11	10.0.2.12	ICMP
21:48:02.599630	10.0.2.12	10.0.1.11	ICMP
21:48:03.599621	10.0.1.11	10.0.2.12	ICMP
21:48:03.599631	10.0.2.12	10.0.1.11	ICMP
21:48:04.599621	10.0.1.11	10.0.2.12	ICMP
21:48:04.599634	10.0.2.12	10.0.1.11	ICMP
21:48:05.599633	10.0.1.11	10.0.2.12	ICMP
21:48:05.599647	10.0.2.12	10.0.1.11	ICMP
21:48:06.599626	10.0.1.11	10.0.2.12	ICMP
21:48:06.599636	10.0.2.12	10.0.1.11	ICMP
21:48:07.599616	10.0.1.11	10.0.2.12	ICMP
21:48:07.599629	10.0.2.12	10.0.1.11	ICMP

Fig. 7: Experimental result: packets captured at MN1

the IP addresses of the nodes as shown in Figure 5. We equipped 3 hosts and 3 SDN switches. Each domain includes one host and one SDN switch. In this case, one controller controls all domain networks. We will explain experimental scenario. At first, a MN1 in D_s is communicating with a CN in D_c. When the MN1 moves to D_d and changes its IP address from 10.0.2.12 to 10.0.3.13, the communication route between MN1 and CN changes as shown in Figure 5. We executed the experiment in the order we show below.

1) MN1 was communicating with CN.
2) MN1 moved and it became MN2.
3) The controller changed the flow entries so that CN and MN2 would communicate.
   - Change the destination of the flow whose destination is MN1 to MN2 at the SDN switch s1.
   - Change the source of the flow sent from MN2 to CN at the SDN switch s3.
4) We checked if the communication between CN and MN1 is continuing apparently.

We set the actions to disguise MN2 as MN1. CN thinks that it's communicating with MN1 while it's actually communicating with MN2.

B. Results

We kept CN pinging to MN1 while we changed the flow entries. As the result, we could confirm that CN continued to

Time	Source	Destination	Protocol
21:48:08.599704	10.0.1.11	10.0.3.13	ICMP
21:48:08.599715	10.0.3.13	10.0.1.11	ICMP
21:48:09.599672	10.0.1.11	10.0.3.13	ICMP
21:48:09.599689	10.0.3.13	10.0.1.11	ICMP
21:48:10.599626	10.0.1.11	10.0.3.13	ICMP
21:48:10.599639	10.0.3.13	10.0.1.11	ICMP
21:48:11.599650	10.0.1.11	10.0.3.13	ICMP
21:48:11.599669	10.0.3.13	10.0.1.11	ICMP
21:48:12.599625	10.0.1.11	10.0.3.13	ICMP
21:48:12.599638	10.0.3.13	10.0.1.11	ICMP
21:48:13.599623	10.0.1.11	10.0.3.13	ICMP
21:48:13.599636	10.0.3.13	10.0.1.11	ICMP
21:48:14.599621	10.0.1.11	10.0.3.13	ICMP

Fig. 8: Experimental result: packets captured at MN2

communicate with MN1, which is actually MN2 disguising as MN1. In Figure 6 we show the packets captured at CN. CN kept communicating with MN1 until we ended capturing the packets. Meanwhile, MN1 stopped communicating with CN and at the same time MN2 started to communicate with CN. Figure 7 shows the packets captured at MN1 and Figure 8 shows the packets captured at MN2. We could confirm the end of communication at MN1 from Figure 7 and the start of communication at MN2 from Figure 8. The time communicating node changed is the timing we changed the flow entries.

From the experimental results, we can tell that it's possible to realize mobility management based on SDN technology.

V. Conclusion

In this paper, we presented a basic design of our proposed architecture and performed a initial experiment to confirm the behavior of our architecture. From the experiment, we confirmed that a MN could continue communicating with a CN after it's movement. Therefore, it is able to realize the mobility management based on SDN.

For future work, we would design an algorithm to find the domain to which a MN belonged before movement and an inter-domain routing algorithm. Furthermore, we would implement the proposed architecture and execute experiments to evaluate this architecture.

References

[1] C. Perkins, "Ip mobility support for ipv4," Nov. 2010, https://tools.ietf.org/html/rfc5944.

[2] D. Johnson *et al.*, "Mobility support in ipv6," Jun. 2004, https://tools.ietf.org/html/rfc3775.

[3] S. Gundavelli *et al.*, "Proxy mobile ipv6," Aug. 2008, https://tools.ietf.org/html/rfc5213.

[4] Y. Li *et al.*, "Software defined networking for distributed mobility management," in *IEEE Globecom Workshops*, dec 2013, pp. 885–889.

[5] Y. Wang *et al.*, "Design and implementation of a software-defined mobility architecture for ip networks," *Mobile Networks and Applications*, vol. 20, no. 1, pp. 40–52, feb 2015.

[6] Y. Wang *et al.*, "A solution for ip mobility support in software defined networks," in *23rd International Conference on Computer Communication and Networks (ICCCN)*, Aug. 2014, pp. 1 – 8.

[7] P. Lin *et al.*, "A west-east bridge based sdn inter-domain testbed," in *Communications Magazine, IEEE (Volume:53 , Issue: 2)*, Feb. 2015, pp. 190–197.