NDR: Name Directory Service in Mobile Ad-Hoc Network

Jae-Hoon Jeong, Jung-Soo Park, Hyoung-Jun Kim
Electronics and Telecommunications Research Institute
{paul,pjs,khj}@etri.re.kr

***Abstract* — This paper proposes the name service architecture called as name directory service (NDR) for the exchange of domain name, IP address and user information among the users of mobile nodes in mobile ad-hoc network. NDR provides DNS service for mobile user in the environment of mobile ad-hoc network where the current dedicated name server is difficult to use for DNS service. It also allows the mobile user to perceive other users in the range of communication and to communicate with the person with whom he or she wants to communicate by providing mobile user with user information (or profile) of the neighbors. It can also solve the address conflict that may be caused by the repetitive partition and mergence of ad-hoc networks. We suggest the design and service scenario of the NDR system.**

***Keywords* — name service, directory service, DNS, user information, autoconfiguration, mobile ad-hoc network, address conflict.**

1. Introduction

Mobile Ad-hoc Network (MANET) consists of mobile nodes that take part in routing so as to communicate with one another in the environment where there is no communication infrastructure [1][2]. Small Office Home Office (SOHO) networks, home-networks, and internal networks of transport vehicles such as airplane, train and bus can be constructed with ad-hoc networking. Up to recently ad-hoc routing protocols for unicasting, multicasting and flooding have been researched but the research of addressing, naming, name service and application in ad-hoc networks have been started over the whole world. In this paper, we focused on naming and name service in MANET, where the current dedicated DNS is impossible to adopt in order to provide name service for mobile nodes. Because MANET has dynamic network topology, the current DNS that has the hierarchical structure cannot be adopted in MANET. Recently, in IETF DNSEXT working group, the name service system for name service in ad-hoc network or home-network has been being developed [3]. The system is called as LLMNR (Link-Local Multicast Name Resolution) and the scope of name service of LLMNR is link-local [4]. It cannot be used to provide name service in multi-hop ad-hoc network yet. Name Directory Service (NDR) proposed in this paper not only provides name service in multi-hop ad-hoc networks, but also allows mobile user in mobile node to perceive the neighbors and their user information that is necessary to decide whether the neighbor is the man with whom the user wants to communicate. NDR can also solve the address conflict that is caused by repetitive partition and mergence of ad-hoc networks. For example, we assume that two mobile nodes of which each is placed in other partitioned ad-hoc network have the same address. Although the IP address of each mobile node is unique in each partitioned ad-hoc network, when two disconnected ad-hoc networks become merged by the movement of mobile nodes, the address conflict occurs. This address conflict problem cannot be detected and settled in the current IP (IPv4 or IPv6) protocol implementation. NDR can solve this address conflict.

We also suggest an autoconfiguration (zero-configuration) technology for generating a unique domain name related to each network device (or interface) of mobile node. The naming suggested in this paper is useful in ad-hoc network where there is no network manager to assign each user a unique domain name. Accordingly, we suggest a name service system called as NDR providing four name services for each mobile node like Figure 1; (a) Automatic name generation, (b) Name-to-address translation, (c) Collection of user information of neighbor nodes, and (d) Detection and settlement of address conflict.

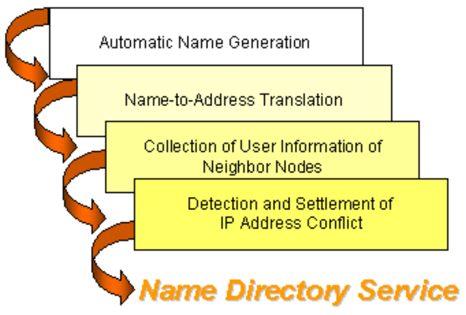

Automatic Name Generation
Name-to-Address Translation
Collection of User Information of Neighbor Nodes
Detection and Settlement of IP Address Conflict
Name Directory Service

Figure 1. NDR Service in each Mobile Node

The rest of the paper is structured as follows; In Section 2, we explain the related work. The organization and operation of the system for name directory service (NDR) are suggested in Section 3. The name services in NDR are also explained. In Section 4, the service scenario of NDR is described. Section 5 shows the comparison between NDR and LLMNR in respect of functionality. In Section 6, we conclude this paper and present future work.

2. Related Work

In this section, we explain LLMNR which provides the name service for nodes in link-local scoped network and the autoconfiguration technology which provides users with the automation of configuration and network services related to elemental networking.

2.1 Link-Local Multicast Name Resolution (LLMNR)

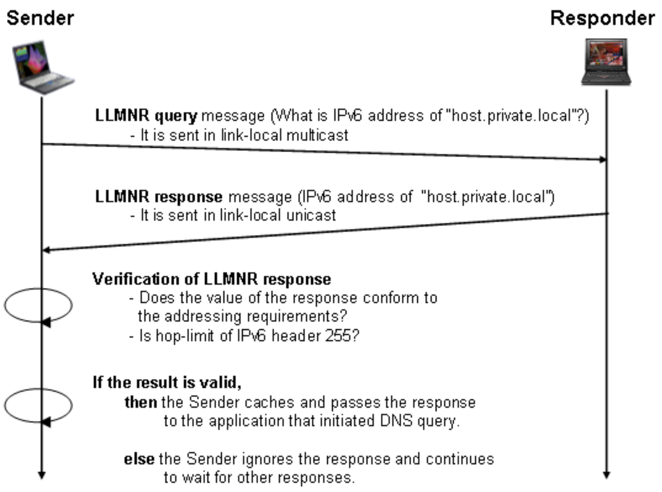

Sender
Responder
LLMNR query message (What is IPv6 address of "host.private.local"?)
- It is sent in link-local multicast
LLMNR response message (IPv6 address of "host.private.local")
- It is sent in link-local unicast
Verification of LLMNR response
- Does the value of the response conform to the addressing requirements?
- Is hop-limit of IPv6 header 255?
If the result is valid,
then the Sender caches and passes the response to the application that initiated DNS query.
else the Sender ignores the response and continues to wait for other responses.

Figure 2. Procedure of the Resolution from Domain Name to IPv6 Address through LLMNR

Link-Local Multicast Name Resolution (LLMNR) has been devised for the resolution between domain name and IP address in the link-local scoped network [4]. Figure 2 shows the procedure of the resolution from domain name to IPv6 address in a subnet through LLMNR. Sender is the resolver that sends LLMNR query in link-local multicast and Responder is the name server that sends the LLMNR response to Sender in unicast. When Sender receives the response, it verifies if the response is valid. If the response is valid, Sender stores it in LLMNR cache and passes the response to the application that initiated the DNS query. Otherwise, Sender ignores the response and continues to wait for other responses. Unless Sender receives any response during a limited amount of time (LLMNR_ TIMEOUT, 1 [sec]), it retransmits LLMNR query by 3 times in order to assure itself that the query has been received by a node which is capable of responding to the query.

2.2 Autoconfiguration Technology

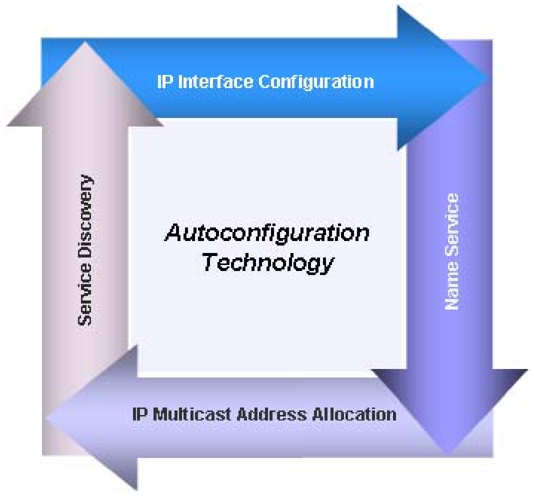

IP Interface Configuration
Name Service
IP Multicast Address Allocation
Service Discovery
Autoconfiguration Technology

Figure 3. Autoconfiguration Technology

IETF Zeroconf working group has defined the technology by which the configuration necessary for networking is performed automatically without manual administration or configuration in the environment, such as small office home office (SOHO) networks, airplane networks and home networks, as zero-configuration (autoconfiguration) [5]. Four main mechanisms regarding the autoconfiguration technology of Figure 3 are being researched; (a) IP interface configuration, (b) Name service (e.g., Translation between host name and IP address), (c) IP multicast address allocation, and (d) Service discovery [6].

3. Name Directory Service (NDR)

Like Figure 4, Name Directory Service (NDR) provides mobile users with not only the name service such as name-to-address translation in multi-hop ad-hoc network, but also the directory service allowing mobiles user to perceive the neighbors and to exchange user information that is necessary to decide whether the neighbor is the man with whom the user wants to communicate. Every mobile node runs NDR Daemon for name directory service like Figure 4. NDR Database contains user information for identifying each other and DNS resource records for name service. The application that needs name resolution can resolve name into IP address through NDR Daemon. NDR Daemon performs not only the function of DNS resolver, but also that of DNS name server.

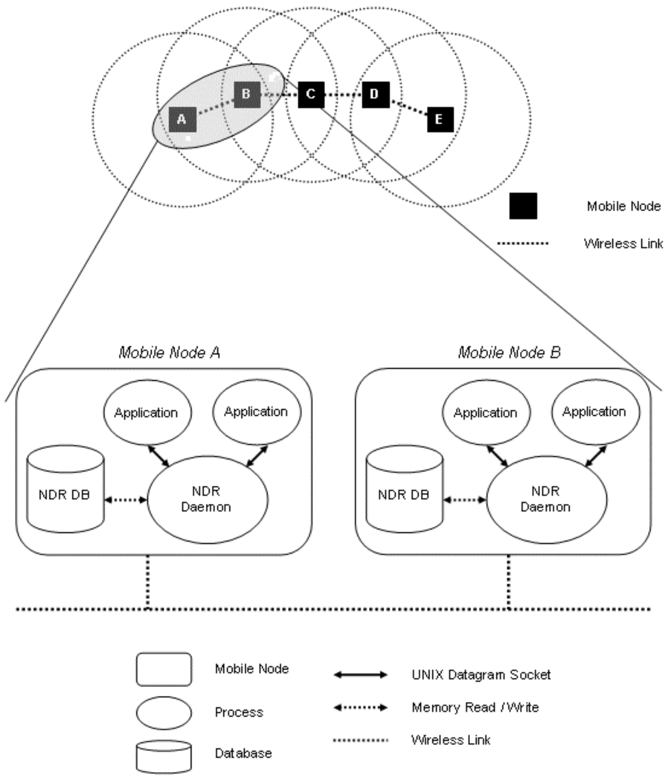


Figure 4. Name Service through NDR in MANET

Mobile Node A and Mobile Node B in Figure 4 can identify the user of the other and resolve the domain name of the other through its NDR Daemon.

3.1 Architecture of NDR System

NDR system consists of two processes so as to perform NDR service like Figure 5; (a) NDR Daemon and (b) NDR Shell.

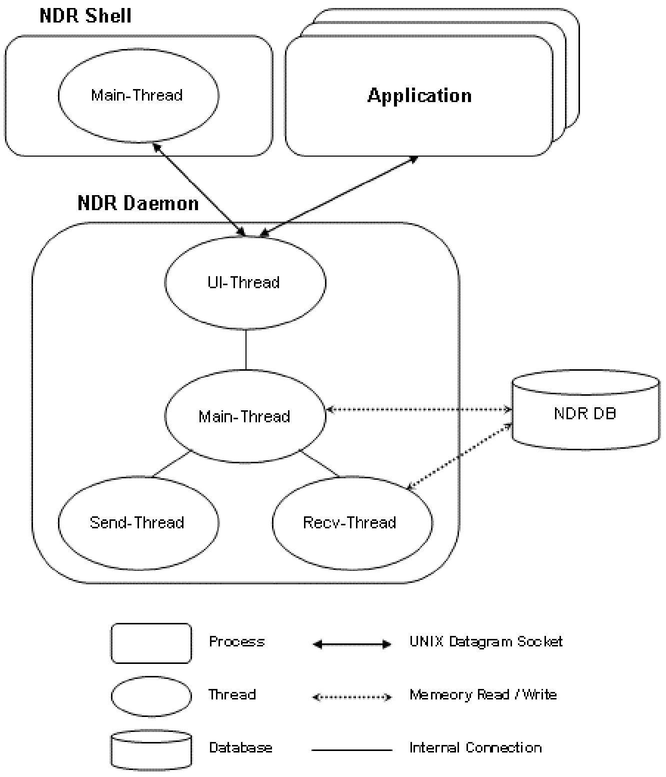


Figure 5. Architecture of NDR System

3.1.1 NDR Daemon

NDR Daemon processes the collection of user information of neighbor nodes, the name-to-address resolution, and the detection and settlement of IP address conflict. It consists of four threads; (a) Main-Thread, (b) Send-Thread, (c) Recv-Thread, and (d) UI-Thread.

(a) Main-Thread : It performs the initiation of NDR Daemon including the generation of unique domain name of mobile node and the registration of user information and domain name into NDR database. It also executes three worker threads (Send-Thread, Recv-Thread, and UI-Thread).

(b) Send-Thread : It generates an NDR message packet for name and directory service and sends it to the neighbor node(s). NDR messages consist of the request message and the response message of user information and DNS resource record.

(c) Recv-Thread : It receives an NDR message packet for name and directory, processes the message, and sends the response to the source node.

(d) UI-Thread : As user-interface thread, it receives a request related to NDR from NDR Shell, relays the request to Main-Thread that processes it and returns the result of the request received from Main-Thread to NDR Shell. Also, it performs the role of DNS resolver, which receives DNS query from an application in the same node and returns DNS response to the application like Figure 5

3.1.2 NDR Shell

NDR Shell is the user-interface process for a user who wants to get the information of neighbors. It receives a request from user, relays the request to UI-Thread of NDR Daemon through UNIX datagram socket. After NDR Daemon processing the request, it receives the result of UI-Thread, it displays the result to the user.

3.2 Procedure of NDR Service

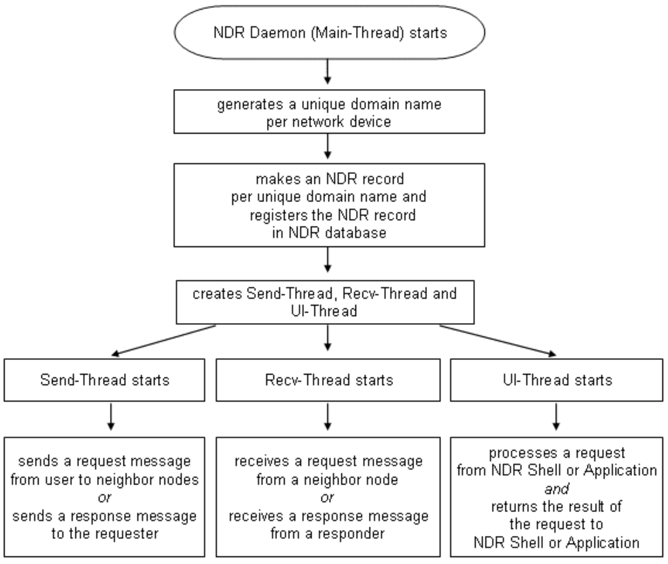

NDR Daemon (Main-Thread) starts
generates a unique domain name per network device
makes an NDR record per unique domain name and registers the NDR record in NDR database
creates Send-Thread, Recv-Thread and UI-Thread
Send-Thread starts
Recv-Thread starts
UI-Thread starts
sends a request message from user to neighbor nodes or sends a response message to the requester
receives a request message from a neighbor node or receives a response message from a responder
processes a request from NDR Shell or Application and returns the result of the request to NDR Shell or Application

Figure 6. Procedure of NDR Daemon's Operation

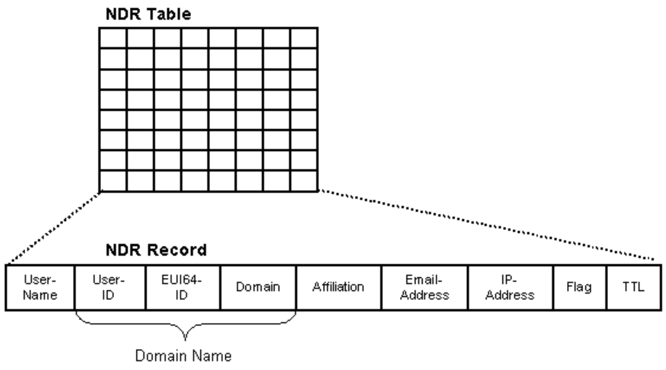
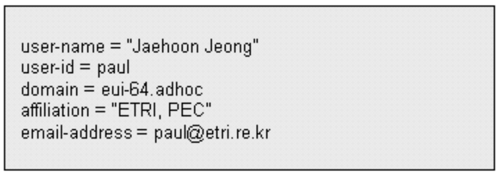


Figure 7. Structure of NDR Table and NDR Record



```
user-name = "Jaehoon Jeong"
user-id = paul
domain = eui-64.adhoc
affiliation = "ETRI, PEC"
email-address = paul@etri.re.kr
```

Figure 8. NDR Configuration File (NDR.conf)

The collection of user information and domain name is performed through NDR Daemon on demand, namely only when user initiates the collection of user information of neighbor nodes through NDR Shell. Figure 6 shows the procedure of NDR Daemon, which consists of five steps.

Step1 : NDR Daemon starts. NDR Daemon runs Main-Thread.

Step2 : Main-Thread generates a unique domain name per network device. Main-Thread makes a unique domain name based on network interface identifier which is unique over the whole world. We explain the mechanism of the name generation in detail in Section 3.3.1.

Step3 : Main-Thread makes an NDR record per unique domain name and registers the NDR record in NDR database. NDR record is a tuple in NDR table which forms NDR database and it contains user information (User Name, Affiliation, Email Address), DNS name information (Domain Name and IP Address), flag (Flag) indicating if the record belongs to this node itself or other node and time-to-live (TTL) that is the caching time of NDR record like Figure 7. Flag can have two values. One is "LOCAL" and the other is "REMOTE". The former indicates that the record belongs to this node itself and the latter indicates that the record belongs to other node. User information and domain of the node are stored in the configuration file called "NDR.conf" like Figure 8.

Step 4 : Main-Thread creates three worker threads; (a) Send-Thread, (b) Recv-Thread, and (c) UI-Thread.

Step 5 : Three worker threads run to perform their own duty.

(a) Send-Thread : It sends a request message received from user or application to neighbor nodes or sends a response message corresponding to a request message to the requester. NDR Daemon uses the multicasting in order to exchange information among one another. In case of IPv4, it uses an administrative scoped IPv4 multicast address, "239.255.1.251" for IPv4 NDR multicast group and in case of IPv6, it uses a site-local scoped IPv6 multicast address, "FF05:0:0:0:0:3::1" for IPv6 NDR multicast group. Send-Thread sends NDR request message to neighbor nodes in multicast and sends NDR response message to the requester in unicast.

(b) Recv-Thread : It receives a request messages from a neighbor node or receives a response message from a responder. Recv-Thread joins NDR multicast group in order to receive the multicast NDR message packets. When it receives a request message, it processes the message and delivers the response message to Main-Thread. Main-Thread delivers the message to Send-Thread, which sends the NDR message packet including the response message. When Recv-Thread receives a response message corresponding to the request message, it delivers the message to Main-Thread. Main-Thread stores each response from each responder in NDR database. After the collection of information, Main-Thread delivers the result to UI-Thread, which sends the result to NDR Shell or Application like Figure 5.

(c) UI-Thread : It receives a request from NDR Shell or Application and returns the result of the request to NDR Shell or Application.

3.3 Name Service through NDR

The name service is composed of name generation, name resolution (name-to-address translation), and the process of IP address conflict.

3.3.1 Name Generation

The mechanism of name generation makes a unique domain name with user-id, device-id (network device's address extended into EUI-64 identifier) and domain like Figure 9 [7].

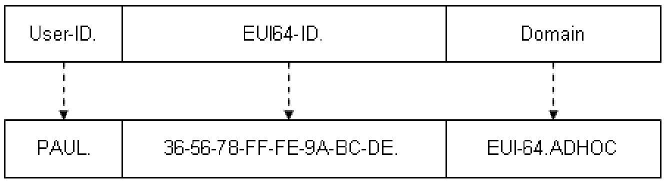


Figure 9. Generation of a Unique Domain Name

user-id is the user identifier selected by user and device-id is EUI-64 identifier derived from the network device's built-in 48-bit IEEE 802 address. domain should include "EUI-64" sub-domain which indicates that the domain name is based on EUI-64. We define the domain for ad-hoc network as EUI-64.ADHOC. For example, supposing that user-id is "PAUL", device-id is "36-56-78-FF-FE-9A-BC-DE", and domain is "EUI-64.ADHOC", a unique domain name would be "PAUL.36-56-78-FF-FE-9A-BC-DE.EUI-64.ADHOC". user-id and domain are registered in NDR.conf of Figure 8.

The merit of the above mechanism guarantees that no name conflict happens although users in other nodes use the same user-id without the procedure of verifying the uniqueness of domain name like dynamic update request of LLMNR [4].

3.3.2 Name Resolution

Each mobile node can resolve a domain name into IP address easily and fast because IP address corresponding to each domain name in NDR record is already listed in NDR table without the additional procedure to resolve name into address unlike the current DNS. The application that needs name resolution can resolve name into IP address through NDR Daemon like Figure 4. When the application sends a DNS query to NDR Daemon through the name resolution function related to NDR Daemon, NDR Daemon processes the query and returns the result to the application. Like this, NDR Daemon performs not only the function of DNS resolver, but also that of DNS name server in the environment of ad-hoc network where there is no DNS name server like LLMNR [4].

3.3.3 Detection and Settlement of IP Address Conflict

NDR can not only provide basic name service to mobile nodes in a connected ad-hoc network, but also detect and settle the address conflict when it perceive that the IP address of its own comes into conflict with that of other node. Because ad-hoc network has dynamic topology, the network is partitioned and merged on and on. The current IP mechanism can detect the address conflict after an IP address is configured in network device (or interface), but can not settle the conflict. As an example of address conflict on the ad-hoc network that uses on-demand ad-hoc routing protocol such as AODV and DSR [8][9], like Figure 10 (a), when these two MANETs (i.e., MANET 1 and MANET 2) that have a node with the same address in their own network (Mobile node A and E have the same address) are merged into one MANET (i.e., MANET3) like Figure 10 (b), one of these two nodes (i.e., Mobile node A and E) tries to communicate with a third-party node (e.g., Mobile node D) and sends a route discovery message, an address conflict happens. Because mobile node A, one of these two nodes, sends a route discovery message packet to start the route discovery like Figure 10 (b), the other node, mobile node E, can perceive that other node which has the same address as its own address has started the route discovery.

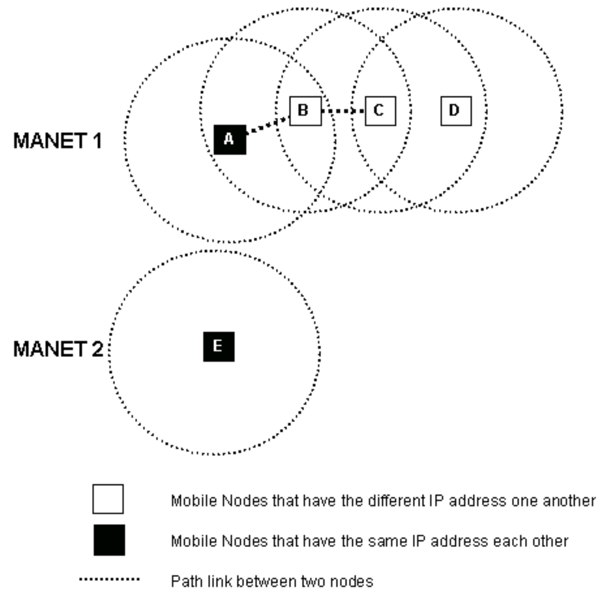

MANET 1
A
B
C
D
MANET 2
E
Mobile Nodes that have the different IP address one another
Mobile Nodes that have the same IP address each other
Path link between two nodes

Figure 10 (a). Two Partitioned Networks : MANET 1 and MANET 2

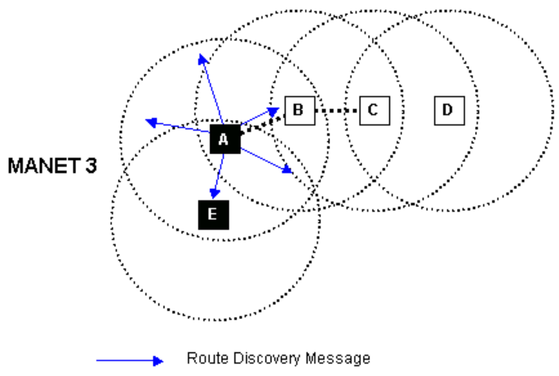


Figure 10 (b). One Merged Network : MANET 3

The current IP stack in ad-hoc network can perceive address conflict in the level of ad-hoc routing protocol implementation, but not settle the conflict, namely not substituting the conflicted address with a new address.

If an ad-hoc routing protocol implementation can detect the address conflict related to its own IP address, it asks NDR Daemon to settle the address conflict. Through ad-hoc stateless address autoconfiguration, NDR Daemon configures a new IP address in the network device where the address conflict happened [10][11]. Unless ad-hoc routing protocol can detect address conflict, NDR can detect the address conflict when NDR Daemon collects the information of neighbor nodes (i.e., user information and DNS name information). When NDR Daemon detects the address conflict, it initiates the reconfiguration of a new IP address for network device configured with the conflicted IP address through ad-hoc stateless address autoconfiguration [10][11].

According to the reconfiguration of IP address, NDR Daemon updates IP Address field of the NDR Record related to the conflicted address.

4. NDR Service Scenario

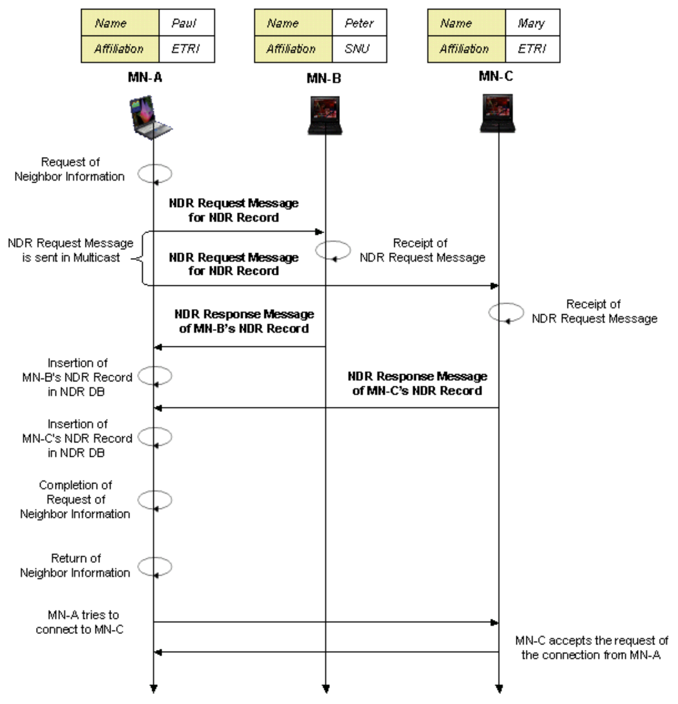

Name
Paul
Affiliation
ETRI
Name
Peter
Affiliation
SNU
Name
Mary
Affiliation
ETRI
MN-A
MN-B
MN-C
Request of Neighbor Information
NDR Request Message for NDR Record
NDR Request Message is sent in Multicast
NDR Request Message for NDR Record
Receipt of NDR Request Message
Receipt of NDR Request Message
NDR Response Message of MN-B's NDR Record
Insertion of MN-B's NDR Record in NDR DB
NDR Response Message of MN-C's NDR Record
Insertion of MN-C's NDR Record in NDR DB
Completion of Request of Neighbor Information
Return of Neighbor Information
MN-A tries to connect to MN-C
MN-C accepts the request of the connection from MN-A

Figure 11. NDR Service Scenario

Figure 11 shows the scenario of the communication among mobile nodes in ad-hoc network through NDR. Mobile nodes MN-A, MN-B and MN-C are joining NDR multicast group. For IPv4 NDR multicast group, an administrative scoped IPv4 multicast address “239.255.1.251” is used and for IPv6 NDR multicast group, a site-local scoped IPv6 multicast address “FF05:0:0:0:0:3::1” is used. In Figure 11, A user in MN-A whose name is “Paul” wants to know the neighbors in the same ad-hoc network. First of all, he issues a command that requests the collection of neighbor information through NDR

Shell. NDR Shell delivers user's request to NDR Daemon and NDR Daemon sends an NDR Request Message for NDR Record of neighbor node in multicast. When Mobile nodes MN-B and MN-C receive the request message, each sends its NDR Record through a NDR Response Message to the requester, MN-A. When MN-A receives a response message from neighbor, it stores the neighbor's NDR Record in its NDR DB. When the collection of neighbor information completes after a predefined collection time, NDR Daemon delivers the information of neighbor nodes to user "Paul" via NDR Shell. User "Paul" finds that the user of MN-C, "Mary", is whom he wants to communicate with and he tries to connect to MN-C. MN-C accepts the request of the connection from MN-A. Like this, mobile users can perceive the neighborhood and can communicate with the person with whom he or she wants to communicate.

5. Comparison between NDR and LLMNR

Table 1. Comparison between NDR and LLMNR

Functionality	NDR	LLMNR
Automatic name generation	yes	no
Need of verifying the uniqueness of domain name	no	yes
Perception of neighborhood	yes	no
Service scope	site-local or link-local	link-local
Detection and settlement of IP address conflict	yes	no
Service of name-to-address translation	yes	yes
Number of the request message packets to resolve the domain names of *n nodes*	O(1)	O(n)

Table 1 shows the comparison between NDR and LLMNR in respect of functionality. An important merit of NDR over LLMNR is that NDR needs one NDR request message packet to resolve the domain names of n neighbor nodes but LLMNR needs n DNS query message packets to do so. Like Table 1, NDR has more functions related to name service than LLMNR in that NDR includes autoconfiguration technology and uses less message packets to resolve domain name [6][7].

6. Conclusion

This paper suggested a name service scheme called name directory service (NDR) in mobile ad-hoc network where the current DNS is difficult to adopt so as to provide mobile users with name service. NDR provides users with not only the name service, but also the exchange of user information that is needed to identify the neighbor in a connected ad-hoc network. NDR also provides the mechanism to generate a unique domain name based network device identifier per network device without the network manager unlike the current DNS and the mechanism to detect and settle the address conflict.

For future work, we will enhance NDR to provide secure name service and simulate to evaluate the performance of NDR and LLMNR in ad-hoc network. Finally, we will implement the Secure NDR (SNDR) and autoconfiguration technologies related to name service in mobile ad-hoc network [12]-[14].

References

[1] Elizabeth M. Royer and Chai-Keong Toh, "A Review of Current Routing Protocols for Ad Hoc Mobile Wireless Networks", IEEE Personal Communications, April 1999.
[2] IETF Manet working group, http://www.ietf.org/html.charters/manet-charter.html
[3] IETF Dnsext working group, http://www.ietf.org/html.charters/dnsext-charter.html
[4] Levon Esibov and Dave Thaler, "Linklocal Multicast Name Resolution (LLMNR)", (work in progress) draft-ietf-dnsext-mdns-12, August 2002.
[5] IETF Zeroconf working group, http://www.ietf.org/html.charters/zeroconf-charter.html
[6] A. Williams, "Requirements for Automatic Configuration of IP Hosts", (work in progress) draft-ietf-zeroconf-reqts-12, September 2002.
[7] Jae-Hoon Jeong, Jung-Soo Park and Hyoung-Jun Kim, "Generation of Unique Domain Name based on Network Device Identifier", draft-jeong-name-generation-00, October 2002.
[8] Charles E. Perkins, Elizabeth M. Belding-Royer and Samir R. Das, "Ad hoc On-Demand Distance Vector (AODV) Routing", (work in progress) draft-ietf-manet-aodv-12, November 2002.
[9] David B. Johnson et al., "The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks (DSR)", (work in progress) draft-ietf-manet-dsr-07, February 2002.
[10] Charles E. Perkins et al., "IP Address Autoconfiguration for Ad Hoc Networks", draft-perkins-manet-autoconf-01.txt, Nov 2001.
[11] Stuart Cheshire, Bernard Aboba and Erik Guttman, "Dynamic Configuration of IPv4 Link-Local Addresses", (work in progress) draft-ietf-zeroconf-ipv4-linklocal-07, February 2003.
[12] ETRI Project for Research and Implementation of IPv6-based Ad-hoc Autoconfiguration Technology, http://www.adhoc.6ants.net
[13] Jaehoon Jeong and Jungsoo Park, "Autoconfiguration Technologies for IPv6 Multicast Service in Mobile Ad-hoc Networks", 10th IEEE International Conference on Networks, Aug. 2002.
[14] Jaehoon Jeong and Jungsoo Park, "Autoconfiguration Technology for IPv6-based Mobile Ad-hoc Network", ICIS'02, Aug. 2002.