Implementation and Evaluation of NTMobile with Android Smartphones in IPv4/IPv6 Networks

Kazuma Kamienoo*, Hidekazu Suzuki*, Katsuhiro Naito † and Akira Watanabe*
* Graduate School of Science and Technology, Meijo University, Aichi 468-8502, Japan
Email: 123430013@ccalumni.meijo-u.ac.jp, {hsuzuki, wtnbakr}@meijo-u.ac.jp
†Graduate School of Engineering, Mie University, Mie 514-8507, Japan
Email: naito@elec.mie-u.ac.jp

***Abstract*—We have been proposing a new IP mobility technology, called "Network Traversal with Mobility" (NTMobile), that solves the NAT traversal problem and provides flexible mobility in IPv4 and IPv6 networks at the same time. In this paper, we have implemented the NTMobile functions in an Android smartphone and performance evaluated in the IPv4 and IPv6 networks. It was confirmed that the NTMobile-compatible nodes can establish a connection with the correspondent node and move between IPv4 and IPv6 networks with low latency.**

***Index Terms*—Mobility, NAT Traversal, IPv4/IPv6 Networks.**

I. INTRODACTION

IP network systems are currently in a transitional period from IPv4 to IPv6. As a result, IPv4 and IPv6 coexisting networks are about to prevail. However, since IPv6 architecture has no compatibility with IPv4, it is not possible to communicate between these networks directly. In the case of IPv4 networks, private networks are generally built using Network Address Translation (NAT) routers, but the NAT goes against the basic principle of the Internet; i.e. it interrupts the end-to-end connectivity. In order to solve this problem, various NAT traversal technologies have been proposed [1]–[3]. However, to ensure the connectivity between IPv4 and IPv6 networks, it is necessary to introduce translator technologies such as NAT-PT, separately. Thus, if you consider the increasingly complex IP networks, it is important to realize mutual connectivity between IPv4 and IPv6 networks based on a single technology.

In the meantime, demand for mobile communication is increasing, with the spread of mobile devices such as smartphones and also due to the development of mobile Internet technologies. However, if the IP address of a mobile node changes along with the changes in networks or wireless interfaces, communication is to be broken, as the IP connection is managed by the IP addresses allocated to the interface of both nodes. The technology to solve this problem is called "mobility technology", and various kinds of mobility technologies have been proposed [4]–[7]. However, most of the conventional mobility technologies assume the networks based on IPv6, and even if their application to the networks based on IPv4 is considered, the existing technologies have problems in that the mobility or communication is restricted by the existence of NAT, and the communication route becomes lengthy.

As a technology to realize mobile transparency in IPv4 and IPv6 networks, Dual Stack Mobile IPv6 (DSMIPv6) [8] is standardized by IETF. This is an extended technology of Mobile IPv6. In DSMIPv6, communication between IPv4 and IPv6 networks is realized through the data-packet relaying function of Home Agent (HA) installed in the home network. Although technologies such as route optimization and distributed installation of HA (Global HA to HA Protocol) [9] are also standardized by IETF as extended technologies of Mobile IPv6, these technologies do not assume that they are applied to IPv4 networks. Accordingly, DSMIPv6 has problems in the points that communications always take a lengthy route via HA and also that they encounter single point failures of HA in IPv4 and IPv6 networks.

We have been proposing a new IP mobility technology called "Network Traversal with Mobility" (NTMobile) that can achieve communication connectivity and flexible mobility simultaneously in IPv4 and IPv6 networks [10]. In NTMobile, connection between the end nodes is established based on virtual IPv4/IPv6 addresses that are independent of the real IP addresses. To forward IP packets, UDP tunnel based on the real IP addresses is created on the optimal path, corresponding to the existence or non-existence of NAT and the type of IP network by which the end nodes have started communication. Then, even if a node moves to another network during communication, UDP tunnel is recreated and the change in the real IP addresses is hidden from application, and in this way, mobility is realized.

In this paper, we have implemented NTMobile in an Android smartphone, and have evaluated the handover latency in IPv4 and IPv6 networks. Hereinbelow, we explain the outline of NTMobile in Chapter II, describe the implementation and evaluation results in Chapter III, and finally conclude this paper in Chapter IV.

II. NTMOBILE

A. Overview

Fig. 1 shows the outline of our NTMobile system together with devices used. The NTMobile system consists of Direction Coordinator (DC), NTMobile-compatible nodes (NTM node) and Relay Server (RS).

- Direction Coordinator
  DC is a device that has roles to assign virtual IP addresses to NTM nodes and instruct them to create a tunnel route

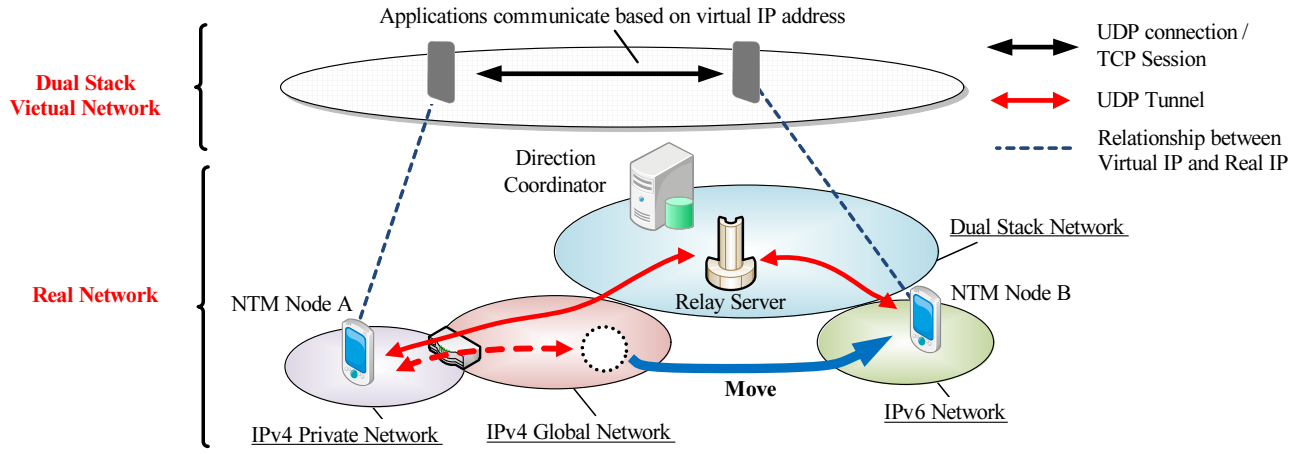


Fig. 1. Overview of NTMobile.

between end nodes. Virtual IP address is a unique address assigned to NTM nodes by DC from the address space allocated for virtual addresses. DC has also a dynamic DNS server function to manage address information of NTM nodes, by registering A/AAAA records of NTM nodes and also records specific to NTMobile (hereinafter called "NTM record"). NTM record is divided into two types; one is NTMv4 record for IPv4 address information and the other is NTMv6 record for IPv6 address information. Each of them keeps such records as Fully Qualified Domain Name (FQDN), real IP address and virtual IP address of each NTM end, real IP address of the global side of the NAT router, and real IP address of DC that manages own address information. DC is installed in the dual stack network. It is possible to install multiple DCs, depending on the scale of the network and NTM nodes.

- NTM Node
  NTM node has two types of IP addresses; i.e., one is real IP address assigned by the connected network, and the other is virtual IPv4/IPv6 addresses assigned by DC. Virtual IP address is an address which is independent of the network and thus, it does not change even if NTM node is assigned a new address from the connected network. NTM node has both virtual IPv4 address and virtual IPv6 address, and applications make communication based on either of them. As NTM node creates a tunnel at the IP layer and forwards the packets of to virtual IP addresses, it can perform end-to-end communication regardless of the difference in real networks.
- Relay Server
  RS is a device that relays data packets between end nodes in specific situations; for example, when communication is made between end nodes both which are under NATs, or when communication is made between IPv4 and IPv6 networks, or when communication is made with an end node which does not have NTMobile functions. RS can also be installed in dual stack network in a distributed manner, and the optimal RS is chosen at the time of creating a tunnel, by taking into account the relaying load and the length of the route.

It is assumed that there are trust relationships between NTM node X and DC_X, between DCs, and between DC and RS. Messages used by NTMobile are encrypted by an encryption key shared between nodes, and authenticated by Message Authentication Code (MAC). Furthermore, tunnel messages between NTM nodes or between NTM node and RS are also encrypted by a common key delivered by DC at the time of creating a tunnel, and authenticated by MAC.

In this Chapter, we describe procedures to make communication between a pair of NTM nodes based on virtual IPv4 application. In the following explanation, we indicate the initiator NTM node as Mobile Node (MN) and the correspondent NTM node as Correspondent Node (CN). In addition, we define real IPv4 address and virtual IPv4 address of NTM node X as $RIP4_X$ and $VIP4_X$ and real IPv6 address as $RIP6_X$, and the DC which manages this address information as DC_X. Path ID used for the communication between NM and CN is defined as PID_{MN-CN} the common key used for encryption and authentication is defined as CK_{MN-CN}. Path ID is an identifier to identify the communication between a pair of NTMs.

B. Registration

Each NTM node registers its address information in its own DC when the device is activated and when the NTM node changes its network. MN sends a Registration Request to DC_{MN}. The Registration Request contains information to be recorded in NTMv4 and NTMv6 records, such as FQDN, $RIP4_{MN}$, $RIP6_{MN}$, and so forth. DC_{MN}, upon receipt of the Registration Request, renew MN's resource record registered in Dynamic DNS server. And if MN exists behind a NAT router, DC_{MN} also registers IPv4 address of the NAT router in NTMv4 record, which is obtained from the source address of the Registration Request message.

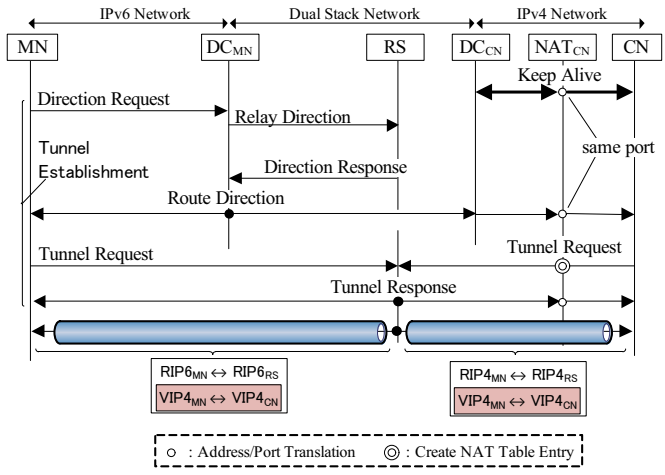


Fig. 2. Tunnel establishment procedure.

C. Tunnel Establishment

In the NTMobile architecture, NTM nodes proceed with tunnel creation process and create a tunnel path, when the initiator node performs DNS's name resolution process, or when either node detects a change in the real IP address during communication. For example, when MN detects DNS's A query on CN's FQDN, MN sends an additional query requesting for AAAA record and NTMv4/NTMv6 records. In this way, MN acquires CN's IP address information (i.e., $VIP4_{CN}$, $VIP6_{CN}$, and $RIP4_{CN}$ and so forth) in DNS's name resolution process. Then, MN temporarily stores the response from DNS server, and executes the tunnel creation process based on the acquired information.

Fig. 2 shows the tunnel creation process. This Figure demonstrates the situation in sequence in which MN, which is connected with IPv6 network, establishes a communication route with CN, which is connected with IPv4 private network.

1) Direction Request: MN sends a Direction Request message to $\mathrm{DC_{MN}}$, requesting for creation of a tunnel to communicate with CN. This message contains the address information described in NTMv4/NTMv6 records such as real IP addresses and virtual IP addresses of both MN and CN (namely, $RIP6_{MN}$, $VIP4_{MN}$, $RIP4_{CN}$, $VIP4_{CN}$, etc.). $\mathrm{DC_{MN}}$ determines the locational relationship between MN and CN from the above information, and chooses the most appropriate tunnel path for communication. In the case of Fig. 2, $\mathrm{DC_{MN}}$ chooses the path through RS because MN and CN exist in different networks; namely, one in IPv6 and the other in IPv4 network.

2) Relay Direction: $\mathrm{DC_{MN}}$ sends a Relay Direction message to RS, requesting it to relay packets between MN and CN. This message contains address information of MN and CN as well as PID_{MN-CN}. RS, upon receipt of the message, sends a Direction Response to $\mathrm{DC_{MN}}$, and waits for Tunnel Requests from both MN and CN.

3) Route Direction: $\mathrm{DC_{MN}}$, upon receipt of the Relay Response from RS, sends Route Directions to both MN and CN instructing them to create a UDP tunnel towards RS. The Route Direction message contains PID_{MN-CN}, address information of the communication partner, real IP address of RS, and CK_{MN-CN}, which is a common key used for encryption for tunnel communication. CN receives the Route Direction via $\mathrm{DC_{MN}}$. As CN periodically sends Keep Alive messages to $\mathrm{DC_{CN}}$, CN is ready to receive packets from $\mathrm{DC_{CN}}$ at any time even if CN is in a private IPv4 network.

4) Tunnel Request/Response: MN and CN send Tunnel Request messages to RS, following the instructions of $\mathrm{DC_{MN}}$. As the result that a Tunnel Request message is sent from CN behind NAT to RS, mapping information for CN and RS to communicate with each other is generated at $\mathrm{NAT_{CN}}$, and a tunnel for IPv4 between CN and RS over NAT can be created.

Through the above process, a route for communication between MN and CN is established. In the case that the process was executed with A record query as a trigger, MN rewrites the IP address contained in the Response kept by DNS Server to $VIP4_{CN}$ and hands it over to DNS resolver. Then, MN's application recognizes $VIP4_{CN}$ as CN's IPv4 address and communication based on virtual IPv4 address is initiated between applications of MN and CN.

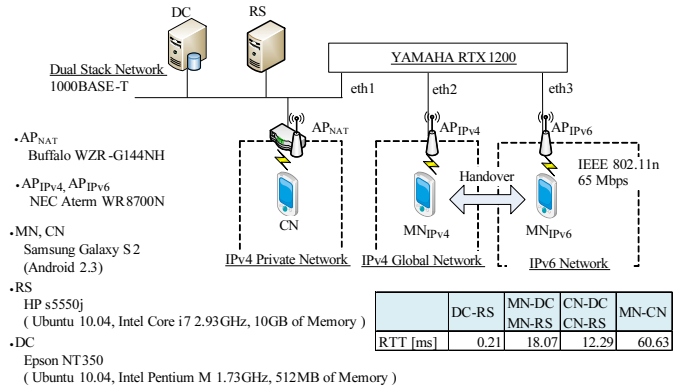
D. Tunnel Communication

In Fig. 2, since applications start communication based on virtual IPv4 addresses, $VIP4_{MN}$ is described at the source address of the packet created by application and $VIP4_{CN}$ is described at the destination address. MN, after encapsulating a packet addressed to $VIP4_{CN}$ with real IPv6 address ($RIP6_{RS}$), sends it to RS. RS, after decapsulating the received packet, forwards it to CN after encapsulating it with real IPv4 address ($RIP4_{CN}$). CN, upon receipt of the encapsulated packet, performs decapsulation and decryption processes, and hands over the extracted packet to the upper application.

Through the above process, MN, which is connected to IPv6 network, and CN, which is connected to IPv4 network can mutually communicate. Because applications communicate based on virtual IPv4 address, they do not receive any influence of the different networks to which NTM nodes are connected. Even if NAT exists on the communication path, applications can make communication without being affected by the NAT because address and port translation by NAT is done against the outer IP/UDP headers of the encapsulated packet.

E. Handover

When MN changes its access network during communication due to movement or switching of wireless interfaces, a tunnel route towards CN is reconstructed by an appropriate direction from DC, based on the locational relationship between MN and CN, in the same way as described in Sec. II-C. On that occasion, tunnel reconstruction alone is executed, without going through the process of DNS name resolution, as CN's own address information is already in hand. Because the applications of MN and CN are making communication based on virtual IP addresses, they are not affected even if real IP addresses change, and you can continue communication.



	DC-RS	MN-DC MN-RS	CN-DC CN-RS	MN-CN
RTT [ms]	0.21	18.07	12.29	60.63

Fig. 3. Evaluation environment.

Meanwhile, the registration process (refer to Sec. II-B) is conducted in parallel with the switching of the tunnel route, and the address information recorded in DC_{MN} is renewed. By always keeping up-to-date address information in DC in that way, reachability to NTM nodes is ensured.

III. Implementation and Performance Evaluation

A. Implementation

We have implemented our NTMobile functions in Android smartphones (Samsung Galaxy S II) and installed DC and RS in Linux PC (Ubuntu 10.04), and conducted operation verification and performance evaluation of our system in IPv4 and IPv6 coexisting networks.

1) NTM Node: NTM node operates with implementation of a kernel module to perform encapsulation and encryption, a user daemon program (NTM daemon) to perform negotiations, and virtual interfaces. At the kernel module, packets are hooked by Netfilter, and encapsulated and encrypted in kernel space. In our NTMobile system, deterioration of throughput is controlled by completing encapsulation process in the kernel space. Because we need such functions as Netfilter and Netlink to implement kernel module of NTMobile, we rebuilt the kernel by effectuating these functions. As regards NTM daemon, we implemented it as a native program, though typical Android application runs on Dalvik virtual machine.

2) Direction Coordinator: DC operates with implementation of a user daemon program to conduct negotiations, and Dynamic DNS program. We adopted BIND9 for the Dynamic DNS program and expanded its capacity so that both NTMv4 and NTMv6 records can be registered.

3) Relay Server: RS operates with implementation of a kernel module to relay tunnel communications and a user daemon program to conduct negotiations. RS executes forwarding process by hooking received packets by Netfilter and handing them over to the correspondent node by the function of kernel module.

B. Experiment environment

We verified the operation of a handover process between IP4 and IPv6 networks by using Android smartphones implemented with our NTMobile function and conducted evaluation of its performance.

Fig. 3 shows the network configuration, specifications of each device and the average round trip times (RTTs) between devices.

The networks under eth1-eth3 are divided into three different networks; namely, Dual Stack network, IPv4 network, and IPv6 network, using in virtual LAN (VLAN) functions of YAMAHA RTX1200. A general broadband router is used for each access point (AP), and the network under the AP_{NAT} is constructed as IPv4 private network. Both AP_{IPv4} and AP_{IPv6} are operated as general access points, by disabling their function of routers. MN and CN are connected to each AP by IEEE 802.11n, and we used WPA/WPA2-PSK (AES) for encryption and authentication functions. As the encryption algorithm, we set AES-CFB to be used at the time of negotiations and AES-CBC to be used at the time of tunnel communications. Both authentication algorithms are based on HMAC-MD5 with a common key of 128 bit of length.

TABLE I
Results of suspended time by handover.

	Time [sec] Case 1 (to IPv6)	Case 2 (to IPv4)
L2 Handover	0.58	0.47
Acquiring IP address	1.71	0.67
Tunnel establishment	0.17	0.12
TCP retransmission	1.17	0.66
Total time	3.63	1.92

C. Performance evaluation

On the occasion of the handover of a terminal, there occurs a communication interruption period caused by L2 handover and acquiring on IP address. Then, packet loss could occur and give influence on the communications between applications.

Therefore, we measured the handover latency caused by the switching of access points during TCP communication between MN with CN. For the TCP communication between MN and CN, iperf is used, and we measured the handover latency based on the change in the sequence numbers of TCP packets. CN was always connected to AP_{NAT} and we switched the access points of MN manually in the following manner

Case 1: Switching the access points during communication via AP_{IPv4} to AP_{IPv6}.

Case 2: Switching the access points during communication via AP_{IPv6} to AP_{IPv4}.

Tab. I shows the measurement results of the handover latency. The values are based on the average of measurement of 10 times. In case 1, it took 1.71 seconds from the time of switching to AP_{IPv6} till the time of acquiring on IP address. Fig. 4 shows the sequence and details of the process time from the time of switching to AP_{IPv6} till the time when the tunnel creation process is initiated. MN, after generation IPv6 address, sends Neighbor Solicitation (NS), and about one second later, the tunnel creation process started. This delay

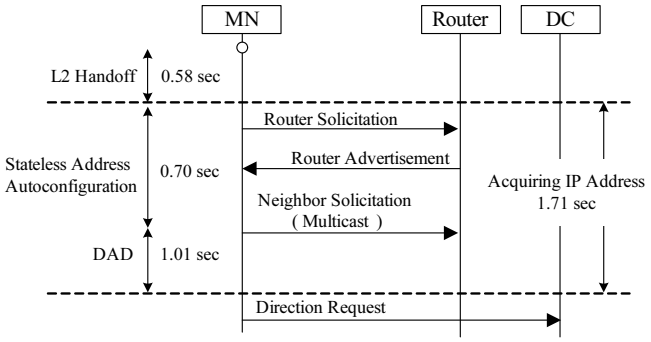


Fig. 4. Handover flow in IPv6.

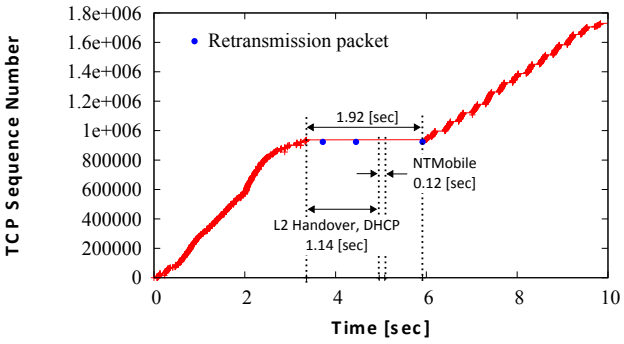


Fig. 5. Changes of TCP sequence number by handover.

is understood to be caused by the DAD (Duplicated Address Detection) [11]. The IPv6 address generated by MN cannot be used temporarily until the DAD process is completed. After the process was completed, the tunnel creation process started, and the time used for that process turned out to be about 5% of the total handover latency.

In case 2, the communication was interrupted for 1.92 seconds after switching to AP_{IPv4}. Fig. 5 shows the changes of TCP sequence numbers. The application packet was retransmitted twice by TCP during the communication interruption period. After switching the access point of MN from AP_{IPv6} to AP_{IPv4}, L2 handover and IP address acquisition process by DHCP was conducted, and it took 1.14 seconds for this process. And, tunnel creation process by NTMobile was performed after the acquisition of IP address, and it took 0.12 second.

After the completion of tunnel reconstruction process, it took 0.66 second for the application to resume its communication. This delay seemed to have been caused by the TCP retransmission control. According to Reference Material [12], it is mentioned that when client acquires IP address by DHCP, the client should wait for 1 to 10 seconds after the connection to the network, before starting processing. In the case of Galaxy S2 which we used this time, the processing by DHCP started 0.3 second after the connection to AP As this waiting time depends on DHCP client, we can assume that it will take up to several times longer to acquire IP address.

Based on the results of our measurement, we found that while the processing time by NTMobile at the time of handover is quite short, it takes 1.14 to 2.29 seconds for L2 handover and the process of acquiring IP address, and such time occupies more than half of the entire communication interruption time. Thus, we found it necessary to realize seamless handover, by utilizing, for instance, wireless authentication high-speed technologies such as IEEE 802.11ai [13] or a combination of plural wireless interfaces such as Wi-Fi and WiMAX.

IV. Conclusions

In this paper, we implemented our NTMobile with functions of mutual connectivity and mobility in IPv4 and IPv6 networks, in Android smartphones, and performed its operation verification and performance evaluation. Based on the operability verification, we confirmed that our system can ensure connectivity and mobility without being affected by the existence of NAT and the difference between IPv4 networks and IPv6 networks. Because communication interruption occurs due to L2 handover and the acquiring IP address at the time of handover, it is necessary, to examine methods to realize a seamless handover in which no communication interruption occurs. As the next step, we plan to evaluate communication performance and scalability of our system in the actual environment such as 3G and WiMAX.

References

[1] J. Rosenberg, R. Mahy, P. Matthews and D. Wing: Session Traversal Utilities for NAT (STUN), *RFC 5389,IETF* (2008).
[2] R. Mahy, P. Matthews and J. Rosenberg: Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN), *RFC 5766,IETF* (2010).
[3] J. Rosenberg: Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols, *RFC 5245,IETF* (2010).
[4] D. Le, X. Fu and D. Hogrefe: A Review of Mobility Support Paradigms for the Internet, *IEEE Communications Surveys*, Vol. 8, No. 1, pp. 38–51 (2006).
[5] D. Johnson, C. Perkins and J. Arkko: Mobility Support in IPv6, *RFC 3775,IETF* (2004).
[6] R. Inayat, R. Aibara, K. Nishimura, T. Fujita and K. Maeda: An End-to-End Network Architecture for Supporting Mobility in Wide Area Wireless Networks, *IEICE Transactions on Communications*, Vol. E87-B, No. 6, pp. 1584–1593 (2004).
[7] M. Takeuchi, H. Suzuki and A. Watanabe: A proposal of Mobile PPC that realizes end-to-end mobility and its implementations, *Transactions of Information Processing Society of Japan,*, Vol. 47, No. 12, pp. 3244–3257 (2006).
[8] H. Soliman: Mobile IPv6 Support for Dual Stack Hosts and Routers, *RFC 5555,IETF* (2009).
[9] R. Wakikawa, R. Kuntz, Z. Zhu and L. Zhang: Global HA to HA Protocol Specification, *draft-wakikawa-mext-global-haha-spec-02,IETF* (2011).
[10] H. Suzuki , K. Kamienoo, H. Nodo, T. Nishio, K. Naito and A. Watanab: A Study on NTMobile that Achieves Both Connectivity and Mobility in IPv4 and IPv6 Networks, *DCIOMO2012*, Vol. 2012, No. 1, pp. 2391–2401 (2012).
[11] S. Thomson and T. Narten: IPv6 Stateless Address Autoconfiguration, *RFC2462,IETF* (1998).
[12] R. Droms: Dynamic Host Configuration Protocol, *RFC2131 ,IETF* (1997).
[13] IEEE 802.ai (Fast Initial Link Set-up), http://www.ieee802.org/11/Reports/tgai_update.htm.