A Study on the Lightweight Mobile IPv6 Protocol Based on Fixed Terminal Node

Yan Pan

College of Information Science and Technology, Bohai University, Jinzhou, China

panyan8011@foxmail.com

Keywords: IPv6; Mobility; Fixed terminal node; Mobile node; Binding

Abstract. Mobile ubiquitous technology and IPv6 technology makes it possible to perform the remote control and information transfer by using the mobile terminal, and it promotes the development of the Internet of things. Therefore, more and more intelligent terminal devices need mobility support, such as intelligent monitoring, smart home and other small smart devices, they do not need to move away from the home link, but they need to communicate with the mobile node, in other words, these devices don't need to support the functions of mobile node. So the fixed terminal node which uses the current complex mobile IPv6 protocol wastes resources and increases costs. Base on the analysis of the the communication process between the terminal smart devices and the mobile node, this paper implemented a lightweight mobile IPv6 protocol on the IPv6 protocol stack of linux, and through the test of the communication between the fixed terminal node and mobile node, the lightweight mobile IPv6 protocol in line with the basic requirements of mobile IPv6 protocol specification RFC 6275.

Introduction

The large-scale commercial deployment in the world of the next generation Internet Protocol IPv6 [1], and the ubiquitous mobile communication network, provides the environment and technical support for the development of the Internet of things [2,3]. The small fixed-terminal smart devices with full functions, low cost and high speed will also be used widely, and their demand for network interconnection is growing day by day. Such as smart home [4,5] equipment, intelligent monitoring equipment, etc. People can use mobile phone and other mobile terminal equipment for remote control. The small fixed terminal devices do not need to support the mobile node and home agent functions, but need to communicate with the mobile terminal. Therefore, the mobile IPv6 protocol [6], which has been implemented to support the mobile node and home agent is obviously a waste of resources. Therefore, it is great practical significance to design and realize the light mobile IPv6 protocol by analyzing the communication process between the fixed terminal node and the mobile node.

Analysis of the Communication Process of Fixed Terminal Node and Modules Design

In order to implement the mobile IPv6 protocol based on fixed terminal, the key is the mobility support implementation of the fixed terminal node, that is to implement the communication process between the fixed terminal node and the mobile node.

The communication process between the fixed terminal node and the mobile node(MN) is mainly have five processes. First, receiving packets from MN, which mainly includes receiving data packet with the home address option(HAO) and processing mobility headers. Second, return routability procedure(RRp), which includes the process of receiving home test init message(HoTI) and care-of test init message(CoTI), as well as the process of sending home test message(HoT) and care-of test message(CoT). Third, processing bindings, which includes receiving binding update(BU), sending binding acknowledgement(BA), sending binding refresh request(BRR) and sending binding error(BE) message. Fourth, sending data packets to MN, which mainly includes examining binding cache for an entry for the destination address, and using a type 2 routing header(RT2) to route the packet to the mobile node(the destination node) by way of its care-of address. Fifth, receiving the ICMPv6[7]

(Internet Control Managemet Protocol version 6) error messages, which mainly includes receving and processing the destination unreachable messages.

Through analyzing the communication between the fixed terminal node and MN, it is necessary to add the modification to the extension header module and the ICMPv6 module base on the IPv6 protocol stack, and design two new modules of processing mobility headers and processing binding cache. The modules shown as Fig. 1.

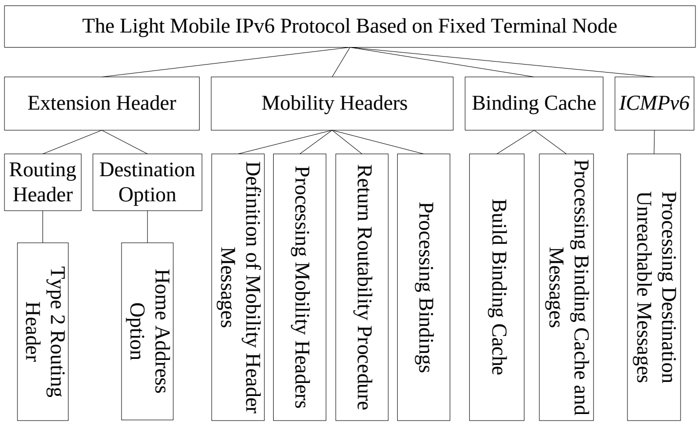
The Light Mobile IPv6 Protocol Based on Fixed Terminal Node
Extension Header
Mobility Headers
Binding Cache
ICMPv6
Routing Header
Destination Option
Type 2 Routing Header
Home Address Option
Definition of Mobility Header Messages
Processing Mobility Headers
Return Routability Procedure
Processing Bindings
Build Binding Cache
Processing Binding Cache and Messages
Processing Destination Unreachable Messages

Figure 1. Modules of the lightweight mobile IPv6 protocol based on fixed terminal node

The Lightweight Mobile IPv6 Protocol Based on Fixed Terminal Node

The lightweight mobile IPv6 protocol base on the special fixed terminal node is described as four parts, processing extension header, processing mobility headers, processing binding cache, and processing ICMPv6 messages[8].

Extension Header. In order to communicate with the mobile node, the mobile IPv6 protocol base on the fixed terminal node depends on extending the extension header to solve the triangle routing problem and the ingress filtering problem of mobile IPv4[9]. In the IPv6 protocol stack extension header, the home address option (HAO) and type 2 routing header option(RT2) are defined.

Once the packet with the HAO which is carried by the destination option extension header(next header value of 60) is received from MN, the fixed terminal node knows that the mobile node is away from home, the source address of the packet is the care-of address of MN and the address in the home address option is the home address of MN. Then the fixed terminal node can send packets with the RT2 to the care-of address of MN directly, while the home address is carried by RT2.

Mobility Header. The Mobility Header is a new extension header of IPv6, which is identified by a next header value of 135. It contains the message types that may be sent using the mobility header.

(1) Defination of mobility header messages

This module defined the contents of mobility header and mobility options, binding refresh request message(BRR), home test init message(HoTI), care-of test init message(CoTI), home test message(HoT), care-of test message(CoT), binding update message(BU), binding acknowledgement message(BA), binding error message(BE), all the messages related to the creation and management of bindings.

(2) Processing mobility headers

A series of operations will be performed,when the fixed terminal node received a message with the mobility header from the mobile node. First, verify the checksum and header length, otherwise, the fixed terminal node discard the message. Second, check the next header field, which value must be 59(no next header), otherwise, the fixed terminal node must discard the message and send message with code 0 (head error) of ICMPv6 parameter error to the source address of the message which is received. Third, the mobility headers type field must legal, otherwise, the fixed terminal node must discard the message and issue a BE message with status field set to 2 (unrecognized mobility headers type value). Forth, according to the mobility headers type field code, turned to the binding update processing or address test init processing.

(3) Return routability procedure

when the message from the fixed terminal node forwarded by the home agent is received, the mobile node will send home test init message and care-of test init message immediately. After receiving the messages, the fixed terminal node generates keygen token for the home test message and care-of test message which will be sent in response to the test init message of the mobile node.

(4) Processing bindings

When received a BU message from MN, the fixed terminal node needs to check the validity of the BU message and verify the type of BU, and then send the message to respons to the BU.

After checking the validity of the mobility header, First, confirm the address either in the home address option or in the source address is a unicast routable home address, and check the validity of the home address, otherwise discard the message. Second, check whether there is a binding cache entry in the binding cache, if not, create a new entry in its binding cache for this home address, otherwise, check the consistency between home registration bit of the BU and home registration flag of binding cache entry, if not, the fixed terminal node will send BA message with status field set to 139 (registration type change disallowed). Third, confirm the sequence number in the BU is greater than it in binding cache entry for the home address, if not, the fixed terminal node will send BA message, with status field set to 135 (sequence number out of window). Fourth, verify the message type, if it is a requests to cache a binding, the fixed terminal node will update binding cache entry with the lifetime field specified in the BU for this home address, but if it is a requests to delete a binding, a binding cache entry for this home address will be deleted. Fifth, after creation or update an binding entry, if the acknowledge(A) bit is set by the sending MN to request a BA, the fixed terminal node will send a BA which the status field is set to a value less than 128. Otherwise, if the fixed terminal node rejects the BU, even the acknowledge(A) bit is not set by the sending MN, it will send a BA which the status field is set to a value greater than or equal to 128.

When the fixed terminal node knows that the binding cache entry, which the lifetime of the MN is almost due, is still in active use, it will send a BRR message to MN in an attempt to extend lifetime. The maximum number of the BRR message that the fixed terminal node send is 3, and the interval between the two BRR is 1 second. But, once the address test init message is received from MN, the fixed terminal node will stop sending BRR message.

Binding Cache. The fixed terminal node maintains a binding cache entry for MN of the communication to support route optimization. So the definition of the binding cache structure directly affects the execution of the protocol and the occupation of the space.

(1)Structure of binding cache

Each binding cache entry contains the following fields, as shown in Table 1.

Table 1 Binding cache entry fields

Fields	Description
home_addr	the home address of the mobile node
coa	the care-of address for the mobile node
callback_time	lifetime
br_callback_time	time sending BRR
seq	maximum value of the sequence number
last_br	the last time of sending BRR
last_destunr	the last time of receiving destination unreachable messages
br_count	the number of sending BRR
destunr_count	the number of receiving destination unreachable messages

Considering the time complexity of the query, insert and delete operations of the binding cache, the structure of the binding cache is determined by the structure of the circular doubly linked list which is provided by Linux kernel and hash table with buckets[10].

In the binding cache structure which is shown as Fig. 2, two circular doubly linked list structures are defined, one is the sorted list which is used for sorting each binding cache entry by the value of the lifetime field, the first node is the recent binding cache entry to expire in this list, so that the purpose of the implementation is to facilitate processing of expired binding entries. When the lifetime of a binding cache entry is updated, the doubly linked list deletes this binding cache entry very flexible, and inserts it into the appropriate position of the sorted list. The other is the hash table which is a fast lookup table, it defines a array with N elements, each array element contains a hash bucket with a list head of doubly linked list. The hash table is used for linking binding cache entries according to the hash function with the key of home address. The hash function is as follows:

$$H(home\_addr)=home\_addr \ \ MOD \ \ N. \quad (1)$$

Each mobile node is always identified by its home address on the home link, so in the Eq. 1, home_addr(home address) is selected as the key of the binding cache entries. N is the number of hash buckets, which value is taken as 2^n ($0<n\leqslant10$). So the operation can be simplified to take the last n bits of the home_addr, which reduced the computation and improved the operation speed.

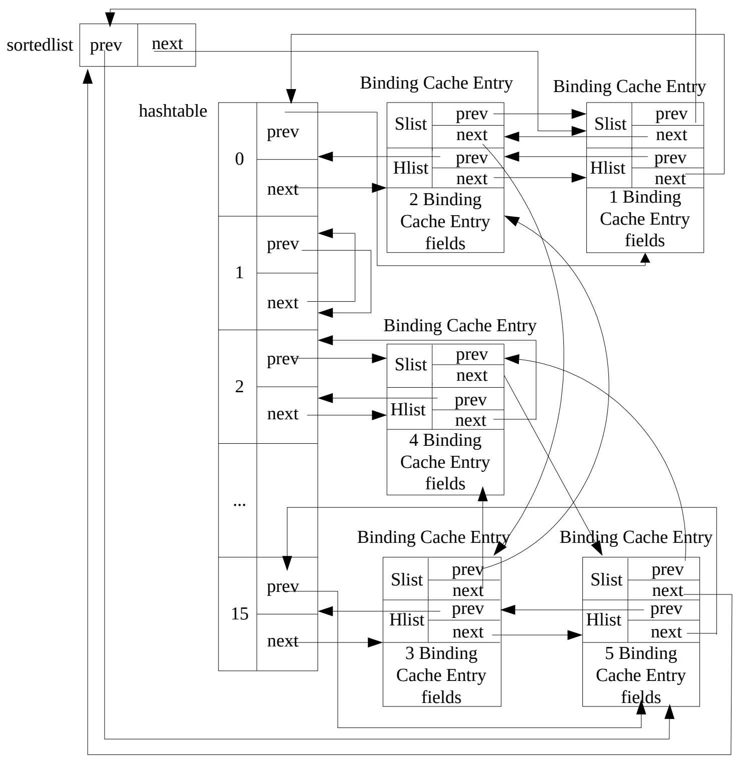

Figure 2. Hash table and sorted list structure of binding cache

(2)Operations of binding cache

The operations of binding cache mainly include: initialize a binding cache structure, add a binding cache entry, delete a binding cache entry, query a binding cache entry, process the expiration time function, and exit binding cache.

ICMPv6 Messages. The ICMPv6 processing module for mobile IPv6 is mainly implemented by modifying the ICMP protocol of the IPv6 protocol stack partially and adding special handling to destination unreachable messages. When the icmpv6_rcv function receives a message with the type of destination unreachable, it goes to the icmpv6_dest_unreach() function for handling, if it receives more than 5 destination unreachable ICMP error messages for a destination in its binding cache persistently, it will go to the function of delete a binding cache entry.

Conclusions

By analyzing the communication process between the mobile node and the fixed terminal node, the lightweight mobile IPv6 protocol is designed and implemented, which has strict requirements on resource. Through the test of the return routability procedure, the binding process and the communication after binding, the lightweight mobile IPv6 protocol base on the fixed terminal node in line with the basic requirements of mobile IPv6 protocol specification RFC 6275.

Acknowledgements

This work is partially supported by LiaoNing Provincial Federation Social Science Circle under Grant Lslgslhl-020, LiaoNing Planning office of philosophy and social science under Grant L14AGL002.

References

[1] Deering, S. and R. Hinden, Internet Protocol version 6(IPv6) Specification, RFC 2460, 1998, 1860–1864.

[2] A.J. Jara, D. Fernandez, P. Lopez, M.A. Zamora, and A.F. Skarmeta: Lightweight MIPv6 with IPSec support Mobile Information Systems, vol. 10, no. 1, 2014, 37-77

[3] A. Shukla and P. Yadav: International Journal on Recent and Innovation Trends in Computing and Communication, Volume: 1, Issue: 4, 2013, 335-3382

[4] R.J. Conejar, R. Jung and H.K. Kim: International Journal of Smart Home, 10(10), 2016, 283-29

[5] R.D. Caytiles and B. Park: International Journal of Smart Home Vol. 6, No. 1, 2012, 25-36

[6] E.C. Perkins, D. Johnson and J. Arkko: Mobility Support in IPv6 Specification, RFC 6275, 2011

[7] A. Conta and S. Deering: Internet Control Message Protocol (ICMPv6) for the Internet Protocol version 6 (IPv6) specification RFC 4443, 2006

[8] J. Davies, Understanding IPv6(Third Edition), Microsoft Press Corp., U.S., 2012, 552-577

[9] S.K. Hussein and K.M. Almustafa, Journal of Communication and Computer 10 (2013) 1554-1565

[10] R. Love, Linux Kernel Development (Third Edition), Pearson Education, U.S., 2010, 85-111