Unified L2 Abstractions for L3-Driven Fast Handover
draft-koki-mobopts-l2-abstractions-04.txt

Status of this Memo

By submitting this Internet-Draft, each author represents that any applicable patent or other IPR claims of which he or she is aware have been or will be disclosed, and any of which he or she becomes aware will be disclosed, in accordance with Section 6 of BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet-Draft will expire on November 5, 2006.

Copyright Notice

Copyright (C) The Internet Society (2006).

Abstract

This draft proposes unified L2 abstractions for L3-driven fast handover. For efficient network communication, it is vital for a protocol layer to know or utilize other layer’s information such as L2 triggers. However, in current operating systems, each protocol layer is designed independently and information exchange between
protocol layers is not allowed. To solve the problem, this draft defines nine kinds of L2 abstractions in the form of "Primitive". This concept can be applied to L3-driven fast handover mechanism using "Primitives".
# Table of Contents

1. Introduction .............................................. 5
2. Terminology ................................................. 6
3. Primitives for L2 Abstractions .............................. 7
4. Definitions of Primitives .................................... 9
   4.1. L2-LinkStatus (Type 1) ................................. 9
   4.2. L2-PeerList (Type 1) .................................. 9
   4.3. L2-PeerFound (Type 2) ................................. 9
   4.4. L2-PeerLost (Type 2) .................................. 9
   4.5. L2-LinkUp (Type 2) .................................. 10
   4.6. L2-LinkDown (Type 2) .................................. 10
   4.7. L2-LinkGoingDown (Type 2) ............................. 10
   4.8. L2-LinkConnect (Type 3) ............................... 10
   4.9. L2-LinkDisconnect (Type 3) ............................ 10
5. Definitions of Static Parameters ............................ 11
   5.1. Network Interface ID ................................. 11
6. Definitions of Dynamic Parameters ........................... 12
   6.1. Peer ................................................ 12
   6.2. Condition ........................................... 12
   6.3. Peer List ............................................ 12
   6.4. Enable/Disable ...................................... 12
   6.5. Ack/Error .......................................... 12
7. Architectural Considerations ................................. 13
8. Security Considerations ..................................... 17
9. References .................................................. 17

**Appendix A.** Example Scenario ................................. 19

**Appendix B.** Example Operation for FMIPv6 .................... 21
   B.1. Example Operation-1 for FMIPv6 ....................... 21
   B.2. Example Operation-2 for FMIPv6 ....................... 23

**Appendix C.** Example Mapping of Primitives and IEEE802.11 .... 26
   C.1. L2-LinkStatus ....................................... 26
   C.2. L2-PeerList ......................................... 26
   C.3. L2-PeerFound ........................................ 26
   C.4. L2-PeerLost ......................................... 27
   C.5. L2-LinkUp ........................................... 27
   C.6. L2-LinkDown ......................................... 27
1. Introduction

In recent years, the environment around computers is not static and changes dynamically. Especially, when a mobile node moves to a different network, its environment considerably changes. For example, in the case of wireless communication, parameters such as radio strength changes widely depends on time or site.

For efficient network communication, it is vital for a protocol layer to know or utilize other layer’s information. There are several proposals for seamless handovers in IPv6 network such as Fast Handovers for Mobile IPv6 (FMIPv6) [1] and Hierarchical Mobile IPv6 (HMIPv6) [2]. In FMIPv6, the network layer must know the indication of a handover from the link layer in advance to achieve seamless handovers. However, each protocol layer is designed independently and information exchange between protocol layers is not allowed.

For further handover enhancements, this document defines nine kinds of L2 triggers in the form of "Primitive". This concept can be applied to L3-driven fast handover mechanism using "Primitives".
2. Terminology

This document uses the following terms.

L3-Driven Fast Handover

The handover which is initiated by the network layer on a mobile node. Since it allows L3 handover preparation during communication, it can reduce packet loss during handover. L3-driven fast handover mechanism requires L2 information as a trigger of handover procedure.

Peer

The attachment point of a mobile node. (e.g., An access point in IEEE 802.11 networks.)

Primitive

A unit of information which is sent from one layer to another. There are four classes of primitives: Request, Confirm, Indication and Response.
3. Primitives for L2 Abstractions

Each layer offers its services by the form of primitives. There are four classes of primitives as shown in Figure 1: Request, Confirm, Indication, and Response. The request is issued by the layer that wants to get services or information from another layer, and the confirm is the acknowledgment of the request. The indication is a notification of information to the layer that requested the service, and the response is the acknowledgment of the indication. In this architecture, each layer can communicate evenly.

```
+------------------+
| Request          |
| Layer N          |
+------------------+
     | / \            |
     +--------+      |
     |       |        |
     +--------+      |
     | / \            |
     | Confirm        |
     | Layer N-m      |
+------------------+
     | / \            |
     +--------+      |
     |       |        |
     +--------+      |
     | / \            |
     | Indication     |
+------------------+
```

Figure 1: Primitives

The primitive consists of five fields: the protocol layer id to which this primitive should be sent, the protocol id to which protocol entity this primitive should be sent, the primitive class (i.e., request, confirm, indication, or response), the primitive name, and parameters.

There are three different usages of "Primitives":

Type 1. To provide L2 information to upper layers immediately

A "Request" primitive is an acquisition request for L2 information. As a "Confirm" primitive, L2 information returns immediately.

Type 2. To notify upper layers of L2 events asynchronously

"Request" and "Confirm" primitives are used just for registration. When an event occurs, an "Indication" primitive is asynchronously delivered to an upper layer.
Type 3. To control L2 actions from upper layers

A "Request" primitive is a request for operation. Ack or
nack returns immediately as a "Confirm" primitive.
4. Definitions of Primitives

In order to obtain and change L2 information, some sorts of Primitives described below are necessary. The following Primitives are defined.

4.1. L2-LinkStatus (Type 1)

The L2-LinkStatus.request is sent to the link layer when an upper layer needs the current information of a link. The L2-LinkStatus.request primitive contains "Network Interface ID" parameter. In response, the L2-LinkStatus.confirm returns. The L2-LinkStatus.confirm primitive contains three parameters: "Network Interface ID", "Peer" and "Condition". "Peer" and "Condition" indicate the current status about the link between a mobile node and a peer.

4.2. L2-PeerList (Type 1)

The L2-PeerList.request is sent to the link layer when an upper layer needs a list of candidate peers. The L2-PeerList.request primitive contains "Network Interface ID" parameter. In response, the L2-PeerList.confirm returns with "Network Interface ID" parameter and "Peer List" parameter. "Peer List" parameter is the list of candidate peers.

4.3. L2-PeerFound (Type 2)

The L2-PeerFound.indication is asynchronously provided to an upper layer when new peers are detected. This primitive carries "Network Interface ID" parameter and "Peer List" parameter. "Peer List" parameter contains the information of new peers detected by a mobile node. In order to use this notification, the registration process using L2-PeerFound.request and L2-PeerFound.confirm is needed in advance. The L2-PeerFound.request primitive has two parameters: "Network Interface ID" and "Enable/Disable". "Enable/Disable" parameter shows whether this notification function is turned on. When this registration succeeds, the L2-PeerFound.confirm returns with "Network Interface ID" parameter and "Ack" parameter in response.

4.4. L2-PeerLost (Type 2)

The L2-PeerLost.indication is asynchronously provided to an upper layer when a peer included in the list of candidate peers is got lost. This primitive carries "Network Interface ID" parameter and "Peer List" parameter. "Peer List" parameter contains the information of the peers which disappeared from the candidates list.
The registration process using the L2-PeerLost.request and the L2-PeerLost.confirm is similar to the L2-PeerFound above.

4.5. L2-LinkUp (Type 2)

The L2-LinkUp.indication is asynchronously provided to an upper layer when a new link is connected. This primitive carries "Network Interface ID" parameter and "Peer" parameter. The L2-LinkUp.request contains "Network Interface ID" parameter and "Enable/Disable" parameter for registration. When the registration succeeds, the L2-LinkUp.confirm with "Network Interface ID" parameter and "Ack" parameter returns.

4.6. L2-LinkDown (Type 2)

The L2-LinkDown.indication is asynchronously provided to an upper layer when an existing link is disconnected. The registration processing is same as the L2-LinkUp above.

4.7. L2-LinkGoingDown (Type 2)

The L2-LinkGoingDown.indication is asynchronously provided to an upper layer when an existing link is to be disconnected. This notification contains two parameters: "Network Interface ID" and "Peer". "Peer" parameter is the attachment point at which the link quality is degrading. In the registration processing, the L2-LinkGoingDown.request carries "Network Interface ID" parameter and "Enable/Disable" parameter.

4.8. L2-LinkConnect (Type 3)

The L2-LinkConnect.request is sent to the link layer when an upper layer has to change or create a new link to the specific "Peer". This primitive carries "Network Interface ID" parameter and "Peer" parameter. This operation begins after sending L2-LinkConnect.confirm with "Ack" by the link layer.

4.9. L2-LinkDisconnect (Type 3)

The L2-LinkDisconnect.request is sent to the link layer when an upper layer has to destroy an existing link to the specific "Peer". This primitive carries "Network Interface ID" parameter and "Peer" parameter. This operation begins after sending L2-LinkDisconnect.confirm with "Ack" by the link layer.
5. Definitions of Static Parameters

This section lists static parameters. Once the values of static parameters are configured, they do not change frequently during communication. The following parameters are transferred as a part of primitives.

5.1. Network Interface ID

"Network interface ID" parameter identifies uniquely the network interface in a node. The identifier is an implementation-specific consideration (e.g., name, index or unique address assigned to each interface). This parameter also contains the network interface type which indicates the kind of technology of the network interface (e.g., IEEE802.11a/b/g, 3GPP, etc.). This parameter is required for all the primitives.
6. Definitions of Dynamic Parameters

This section lists dynamic parameters. The values of dynamic parameters change frequently during communication. The following parameters are transferred as a part of primitives.

6.1. Peer

"Peer" parameter uniquely identifies the peer.

6.2. Condition

"Condition" parameter consists of the following sub-parameters: available bandwidth and link quality level. These sub-parameters are the abstracted information for considering current quality-of-service. The abstraction algorithms of sub-parameters depend on hardware device and software implementation. The useful range of link quality is divided into five levels; EXCELLENT, GOOD, FAIR, BAD, NONE in descending order.

6.3. Peer List

"Peer List" parameter consists of arbitrary couples of two sub-parameters: "Peer" and "Condition". This parameter shows a list of peers and its condition.

6.4. Enable/Disable

"Enable/Disable" flag is used for turning event notification on/off. When an upper layer needs notifications, the "Request" primitive with "Enable" is sent to the link layer as registration. When an upper layer needs no more notifications, the "Request" primitive with "Disable" is sent.

6.5. Ack/Error

When an upper layer requests some notifications, the link layer receives and confirms this "Request". If the "Request" is valid, the "Confirm" primitive with "Ack" is sent. And if it is invalid, the "Confirm" with "Error" is sent to the upper layer.
7. Architectural Considerations

The IAB document "Architectural Implications of Link Indications" [3] discusses the role and the issues of link indications within the Internet Architecture. This section discusses the architectural considerations mentioned in Section 2 of the IAB document.

[1] Proposals should avoid use of simplified link models in circumstances where they do not apply.

The information in each layer should be abstracted before it is sent to another layer. For example, in IEEE 802.11, the Received Signal Strength Indicator (RSSI), the number of retransmissions and existence of association between the mobile node and the access point are used so that the link layer indications can adjust themselves to various environments or situations. The thresholds needed for some link indications are defined from empirical study.

In the conventional protocol layering model, the Protocol Entity (PE) is defined as the entity that processes a specific protocol. Our proposal introduced the Abstract Entity (AE) to achieve link independency of the link indications. An AE and a PE make a pair. An AE abstracts the PE-dependent information to the PE-independent information.

Figure 2 shows AEs and PEs using primitives.

[2] Link indications should be clearly defined, so that it is understood when they are generated on different link layers.

To make the link information/indications clear, our proposal defines the 4 types of primitives; request/confirm and indication/response as described in Section 3. The request is used to obtain the information of another layer. The confirm is the reply to the request and it includes the requested information. The indication is generated when a particular event occurred. The response is the reply to the indication.

In our proposal about IEEE 802.11b, L2-LinkUp is defined as the status in which an association to the AP is established, and L2-LinkDown is defined as the status in which an association to the AP is not established. L2-LinkGoingdown is generated when the link quality becomes below the predefined threshold. Since the...
Received Signal Strength Indicator (RSSI) and the number of retransmissions are used to abstract and evaluate the link quality, L2-LinkGoingDown represents the link quality in both directions. It should use an average of RSSI or the number of retransmissions damped for one second or more to cope with transient link conditions.


As described above, our proposal uses the RSSI, the number of retransmissions, and the existence of an association to calculate the link status. Further work is necessary to define appropriate parameters and calculation.

[4] Upper layers should utilize a timely recovery step so as to limit the potential damage from link indications determined to be invalid after they have been acted on.

The proposed L3-driven handover described in Appendix D uses the L2-LinkGoingDown indication as the trigger for starting handover. L2-LinkGoingDown is indicated when the link quality goes down below the specific threshold. This indication is not canceled even if the link quality goes up soon. In the circumstance where the link quality is changing frequently and widely, the invalid indication may cause redundant handover and degrade the link quality further. Further work is necessary to examine this problem.

[5] Proposals must demonstrate that effective congestion control is maintained.

In the circumstance where the link quality is changing frequently and widely or a mobile node is moving rapidly, the proposed mechanism may continuously generate many and various indications. Some traffic may be generated with these indications, or frequent handovers may negatively affect to the upper layers. Both of them can cause congestion. Further work is necessary to solve this problem.
Proposals must demonstrate the effectiveness of proposed optimizations.

In IPv6 mobility, a L3-driven handover mechanism using link indications can dramatically reduce gap time due to handover. The L3-driven handover mechanism needs the L2-LinkToBeDown indication to predict disconnection. But L2-LinkGoingDown is sometimes untrusted because it is difficult to abstract the link quality. Invalid L2-LinkGoingDown may cause redundant handover. A handover mechanism using only L2-LinkUp/L2-LinkDown can also reduce gap time modestly. An example of an implementation and an evaluation of a L3-driven handover mechanism is described in Appendix D.

Link indications should not be required by upper layers, in order to maintain link independence.

Interoperability between the ordinary host and the host which implements the proposed indications is kept. The ordinary host can also start a handover by receiving the Router Advertisement message although there may be long gap time due to a handover. The gap time due to a handover can be reduced by using L2-LinkUp or L2-LinkDown. In addition, L2-LinkGoingDown can dramatically reduce the gap time.

Proposals should avoid race conditions, which can occur where link indications are utilized directly by multiple layers of the stack.

Since our proposal defines the link indications to only the IP layer, race conditions between multiple layers never happen. Further work is necessary when this proposal defines link indications to the transport and upper layers.

Proposals should avoid inconsistencies between link and routing layer metrics.

Our proposal does not deal with routing metrics.
Overhead reduction schemes must avoid compromising interoperability and introducing link layer dependencies into the Internet and Transport layers.

As described above, the link indications in our proposal are abstracted to the information independent of link types to reduce the gap time due to a handover, and the ordinary host can execute handover without using the link indications defined in our proposal.

Proposals advocating the transport of link indications beyond the local host need to carefully consider the layering, security and transport implications. In general, implicit signals are preferred to explicit transport of link indications since they add no new packets in times of network distress, operate more reliably in the presence of middle boxes such as NA(P)Ts, are more likely to be backward compatible, and are less likely to result in security vulnerabilities.

Our proposal does not define exchange of link indications between nodes.
Figure 2: AE and PE with Primitives
8. Security Considerations

The IAB document "Architectural Implications of Link Indications" [3] discusses the role and the issues of link indications within the Internet Architecture. This section discusses the security considerations mentioned in Section 4 of the IAB document.

[12] Spoofing

In the proposed link indication architecture, the link layer information is obtained from frames sent by the AP. This proposal relies on these frames. Therefore, it is necessary to make the link layer protocol secure to avoid spoofing.

[13] Indication validation

Transportation of the link indications between nodes is not assumed, hence this consideration is out of scope in our proposal.

[14] Denial of service

In the proposed link indication architecture, the link layer information is obtained from frames sent by the AP. This proposal relies on these frames. Therefore, it is necessary to make the link layer protocol secure to avoid spoofing.

9. References


Appendix A. Example Scenario

For example, the picture below shows L3-driven fast handover mechanism using the L2 triggers on MN.

```
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>L2</td>
<td>L3</td>
</tr>
<tr>
<td>-----------------</td>
<td>----------------</td>
</tr>
<tr>
<td>&lt;--------LinkUP.req-------</td>
<td>&lt;--------LinkUP.cnf-------</td>
</tr>
<tr>
<td>&lt;--------LinkGoingDown.req-------</td>
<td>&lt;--------LinkGoingDown.cnf-------</td>
</tr>
<tr>
<td>=</td>
<td>=</td>
</tr>
</tbody>
</table>
| Low Signal-----LinkGoingDown.ind------>
| <--------PeerList.req------- | <--------PeerList.cnf------- >Handover Preparation |
| <--------LinkConnect.req------- | L2 Handover--LinkConnect.cnf-------->|
| finish--------LinkUp.ind----->L3 Handover |
| finish |
```

L2: Link Layer on MN
L3: Network Layer on MN
req: Request
cnf: Confirm
ind: Indication

Figure 3: L3-Driven Fast Handover Mechanism

First, the L3 issues LinkUp.request to receive LinkUp.indication when the link becomes available. The L3 also issues LinkGoingDown.request to receive LinkGoingDown.indication when the link quality goes down below the threshold.

In the beginning of the L3-driven handover procedure, the L2 detects the radio signal strength is going down. Then the L2 sends L2-LinkGoingDown.indication to the L3. The L3 prepares for handover (e.g., CoA generation, DAD, ND cache creation, and routing table setting) and sends L2-PeerList.request to the L2 if the list of access points is needed.
If the L3 decides to perform handover according to some rules, the L3 sends L2-LinkConnect.request with some parameters about candidate access points to request L2 handover. L2 handover begins after the L3 sends L2-LinkConnect.confirm to the L2. When the L2 handover finished, the L2 sends L2-LinkUp.indication to notify the L3. Finally, the L3 performs handover (e.g., sending BU).

One of the important features of L3-driven fast handover using Primitives is that the L3 handover preparation can be done during communication. So, it can reduce disruption time during handover.
Appendix B. Example Operation for FMIPv6

Here shows 2 scenarios of L3 driven fast handover for FMIPv6. Scenario 2 is differ from scenario 1 for the timing of sending some messages.

B.1. Example Operation-1 for FMIPv6

Figure 4 shows the predictive mode of FMIPv6 operation with L3-driven link switching mechanism.
When the MN establishes link connectivity with the PAR, it performs router discovery. Then, the MN can send RtSolPr message to the PAR in order to resolve access point identifiers to subnet router information. To send RtSolPr, a MN discovers one or more access
points by sending L2-PeerList.request to the link layer. As a response to L2-PeerList.request, L2-PeerList.confirm returns with "Peer List" parameter which contains identifiers and conditions of nearby access points. After initial AP discovery, L2-PeerFound.indication with "Peer List" is also sent from the link layer when one or more access points are discovered.

When the link layer of the MN detects radio signal strength is going down, it sends L2-LinkGoingDown.indication to the network layer. Then, the MN sends FBU to PAR as the beginning of L3 handover procedure. The NCoA required for FBU message is determined according to the MN’s policy database and received PrRtAdv message. As a response to FBU, the MN receives FBack and then switches its link by L2-LinkConnect.request with specific "Peer" parameter immediately. The handover policy of the MN is outside the scope of this document.

After L2 handover, the link layer of the MN sends L2-LinkUp.indication to the network layer. The MN sends FNA message to NAR immediately. The NAR will send tunneled and buffered packets to the MN.

B.2. Example Operation-2 for FMIPv6

Figure 5 shows the predictive mode of FMIPv6 operation with L3-driven link switching mechanism.
Differ from the scenario 1, MN in scenario 2 sends LinkConnect.req from network layer to link layer after MN sends FBU. Actually, as MN sends FBack to both AR(not only PAR but also NAR), MN can get the FBack that sent out by PAR and passed through NAR, after it moves to
under the NAR.
Appendix C. Example Mapping of Primitives and IEEE802.11

This section shows examples of the mapping between "Primitives" and IEEE802.11 parameters.

C.1. L2-LinkStatus

Most of parameters for L2-LinkStatus are related to the configuration of network interface hardware. So, the operating system and the device driver module can easily collect such information. However, to create the "Condition" parameter, the MN should maintain statistics and parameters related to current wireless environment.

There are two sub-parameters of the "Condition" parameter: available bandwidth, link quality level. An available bandwidth of the current peer can be obtained by maintaining transmission rate indication and statistics of frame transmission at every time the frame transmission occurs. A link quality level can be updated by maintaining the following parameters and statistics at every time the frame reception occurs: receive signal strength indication (RSSI), transmission/reception rate indication, transmit/receive latency, bit error rate, frame error rate and noise level. Link quality level is divided into five levels: EXCELLENT, GOOD, FAIR, BAD, and NONE in descending order. Some parameters can be managed by setting threshold from software. When the parameters cross the threshold, an interrupt is generated for the software.

C.2. L2-PeerList

In IEEE 802.11 networks, L2-PeerList returns detected APs which quality is exceeding the provided threshold as peer candidates (by "Peer List" parameter). Therefore, the MN should always maintain the database of available access points according to reception of beacon frame, probe response frame and all frames. This AP database is managed in consideration of time, number of frames and signal strength. There are some network interface hardwares that can notify events to operating system only when the desired event occurs, by setting threshold from software. Moreover, the MN can transmit probe request frame for access point discovery, if needed.

C.3. L2-PeerFound

In IEEE 802.11 networks, L2-PeerFound is notified when new peers that link quality level exceeds the provided threshold are detected by listening beacons or scanning APs. If the received frame (mainly beacon or probe response) is not in the AP database described in Appendix C.2, then the link layer can create L2-PeerFound.indication. For example, if L2-PeerFound’s provided threshold of link quality is
GOOD, the L2-PeerFound.ind is made and sent to upper layer when peer’s link quality becomes stronger than to the level of GOOD.

C.4. L2-PeerLost

In IEEE 802.11 networks, L2-PeerLost is notified when an AP included in the list of candidate APs is got lost by listening beacons or scanning APs. If the entry in the AP database described in Appendix C.2 expires, or link quality level is under threshold, then the link layer can create L2-PeerLost.indication. To calculate the link quality level, signal strength of the received frame (mainly beacon or probe response) can be used. For example, if L2-PeerLost’s provided threshold of link quality is BAD, the L2-PeerLost.ind is made and sent to upper layer when peer’s link quality becomes weaker than the level of BAD.

C.5. L2-LinkUp

In IEEE 802.11 networks, L2-LinkUp is notified when association or reassociation event occurs. When such event occurs, the MN receives association response frame or reassociation response frame. Immediately after receiving it, the link layer can create L2-LinkUp.indication.

C.6. L2-LinkDown

In IEEE 802.11 networks, L2-LinkDown is notified when disassociation event occurs or when getting no beacon during the certain time. When such event occurs, the MN sends disassociation frame to AP, or the entry of the specific AP is deleted from the AP database described in Appendix C.2. By detecting such events, the link layer can create L2-LinkDown.indication.

C.7. L2-LinkGoingDown

In IEEE 802.11 networks, L2-LinkGoingDown is notified when radio signal strength of the connected AP is going down and becomes under of the provided threshold. By maintaining parameters for a link quality level like the description in Appendix C.1, the link layer can create L2-LinkGoingDown.indication.

C.8. L2-LinkConnect

In IEEE 802.11 networks, each AP is identified by BSSID, the service set of several APs is identified by SSID. When L2-LinkConnect is used to connect specific AP or service set, the link layer should set BSSID or SSID. Also, the channel should be set appropriately at the same time by searching database described in Appendix C.2. To
connect to the AP, MN should be authenticated by the AP, by sending authentication message from MN's link layer, and then MN sends association message to associate with the AP.

C.9. L2-LinkDisconnect

In IEEE802.11 networks, the MN sends disassociation message to the AP for disconnection. When L2-LinkDisconnect is used for disconnection from the current AP, the link layer should send disassociation message to the appropriate AP, and stop data transmission.
Appendix D. Implementation and Evaluation of the Proposed Model

This section describes an implementation of the proposed link indication architecture and its evaluation.

An IEEE802.11a wireless LAN device driver was modified to provide abstract link layer information in the form of primitives defined in Sec. 4. The modified driver has two AP lists. One contains the device dependent information such as the RSSI, the retransmission count, various AP parameters and various statistics. The device dependent information except for the AP parameters is updated whenever the device receives a frame. If the received frame is the management frame, the AP parameters are also updated based on the parameters in the frame.

Another AP list contains the abstract information. The abstract information is updated periodically by using the device dependent information. In the abstraction processing, for example, the RSSI or the retransmission count is converted to the common indicator "link quality".

L2-PeerList and L2-LinkStatus were implemented by using only the abstract information. Thus, the upper layers can use information of current AP and adjacent APs without depending on the devices. L2-PeerFound and L2-PeerLost is notified if the link quality went up or down beyond the thresholds when the abstract information is updated. If the link quality of the current AP went down below the specific threshold, L2-LinkGoingDown is notified. L2-LinkUp and L2-LinkDown are notified when an association with an AP is established or destroyed. To realize L2-LinkConnect and L2-LinkDisconnect, functions to connect or disconnect to the specified AP were implemented. In these functions, since only abstract information is needed to specify the AP, other layers need not to know the device dependent information.

There are eight ARs, each of which is located 80-100m interval. The AP is collocated at the AR. IEEE802.11a was used as the link layer. The same wireless channel was used at all the APs. Thus there are eight wireless IPv6 subnets in the testbed. The mobile node moved at a speed of 30-40 km/h. When link quality of current AP goes down, the mobile node executes L3-driven handover described in Appendix A. In this measurement, assume that the mobile node can know the static information of adjacent AR/AP. An application called DVTS was running on the mobile node and sends digital video data at about a 15Mbps data rate through the wireless IPv6 subnet to the correspondent node, which replays received digital video data. In this environment, the L2 handover needs less than 1 msec and mobility protocol LIN6 [4] needs a few msec for location registration. And
the total gap time due to the handover was 3-4 msec. In most case, there was no influence to the replayed pictures over handover.
Appendix E. Changes from 03

- The primitive name "LinkToBeDown" was changed to "LinkGoingDown".

- Several typos were fixed.
Authors’ Addresses

Fumio Teraoka
Graduate School of Science and Technology, KEIO University
3-14-1 Hiyoshi, Kohoku-ku
Yokohama, Kanagawa 223-8522
Japan
Phone: 45-566-1425
Email: tera@ics.keio.ac.jp
URI:  http://www.tera.ics.keio.ac.jp/

Kazutaka Gogo
Faculty of Science and Technology, KEIO University
3-14-1 Hiyoshi, Kohoku-ku
Yokohama, Kanagawa 223-8522
Japan
Phone: 45-566-1425
Email: gogo@tera.ics.keio.ac.jp
URI:  http://www.tera.ics.keio.ac.jp/

Koshiro Mitsuya
Jun Murai Lab, Shonan Fujisawa Campus, KEIO University
5322 Endo
Fujisawa, Kanagawa 252-8520
Japan
Phone: +81-466-49-1100
Email: mitsuya_at_sfc.wide.ad.jp
URI:  

Rie Shibui
Graduate School of Science and Technology, KEIO University
3-14-1 Hiyoshi, Kohoku-ku
Yokohama, Kanagawa 223-8522
Japan
Phone: 45-566-1425
Email: shibrie@tera.ics.keio.ac.jp
URI:  http://www.tera.ics.keio.ac.jp/
Intellectual Property Statement

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Disclaimer of Validity

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Copyright Statement

Copyright (C) The Internet Society (2006). This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

Acknowledgment

Funding for the RFC Editor function is currently provided by the Internet Society.