Design of an Application Based IP Mobility Scheme on Linux Systems

Kohei Tanaka†, Fumihito Sugihara‡, Katsuhiro Naito†, Hidekazu Suzuki*, and Akira Watanabe*

†Faculty of Information Science, Aichi Institute of Technology, Toyota, Aichi 470-0392, Japan
{kohei, naito}@pluslab.org
‡Department of Electrical and Electronic Engineering, Mie University, Tsu, Mie 514-8507, Japan
*Graduate School of Science and Technology, Meijo University, Nagoya, Aichi 468-8502, Japan

Abstract - Internet of Thing (IoT) systems have been attracting attention as one of the solutions for new services in the Internet. They usually employ a client-server model due to a difficulty of end-to-end communication in practical networks. However, scalability will be a major issue in IoT systems because billions of IoT devices will be installed around the world in recent years. The authors have been proposed a new IP mobility mechanism called NTMobile (Network Traversal with Mobility) to realize end-to-end communication in IoT systems because end-to-end communication can improve system scalability by reducing traffic through servers. Conventional implementation employed a kernel module mechanism for NetFilter because the kernel module implementation is the best way of realizing high throughput performance. On the contrary, the kernel module should be maintained according to changes in NetFilter specifications. This paper designs an application based IP mobility scheme on Linux systems, where the developed IP mobility library can realize the IP mobility function in an application layer on Linux systems. As a result, developers can realize an end-to-end communication model by employing the enhanced IP mobility library. The proposed design ensures compatibility between the developed library and the conventional NTMobile. Therefore, developers can select the implementation scheme according to the required performance.

Keywords: IP Mobility, Accessibility, Application library, Linux, NTMobile.

1 INTRODUCTION

Recent microcomputer boards implement some network interfaces to cooperate with another microcomputer boards according to the increase of demand for IoT (Internet of Things) systems [1], [2]. Almost all microcomputer boards for IoT systems are usually installed in a private network due to a security policy and limitation of assignable global IP addresses. Typical private networks prohibit access from the global Internet to a node in their private networks. Therefore, interconnectivity between IoT devices is a big issue even if they should communicate with each other to realize their specific service. Additionally, operating systems select an interface to access to the Internet according to network condition of each interface and access policies when an IoT device is a mobile node [3]. Therefore, IoT systems also require a seamless connectivity scheme because their IP address typically changes when they switch the interface even if they should communicate with each other to realize their specific service.

IP mobility protocols are a solution to the requirement for inter-connectivity and seamless connectivity in IoT systems because they can realize continuous communication when an IP address for an interface changes due to switching of access networks [4]–[9]. They are classified into three types: IP mobility schemes for IPv4, IP mobility schemes for IPv6, and IP mobility schemes for both IPv4 and IPv6. Mainstream of IP mobility schemes is for IPv6 because IPv6 is suitable for mobility. On the contrary, the number of implementations for IPv4 is quite few though some mechanisms have been proposed [10]–[12]. DSMIPv6 (Dual Stack Mobile IPv6) [13] supports both IPv4 and IPv6 networks. However, it still does not support inter-connectivity between IPv4 and IPv6.

IPv4 continues to be the mainstream protocol under the present circumstances of the Internet. Additionally, some Internet service providers start the service with large scale network address translator (LSN) in order to meet the shortage of IPv4 global addresses [14]–[16]. As a result, IP mobility in a private network behind a NAT becomes an important issue. STUN (Simple Traversal of UDP through NATs) [17], TCP hole punching [18], TURN (Traversal Using Relay NAT) [19], and ICE (Interactive Connectivity Establishment) [20], [21] have been proposed for a NAT traversal. These conventional work requires implementation of the special mechanisms in an application. Additionally, some IoT devices may use IPv6 addresses because IPv6 networks have enough number of IP addresses and some devices are suitable for an IPv6 protocol stack. Therefore, inter-connectivity between IPv4 and IPv6 networks is also an essential function to realize inter-operation between IoT devices.

The authors have developed a new IP mobility technology called NTMobile (Network Traversal with Mobility) [22]–[24]. The features of NTMobile are an IP mobility and an accessibility in both IPv4 and IPv6 networks. Therefore, each client of NTMobile can communicate with each other even when they use a different IP protocol version because they can communicate with virtual IP addresses that are independent addresses from physical IP addresses. NTMobile systems have some servers: account server (AS), direction coordinator (DC), notification server (NS), relay server (RS), and cache server (CS). DC and RS serve an IP mobility and an accessibility functions for each client, AS serves an authentication service, and CS serves an offline data-exchange service. Our implementation assumes Linux systems. The implementation design is classified into a signaling daemon and a packet manipulation module for the Linux kernel. As a result, we have confirmed that our implementation scheme

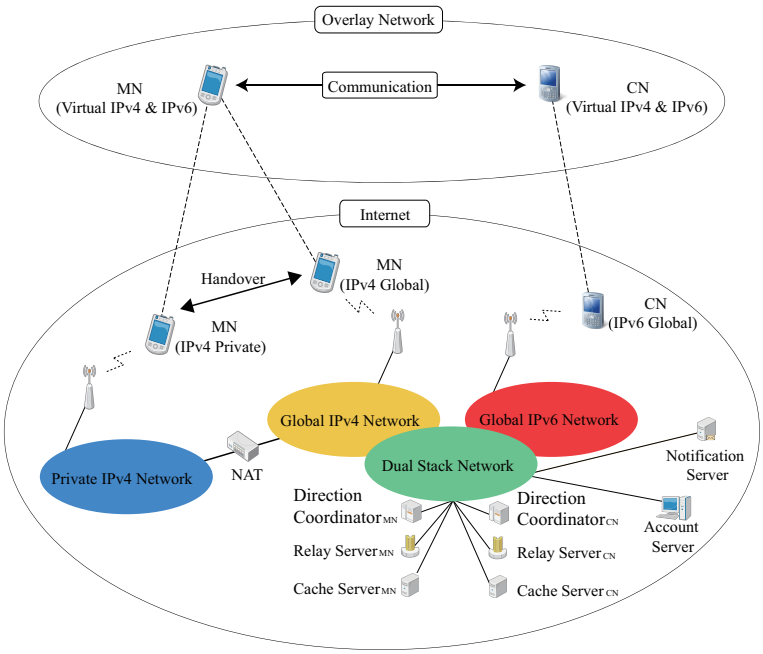


Figure 1: System model of NTMobile

can realize high throughput performance. On the contrary, we find a maintenance issue for the packet manipulation module because netfilter module in Linux systems sometimes changes their specification according to the upgrade of a Linux kernel.

This paper proposes a new design of an application based IP mobility for NTMobile nodes. The original design can provide NTMobile functions by an application library instead of the special kernel module. Therefore, application programmers can obtain IP mobility and accessibility functions by using of proposed application library.

2 NTMOBILE

NTMobile can realize IP mobility and accessibility for IPv4 and IPv6 networks. Figure 1 shows the overview of NTMobile system model. The NTMobile system consists of an account server (AS), some direction coordinators (DCs) with some relay servers (RSs), cache servers (CSs), Notification Servers (NSs) and NTMobile nodes. AS serves authentication service to all DCs, and each DC controls their RSs, CSs, NSs and NTMobile nodes. NTMobile node has IP mobility and accessibility functions by communicating with AS and its DC. Each DC has a virtual IP address pool for its NTMobile nodes, and assigns an address to each NTMobile node in a registration process. Each NTMobile node constructs a UDP tunnel between NTMobile nodes according to a signaling direction from its DC, and communicate with each other by using their virtual IP addresses. As a result, each NTMobile node can communicate continuously even if a real IP address at interfaces is changed because the virtual IP addresses are independent from physical IP addresses. The details of the system components are as followings.

2.1 Account Server (AS)

AS is an individual server that manages authentication information. Therefore, AS can distribute node information of each NTMobile node to initialize a setting for NTMobile nodes. Additionally, it bears responsibility for authentication by replying login response message when a NTMobile node makes inquiries about its authentication to AS. AS also distributes a shared key for encryption between the NTMobile node and DC.

2.2 Direction Coordinator (DC)

DC manages location information of each NTMobile node and indicates signaling processes for tunnel construction between NTMobile nodes. Each DC also owns the DNS (Domain Name Server) function. Therefore, DC can easily find another DC by searching a NS (Name Server) record in DNS. In addition, DC manages virtual IP addresses for NTMobile nodes to prevent a duplication of an address assignment. DC assigns them to each NTMobile node in a registration process. In the tunnel construction process, it indicates the tunnel construction processes to both NTMobile nodes.

2.3 Notification Server (NS)

NTMobile typically uses a UDP protocol to communicate between a NTMobile node and its DC. Therefore, the NTMobile node should send keep-alive messages to its DC when it uses a private address under a NAT router. NS can provide a notification service with a TCP connection between NS and the NTMobile node. Therefore, the employing NS can reduce the amount of messages for keep-alive. Additionally, NS also supports APNS (Apple Push Notification Service) for iOS and GCM (Google Cloud Messaging) for Android OS.

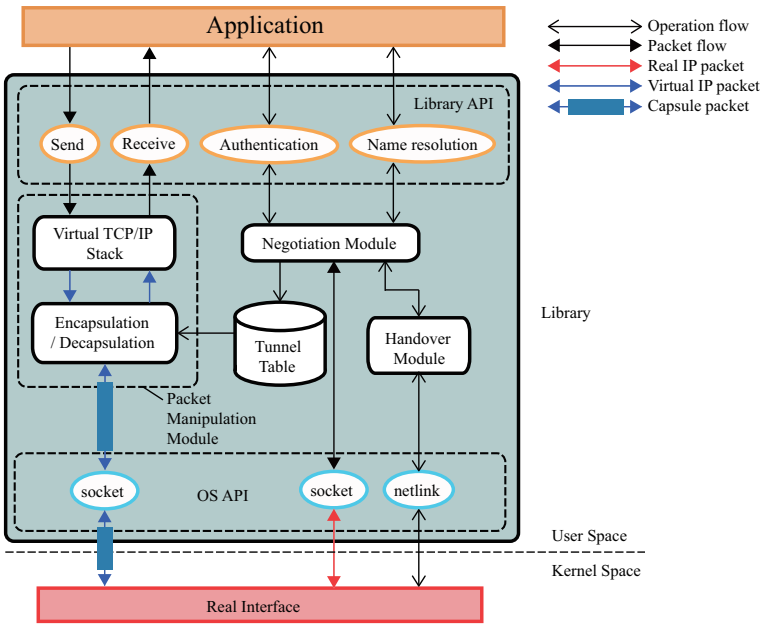


Figure 2: Overview of application library design

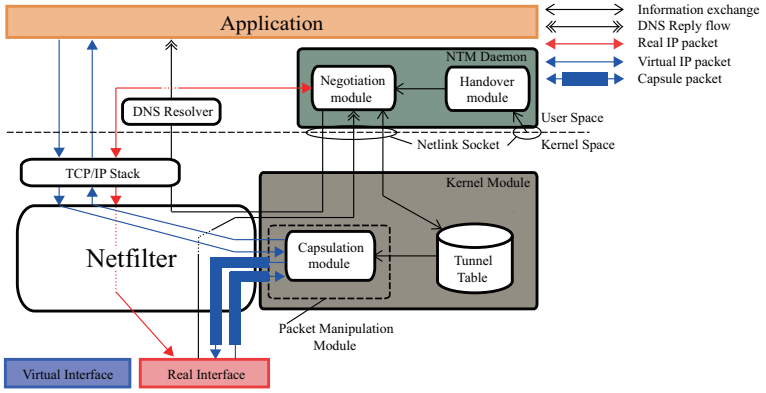


Figure 3: Overview of kernel module design

2.4 Relay Server (RS)

The relay server function is to relay tunnels between NTMobile nodes when both NTMobile nodes exist under different NAT routers or exist under different version networks such as IPv4 and IPv6 networks. DC manages some relay servers to realize load balancing and to avoid a single point of failure. It also chooses a relay server to activate the relay function for dedicated NTMobile nodes.

2.5 Cache Server (CS)

The function of cache servers is storing data temporary for non-realtime data exchange between NTMobile nodes. A correspondent NTMobile node can download the stored data since CS stores the data for a certain period.

2.6 NTMobile Node

Functions of NTMobile nodes are to realize IP mobility and accessibility in IPv4 and IPv6 networks. They obtain their own information from AS in the initialization phase when they connect to the NTMobile network at first. Then, they inform their own network information to their DC because DC should manage network information of each NTMobile node to realize IP mobility and accessibility. They also update the own network information when they switch access networks.

3 APPLICATION BASED IP MOBILITY

This paper proposes a new design of an application based IP mobility and accessibility for NTMobile nodes. The proposed design provides APIs (Application Programming Interface) for NTMobile functions: an initialization, an authentication, a tunnel construction request, and network sockets for NTMobile communication. It also provides a virtual IP address for the network sockets. Our API provides BSD compatible network sockets for application developers. Therefore, application developers can easily apply their network application with virtual IP addresses for NTMobile network.

3.1 System Model

Figure 2 shows the overview design of the proposed application based IP mobility and accessibility system for NTMobile nodes. Figure 3 shows the overview design of conventional kernel module. The conventional kernel module performs IP capsulation/decapsulation and encryption/decryption processes in Linux kernel. The design also provides a virtual network interface for applications. The conventional implementation consists of a shared library for common NTMobile functions, daemon programs for a NTMobile node, an account server, a direction coordinator, a relay server, and a notification server, and a kernel module for a NTMobile node. The source size of the conventional shared library, daemon program for a NTMobile node, and the kernel module is 23K lines, 10K lines and 12K lines respectively. Therefore the overhead of the NTMobile program is acceptable size for a small resource hardware.

On the contrary, the proposed application library is inserted between an original network socket in Linux OS and a developer's application. In the proposed design, we also employ the shared library to implement the common NTMobile functions. Additionally, we develop a new application library including functions for a transport layer, a daemon program of a NTMobile node and a kernel module. We employ lwIP [25] as a transport layer. The source size of lwIP and application library is 55K lines and 10K lines respectively. As a result, we think that our proposed design is still acceptable for a small resource hardware such as M2M and IoT devices. The application can request to the proposed application library for initialization of NTMobile node functions, an authentication for NTMobile network, and a tunnel construction request to a correspondent node. The actual data communication is performed with special network sockets for NTMobile functions. The application library can perform IP capsulation/decapsulation and encryption/decryption processes between the special network sockets and the original Linux network sockets.

3.2 Application Programming Interface

The proposed design provides APIs (Application Programming Interface) for NTMobile functions: an initialization, an authentication, a tunnel construction request, and network sockets for NTMobile communication. The following is the APIs for an application.

- Authentication API

  The proposed library assumes a specific authentication scheme with a user account and a password. Therefore, an application should request authentication with the user account information through the authentication API. The authentication API requests the negotiation module for authentication processes. The negotiation module should obtain the NTMobile node information when the application library has no configuration. Then, it also requests AS to authenticate with the account information. Finally, it registers the network information to the DC.

- Name Resolution API

  NTMobile network employs FQDN (Fully Qualified Domain Name) as an identifier for a NTMobile node. Therefore, an application should request a tunnel construction with a FQDN of a correspondent node through the name resolution API. The name resolution API requests the negotiation module to construct a UDP tunnel to the correspondent node. The negotiation module also requests to NTMobile node's DC to construct a UDP tunnel. Finally, it also reports to the application about the completion of the tunnel construction process.

- NTMobile Network Socket API

  NTMobile network socket API is connected to an original Linux socket when the tunnel construction process is completed. The application starts communication over the UDP tunnel when they receive this information about completion of the tunnel construction process from the negotiation module.

3.3 Library Modules

The proposed library consists of three main modules for negotiation, handover, and packet manipulation. The following is the functions of each module.

- Negotiation Module

  The negotiation module serves as a signaling function for NTMobile communication. Therefore, application developers do not care about the detail process of NTMobile communication because the negotiation module can handle IP mobility and accessibility functions in NTMobile. The negotiation module registers the specific information for packet manipulation into the tunnel table.

- Handover Module

  The handover module should check the network interface status to detect a change of the access network. It also handles reconstruction of the UDP tunnel to the correspondent node when IP address changes due to an access network change.

- Packet Manipulation Module

  The packet manipulation module consists of two main functions: a virtual TCP/IP stack and a capsulation function. The virtual TCP/IP stack provides a transport layer function such as TCP and UDP to an application. The capsulation function handles the encapsulation and decapsulation for a UDP tunnel according to the information in the tunnel table.

3.4 Signaling Process

Figures 4, 5, 6 and 7 show the signaling processes in NTMobile. These figures are assumed that NTMobile node MN in a global IPv4 network requests to communication with NTMobile node CN in a private IPv4 network. Then, NTMobile node MN changes the network from the global IPv4 networks

to a private IPv4 network. Direction coordinators $\mathrm{DC_{MN}}$ and $\mathrm{DC_{CN}}$ are the management servers for NTMobile node MN and CN respectively. The detail signaling process is as follows.

- Authentication

  Figure 4 shows the signaling processes for login in NTMobile. MN should request authentication to join the NTMobile network. First, MN connects AS to obtain the configuration for the authentication of MN. Then, MN requests authentication to AS with MN's account information and the obtained information from AS. After authentication, MN registers the information about the access network to $\mathrm{DC_{MN}}$ for IP mobility. Finally, MN registers the information of MN to $\mathrm{NS_{MN}}$ to receive the notification of the tunnel request from CN.

  1. The application calls the authentication API with the account information.
  2. The negotiation module on MN connects AS to obtain the configuration of MN when the configuration of NTMobile has not completed.
  3. AS replies the configuration information when the account information is valid.
  4. The negotiation module registers the obtained configuration information.
  5. The negotiation module requests authentication to AS by transmitting the login request message.
  6. AS generates a shared key for encryption between MN and $\mathrm{DC_{MN}}$ and distributes to $\mathrm{DC_{MN}}$ the shared key.
  7. $\mathrm{DC_{MN}}$ replies the acknowledgement message to AS.
  8. AS informs the shared key for encryption between MN and $\mathrm{DC_{MN}}$, and a FQDN of $\mathrm{DC_{MN}}$ by replying the login response message.
  9. The negotiation module registers the information about the access network to $\mathrm{DC_{MN}}$ by transmitting the registration request message.
  10. $\mathrm{DC_{MN}}$ requests $\mathrm{NS_{MN}}$ to provide a notification service using TCP connection for MN.
  11. $\mathrm{NS_{MN}}$ replies the acknowledgement message to $\mathrm{DC_{MN}}$ when it prepares the notification service for MN.
  12. $\mathrm{DC_{MN}}$ updates the network information of MN, and replies the registration response message.
  13. The negotiation module transmits a keep alive message to keep the NAT table on the route.
  14. MN registers the information of MN to $\mathrm{NS_{MN}}$.
  15. $\mathrm{NS_{MN}}$ replies the acknowledgement message to MN.
  16. The negotiation module transmits a keep alive message to keep the NAT table on the route.
  17. The negotiation module on MN returns the result of the authentication to the application.

- Tunnel Construction

  Figure 5 shows the signaling processes of tunnel construction in NTMobile. MN requests the tunnel construction to $\mathrm{DC_{MN}}$ to communicate to CN by transmitting the direction request message containing the FQDN of CN. Then, $\mathrm{DC_{MN}}$ finds $\mathrm{DC_{CN}}$ with the NS record of CN's FQDN. $\mathrm{DC_{MN}}$ accesses $\mathrm{DC_{CN}}$ to obtain the information about the access network of CN. $\mathrm{DC_{MN}}$ selects a tunnel route from the obtained information of CN. In this situation, $\mathrm{DC_{MN}}$ selects the direct communication route. Each DC indicates a tunnel construction processes to each NTMobile nodes. Finally, CN under the NAT router constructs the tunnel to MN.

  1. The application calls the name resolution API to construct a UDP tunnel to NTMobile CN.
  2. The negotiation module requests the tunnel construction to $\mathrm{DC_{MN}}$ by transmitting the direction request message. The direction request message contains the FQDN of CN.
  3. $\mathrm{DC_{MN}}$ makes inquiries about NS record for the FQDN of CN since $\mathrm{DC_{CN}}$ is the domain name server and manages the domain for the FQDN of CN.
  4. $\mathrm{DC_{MN}}$ generates a shared key for encryption between $\mathrm{DC_{MN}}$ and $\mathrm{DC_{CN}}$ and distributes to $\mathrm{DC_{CN}}$ the shared key.
  5. $\mathrm{DC_{CN}}$ replies the acknowledgement message to $\mathrm{DC_{MN}}$.
  6. $\mathrm{DC_{MN}}$ requests the information for CN to $\mathrm{DC_{CN}}$ by transmitting the NTM information request message.
  7. $\mathrm{DC_{CN}}$ replies the information about CN by replying the NTM information response message.
  8. $\mathrm{DC_{MN}}$ requests the tunnel construction to CN to $\mathrm{DC_{CN}}$ by transmitting the route direction message.
  9. $\mathrm{DC_{CN}}$ requests the notification to CN to $\mathrm{NS_{CN}}$.
  10. $\mathrm{NS_{CN}}$ sends the notification to CN using a type compatible with CN's device.
  11. CN replies the acknowledgement message to $\mathrm{DC_{CN}}$.
  12. $\mathrm{DC_{CN}}$ forwards the route direction message to CN.
  13. CN replies the acknowledgement message to $\mathrm{DC_{CN}}$.
  14. $\mathrm{DC_{CN}}$ also replies the acknowledgement message to $\mathrm{DC_{MN}}$.
  15. $\mathrm{DC_{MN}}$ indicates the tunnel construction to CN to MN by transmitting the route direction message.
  16. The negotiation module on MN replies the acknowledgement message to $\mathrm{DC_{MN}}$.

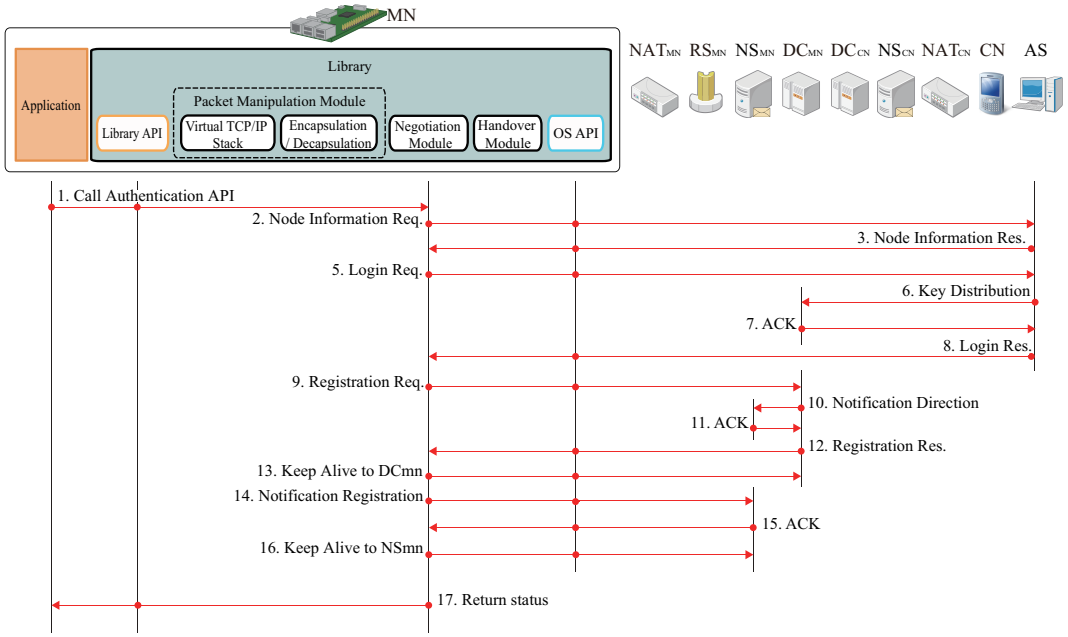


Figure 4: Signaling processes of the authentication

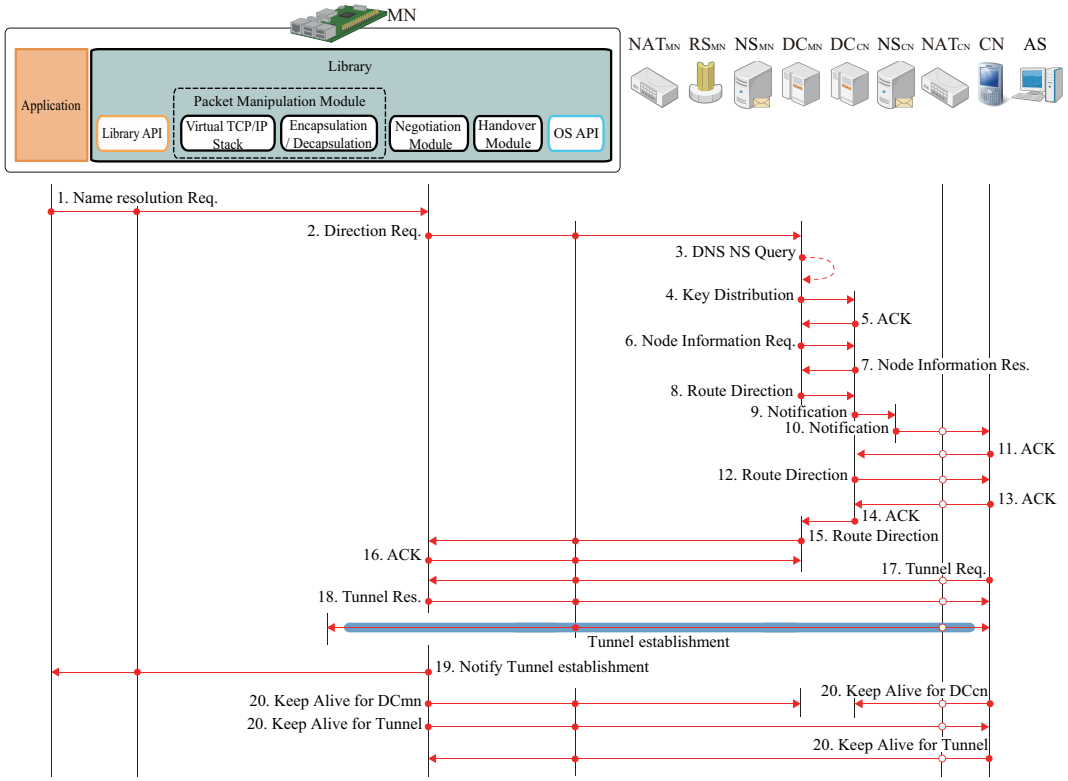


Figure 5: Signaling processes of the tunnel construction

17. CN transmits the tunnel request message to MN according to the direction from DC_{MN} because MN has a global IP address in this case.
18. MN replies the tunnel response message to CN to complete the tunnel construction process.
19. The negotiation module notifies the completion of the tunnel construction process to the application.
20. The negotiation module transmits a keep alive message to keep the NAT table on the route.
21. The application starts the communication through the constructed UDP tunnel. Then, it transmits packets by using the special NTMobile socket.

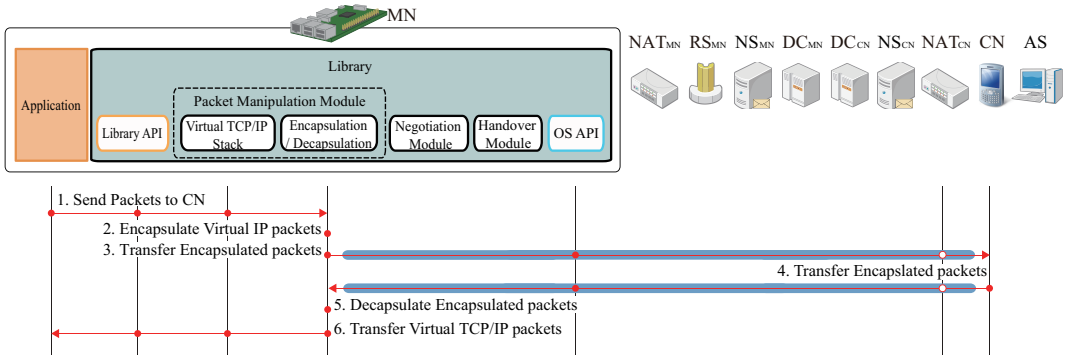


Figure 6: Signaling processes of the data communication

- Data Communication

  Figure 6 shows the signaling processes of data communication in NTMobile. The data communication between MN and CN is performed as encapsulated packet through the constructed tunnel. Therefore, the capsulation module performs the capsulation and decapsulation processes to conceal the change of the real IP address. The application uses the special socket API to perform the data communication.

  1. The virtual TCP/IP stack in the packet manipulation module serves the transport layer function to the application.
  2. The encapsulation and decapsulation module performs the encapsulation of virtual IP packets to CN.
  3. The encapsulated packet is transmitted to CN.
  4. The encapsulated packet is transmitted from CN.
  5. The encapsulation and decapsulation module performs the decapsulation the encapsulated packet from CN.
  6. The virtual TCP/IP stack in the packet manipulation module sends the decapsulated packet to the application.

- Change The MN Network

  Figure 7 shows the signaling processes when MN changes the access network. The handover module of NTMobile always observes the IP address of own network interface to detect the change of an access network. MN registers the information about the access network to DC_{MN} when it detects the change of the access network. Then, the same tunnel construction procedures will be performed to reconstruct the tunnel. In this situation, DC_{MN} selects the relay communication through RS. Each DC indicates the tunnel construction process to each NTMobile nodes. Finally, MN constructs the tunnel to CN through RS.

  1. Linux kernel can notify the change of network configuration through netlink socket.
  2. The handover module notifies the negotiation module that the access network of the device has changed.
  3. The negotiation module registers the new information about the access network to DC_{MN} by transmitting the registration request message.
  4. DC_{MN} requests NS_{MN} to provide a notification service using TCP connection for MN.
  5. NS_{MN} replies the acknowledgement message to MN.
  6. DC_{MN} updates the network information of MN, and replies the registration response message.
  7. The negotiation module transmits a keep alive message to keep the NAT table on the route.
  8. MN registers the information of MN to NS_{MN}.
  9. NS_{MN} replies the acknowledgement message to MN.
  10. The negotiation module transmits a keep alive message to keep the NAT table on the route.
  11. The negotiation module requests the tunnel construction to DC_{MN} by transmitting the direction request message. The direction request message contains the FQDN of CN.
  12. DC_{MN} makes inquiries about NS record for the FQDN of CN.
  13. DC_{MN} generates a shared key for encryption between DC_{MN} and DC_{CN} and distributes to DC_{CN} the shared key.
  14. DC_{CN} replies the acknowledgement message to DC_{MN}.
  15. DC_{MN} requests the information for CN to DC_{CN} by transmitting the NTM information request message.
  16. DC_{CN} replies the information about CN by replying the NTM information response message.
  17. DC_{MN} requests RS to relay the communication between both NTMobile nodes.
  18. RS prepares the tunnel forwarding between both NTMobile nodes, and replies the acknowledgement message to MN.

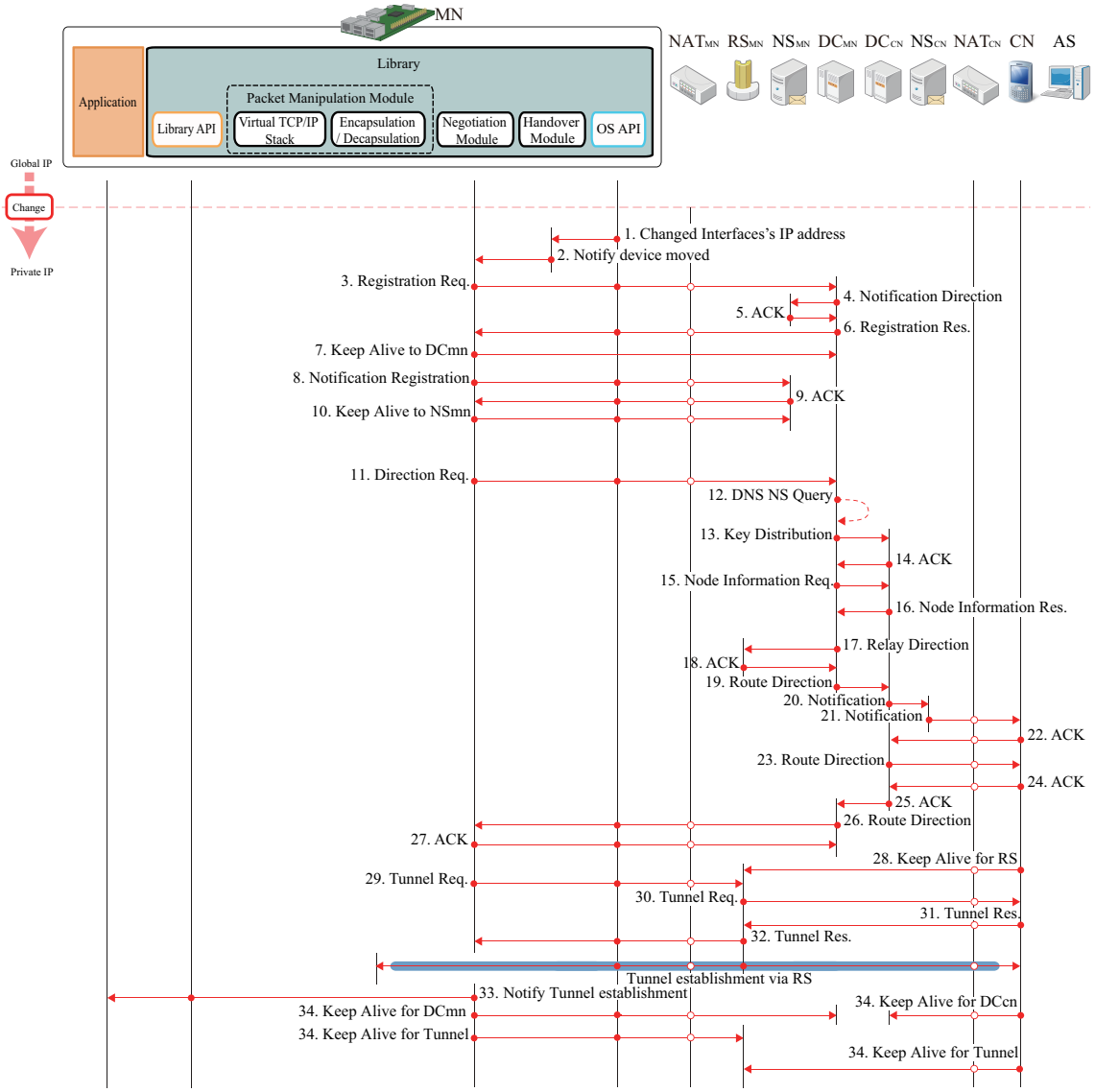


Figure 7: Signaling processes of the change the MN network

19. DC_{MN} requests the tunnel construction toward CN to DC_{CN} by transmitting the route direction message.
20. DC_{CN} requests NS_{CN} to transmit a notification to CN.
21. NS_{CN} sends the notification to CN using a type compatible with CN's device in response to a request from DC_{CN}.
22. CN replies the acknowledgement message to DC_{CN}.
23. DC_{CN} forwards the route direction message to CN.
24. CN replies the acknowledgement message to DC_{CN}.
25. DC_{CN} also replies the acknowledgement message to DC_{MN}.
26. DC_{MN} indicates the tunnel construction to CN to MN by transmitting the route direction message.
27. The negotiation module on MN replies the acknowledgement message to DC_{MN}.
28. CN transmits a keep alive message to keep the NAT table on the route.
29. MN transmits the tunnel request message to RS according to the direction from DC_{MN} because both NTMobile nodes do not have a global IP address in this case.
30. RS transmits the tunnel request message to CN according to the direction from DC_{MN}.
31. CN replies the tunnel response message to RS to complete the tunnel construction process.
32. RS replies the tunnel response message to MN to complete the tunnel construction process.
33. The negotiation module notifies the completion of the tunnel construction process to the application.
34. The negotiation module transmits a keep alive message to keep the NAT table on the route.

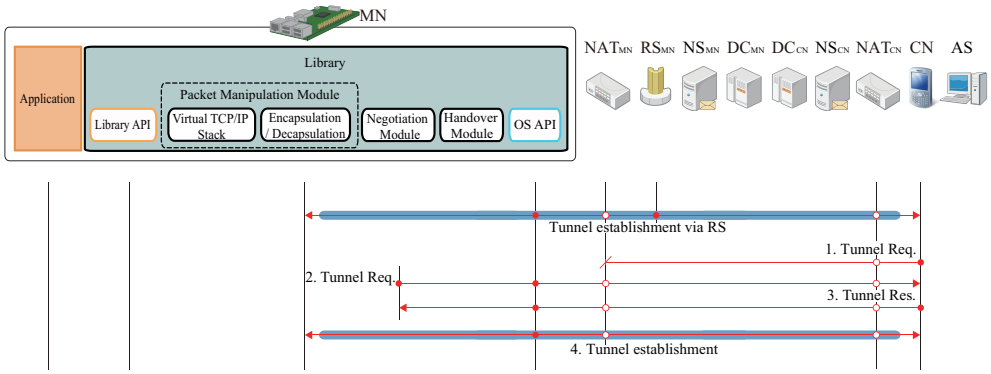


Figure 8: Signaling route optimization of library

- Route Optimization

  Figure 8 shows the signaling processes of route optimization in NTMobile. MN may be able to optimize the communication route by changing the route directly when it uses RS for the relay communication. Each NTMobile node transmits the tunnel request message to its own corresponding node to check the direct accessibility. They can change the route from the relay communication to the direct communication when either of NTMobile node can receive the tunnel request message from its own corresponding node. When neither NTMobile nodes can receive the tunnel request message, they use RS for the relay communication.

  1. CN transmits the tunnel request message to MN. In this case, the message cannot reach MN.
  2. MN transmits the tunnel request message to CN. In this case, the message can reach CN.
  3. CN replies the tunnel response message to MN to complete the tunnel construction process for the direct communication.
  4. Both NTMobile nodes change the tunnel route from the route through RS to the direct route.

- Tunnel communication in a same real private address

  Figure 9 shows the tunnel communication where a same real private IP address RIP is assigned to both MN and CN. In this situation, MN and CN use virtual IP addresses $\mathrm{VIP_{MN}}$ and $\mathrm{VIP_{CN}}$ respectively. Therefore, the application also applies these virtual IP addresses for communication. In the tunnel communication, the IP packet including these virtual IP addresses is encapsulated in the packet manipulation module. As a result, the packet manipulation module changes the IP address field from these virtual IP addresses to the real IP addresses RIP and $\mathrm{RIP_{RS}}$ The source address changes from RIP to $\mathrm{RIP_{NATMN}}$ through $\mathrm{NAT_{MN}}$. RS transfer the received packet from MN. Hence, the addresses changes from the pair of $\mathrm{RIP_{NATMN}}$ and $\mathrm{RIP_{RS}}$ to the pair of $\mathrm{RIP_{RS}}$ and $\mathrm{RIP_{NATCN}}$. The destination address changes from $\mathrm{RIP_{NATCN}}$ to RIP through $\mathrm{NAT_{CN}}$. Finally, CN receives this packet from MN through RS.

- Tunnel communication with cascading NATs

  Figure 10 shows the tunnel communication where a same real private IP address RIP is assigned to both MN and CN and CN exists under the cascading NAT routers. In the tunnel communication, the IP packet including virtual IP addresses is encapsulated in the packet manipulation module. As a result, the packet manipulation module changes the IP address field from these virtual IP addresses to the real IP addresses in the similar manner in Fig. 9. Finally, CN receives this packet from MN through RS.

4 EVALUATION

4.1 Processing Overhead

We have measured the processing overhead in a NTMobile network. We have prepared the evaluation environment in Fig. 11. Table 1 shows the detail information about each component.

In the evaluation, we have measured each period for the NTMobile signaling. The results represent an average of 100 trials. In the measurement, we used ping tool to measure round trip time (RTT) and NSLOOKUP tool to measure a name resolution period for an AAAA record. Tables 2 and 3 show the RTT for each link. We can find that the RTT between servers is around 0.1[ms] and RTT between a server and a NTMobile node is around 1[ms]. The difference comes from the hardware performance.

Table 4 shows the processing period for each NTMobile signaling. The results show that the developed implementation requires about 200[ms] for authentication and a registration. The authentication is required when a NTMobile node joins a NTMobile network. Therefore, 200[ms] are acceptable time for the login process. The registration is required when a NTMobile node changes an access network. It is also acceptable for updating an own network information. The tunnel construction period is an important for a usability. From the results, we can find that the tunnel construction period is less that 200[ms]. It is a known fact that TCP connection setup requires a three-way handshake. Being compared to the period for the three-way handshake, the tunnel construction period is also acceptable.

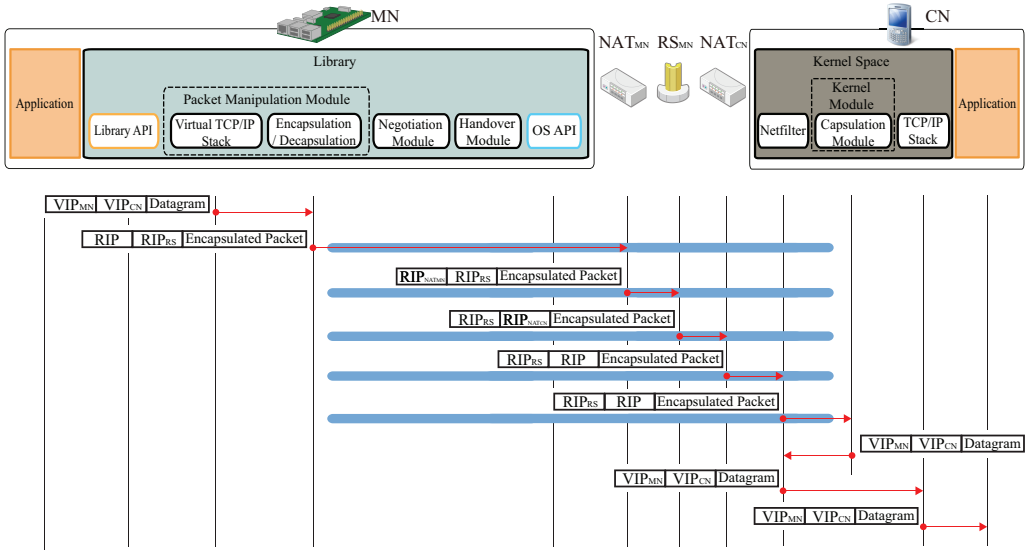


Figure 9: Tunnel communication in a same real private address

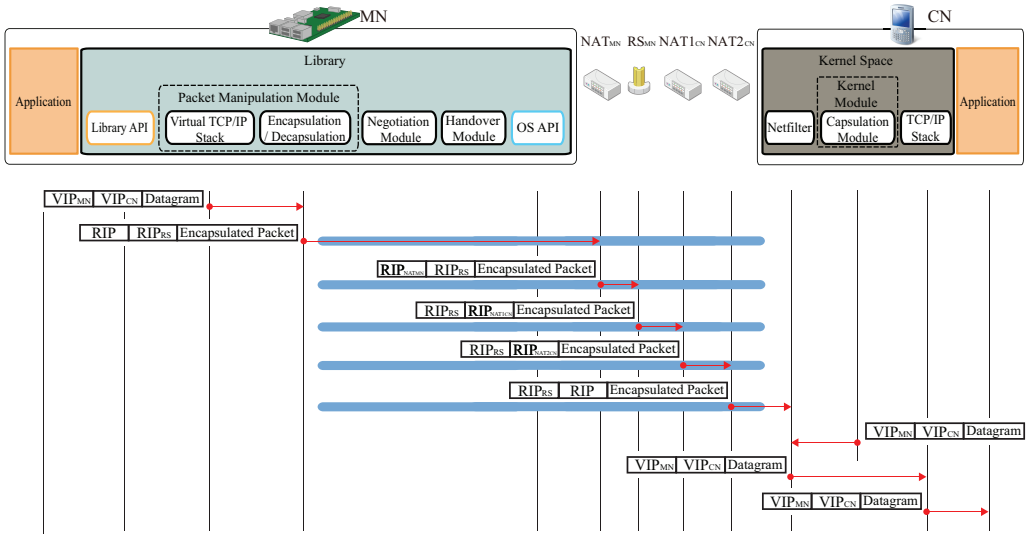


Figure 10: Tunnel communication with cascading NATs

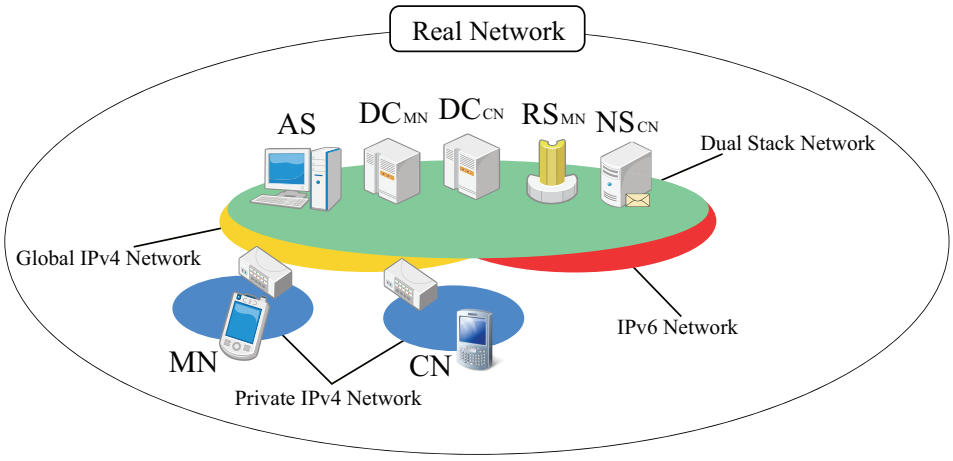


Figure 11: Evaluation environment

Table 1: Machines specification

	AS, DC_{MN}, DC_{CN}, RS_{MN}, NS_{CN}	MN, CN
OS	CentOS 6.7	Ubuntu 12.04
Kernel	Linux 2.6	Linux 3.2
CPU	Intel Xeon 2.8GHz	Virtual 1core 1.8GHz
Memory	512MB	512MB

Table 4: Processes period

	Delay time [ms]		
Process	min	avg	max
Authentication	152.0	179.1	226.6
Registration & Notification Registration (without NS)	57.4	64.7	71.3
Registration & Notification Registration (with NS)	172.3	194.6	233.5
Notification Registration - ACK (TCP)	48.9	55.1	63.4
Tunnel Construction (without NS)	137	169	191
Direction Request - Route Direction	112.0	134.6	151.8
Tunnel Request - Tunnel Response	13.8	29.3	45.5
Tunnel Construction (with NS)	170	197	226
Direction Request - Route Direction	133.5	154.3	174.2
Tunnel Request - Tunnel Response	13.8	29.3	45.5

Table 2: RTT between servers

	RTT [ms]		
Connection pair	min	avg	max
AS - DC_{MN}	0.117	0.140	0.170
AS - DC_{CN}	0.110	0.121	0.174
DC_{MN} - RS_{MN}	0.128	0.148	0.173
DC_{CN} - NS_{CN}	0.133	0.155	0.177

Table 3: RTT between servers and nodes

	RTT [ms]		
Connection pair	min	avg	max
AS - MN	0.99	1.45	1.87
AS - CN	0.91	1.39	2.20
DC_{MN} - MN	0.91	1.38	1.97
DC_{CN} - CN	0.97	1.42	2.26
RS_{MN} - MN	0.91	1.39	1.81
RS_{MN} - CN	1.00	1.38	2.22
NS_{CN} - CN	0.92	1.35	3.04

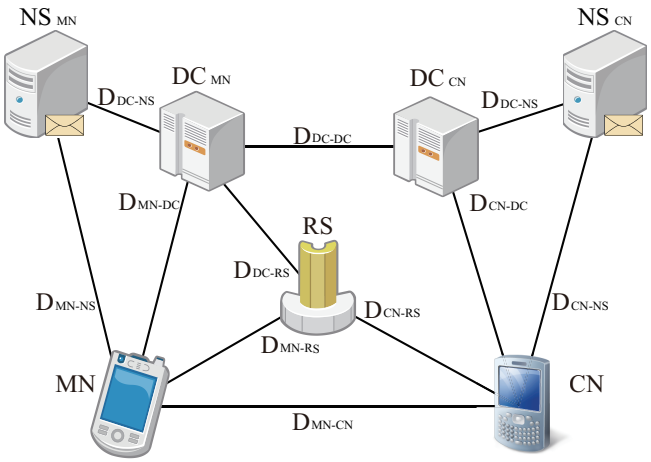


Figure 12: Transmission delay in NTMobile network

4.2 Processing Transmission Delay

We have measured transmission delays in real IP networks to evaluate the overhead due to transmission delay in practical networks. Figure 12 shows the definition of each transmission delay in NTMobile network.

Table 5 shows the number of each transmission delay to construct a communication tunnel. Therefore, the total overhead due to the transmission delay is the summation of each delay. Tables 6 and 7 show the measurement results of transmission delay on ping tool in real wired networks and in LTE networks. The results are the average of 1,000 trials and 100 trials respectively. Table 8 shows the query period for the DNS mechanism with NSLOOKUP tool. From the evaluations in Tabs. 5, 6 and 7, the estimated tunnel construction period is 491 [ms] for direct communication and 689 [ms] for relay communication. The main factor of the overhead comes from the transmission delay in LTE networks. We think that the overhead can be acceptable in practical situations because it occurs only when users try to start a communication.

5 CONCLUSION

This paper has extend the IP mobility mechanism called NTMobile (Network Traversal with Mobility) to designed an application based IP mobility scheme on Linux systems, where the developed IP mobility library can realize the IP mobility function in an application layer on Linux systems. As a result, developers can realize an end-to-end communication model by employing the enhanced IP mobility library.

Table 5: Number of delay path for tunnel construction process

Connection path	Route name	Times
Direct	D_{MN-DC}	2
	DNS	1
	D_{DC-DC}	6
	D_{DC-NS}	1
	D_{CN-NS}	1
	D_{CN-DC}	3
	D_{MN-CN}	1
Relay	D_{MN-DC}	2
	DNS	1
	D_{DC-DC}	6
	D_{DC-RS}	2
	D_{DC-NS}	1
	D_{CN-NS}	1
	D_{CN-DC}	3
	D_{MN-RS}	2
	D_{CN-RS}	3

Table 6: Transmission delay in wired networks

Target host	Trial	Average time [msec]
google.com	1,000	6.222
yahoo.co.jp	1,000	7.3215

Table 7: Transmission delay in LTE networks

Target host	Trial	Average time [msec]
google.com	100	59.365
yahoo.co.jp	100	63.79

Table 8: Query period of DNS lookup

Target domain	Trial	Average time [msec]
aitech.ac.jp	100	12.54
yahoo.co.jp	100	12.9

ACKNOWLEDGEMENT

This work is supported in part by the Grant-in-Aid for Scientific Research (26330103, 15H02697), Japan Society for the Promotion of Science (JSPS) and the Integration research for agriculture and interdisciplinary fields, Ministry of Agriculture, Forestry and Fisheries, Japan, and the Science Research Promotion Fund from The Promotion and Mutual Aid Corporation for Private Schools of Japan.

REFERENCES

[1] M. Buddhikot, G. Chandranmenon, S. Han, Y. W. Lee, S. Miller, and L. Salgarelli, "Integration of 802.11 and third-generation wireless data networks," Proceedings of the IEEE INFOCOM 2003, Volume 1, pp. 503 - 512, March/April (2003).

[2] Q. Zhang, C. Guo, Z. Guo, and W. Zhu, "Efficient mobility management for vertical handoff between WWAN and WLAN," IEEE Communications Magazine, Volume 41, No. 11, pp. 102 - 108, November (2003).

[3] L. A. Magagula and H. A. Chan, "IEEE802.21-Assisted Cross-Layer Design and PMIPv6 Mobility Management Framework for Next Generation Wireless Networks," Proc. IEEE WIMOB '08, pp. 159 - 164, October (2008).

[4] D. Le, X. Fu and D. Hogrere, "A Review of Mobility Support Paradigms for the Internet," IEEE Communications surveys, 1st quarter 2006, Volume 8, No. 1, pp. 38 - 51, First Quarter (2006).

[5] A. C. Snoeren and H. Balakrishnan, "An End-to-End Approach to Host Mobility," Proceedings of the ACM Mobicom, pp. 155 - 166, August (2000).

[6] C. Perkins, "IP Mobility Support for IPv4, Revised," IETF RFC 5944, November (2010).

[7] M. Ishiyama, M. Kunishi, K. Uehara, H. Esaki, and F. Teraoka, "LINA: A New Approach to Mobility Support in Wide Area Networks," IEICE transactions on communications, Volume E84-B, No.8, pp. 2076 - 2086, August (2001).

[8] http://www.mip4.org, retrieved: February (2012).

[9] C. Perkins and D. Johnson, "Route Optimization in Mobile IP," draft-ietf-mobileip-optim-11.txt, Work in progress, September (2001).

[10] S. Salsano, C. Mingardi, S. Niccolini, A. Polidoro, and L. Veltri, "SIP-based Mobility Management in Next Generation Networks," IEEE Wireless Communication, Volume 15, Issue 2, pp. 92 - 99, April (2008).

[11] M. Bonola, S. Salsano, and A. Polidoro, "UPMT: universal per-application mobility management using tunnels," In Proc. of the 28th IEEE conference on Global telecommunications (GLOBECOM'09), pp. 1 - 8, November/December (2009).

[12] M. Bonola and S. Salsano, "S-UPMT: a secure Vertical Handover solution based on IP in UDP tunneling and IPsec," GTTI Riunione Annuale 2010, (online), http://www.gtti.it/GTTI10/papers/gtti10_submission_29.pdf, June (2010).

[13] H. Soliman, "Mobile IPv6 Support for Dual Stack Hosts and Routers," IETF RFC 5555, June (2009).

[14] E. Fogelstroem, A. Jonsson, and C. Perkins, "Mobile IPv4 Regional Registration," IETF RFC 4857, June (2007).

[15] H. Levkowetz and S. Vaarala, "Mobile IP Traversal of Network Address Translation (NAT) Devices," IETF RFC 3519, April (2003).

[16] G. Montenegro, "Reverse Tunneling for Mobile IP, revised," IETF RFC 3024, January (2001).

[17] J. Rosenberg, R. Mahy, P. Matthews, and D. Wing, "Session Traversal Utilities for NAT (STUN)," IETF RFC 5389, October (2008).

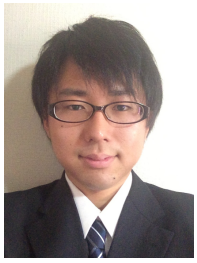
[18] P. Srisuresh, B. Ford, and D. Kegel, "State of Peer-to-Peer (P2P) Communication across Network Address Translators (NATs)," IETF RFC 5128, March (2008).

[19] R. Mahy, P. Matthews, and J. Rosenberg, "Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN)," IETF RFC 5766, April (2010).

[20] J. Rosenberg, "Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols," IETF

RFC 5245, April (2010).
[21] M. Westerlund and C. Perkins, "IANA Registry for Interactive Connectivity Establishment (ICE) Options," IETF RFC 6336, July (2011).
[22] K. Naito, K. Kamienoo, T. Nishio, H. Suzuki, A. Watanabe, K. Mori, and H. Kobayashi, "Proposal of Seamless IP Mobility Schemes: Network Traversal with Mobility (NTMobile)," IEEE GLOBECOM 2012, pp. 2572 - 2577, December (2012).
[23] K. Naito, K. Kamienoo, H. Suzuki, A. Watanabe, K. Mori, and H. Kobayashi, "End-to-end IP mobility platform in application layer for iOS and Android OS," IEEE Consumer Communications & Networking Conference (CCNC 2014), pp. 92 - 97, January (2014).
[24] http://www.ntmobile.net
[25] lwip - A Lightweight TCP/IP stack, http://www.nongnu.org/lwip/

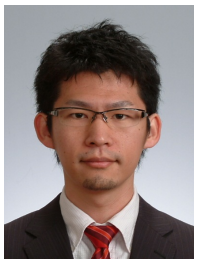
(Received October 30, 2015)
(Revised February 7, 2016)



Kohei Tanaka is currently working towards the B.S. degree in the Faculty of Information Science at Aichi Institute of Technology, Japan. His research interests include network mobility systems and Internet of Things (IoT).

Fumihito Sugihara received the B.S. degree in Electrical and Electronic Engineering from Mie University, Japan in 2014. He is currently working towards the M.S. degree in the Department of Electrical and Electronic Engineering at Mie University, Japan. His research interests include network mobility systems and Machine to Machine (M2M).

Katsuhiro Naito received the B.S. degree in Electronics Engineering from Keio University, Japan in 1999, and received the M.S. and Ph.D. degrees in Information Engineering from Nagoya University, Japan in 2001 and 2004, respectively. From 2004 to 2014, Dr. Naito was an assistant professor in the electrical and electronic engineering department of Mie university. He was a visiting scholar in the computer science department of University of California, Los Angeles (UCLA) in 2011. Since 2014, he has been an associate professor in the information science department of Aichi Institute of Technology. His research interests include 5G technologies, vehicular communication systems, Internet of Things (IoT) and Machine to Machine (M2M) systems, overlay networks, and network protocols and architectures.



Hidekazu Suzuki received his B.S., M.E., and Ph.D. degrees in Information Engineering from Meijo University, Japan in 2004, 2006 and 2009, respectively. From 2008 to 2010, he was a Postdoctoral Research Fellow of the Japan Society for the Promotion of Science. He was an Assistant Professor in the Department of Information Engineering in Meijo University from 2010 to 2015. He has been an Associate Professor in the same department from 2015. His research interests are in the fields of network security, mobile networks and home networks, and so on. He is also a member of IEEE, ACM, IEICE and IPSJ.

Akira Watanabe received his B.E. and M.E. degrees from Keio University in 1974 and 1976, respectively. He joined Mitsubishi Electric Corporation in 1976, and engaged in R&D of network devices. He received the Ph.D. degree in communication systems from Keio University in 2001. Since joining Meijo University in 2002, he has been researching on network connectivity and mobility.