IPv6 OLSR Implementation on Zebra Platform

Kouji Okada
okada@sfc.wide.ad.jp

Ryuji Wakikawa
ryuji@sfc.wide.ad.jp

Jun Murai
jun@sfc.wide.ad.jp

Graduate School of Media and Governance, Keio University
Faculty of Environmental Information, Keio University
Keio University Shonan Fujisawa Campus
5322 Endo, Fujisawa Kanagawa, 252-8520 Japan

Abstract

Implementations of MANET routing protocols are now released from many organizations and MANET is now feasible in the real situation. To put MANET to practical use, it is necessary to consider some issues such as the cooperation between MANET and the Internet and the combination of several MANET routing protocols. In this research, we implemented IPv6 OLSR on GNU Zebra, an integrated routing package, so that OLSR interacts with other routing protocols. This implementation also includes an approach to obtain global routable addresses for MANET routers.

1. Introduction

While we have implemented OLSR for IPv6, we encounter several issues to implement IPv6 version of OLSR. This paper summarizes all the problems and explains how we solve the issue in our implementation. In general, routing protocols disregard differences between IPv4 and IPv6, and suggest both can be supported if routing messages for both protocols are defined. However, there are some differences between IPv4 and IPv6 which may affect how the routing protocols should work with IPv6. Addressing in MANET is a problem for both IPv4 and IPv6. When MANET connects to the global Internet, IPv4 might do NAT, while global addresses might be suitable for IPv6.

2. OLSR for IPv6

In order to implement OLSR for IPv6, we need several modifications to the specification. The original specification[1] is designed for IPv4 and must be updated to support IPv6.

- 128 bit Address length

  OLSR must carry IPv6 addresses instead of IPv4 (32bit). The address field of all the signal messages such as HELLO, TC, HNA and MID is expanded to 128 bits.

- Limited Broadcast Address
  In IPv4, 255.255.255.255 is used to flood control messages to entire MANET. It is obvious that we cannot use this address in IPv6 network and decided to use all node multicast address (ff02::1) for IPv6 MANET although there is a discussion about the MANET link-local multicast address[2]. The all node multicast address works like the IPv4 limited broadcast address and is used for many operations of IPv6 such as NDP. Since the scope of "ff02::1" is the link local address, it is used only for communications with 1-hop neighbors. However, if the flooded packet is used to setup a route for the sender node (i.e. reverse route), it should use the address except for link-local address as a source address. This is because routes for link local addresses are prohibited on multi-hop networks.

- Address Scope
  IPv6 has a notion of "scope" to control validity of each IPv6 address. In a MANET, we designed that a link-local scope is used to identify a physical link and each MANET node is identified with the unique-local scope while multi-hop communications in the MANET. Each MANET node generates a link local address according to the IPv6 specification[3] and also obtains a unique local address from the unique local prefix of the network. Communications between nodes inside of a MANET should be done with the unique local scope addresses, and each MANET node acquires a global unicast address for global communication.

- HNA message
  IPv6 address uses prefix length to identify network part of an IPv6 address. Therefore, we dismiss the netmask

field and add a prefix length field for IPv6. The modified HNA message is described in Figure 1.

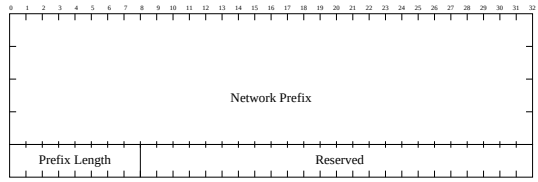

Figure 1. Modified HNA message

3. MANET Global Connectivity

According to [4], an internet gateway can advertise a global prefix to an entire MANET. [4] recommends to distribute a global prefix by either sending modified router advertisement or sending a new defined message of OLSR. In our research, a new OLSR message named Router Advertisement (RA) message is defined to configure a global routable addresses. The TLV based OLSRv2 message format described in [4] is modified into an OLSRv1 message format shown in Figure 2. The message type 5 is assigned for this RA message.

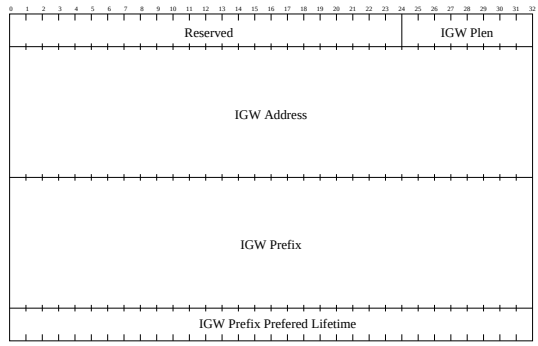

Figure 2. Internet Gateway Advertisement Message Format

An internet gateway in OLSR network advertises this RA message as one of OLSR signalings. In IPv6, a node generates a global IPv6 address from the received global prefix when it receives a router advertisement message of NDP. Likewise, a receiver of OLSR RA message can generate a global IPv6 address from the global prefix advertised by the internet gateway. In our implementation, the internet gateway is configured to disseminate this RA message every 5 seconds.

4. Zebra OLSR

In our system, multiple routing protocols can be run on a machine and manage a routing table. The advantages of using multiple routing protocols is to reduce a number of routing entries for a network. For example, OLSR maintains routing entries for nodes inside 5-hop distance, and DYMO proactively discovers a node which hop count is more than 5 hops. Thus, a total number of routing signal overhead is expected to be reduced. Alternatively, if an internet gateway receives a route of a mobile prefix carrying by a MANET node, it can advertise the mobile prefix to the Internet by using IGP. To do so, multiple routing protocols must be able to run simultaneously on an internet gateway.

GNU Zebra[5] is a routing stack package for UNIX such as Linux, FreeBSD, NetBSD, OpenBSD and MAC OS X. It supports BGP4, RIPv1, RIPv2 and OSPFv2, and is capable of running multiple routing protocols simultaneously. Zebra is freely distributed under the open source license (GPL). Since GNU Zebra also provides the software development kit, several newly developed routing protocols are implemented on the Zebra. GNU Zebra consists of a zebra daemon and each routing daemon. The zebra daemon manages all the operations between kernel and each routing daemons, described in Figure 3.

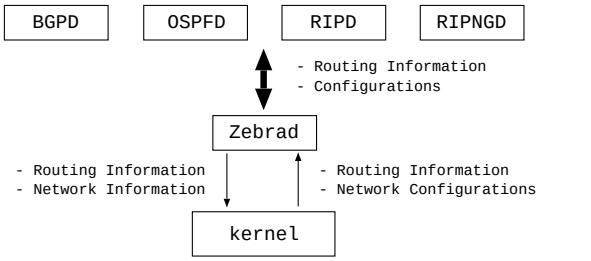

Figure 3. Overview of Zebra

The functionalities provided by GNU Zebra are listed below.

- The Interface to the Routing Table in the Kernel

  When a routing daemon needs to manipulate a routing table, the zebra daemon receives requests from the routing daemon and updates it as it requested. Furthermore, the zebra daemon notifies the event of how the routing table is modified to all the running routing daemons. The zebra daemon also notifies the changes of routing table which is done by third person (non ZEBRA related daemons or static routing) and status changes of each network interface such as address additions or address changes.

- Route Re-Distribution

In GNU Zebra, each routing daemon exchanges route information each other, the zebra daemon re-distribute routes if necessary. For example, the DYMO daemon utilizes route information acquired by the OLSR daemon. The Internet gateway advertises OSPF route information (i.e. route information of the Internet) to an OLSR network by re-distributing OSPF routes.

- Routing Protocol Development Kit

  GNU Zebra provides consistent API for manipulating a routing table. Therefore, each routing daemon can co-operate each other through the API.

  Moreover, GNU Zebra provides a user interface to each daemon based on telnet user interface. This user interface is similar to one's of Internet Operating System (IOS) developed by Cisco systems. It implies that network operators are familiar with the user interface of GNU Zebra.

5. Architecture

5.1. General Architecture

We have implemented IPv6 based OLSR and Global Connectivity on GNU Zebra 0.95-pre2 with C language. The OLSR implementation supports all the fundamental functions of OLSR such as HELLO message exchanges and the routing message flooding. In addition to those functions, it supports the global connectivity management that is internet gateway discovery and IP address acquisition. A MANET node running this OLSR implementation can access to the Internet if an internet gateway is available nearby.

5.2. Thread

GNU Zebra library includes thread management methods. We realized multi-processing functionality on the implementation with GNU Zebra library. There are two types of methods in the OLSR daemon, periodic methods which is called periodically like the hello message send method, and the event-driven read method which reads data from the socket for OLSR control messages. GNU Zebra thread management system sort the periodic methods in the right order and put them into the thread list. The OLSR daemon fetches the methods in the list and calls the head method in the list at the right timing. When called from the OLSR daemon, periodic methods first register the wait interval for the next timing, and generate a OLSR control message. The OLSR daemon calls the read method when the data arrive at the receiving socket. The read method calls the packet parser to retrieve the messages contained in the packet, then appropriate message processors are called from the packet parser.

5.3. Command Line Interface

Our implementation has a command line interface written with the Zebra library. The commands on the Command Line Interface (CLI) is categorized into two groups, the "display" commands and the "configure" commands.

Display commands show the status of the OLSR daemon, and the are implemented as subcommands of "show" commands as shown in Figure 4. The "neighbor" command shows main addresses and Willingness values of each neighbor and the status of the neighbor, whether the neighbor is a symmetric neighbor or not. Likewise, other commands list the OLSR statistics information.

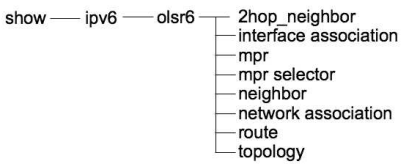

Figure 4. olsr6d Command Line Interface (show)

Configure commands are consists of two subcommands groups, the "router" subcommands and the "interface" subcommands as shown in Figure 5. The "router olsr6" commands is to configure the behavior of the OLSR daemon. Each interval commands is set to configure the send interval of correspondent messages by the second. The "interface" command requires the name of a interface to configure. The "olsr6" command enables to exchange OLSR routing messages on the interfaces. The default value for "olsr6" command is "disable".

An example of the configuration file is shown in Figure 6. The default configuration values are hidden in the file. Among three network interfaces, the OLSR is running on the Prism II wireless interface. The hello message interval is set to 1 second from default value 2. And the router is advertising network prefix 2001:200:1::/64 by HNA messages.

6. Implementation Restriction

There are several restrictions on our OLSR implementation. The first one is GNU Zebra problem that GNU Zebra thread scheduler manage thread timing only on the second time scale. However, there are requirements to signal routing messages on milli-second time scale like vehicle safety drive support systems. To realize quick topology conver-

```
configure —— terminal

        router —— olsr6
                  hello-interval <seconds>
                  hna-interval <seconds>
                  mid-interval <seconds>
                  tc-interval <seconds>
                  prefix <Prefix> (<Plen>)
                  willingness <willingness>
                  global6 mode <gateway|client>

      interface —— <Interface Name>
                  olsr6  <enable|disable>
                  global6 advertisement <enable|disable>
```

Figure 5. olsr6d Command Line Interface (configure)

gence, the precise messaging scheduler which can run on milli-second time scale.

The next problem is concerned with Neighbor Cache with asymmetric links. In case there are asymmetric links with non link-local scope addresses, unique local addresses or global addresses, the nodes which receives hello messages from the nodes located on the other side sends the Neighbor Solicitation to the hello message sender. The node which receives Neighbor Solicitation packets creates a neighbor cache entry of the sender according to the IPv6 specification even when the routing protocol decides that the route for the destination is 2 hop node. BSD systems transfer data packets according to the neighbor cache with unavailable links rather than available host routes in the routing table, and it causes the packet losses. We are now investigating the detail of the issue. Some architecture to configure the preference between the neighbor cache and the routing table is necessary.

7. Conclusion

In this paper, we introduces our OLSR implementation for IPv6 on the GNU Zebra software. OLSR is modified to carry IPv6 host and network routes. This implementation also contains the capability of Internet access by using Internet Gateway. A new OLSR message called RA message is defined for dissemination of Internet route (i.e. default route "::"). The implementation information is also available in the paper. GNU Zebra provides several API and user interfaces to develop new routing protocols. Only few documentations are available from GNU Zebra, this paper may complement how to develop the routing protocols on GNU Zebra.

```
!
! Zebra configuration saved from vty
! 2006/07/25 05:23:33
!
hostname olsr6d@localhost
password 8 XXXX
enable password XXXX
log file olsr6d.log
log stdout
service advanced-vty
!
!
interface fxp0
!
interface lo0
!
interface wi0
 olsr6 enable
!
!
router olsr6
 hello-interval 1
 prefix 2001:200:1:: 64
!
line vty
!
```

Figure 6. olsr6d Configuration Example

References

[1] T. Clausen and P. Jacquet. Optimized Link State Routing Protocol OLSR. Request for Comments (Experimental) 3561, Internet Engineering Task Force, October 2003.

[2] I. Chakeres MANET IANA Needs (work in progress, draft-chakeres-manet-iana-01). Internet Draft, Internet Engineering Task Force, September 18, 2006

[3] T. Narten, E. Nordmark, and W. Simpson Neighbor Discovery for IP Version 6 (IPv6) Request for Comments (Draft Standard), Internet Engineering Task Force, December 1998.

[4] R. Wakikawa, J. Malinen, C. Perkins, A. Nilsson, and A. Tuominen. Global Connectivity for IPv6 Mobile Ad Hoc Networks (work in progress, draft-wakikawa-manet-globalv6-05). Internet Draft, Internet Engineering Task Force, March 2005.

[5] Zebra Project Home Page http://www.zebra.org/