End-to-end IP mobility platform in application layer for iOS and Android OS

Katsuhiro NAITO, Kazuo MORI
and Hideo KOBAYASHI
Graduate School of Engineering
Mie University
1577 Tsu, Mie, Japan
Email: naito, kmori, koba@elec.mie-u.ac.jp

Kazuma Kamienoo, Hidekazu SUZUKI
and Akira WATANABE
Graduate School of Science and Technology
Meijo University
Tenpaku-ku, Nagoya, Aichi, Japan
Email: hsuzuki@meijo-u.ac.jp, wtnbakr@meijo-u.ac.jp

Abstract—**Smartphones are a new type of mobile devices that users can install additional mobile software easily. In the almost all smartphone applications, client-server model is used because end-to-end communication is prevented by NAT routers. Recently, some smartphone applications provide real time services such as voice and video communication, online games etc. In these applications, end-to-end communication is suitable to reduce transmission delay and achieve efficient network usage. Also, IP mobility and security are important matters. However, the conventional IP mobility mechanisms are not suitable for these applications because most mechanisms are assumed to be installed in OS kernel. We have developed a novel IP mobility mechanism called NTMobile (Network Traversal with Mobility). NTMobile supports end-to-end IP mobility in IPv4 and IPv6 networks, however, it is assumed to be installed in Linux kernel as with other technologies. In this paper, we propose a new type of end-to-end mobility platform that provides end-to-end communication, mobility, and also secure data exchange functions in the application layer for smartphone applications. In the platform, we use NTMobile, which is ported as the application program. Then, we extend NTMobile to be suitable for smartphone devices and to provide secure data exchange. Client applications can achieve secure end-to-end communication and secure data exchange by sharing an encryption key between clients. Users also enjoy IP mobility which is the main function of NTMobile in each application. Finally, we confirmed that the developed module can work on Android system and iOS system.**

I. Introduction

Smartphones are new types of mobile devices that support multiple wireless interfaces such as 3G, LTE, WiFi etc. [1], [2]. Operating systems select a wireless interface to access to the Internet according to received signal intensity of each interface and access policies [3]. Various smartphone applications have been distributed for smartphone platform such as iOS, Android etc. Almost all smartphone applications employ client-server models because of the restriction of real networks.

Recently, some smartphone applications provide a real time communication service such as voice and video communication, online games etc. In the real time communication, end-to-end communication is a good way to reduce round trip time (RTT) and to achieve effective network usage. However, these smartphone applications also employ server-client models because end-to-end communication is difficult for smartphones in real networks because NAT, which is normally used in IPv4 networks, prevents end-to-end communication. In server client systems, servers have to be expanded to reduce transmission delay when the number of users increase. As the result, maintenance cost for the server resources is also increased, and the increased cost causes difficulty of extensive software developments. From the viewpoint of mobile carriers, a large amount of traffic passes through boarder gateway routers and consumes network resource in mobile carriers. Therefore, software developers should also consider the effective network usage to achieve continuous growth of smartphone applications.

In addition, real time communication services generally require continuous communication. However, operating systems may change wireless interfaces due to client movement or access policies. The change of interfaces causes the change of the source IP address for all applications. As the results, the real time communication is disconnected due to the change of the source IP address.

IP mobility mechanisms are a good way to solve these issues in real time applications on smartphone devices. They can provide continuous communication service even when IP addresses change due to client movements or switching of access networks [4], [5], [6], [7]. In most IP mobility mechanisms, location information of each client is uploaded to management servers, by which, clients can find a destination IP address with a host identifier such as a host name, a fully qualified domain name (FQDN) etc.

As for IP mobility mechanisms, some implementations for IPv6 have been proposed. On the contrary, the number of implementations for IPv4 is quite few though some mechanisms have been proposed [8], [9], [10]. Generally, IPv4 is used in mobile carrier networks because IPv4 is the mainstream protocol in current networks. Recently, some mobile carriers start to employ large scale network address translator (LSN) in order to meet the shortage of IPv4 global addresses because smartphone devices require more and more IP addresses [11], [12], [13]. Therefore, IP mobility in the case when NAT exists between end nodes is also an important issue. Additionally, IPv6 addresses have been introduced to solve the problem on a long term basis. As the results, IP mobility mechanisms for IPv4 and IPv6 networks will become important functions for smartphone applications. However, it is difficult for software developers to implement own IP mobility mechanisms in each smartphone application because IP mobility mechanisms are generally constructed on complicated systems including server functions and client functions.

The authors have developed a new type of a novel IP mobility technology called NTMobile (Network Traversal with Mobility) [16], [17], [18]. In NTMobile, each client has a virtual IP address that is assigned from the direction coordinator. Applications in the node create a connection by using virtual IP addresses. The packets having virtual IP addresses are encapsulated by UDP packets with real

IP addresses. Therefore, the applications can continue the communication even if a real IP address is changed due to switching of access networks. In NTMobile systems, we set two types of equipments: direction coordinators (DCs) and Relay Servers (RSs). DCs store the location information of clients and manage connection between clients. RSs are the special servers to relay packets between clients when both clients cannot communicate directly due to NAT routers or clients use different protocol versions such as IPv4 and IPv6. We have implemented NTMobile on Android smartphone devices. However, the NTMobile implementation requires root authority in Linux system to install the IP mobility functions, and is insufficient to meet the requirement for smartphone applications because smartphone applications are generally installed by user authority.

The goal of this paper is to provide secure end-to-end communication framework that can be installed by user authority for smartphone applications. For realtime communication, we employ NTMobile as the base system because NTMobile can support IP mobility in IPv4 and IPv6 networks and secure data communication between clients. Then, we extend it to support notification mechanisms of smartphone OS to reduce keep alive operations. Generally, smartphone device may be offline due to out of service, shutdown etc. Especially, iOS does not permit background operation of smartphone applications to the Apple's policy even if the device is connected to networks. Therefore, non-realtime communication is also required in smartphone applications. Hence, we also propose secure data exchange mechanisms that are suitable for non-realtime communication. In conventional server-client models, data are just encrypted over networks, not on servers. Our secure data exchange mechanisms provide end-to-end encryption of data. As the results, servers cannot decrypt data because encryption key is shared between only clients. In the implementation, we design NTMobile application module that supports the end-to-end communication and the secure data exchange mechanisms. Therefore, software developers can employ the module to support IP mobility and secure data exchange. Moreover, the developed smartphone applications employing the module can be installed with user authority, not root authority. To the best of our knowledge, NTMobile application module is the first end-to-end mobility framework in the application layer for smartphone applications. By conducting experiments, we confirmed that the developed module can work on Android system and iOS system.

II. System Model

Fig. 1 shows the system model of the proposed end-to-end IP mobility platform. The server systems consist of an account server (AS), some direction coordinators (DCs) with some relay servers (RSs), cache servers (CSs) and NTMobile clients. All DCs are linked to the AS, and RSs, CSs and NTMobile clients are also linked to their DC. NTMobile client is realized with NTMobile application module which is installed in the application layer. Therefore, the proposed platform can be extended easily with the increase of users. In NTMobile, each client has a virtual IP address for applications, and creates a UDP tunnel between clients according to a direction from its DC. Therefore, each client can communicate each other by using the virtual IP addresses even if real IP addresses are changed. The details of the system components are as followings.

- Account Server
  AS is an only equipment that manages authentication information for users. Hence, AS replies an authentication reply when a DC makes inquiries about their client authentication to AS. In addition, AS also distributes authentication information to CSs because CSs require to authenticate clients when data are uploaded.
- Direction Coordinator
  In NTMobile, DC manages its clients with NTMobile application module. Hence, each DC has location information for clients to achieve IP mobility. DC also manages its RSs and its CSs to achieve IP mobility and data exchange. In addition, DC relays an authentication request from a client to AS because DCs do not have authentication information for users. Additionally, DC suports multiple smartphone applications. Therefore, software developers can share DC to provide own application services.
- Relay Server
  The function of relay servers is to relay packets between clients when both clients exist under different NAT routers or exist under different version networks such as IPv4 and IPv6 networks. The relay function is activated by its DC when DC decides to use a RS to create a UDP tunnel between clients.
- Cache Server
  The function of cache servers is to store exchanged data for clients in the secure data exchange mechanisms. Smartphone devices may be offline due to out of service in 3G systems and WiFi in the real usage situation. Especially, iOS systems do not permit background operation of applications. In NTMobile application module, clients can exchange a shared key between clients by using their DCs and CS. Hence, the module can provide secure data exchange mechanisms for developers.

III. Realtime communication

In NTMobile application module, we employ NTMobile to achieve realtime communication between clients. Therefore, we employ the same packet structure and signaling processes in NTMobile[16], [17]. In NTMobile, each client should register its location information such as an IP address, a port number etc. to its DC. The registered information is used to create a UDP tunnel between clients. Additionally, NTMobile has own secure encryption key exchange mechanism between clients. Therefore, secure realtime communication can be achieved by using the exchanged shared key.

Fig. 2 shows the overview of UDP tunnel creation process between clients. In the figure, MN has a global IP address and CN has a private IP address under NAT router $\mathrm{NAT_{CN}}$. Then, MN has a private IP address under NAT router $\mathrm{NAT_{MN}}$ due to client movements. For application communication, each client uses its virtual IP address assigned from its DC. As the results, the applications can continue to communicate based on virtual IP address though real IP addresses change. The followings are processes to create and recreate end-to-end tunnels.

- Location registration
  MN transmits the registration request message to its $\mathrm{DC_{MN}}$ to register its own location information. $\mathrm{DC_{MN}}$ replies the registration response message as the acknowledgement.
- Tunnel creation request
  MN transmits the direction request message to $\mathrm{DC_{MN}}$

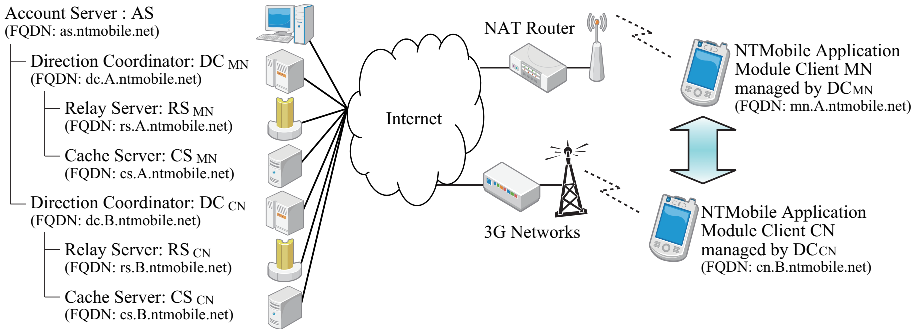
Account Server : AS
(FQDN: as.ntmobile.net)
Direction Coordinator: DC MN
(FQDN: dc.A.ntmobile.net)
Relay Server: RS MN
(FQDN: rs.A.ntmobile.net)
Cache Server: CS MN
(FQDN: cs.A.ntmobile.net)
Direction Coordinator: DC CN
(FQDN: dc.B.ntmobile.net)
Relay Server: RS CN
(FQDN: rs.B.ntmobile.net)
Cache Server: CS CN
(FQDN: cs.B.ntmobile.net)
Internet
NAT Router
3G Networks
NTMobile Application
Module Client MN
managed by DCMN
(FQDN: mn.A.ntmobile.net)
NTMobile Application
Module Client CN
managed by DCCN
(FQDN: cn.B.ntmobile.net)

Fig. 1. System model.

to request the tunnel creation between MN and CN. In NTMobile, each DC has a DNS function and FQDN of each DC is registered as Name Server (NS) record. Therefore, $\mathrm{DC}_{\mathrm{MN}}$ transmits the DNS NS query to find the FQDN of $\mathrm{DC}_{\mathrm{CN}}$ which is the manager of CN because each DC is linked to its domain name in NTMobile. Then, $\mathrm{DC}_{\mathrm{MN}}$ transmits the NTM Information Request message to $\mathrm{DC}_{\mathrm{CN}}$ to obtain the client information of CN. In this figure, $\mathrm{DC}_{\mathrm{MN}}$ decides to create the direct connection between MN and CN because MN has the global IP address. Then, $\mathrm{DC}_{\mathrm{MN}}$ transmits the route direction messages to MN and CN via $\mathrm{DC}_{\mathrm{CN}}$. The reason to relay the messages at $\mathrm{DC}_{\mathrm{CN}}$ is that $\mathrm{DC}_{\mathrm{MN}}$ does not have confidential relationship with CN because only a client and its DC have confidential relationship in NTMobile. Additionally, the route direction messages includes a temporal key for encrypting a shared key. $\mathrm{DC}_{\mathrm{CN}}$ transmits the notification request message to push notification servers such as GCM (Google Cloud Messaging) [14] and APNS (Apple Push Notification Service) [15] to notify the arrival of the route direction message. CN transmits the route direction request message to $\mathrm{DC}_{\mathrm{CN}}$ to request the route direction message. $\mathrm{DC}_{\mathrm{CN}}$ replies the route direction message to CN. As the results, both MN and CN can recognize how to create the tunnel between end-clients.

- Tunnel establishment
In NTMobile, client having a global IP address should receive a tunnel request message because client having a private IP address cannot receive any packets from the Internet. In this figure, CN having the private IP address transmits the tunnel request message to MN having the global IP address to create the UDP tunnel between end-clients. MN replies the tunnel response message to CN to establish the tunnel. Finally, both MN and CN can be connected through the established tunnel. In this process, each client also exchange a shared key encrypted by the temporal key from $\mathrm{DC}_{\mathrm{MN}}$. Both clients establish a tunnel to RS when both clients cannot communicate directly due to NAT routers or clients use different protocol versions such as IPv4 and IPv6.
- Location reregistration
Client should reregister its own location information when the access IP address changes due to client movements or switching of access networks. In this figure, the IP address of MN has changed from the global IP address to the private IP address. Therefore, MN transmits the registration request message to its $\mathrm{DC}_{\mathrm{MN}}$ to register its own location information. $\mathrm{DC}_{\mathrm{MN}}$ replies the registration response message as the acknowledgement.
- Tunnel recreation request
MN should request to recreate the UDP tunnel between MN and CN due to change of the access IP address. In the similar manner, MN transmits the direction request message to $\mathrm{DC}_{\mathrm{MN}}$ In this figure, $\mathrm{DC}_{\mathrm{MN}}$ decides to create the new relay connection through the relay server $\mathrm{RS}_{\mathrm{MN}}$ between MN and CN because both clients have the private IP addresses and exist under different NAT routers. Then, $\mathrm{DC}_{\mathrm{MN}}$ transmits the route direction messages to MN and CN via $\mathrm{DC}_{\mathrm{CN}}$. In the tunnel recreation processes, $\mathrm{DC}_{\mathrm{CN}}$ transmits the route direction message to CN because notification may be delayed according to the status of notification servers.
- Tunnel establishment
In NTMobile, a relay server should receive a tunnel request message when both clients have private IP addresses. In this figure, both MN and CN transmit the tunnel request message to the relay server to recreate the UDP tunnel. $\mathrm{RS}_{\mathrm{MN}}$ replies the tunnel response messages to MN and CN respectively to establish the tunnel. Finally, both MN and CN can be connected through $\mathrm{RS}_{\mathrm{MN}}$.

IV. Non-realtime communication

In the non-realtime communication, we have introduced cache servers to store exchanged data between clients. In the conventional systems, encryption of data is performed over networks. As the results, servers can recognize all data. Proposed NTMobile application module provides secure data exchange mechanisms for software developers. In the module, each client can exchange a shared key for data exchange beforehand. Then, they can exchange encrypted data through a cache server. As the results, all servers cannot decrypt data through servers because the encryption key is shared only in clients.

A. Key exchange processes

In NTMobile application module, all data should be encrypted. Therefore, key exchange mechanisms are required to share a shared encryption key between clients. Fig. 3 shows the key exchange process. In the proposed mechanisms, MN, which is the source client, creates a temporal key and a shared encryption key. The temporal key is used to encrypt the shared encryption key. Therefore, the both keys should be transfered separately. In the proposed platform, the temporal

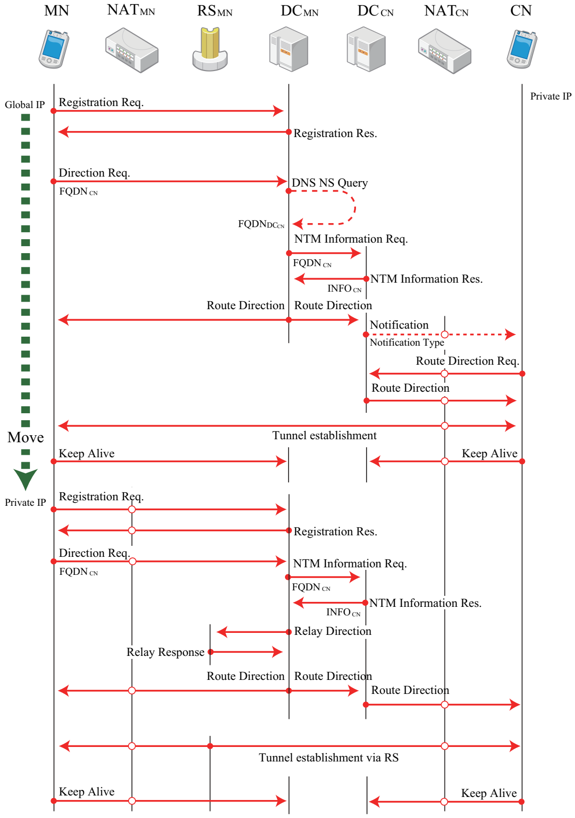

Fig. 2. Tunnel creation process for realtime communication.

key is transfered through direction coordinators, and the shared encryption key is transfered through a cache server. The detail procedures are described as followings.

- Upload of temporal key
  MN creates the temporal key for encryption of the shared encryption key, and uploads it to its own $\mathrm{DC}_{\mathrm{MN}}$. The temporal key is assumed to be encrypted by another shared key between MN and $\mathrm{DC}_{\mathrm{MN}}$ because clients share the key with its DC in the login process.
- Transfer of temporal key
  $\mathrm{DC}_{\mathrm{MN}}$ transmits the DNS NS query to find the FQDN of $\mathrm{DC}_{\mathrm{CN}}$ which is the coordinator of CN. Then, it transfers the uploaded temporal key to $\mathrm{DC}_{\mathrm{CN}}$. In NTMobile, communication between servers can be encrypted because each server has own electronic signature. Then,the uploaded temporal key is also encrypted by the another shared key between the servers.
- Upload of shared key
  MN creates a shared key for encryption of upload data, and uploads it to its own $\mathrm{CS}_{\mathrm{MN}}$ as the encryption key upload request message. The shared key is assumed to be encrypted by another shared key between MN and $\mathrm{CS}_{\mathrm{MN}}$ because MN shares the key in the login process.
- Notification of shared key
  $\mathrm{CS}_{\mathrm{MN}}$ transmits the encryption key exchange request message to $\mathrm{DC}_{\mathrm{CN}}$. $\mathrm{DC}_{\mathrm{CN}}$ transmits the notification request message to push notification servers such as GCM and APNS to notify the arrival of the encryption key exchange request message. $\mathrm{DC}_{\mathrm{CN}}$ replies the encryption key exchange response message to $\mathrm{DC}_{\mathrm{MN}}$. $\mathrm{DC}_{\mathrm{MN}}$ also replies the encryption key upload response message to MN.
- Download of temporal key
  CN transmits the temporal key list download request message to its $\mathrm{DC}_{\mathrm{CN}}$. $\mathrm{DC}_{\mathrm{CN}}$ replies the temporal key list download response message including the hash values of the temporal keys. Then, CN transmits the temporal key download request message with the hash value to $\mathrm{DC}_{\mathrm{CN}}$. $\mathrm{DC}_{\mathrm{CN}}$ replies the temporal key download response including the temporal key.
- Download of shard key
  CN transmits the cache list download request message to its $\mathrm{DC}_{\mathrm{CN}}$. $\mathrm{DC}_{\mathrm{CN}}$ replies the cache list download response message including the FQDN of cache servers and the hash values of the uploaded data. CN transmits the encryption key download request message including the hash value to $\mathrm{CS}_{\mathrm{MN}}$. $\mathrm{CS}_{\mathrm{MN}}$ replies the encryption key download response message including the shared key. Finally, CN decrypts the encrypted shared key with the temporal key.

B. *Data exchange processes*

After the key exchange processes, both MN and CN can share the shared key to encrypt the data. The detail of data exchange procedures are described as followings.

- Upload of encrypted data
  MN encrypted the data with the shared key that is shared in the key exchange processes. Then, it uploads the encrypted data to its own $\mathrm{CS}_{\mathrm{MN}}$ as the encrypted data upload request message.
- Notification of encrypted data
  $\mathrm{CS}_{\mathrm{MN}}$ transmits the DNS NS query to find the FQDN of $\mathrm{DC}_{\mathrm{CN}}$ which is the coordinator of CN. $\mathrm{CS}_{\mathrm{MN}}$ transmits the encrypted data exchange request message to $\mathrm{DC}_{\mathrm{CN}}$. $\mathrm{DC}_{\mathrm{CN}}$ transmits the notification request message to push notification servers such as GCM and APNS to notify the arrival of the encrypted data exchange request message. $\mathrm{DC}_{\mathrm{CN}}$ replies the encrypted data exchange response message to $\mathrm{CS}_{\mathrm{MN}}$. $\mathrm{DC}_{\mathrm{MN}}$ also replies the encrypted data upload response message to MN.
- Download of encrypted data
  CN transmits the cache list download request message to its $\mathrm{DC}_{\mathrm{CN}}$. $\mathrm{DC}_{\mathrm{CN}}$ replies the cache list download response message including the FQDN of cache servers and the hash values of the uploaded data. CN transmits the encrypted data download request message including the hash value to $\mathrm{CS}_{\mathrm{MN}}$. $\mathrm{CS}_{\mathrm{MN}}$ replies the encrypted data download response message including the encrypted data. Finally, CN decrypts the encrypted data with the shared key that is shared in the key exchange processes.
- Confirmation of encrypted data
  CN transmits the encrypted data confirmation request message including the hash value and its own FQDN to $\mathrm{CS}_{\mathrm{MN}}$. $\mathrm{CS}_{\mathrm{MN}}$ replies the encrypted data confirmation response message. Then, $\mathrm{CS}_{\mathrm{MN}}$ updates the delivery status of the encrypted data.

V. Implementation

Fig. 5 shows the implementation model. We have implemented the server function by using C for NTMobile functions, perl for HTTP functions and mySQL over Cent OS 6.4. Additionally, we have developed the NTMobile application module

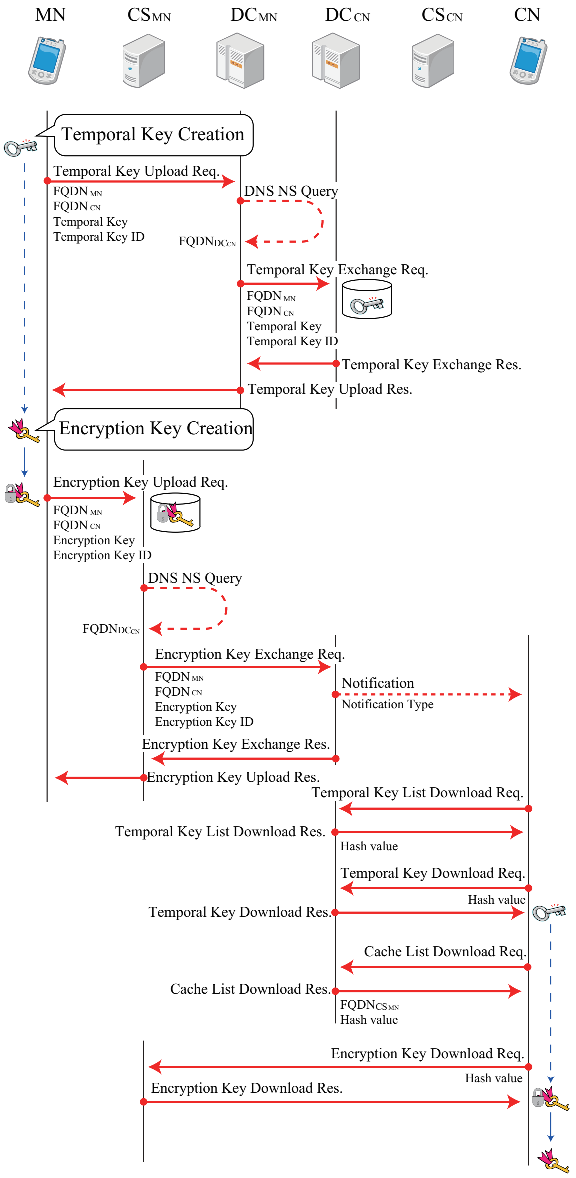

Fig. 3. Key exchange process.

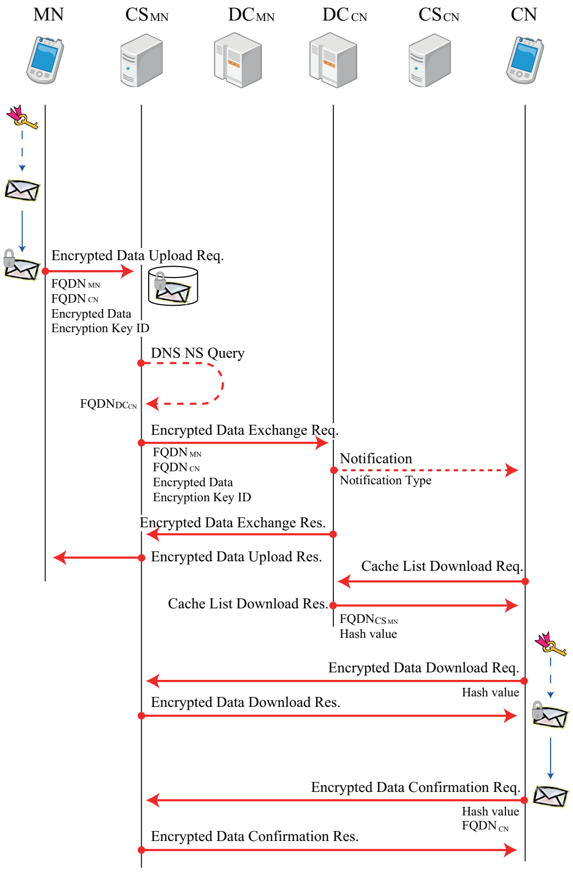

Fig. 4. Data exchange process.

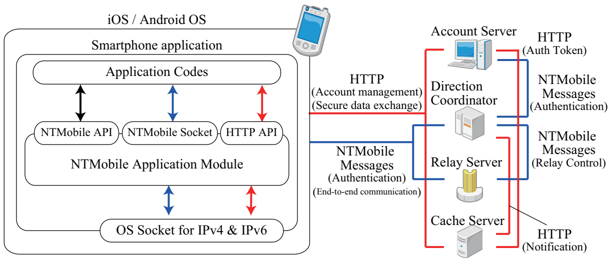

Fig. 5. Implementation model.

for smartphone application by using core NTMobile functions. In the application module, many useful APIs, such as authentication, account management, and socket communication are prepared for software developers. Additionally, secure data exchange mechanisms are implemented over HTTP. Therefore, they need not to aware the details of IP mobility and secure data exchange mechanisms to develop their applications.

A. Extension of NTMobile

We have extended NTMobile implementation to support the proposed communication. The followings are extended parts.

- Authentication mechanisms
  We introduce the account server that has authentication information. Therefore, all DCs are linked to AS and exchange authentication data in the login process.
- Distribution of authentication token
  Cache servers authenticate users by an authentication token. The authentication token is distributed by the account server that has authentication information. In the implementation, the account server pushes the authentication token via HTTP when the authentication data is updated.
- Transmission of notification
  The notification service such as APNS and GCM is important function for smartphones to reduce consumed power. In the proposed platform, we employ the notification service to notify information for realtime communication functions and non-realtime communication to users. In the implementation, direction coordinators collect information for notification service from clients, and transmit a notification request message to notification servers.
- Storing of temporal keys
  NTMobile has own secure encryption exchange mecha-

nisms. However, the proposed secure non-realtime communication mechanisms require storing temporal keys at direction coordinators. Therefore, direction coordinators have been extended to store temporal keys for non-realtime communication.

B. Non-realtime communication functions

In the proposed platform, we append the secure data exchange mechanisms to the extended NTMobile functions to achieve non-realtime communication functions. In the smartphone platforms, many useful APIs are prepared for developers. Therefore, we employ HTTP API for the non-realtime communication. In the implementation by HTTP, we inserted the following information.

- Version field
  The version field is the development version of NTMobile application module.
- Authentication token
  The authentication token is used to authenticate clients at HTTP servers. it is assumed to be exchanged in the authentication process between a client and its DC.
- Own FQDN
  In NTMobile, each client has a FQDN to identify client. The FQDN is assigned by its DC.
- Destination FQDNs
  The destination FQDNs is the FQDN list of destination clients.
- Application ID
  The application ID is used to recognize each smartphone application.
- Service ID
  The service ID is used to recognize each function in an application.
- Service item ID
  The service item ID is used to recognize each item in a function.
- Notification message
  The notification message is shown on smartphone device through the notification service such as GCM and APNS.

C. Evaluation

We have developed sample applications for iOS and Android OS. The application can achieve end-to-end communication by using UDP and exchange data by using HTTP. In the evaluation, we use iPhone 5 (iOS 6.1) and Galaxy Nexus (Android 4.2). These devices are connected to an IPv4 private network, an IPv4 global network, and an IPv6 global network through WiFi. In the experiments, we can confirm that the end-to-end communication is achieved in the all network connection patterns. Additionally, we can find that only a few second is enough to reestablish the end-to-end connection by IP mobility functions. From the results, we think our platform has practical performance for smartphone usages.

VI. Conclusion

We have proposed the secure end-to-end communication framework that can be installed by user authority for smartphone applications. As the realtime communication, we have extended NTMobile to support notification mechanisms in smartphone OS to reduce keep alive operations. Then, the extended mechanisms can provide secure data communication and IP mobility in IPv4 and IPv6 networks to smartphone applications. As the non-realtime communication, we have introduced cache servers into NTMobile to provide secure data exchange mechanisms. Our secure data exchange mechanisms provide end-to-end encryption of data for smartphone applications. In the implementation, we design NTMobile application module for iOS and Android systems to provide secure realtime and non-realtime communication. The developed application module can be installed with user authority in iOS and Android OS. As the results, applications can employ the developed application module to achieve secure IP mobility over smartphone applications. In the final manuscript, we will show the more detail evaluation results based on the voice communication functions over the end-to-end communication and chat functions over the secure data exchange mechanisms.

Acknowledgment

This work is supported in part by the SCOPE in 2013 by the Ministry of Internal Affairs and Communications and Grant-in-Aid for Young Scientists (B)(23700075), Japan Society for the Promotion of Science (JSPS).

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, Vol. 1, pp. 503-512, 2003.

[2] Q. Zhang, C. Guo, Z. Guo and W. Zhu, “Efficient mobility management for vertical handoff between WWAN and WLAN,” IEEE Communications Magazine, Vol. 41, No. 11, pp. 102-108, 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, Oct. 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, 2006.

[5] C. Perkins, “IP Mobility Support for IPv4, Revised,” RFC 5944, IETF (2010).

[6] M. Ishiyama, M. Kunishi, K. Uehara, H. Esaki, and F. Teraoka, “LINA: A New Approach to Mobility Support in Wide Area Networks,” IEICE Trans. Comm., Vol.E84-B, No.8, pp.2076– 2086, 2001.

[7] http://www.mip4.org, retrieved: February , 2012.

[8] S. Salsano, C. Mingardi, S. Niccolini, A. Polidoro and L. Veltri ”SIP-based Mobility Management in Next Generation Networks,” IEEE Wireless Communication, Vol. 15, Issue 2, April 2008.

[9] 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) 2009.

[10] 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, 2010.

[11] E. Fogelstroem, A. Jonsson, and C. Perkins, “Mobile IPv4 Regional Registration,” RFC 4857, June 2007.

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

[13] G. Montenegro, “Reverse Tunneling for Mobile IP, revised,” RFC 3024, January 2001.

[14] http://developer.android.com/google/gcm/

[15] http://developer.apple.com/library/mac/#documentation/NetworkingInternet/Conceptual/RemoteNotificationsPG/Introduction.html

[16] Kazuma Kamienoo, Hidekazu Suzuki, Katsuhiro Naito, Akira Watanabe, “Implementation and Evaluation of NTMobile with Android Smartphones in IPv4/IPv6 Networks,” IEEE Global Conference on Consumer Electronics (GCCE2012), pp.125 − 129, October 2012.

[17] Katsuhiro Naito, Kazuma Kamienoo, Takuya Nishio, Hidekazu Suzuki, Akira Watanabe, Kazuo Mori, Hideo Kobayashi, “Proposal of Seamless IP Mobility Schemes: Network Traversal with Mobility (NTMobile),” IEEE GLOBECOM 2012, December 2012.

[18] http://www.ntmobile.net