An IP-Decoupling Approach to Host Mobility

Chun-Chieh Wang
Department of Computer Science and Information Engineering
National Chiao Tung University, Hsinchu, Taiwan 30050
Email: jjwang@csie.nctu.edu.tw
Phone: +886-3-5731892
Fax: +886-3-5724176

Chien-Chao Tseng
Department of Computer Science and Information Engineering
National Chiao Tung University, Hsinchu, Taiwan 30050
Email: cctseng@csie.nctu.edu.tw
Phone: +886-3-5731867
Fax: +886-3-5724176

Jen-Chi Liu
Computer and Communications Research Laboratories
Industrial Technology Research Institute, Hsinchu, Taiwan 310
Email: jcliu@itri.org.tw
Phone: +886-3-5914663
Fax: +886-3-5820263

Kuang-Hui Chi
Computer and Communications Research Laboratories
Industrial Technology Research Institute, Hsinchu, Taiwan 310
Email: chikh@itri.org.tw
Phone: +886-3-5917946
Fax: +886-3-5820310

An IP-Decoupling Approach to Host Mobility

Chun-Chieh Wang Chien-Chao Tseng
Department of Computer Science and Information Engineering
National Chiao Tung University
1001 Ta-Hsueh Road, Hsinchu, Taiwan 30050
E-mail: {jjwang, cctseng}@csie.nctu.edu.tw

Jen-Chi Liu Kuang-Hui Chi
Computer and Communications Research Laboratories
Industrial Technology Research Institute, Hsinchu, Taiwan 310
Email: {jcliu,chikh}@itri.org.tw

ABSTRACT

This paper presents the design, implementation, and performance evaluation of an approach to supporting host mobility in wireless Internet. We observe that an IP address conventionally serves dual purposes: a routing directive in the network-layer and an end-point identity in transport or upper-layers. Such a coupling causes session disruptions when hosts move and affiliate with new IP addresses due to changing points of attachment to the system. As a remedy, we propose to support host mobility by decoupling the IP layer routing directive from the transport layer identification and by introducing a table to maintain the binding of IP addresses to transport-layer identities or vice versa. The binding information can be queried or modified using the current DNS system with secure DNS updates, and can be exchanged directly between two communication peers without any third party's interventions. Therefore the routing path of our approach is optimized, as opposed to the well-known IETF standard Mobile IPv4. As further quantitative comparisons, we contrast our scheme to Mobile IPv4 and Mobile IPv6 in terms of overhead of packet size and extra processing time in TCP/IP protocol stack, and packet loss during handoffs. Simulation results show that our approach is promising, incurring comparatively least packet overhead than counterpart schemes do. Concerning applicability, our approach is susceptible to easy deployment in a setting of IPv6 networks, or IPv4 and IPv6 co-existent networks. Additionally, our proposal can also be used to support mobility in multi-tier environment such as Wireless LANs and GPRS overlay networks.

1 Introduction

In recent years, the proliferation of mobile communication changes people's life vastly. People can use mobile devices to connect to the Internet anytime and anywhere even when traveling on vehicles. The research community has identified host mobility management as a very important issue, aiming to allow mobile devices to retain connections while changing their locations and points of attachment to the network. It is not trivial to support host mobility in today's networking infrastructure since traditional communication paradigms were not designed for mobile environment – drastic performance degradation thereby results from unreliable wireless communication links and tight resource constraints on mobile nodes.

Most of the existing schemes add some new entities, mobility agents, to support host mobility and avoid modifying protocol fabrics in correspondent nodes. Among others, IETF has proposed Mobile IP as a standard. These previous schemes are complex and the resultant performance could be very challenging. Besides, these schemes require that host mobility be supported only if there is a mobility agent in the visited network. This limits wide-ranging applicability. As a remedy, we propose an approach, referred to as the IP-Decoupling approach in this text, which operates without any mobility agent. Our approach modifies the original TCP/IP protocol stack such that the protocol suite supports host mobility. We separate the use of a single IP address shared by all protocol layers into two levels: host identifier address and network identifier address. The former is similar to a domain name identifying a unique end host whereas the latter a routing directive which is used to route packets to a correct destination. We employ the network identifier address in IP layer to route packets and the host identifier address in upper layers. A table is introduced to record the mapping between two addresses.

The remainder of this paper is organized as follows. Section 2 describes previous schemes supporting host mobility and summarizes their potential flaws. Section 3 introduces our approach. Implementation issues are provided in Section 4. Section 5 shows the performance evaluation of our approach, in comparison with two counterparts, Mobile IPv4 and Mobile IPv6. Section 6 discusses several related issues of our approach. Lastly, conclusion and future work are drawn in Section 7.

2 Related Work

Mobile IP [9] is an IETF standard specified to support host mobility and transparent routing of datagrams between stationary or mobile entities in the Internet. A mobile node (MN) is identified by its home address, regardless of its current point of attachment to the system. A correspondent node (CN) sends IP datagrams to the MN at its home address in the same way as with any other destinations. When visiting a new network, the MN registers a locally acquired temporary IP address, namely care-of address, with its home agent. The home agent

then intercepts and tunnels datagrams addressed to the mobile node's home address by some encapsulation mechanism [10,11] to the mobile node's care-of address.

Mobile IP is subject to triangle routing and ingress filtering. The former refers to an issue of routing inefficiency that packets destined for an MN away from home need to be delivered indirectly via the HA. This issue is dealt with by a route optimization scheme [12] which utilizes Binding Update messages to inform CNs to route packets directly to the MN, while at the expense of modifying protocols on wide-ranging CNs. Secondly, ingress filtering [2], originally designed to avoid denial-of-service attacks, mandates routers not to forward outwards packets with Source IP Addresses foreign to the local network. In order to enable MNs to originate packets invariably using home addresses, it is required that packets from a visiting MN be first tunneled back to the HA from which packet delivery towards destination CNs proceeds. Such a reverse tunneling approach, however, makes routing paths even longer since packets to and from the MN are required to be routed through the HA. In addition, tunneling overhead due to encapsulation and decapsulation becomes double.

Loose Source Routing (LSR) scheme [4] using IP Loose Source Route option for packet delivery instead of using encapsulation mechanisms necessitates supports from mobility agents. A major drawback of this scheme is that most of the Internet hosts do not perform correct route reversal, and therefore CNs may not deliver packets back to the MN correctly. Besides, packets with LSR option may receive poor service from common IP routers.

The end-to-end approach [13] by Snoeren and Balakrishnan exploits dynamic updates to the Domain Name System (DNS) to track every MN's location. A *connection migration* procedure is proposed to retain existing TCP connections when an MN changes its point of attachment from one network to another. This approach adds a MIGRATE_WAIT state in TCP state transition diagram and a new Migration Option in a SYN segment. An important flaw of this approach is that it is suitable for TCP applications only.

Gupta and Reddy [3] used an ICMP-like redirection message to redirect packets to MNs' current locations. While this scheme requires supports from Has, our approach queries DNS to locate a concerned MN without any mobility agent interventions.

3 The IP-Decoupling Approach

3.1 Overview

Given deficiencies in Mobile IP, we propose another orthogonal scheme to support host mobility. Our proposal operates independently of Mobile IP and, in our setting, does not require any mobility agent to mediate in communication activities. The main idea behind the proposed IP-Decoupling (IPD) approach is to separate, with respect to a host, a single IP address shared by all protocol layers into two levels: network-identifier (NID) address and

host-identifier (HID) address. The NID address is used to identify the host's location to which packets are routed correctly, and changes whenever the host moves. In contrast, the HID address is used in upper layer such as TCP or UDP to uniquely identify the end host. Notice that the HID address remains unchanged throughout a connection.

In terms of interfaces with TCP/IP, we develop a software module operating between transport and network layers, as illustrated below. We mandate that all originally in-between data traffic be processed before delivery in communication peers.

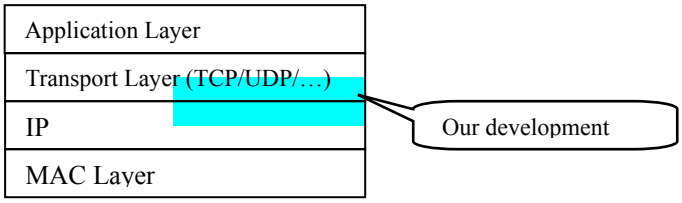


Figure 1: IPD module in original protocol stack

3.2 HID Option

Since we separate a single IP address shared by all protocol layers into NID and HID addresses, and use NID addresses in Source and Destination IP Address fields of IP header, we need to add a new IP option in outgoing packet headers, to inform receivers of source and destination HID addresses of packets. Generally an IP datagram with HID Option can be illustrated in Figure 2.

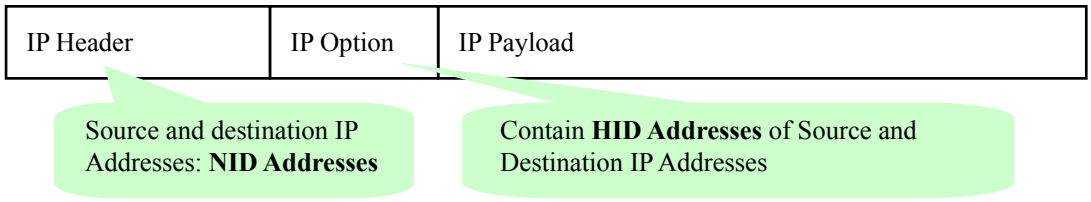


Figure 2: IP Datagram with HID Option

HID Option format is indicated in Figure 3. When the Length field of the option is 5, the option carries HID address of a source host if the Src/Dst is 0, or that of a destination host otherwise. When the Length field is 8, both the source and the destination hosts' HID addresses are present.

	Type = 42	Length = 5	Src/Dst = 0 or 1
Source/Destination HID Address			

	Type = 42	Length = 8
Source HID Address		
Destination HID Address		

Figure 3: HID Option, a new IP option type

3.3 NID and HID Addresses Assignment

For correct routing and avoiding ingress filtering, we specify that each packet header carries the current NID addresses of the source and destination hosts in Source and Destination IP Address fields, respectively. We assume that a host can obtain a new NID address from some local DHCP server when moving to a new network.

An HID address, representing the unique identifier of a mobile host, is used in upper layers to identify a connected peer. For example, source and destination HID addresses and port numbers are used in a TCP socket.

We further classify the HID address into two groups: fixed or dynamic HID addresses. A fixed HID address, once assigned to some host, is employed in all subsequently initiated sessions. Observe that a fixed HID address appears similar to the domain name of a mobile node but of a 4-tuple IP address format. On the other hand, a dynamic HID address allocated to a host does not change for a session, while the host may utilize different HID addresses session by session. In our subsequent development, we use dynamic HID addresses mainly due to backward compatibility consideration, since applying fixed HID addresses makes IPD-enabled and IPD-incapable node unable to setup TCP or UDP connections with each other. For example, a Checksum field in TCP header of a TCP datagram is a mandatory field that must be calculated and stored by the sender, and then verified by the receiver. The TCP checksum is calculated with a pseudo-header which contains the source and destination IP addresses. As clarified in Figure 4, since the IPD-incapable CN uses NID addresses to calculate the TCP checksum while the IPD-enabled MN uses HID addresses, TCP checksum can not be verified correctly. Instead, if HID addresses are assigned in a dynamic manner, communication among IPD-enabled or IPD-incapable hosts, can easily be maintained. Furthermore, the assignment of fixed HID addresses is quite complex and modifications to the current DNS system are required.

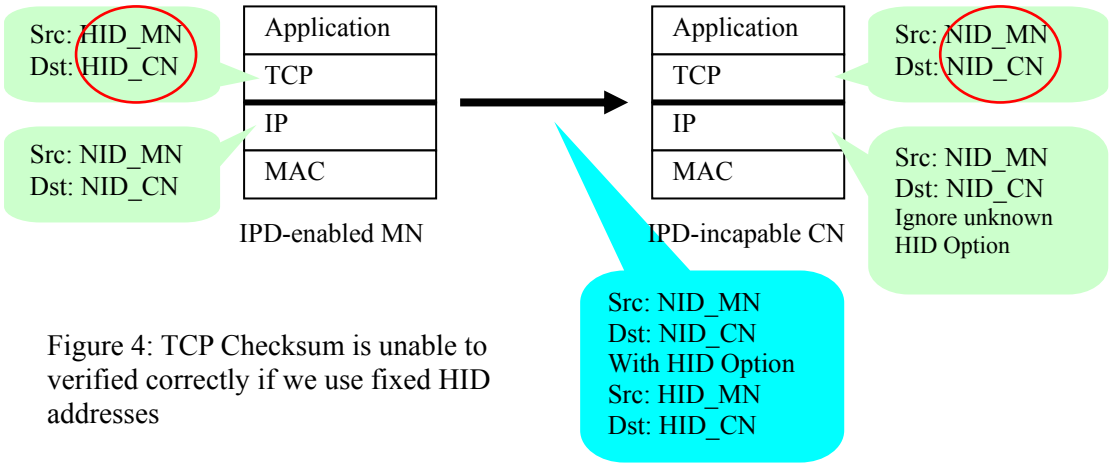


Figure 4: TCP Checksum is unable to verified correctly if we use fixed HID addresses

We stipulate that each mobile node uses its current NID address as the HID address for every newly activated session and an HID address will not change during the lifetime of

assigned session. A simple case of dynamic HID address assignment is shown in Figure 5, assuming that the original NID address of MN is *IPn*.

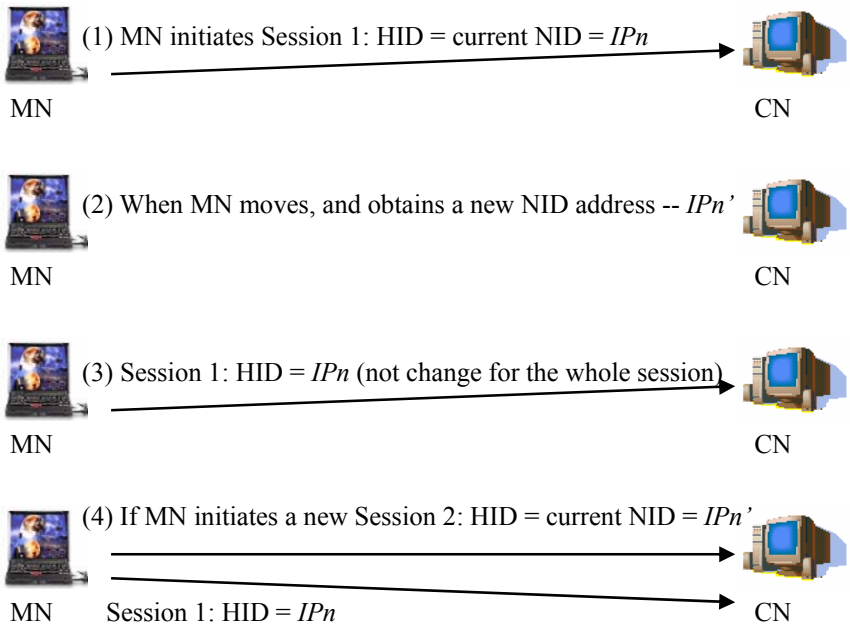


Figure 5: Dynamic HID Address Assignment

As illustrated in Figure 6, when communicating with an IPD-incapable host, a mobile host uses its current NID address as the current HID address. Hosts without IPD capabilities use the NID address in upper layers. Such a machinery equates the HID address to the NID address such that IPD-incapable hosts can verify TCP checksum correctly, thereby retaining backward compatibility.

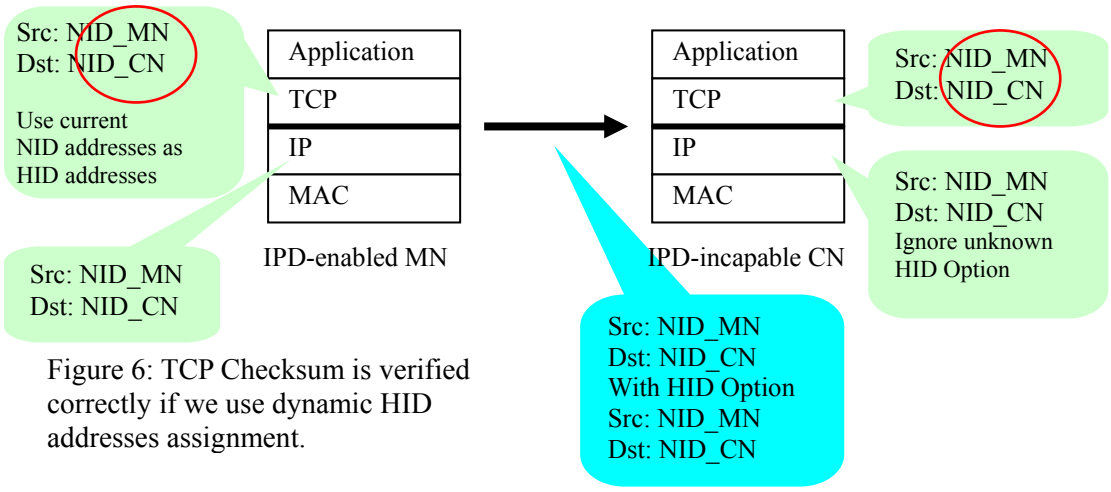


Figure 6: TCP Checksum is verified correctly if we use dynamic HID addresses assignment.

3.4 Locating a Mobile Node

Our approach adopts today's Domain Name System (DNS) [7,8] to locate a mobile node. When a host moves and detects its change of attachment to the network, it obtains a new NID address locally from a DHCP server. Then the host employs a secure DNS update [1] procedure to update the corresponding entry in the root DNS server. Therefore, in response to a DNS query, the server always returns the latest NID address for the concerned host. To avoid the cache for DNS record, we set the time-to-live (TTL) field for the A–record of the mobile node to zero. This approach is also applied in [13].

If a mobile node moves after a connection is established, it will send an ICMP Echo Request with HID Option (similar to Binding Update messages in Mobile IPv6) to every correspondent node to update the MN's location information. We designate the original ICMP Echo Request to carry an additional HID Option in the IP header to inform the correspondent node of some new NID address. The message format is shown in Figure 7.

<table>
<tr><td>Version</td><td>Length=7</td><td>TOS</td><td colspan="2">Total Length</td></tr>
<tr><td colspan="3">Identification</td><td>Flags</td><td>Fragment Offset</td></tr>
<tr><td colspan="2">TTL</td><td>Protocol</td><td colspan="2">Header Checksum</td></tr>
<tr><td colspan="5">Source IP Address (NID Address)</td></tr>
<tr><td colspan="5">Destination IP Address (NID Address)</td></tr>
<tr><td>Type = 42</td><td>Length = 5</td><td>Src/Dst=0</td><td colspan="2"></td></tr>
<tr><td colspan="3">Source HID Address</td><td colspan="2">Padding = 0</td></tr>
<tr><td>Type = 8</td><td>Code = 0</td><td colspan="3">Checksum</td></tr>
<tr><td colspan="2">Identifier</td><td colspan="3">Sequence Number</td></tr>
</table>

Figure 7: ICMP echo request with HID Option in IP header.

3.5 Address Mapping

In our approach, an AMT (Address Mapping Table) is required in each host. The AMT is to relate an NID address to an HID address and vice versa. To this end, we propose three schemes with their respective AMT structures. We will also discuss their advantages and disadvantages.

3.5.1 Host-oriented Method

An AMT of the host-oriented method is depicted in Figure 8, each entry of which consists of a HID field, a NID field, and a Timeout field. HID and NID fields record the HID and the NID addresses of some host, respectively. The Timeout field, recording the time duration for which the entry is valid, is refreshed whenever the entry is accessed or modified.

HID1	NID1	Timeout1
HID2	NID2	Timeout2
HID3	NID3	Timeout3
……	……	……

Figure 8: Host-oriented AMT

In the beginning, we create an entry for the host itself in the table, whose HID and NID fields are both set to the host's local address. When starting communication with or receiving a packet from a new node, we create another entry for the correspondent node. For instance, to start a session with a new node, the host performs a DNS query and resolves the correspondent IP address 140.113.215.199. A new entry is then created on the host AMT, with HID and NID being set to 140.113.215.199.

As a host moves and switches its network-layer address, say from *IPn* to *IPn'*, all the NID fields in the local AMT matching *IPn* are replaced with *IPn*. Additionally, the mobile host will create a new entry with HID and NID fields being set to *IPn* unless an identical entry has existed. Next, the mobile host will send an ICMP echo request with HID Option to every correspondent node, to update accordingly all the *IPn*-valued NID fields in the AMTs of these correspondent nodes.

As an example, Figure 9 shows the changes of AMTs on a mobile host and its correspondent node over successive six events. For simplicity, an AMT entry is represented as a form of an address pair (HID, NID) only. Suppose that initially the local IP addresses of the host and the correspondent node are IPn and IPc, respectively.

Event Sequence	AMT of MN (HID,NID)	AMT of CN (HID,NID)
(1) Initially	(*IPn*, *IPn*)	(*IPc*, *IPc*)
(2) The MN tries to make a connection with a CN	(*IPn*, *IPn*) (*IPc*, *IPc*)	(*IPc*, *IPc*)
(3) The CN receives first packets from the MN	(*IPn*, *IPn*) (*IPc*, *IPc*)	(*IPc*, *IPc*) (*IPn*, *IPn*)
(4) The MN moves and obtains a new IP address IPn'	(*IPn*, *IPn'*) (*IPc*, *IPc*) (*IPn'*, *IPn'*)	(*IPc*, *IPc*) (*IPn*, *IPn*)
(5) The CN receives an ICMP echo request with HID option from the MN	(*IPn*, *IPn'*) (*IPc*, *IPc*) (*IPn'*, *IPn'*)	(*IPc*, *IPc*) (*IPn*, *IPn'*)
(6) The MN initiates another connection to the CN after movement	(*IPn*, *IPn'*) (*IPc*, *IPc*) (*IPn'*, *IPn'*)	(*IPc*, *IPc*) (*IPn*, *IPn'*) (*IPn'*, *IPn'*)

Figure 9: An example of the host-oriented method.

We approach message delivery by means of the foregoing example. Given an outgoing packet, we perform local AMT look-ups using HIDs as indices and change the Source and Destination IP Addresses fields of the packet to the mapped NID addresses of the sending and recipient hosts, respectively. Then normal routing proceeds. On reception, the correspondent node examines the Source and Destination IP Addresses fields of the packet against the local AMT using NIDs as indices, and then replaces the two fields with the mapped HID addresses, to restore the packet such as delivered through legacy routing.

We contend with a possibility that multiple HID addresses are mapped to a common NID address as Case (6) of Figure 9 indicates, causing resolution conflicts during processing an incoming packet. Such conflicts occur when a host moves while maintaining sessions established respectively from different networks. To distinguish packets meant for distinct sessions, we employ HID Option introduced in Section 3.2 in the event that one of the communication peers (source or destination sites) uses distinct HID or NID address. In this case, HID Option accommodates the HID addresses of concern.

Figure 10 illustrates the Source and Destination IP Addresses and HID Option conveyed in the IP packet headers for the two concurrent connections after Step (6) of Figure 9, whereupon the CN can identify correct traffic sources from received HID Option.

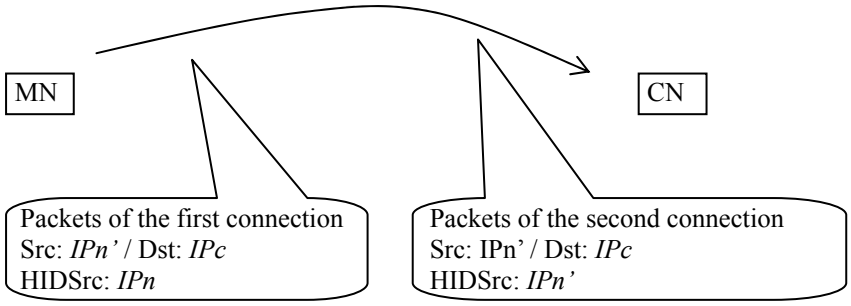


Figure 10: Using HID Option to signify packets intended for different sessions

Given host-oriented address mapping, our IPD approach is summarized in the following three procedures:

(A) Upon changing network-layer addresses from *IPn* to *IPn'*, a mobile host executes the steps below.

Step 1. Update each AMT entry recording *IPn* in the NID field to *IPn'*.

Step 2. Create a new entry whose HID and NID fields are both set to *IPn'* unless an identical entry has existed.

Step 3. Send an ICMP echo request with HID Option (Figure 7) to every correspondent node to update outdated NID fields in their AMTs.

(B) On receipt of a packet originated from upper layer to be destined for some other site, we

examine the Source and Destination IP Addresses, say *IPn* and *IPc*, respectively, of the packet. Given *IPn* and *IPc* as HID addresses, the local host performs subsequent steps with reference to the local AMT:

Step 1. If there exists an entry whose HID field records *IPc*, replace the Destination IP Address of the packet header with the discovered NID address.
Otherwise, add a new entry with HID and NID fields being both set to *IPc*.

Step 2. If there exists an entry whose HID field matches *IPn*, replace the Source IP Address of the packet header with the mapped NID address.

Step 3. If the host's current IP address is different from *IPn*, add into the packet header an HID Option with *IPn* assigned to the Source HID Address field.

Step 4. If the newly replaced Destination IP Address in the packet header is different from *IPc*, add into the header an HID Option with *IPc* assigned to the Destination HID Address field.

(C) Given an incoming packet to be destined for upper layers, a recipient node executes actions below.

Step 1. If the packet does not carry any HID Option, normal packet processing proceeds, bypassing Steps 2, 3 and 4. This is used for backward compatibility whence nodes without IPD-capabilities can still communicate with mobile nodes.

Step 2. Otherwise, if the Source HID Address, say *IPc*, is present, check whether the HID field of any AMT entry matches.

(a) If so and the packet header carries a Source Address, say *IPc'*, different from the NID value of this entry, replace the NID field with *IPc'*.

(b) If not, create a new entry whose HID and NID fields are set to *IPc* and Source IP Address in the packet header, respectively.

Step 3. Replace the Source IP Address of the packet with the Source HID Address, if present, in HID Option.

Step 4. Replace the Destination IP Address of the packet with the Destination HID Address, if present, in HID Option.

Observe that in Step 2(a) of Case (C), we can use an ICMP echo request with HID option or piggyback HID Option to update correspondent nodes' AMTs.

3.5.2 Session-oriented Method

A session-oriented AMT is depicted in Figure 11 each entry of which contains both the source and the destination HID and NID addresses and the port information.

Session-oriented AMTs are only used for TCP/UDP packets because only the TCP and the UDP require port information. The basic procedure of the session-oriented method is similar

Src HID1	Src NID1	Dst HID1	Dst NID1	Src port1	Dst port1	Timeout1
Src HID2	Src NID2	Dst HID2	Dst NID2	Src port2	Dst port2	Timeout2
……	……	……	……	……	……	……

Figure 11: Session-oriented AMT

to that of the host-oriented method. For every new TCP/UDP session, we create a new entry in AMT. When the mobile node moves to a new network and the NID address of the local host changes, AMT will be updated and an ICMP echo request with HID Option will be sent to inform all correspondent nodes with active sessions (similar to the host-oriented AMT).

As an example, Figure 12 shows the changes of AMTs on a mobile host and its correspondent node over successive six events. For simplicity, an AMT entry is represented as a form of an address and port pair (Src HID, Src NID, Src port, Dst HID, Dst NID, Dst port) only. Suppose that initially the local IP addresses of the mobile host and the correspondent node are *IPn* and *IPc*, respectively and the mobile host tries to telnet to the correspondent node's port 23.

Event Sequence	AMT of MN (Src HID, Src NID, Src port, Dst HID, Dst NID, Dst port)	AMT of CN (Src HID, Src NID, Src port, Dst HID, Dst NID, Dst port)
(1) Initially	None	None
(2) The MN tries to make a connection with a CN	(*IPn*, *IPn*, 1234, *IPc*, *IPc*, 23)	None
(3) The CN receives first packets from the MN	(*IPn*, *IPn*, 1234, *IPc*, *IPc*, 23)	(*IPc*, *IPc*, 23, *IPn*, *IPn*, 1234)
(4) The MN moves and obtains a new IP address IPn'	(*IPn*, *IPn'*, 1234, *IPc*, *IPc*, 23)	(*IPc*, *IPc*, 23, *IPn*, *IPn*, 1234)
(5) The CN receives an ICMP echo request with HID option from the MN	(*IPn*, *IPn'*, 1234, *IPc*, *IPc*, 23)	(*IPc*, *IPc*, 23, *IPn*, *IPn'*, 1234)
(6) The MN initiates another connection to the CN after movement	(*IPn*, *IPn'*, 1234, *IPc*, *IPc*, 23) (*IPn'*, *IPn'*, 1235, *IPc*, *IPc*, 23)	(*IPc*, *IPc*, 23, *IPn*, *IPn'*, 1234) (*IPc*, *IPc*, 23, *IPn'*, *IPn'*, 1235)

Figure 12: An example of the session-oriented method

Compared with the host-oriented method, the CN can distinguish incoming packets with same NID source and destination addresses by source and destination port information after

Case (6) in Figure 12, i.e., we can use the source and destination NID addresses and port numbers as the key to find the corresponding source and destination HID addresses in the session-oriented AMT. As a consequence, we do not add HID Option in the IP header for outgoing packets in session-oriented method.

Given session-oriented address mapping, our IPD approach is summarized in the following three procedures:

(D) Upon changing network-layer addresses from *IPn* to *IPn'*, a mobile host executes the steps below.

Step 1. Update each AMT entry recording IPn in the NID field to *IPn'*.

Step 2. Send an ICMP echo request with HID Option (Figure 7) to every correspondent node to update outdated NID fields in their AMTs.

(E) On receipt of a packet originated from upper layer to be destined for some other site, we examine the Source and Destination IP Addresses and ports, say *IPn*, *IPc*, *portN*, and *portC*, respectively, of the packet. Given *IPn* and *IPc* as HID addresses, the local host performs subsequent steps with reference to the local AMT:

Step 1. If there exists an entry that all the Source and Destination HID Address and Port fields are exactly the same as *IPn*, *IPc*, *portN*, and *portC*, replace the Source and Destination IP Addresses of the packet header with the discovered NID addresses.

Step 2. Otherwise, add a new entry with Source HID and NID fields being both set to IPn, Destination HID and NID fields being both set to *IPc*, and Source and Destination Port fields being set to *portN* and *portC*, respectively.

(F) Given an incoming packet to be destined for TCP or UDP layer, a recipient node executes actions below.

Step 1. If there exists an entry that all the Source and Destination NID Address and Port fields are exactly the same as the Source and Destination IP Addresses in IP header, say *IPc* and *IPn*, and Source and Destination Ports in TCP or UDP header, say *portN* and *portC*, replace the Source and Destination IP Addresses of the packet header with the discovered HID addresses.

Step 2. Otherwise, add a new entry with Source HID and NID fields being both set to *IPn*, Destination HID and NID fields being both set to *IPc*, and Source and Destination Port fields being set to *portN* and *portC*, respectively.

(G) Given an incoming packet with HID Option

Step 1. If the Source HID Address, say *IPc*, is present, check whether the HID field of any AMT entry matches.

(a) If so and the packet header carries a Source Address, say *IPc'*, different from the NID value of this entry, replace the NID field with *IPc'*.

Notice that in Step 1 of Case (G), we can use an ICMP echo request with HID Option or

piggyback HID Option to update correspondent nodes' AMTs.

The advantage of the session-oriented AMT is that packets require no more HID Option overhead. We can use the source and destination NID addresses and port numbers to uniquely identify their HID addresses.

3.5.3 Hybrid Method

It may be beneficial to combine host-oriented with session-oriented methods. To realize, we use the session-oriented AMT for TCP/UDP packets and the host-oriented AMT for others. In this way, there is no HID Option overhead for TCP/UDP packets while host mobility can still be supported for non-TCP/UDP packets.

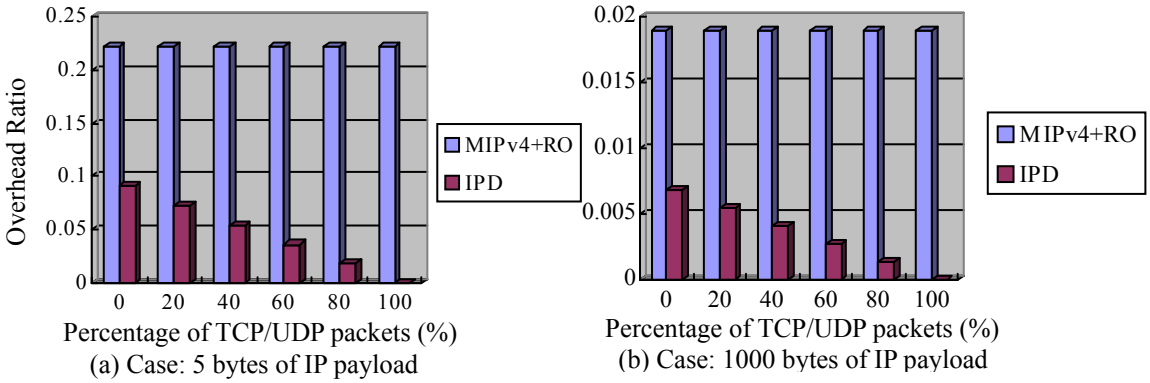
The host-oriented AMT will function as previous three cases and the non-TCP/UDP packets will be delivered correctly. The outgoing TCP/UDP packets should be replaced with the NID address according to the session-oriented AMT in TCP/UDP layer and no extra action is needed in the IP layer. Besides, due to Step 1 in case (C), the incoming TCP/UDP packets will be delivered to the upper layer without modification because the incoming TCP/UDP packets do not contain the HID Option. The replacement of the HID and the NID address are completed in TCP/UDP layer for TCP/UDP packets.

4 Implementation Issues

We have implemented our proposal in FreeBSD 4.4.1-Release. We implement the host-oriented AMT and the session-oriented AMT in the TCP/IP layer. For non-TCP/UDP packets, we add two new functions ip_output_new() and ip_input_new(). Function ip_output_new() is similar to the original ip_output() but replaces the HID address with the NID address according to the host-oriented AMT and adds HID Option field in IP header. Function ip_input_new() is also similar to the original ip_input() but invokes the modified ip_dooptions() which will process HID Option correctly. For the incoming TCP/UDP packets, tcp_input() and udp_input() will replace the NID addresses (both source and destination addresses) with the HID addresses according to the session-oriented AMT. For the outgoing TCP/UDP packets, the HID addresses will be changed to the NID addresses before calling the ip_output() function. Note that both the host-oriented AMT and the session-oriented AMT should be accessible in the IP layer because, when receiving an ICMP Echo Request message with HID Option, both AMTs should be updated.

5 Performance Evaluation

This section compares our approach with other counterpart schemes in terms of three dimensions: overhead of packet size and operation time, and packet loss during handoffs.



(a) Case: 5 bytes of IP payload

(b) Case: 1000 bytes of IP payload

Figure 13: Comparison between IPD and MIP+RO.

The overhead of our IPD approach results from two categories: additional HID Option and operation in the TCP/IP protocol stack. Since an HID Option is added in the IP header to identify source and destination HID addresses for non-TCP/UDP packets, each such packet carry this extra information of either 7 or 10 bytes. Given that TCP/UDP packets account for substantial part of traffic in the Internet, only a small portion of the packets will incur this overhead. Compared with Mobile IPv6 [5] which adds Destination Option (home address option) and Routing Option in every packet for mobile nodes, the incurred overhead in our scheme is relatively small. Here we formulate the performance index of concern as follows:

$$\text{Overhead ratio} = X / (\text{original IP header size} + X + \text{IP payload size}),$$

where X denotes the size of extra bytes in every packet for mobility support purposes. For instance, X in the context of our IPD approach is 7 if only one of the source and destination HID addresses differs from the corresponding NID address, or 10 if both HID addresses are distinct from the source and destination NID addresses.

Figure 13 shows the comparisons between the overhead of the IPD approach and that of the Mobile IP with route optimization under the network with various percentages of TCP/UDP packets. We assume the IP-in-IP encapsulation [10] is used for Mobile IP and hence X is 20 thereon. Figure 14 shows the comparisons of the overhead between Mobile IPv6 and the IPD approach in IPv6. We assume that HID Option size in IPv6 is 19 bytes (1byte for Next Header, 1 for Length, 1 for Src/Dst, and 16 bytes for the IPv6 HID Address) that is almost the same as the size of Mobile IPv6 Destination Options header and routing header extensions. From Figure 13 and Figure 14, we can see that of our IPD approach incur less overhead than Mobile IP with Route Optimization and Mobile IPv6, especially when the percentage of the TCP/UDP packets becomes higher.

Another overhead is the operation involved in the TCP/IP protocol stack to map an HID into a NID address or vice versa. This address translation is only a simple table lookup procedure and therefore requires very little operation time. An AMT can be implemented as a binary tree in which the search and insertion times are both of $O(\log n)$ where n denotes the number of entries in AMT. Besides, the maximum number of concurrent connections is

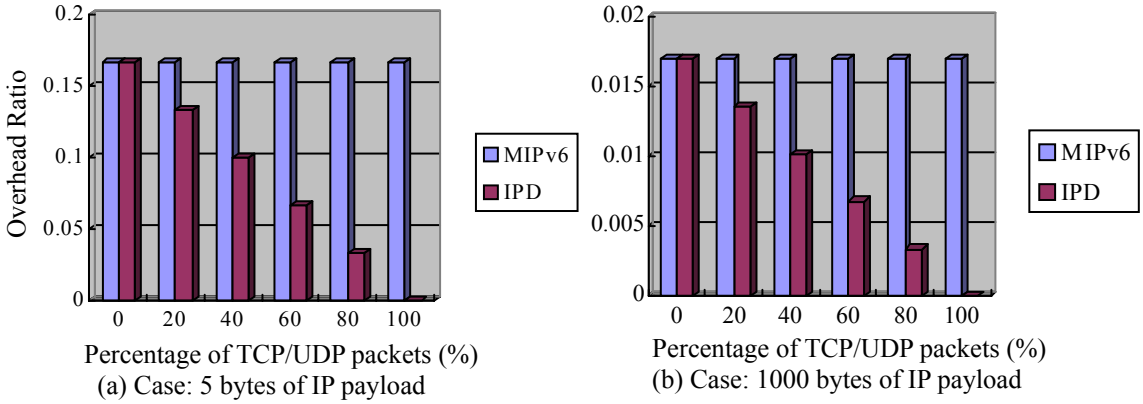


Figure 14: Comparison between IPD and MIPv6.

limited in the current operating systems and therefore the address mapping operation time will not be a serious overhead in our system.

Packet loss during handoffs between different networks is a deficiency of our IPD approach. Packets sent by a correspondent node after a mobile node has moved and before the correspondent node receives the ICMP echo request with HID Option sent by the mobile node will be lost in our system. Such packet loss does not cause connection breakages since the upper layer will retransmit the lost packets but will cause performance degradation. Packet loss during handoffs is inevitable in mobility environment. The maximum number of lost packets for the IPD approach is the Round Trip Time (RTT) between a concerned mobile host and its correspondent node times the maximum packet transmission rate, whilst the same measure for Mobile IP is RTT between the mobile node and the home agent times the maximum packet transmission rate. Our approach will perform better when the mobile node is closer to the correspondent node than the home network.

6 Discussions and Remarks

6.1 Multi-tier System with our IPD Approach

The IPD approach exhibits several important advantages. First, the routing path of the IPD approach is optimized and no mobility agent is required. Besides, the IPD approach is particularly apt for a multi-tier architecture where a multi-mode mobile node with different wireless networking devices such as wireless LAN card and GPRS device may roam between two different networks. Because the change of NID addresses between different devices will not affect the HID address in the upper layer (Figure 15), and therefore, the connection will not be disrupted.

6.2 Mobility between IPv4 and IPv6 Environment

Our proposal is apropos to be employed to support mobility in IPv6 network. We only need

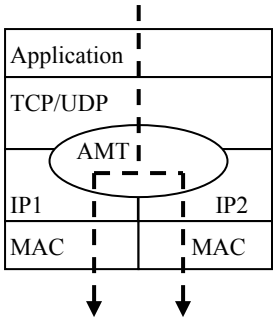


Figure 15: Two-tier architecture with the IPD approach.

to change the IP address size to 128 bits and use ICMPv6 echo request as the update message. The functionality of the AMT and the other procedures remain the same as IPv4 environment. Concerning mobility between IPv4 and IPv6 networks, there are still no standard solutions to integrate Mobile IPv4 and Mobile IPv6 [16]. But with our approach, dual stack mobile node (which has both the IPv4 and IPv6 protocol stack) is able to roam between IPv6 and IPv4 networks because the functions of the IPD approach are almost the same in both IPv4 and IPv6 environments and the changes in NID addresses between IPv4 and IPv6 will not affect the HID address in the upper layer.

6.3 Backward Compatibility

Due to the dynamic HID address assignment, we can still connect to the hosts that do not support the IPD approach. In TCP/UDP layer, the IPD hosts use the HID address while non-IPD hosts use the original IP address to calculate the TCP/UDP checksum and identify the end point of the connections. With the dynamic HID address assignment method, the HID address is the same as the NID address and the original IP address, respectively, in the IPD hosts and in the non-IPD hosts. Therefore, the checksum in both peers will be verified correctly. But under this circumstance, the connection will be broken when the mobile node moves. Another limitation of our approach is that both peers cannot move simultaneously because we do not have the help of mobility agent like Mobile IP does.

6.4 Security Issue

Most security problems can be solved with the same approaches adopted in Mobile IP or Mobile IPv6. We use a straightforward mechanism to solve the connection hijacking problem, as clarified below. More sophisticated solutions are still required further study.

The connection hijacking problem, as exemplified in Figure 16, is that without any authentication mechanism, everyone could claim he is the original mobile node by sending ICMP echo request with HID Option and then intercept the original connection.

The simplest solution is to apply the public/private key mechanism. Every MN generates its own public/private key pair and records every correspondent node' public key in the AMT.

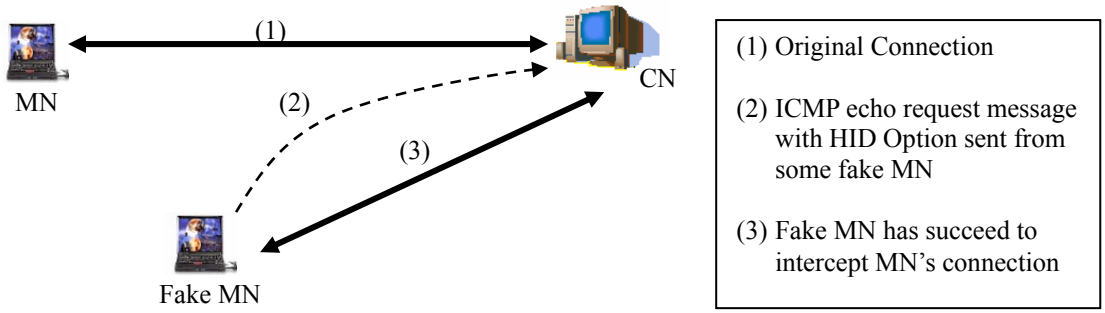


Figure 16: Connection Hijacking Problem

Figure 17 shows a simple authentication procedure to avoid connection hijacking problem.

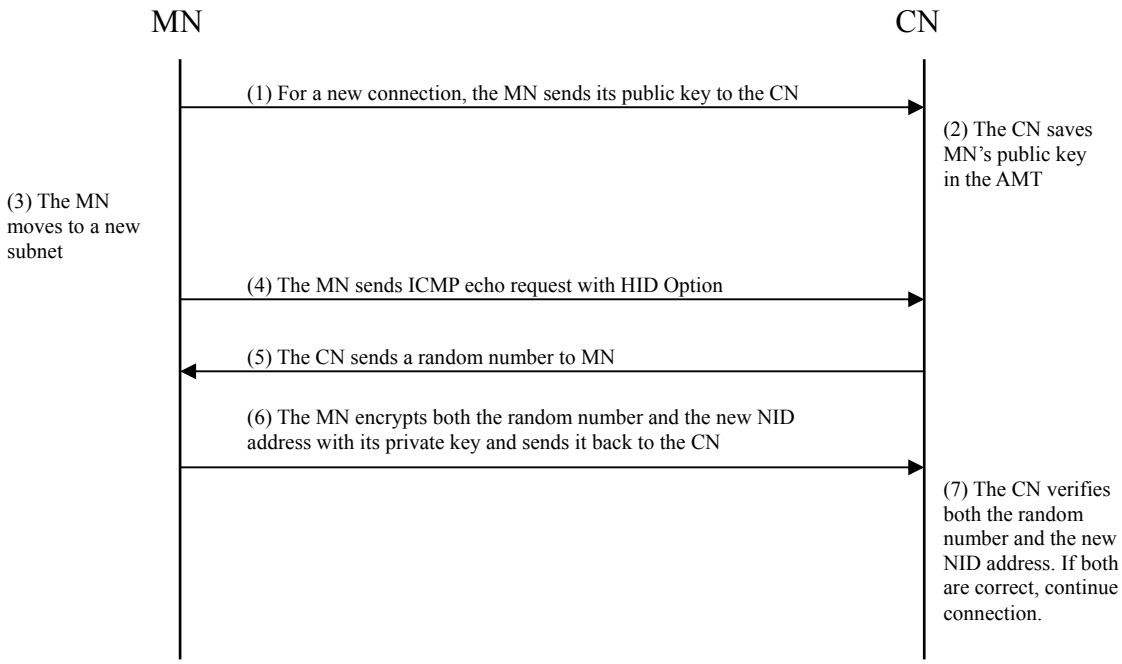


Figure 17: Authentication Procedure.

6.5 Potential Flaw of Our Approach

There are some applications, e.g. FTP, that contain IP address information within the data. Most of them will operate correctly in the IPD architecture, since the HID address used in the upper layer will be mapped to the latest NID address. However there are still some applications that are required some modifications to support our method. For instance, NAT (Network Address Translation) server should update the private and public IP address mapping when intercepting ICMP echo request with HID Option sent by a mobile node in private network.

7 Conclusions and Future Work

This paper presented a scheme to separate the dual purposes of the original IP addresses into HID and NID addresses, and to use this address separation to support host mobility. Unlike previous approaches, the IPD approach does not require any mobility agents. To this end, a table maintaining the binding of IP addresses to transport-layer identities or vice versa is introduced. The binding information is queried or modified using the current DNS system with secure DNS updates, and exchanged directly between two communication peers without any third party's interventions. Routing paths between mobile nodes and correspondent nodes are saliently optimized. Our approach elegantly lends itself to multi-tier system and can support mobility in an IPv4 and IPv6 co-existent network. Simulation results show that our proposal incurs insignificant overhead since the only extra processing time required results from translations between HID and NID addresses. For most packets (TCP or UDP packets), not any HID Option overhead is required and the size of IP datagrams is exactly the same as that of the original ones.

To avoid the connection hijacking problem, we could add an authentication field in HID Option when sending an ICMP echo request to update a correspondent node's AMT. This allows the correspondent node to determine whether the packet with a new NID address belongs to the original peer. The security problem, smooth handoff issues, mobility support in multi-tier system and IPv4 and IPv6 co-existed networks will be further investigations in the future.

References

[1] D. Eastlake 3rd, "Secure domain name system dynamic update," *RFC 2137*, IETF, Apr 1997.

[2] P. Ferguson and D. Senie, "Network ingress filtering: Defeating denial of service attacks which employ IP source address spoofing," *RFC 2267*, IETF, Jan 1998.

[3] S. Gupta and A.L.N. Reddy, "A client oriented, IP level redirection mechanism," *Proc. IEEE Infocom '99*, Mar 1999.

[4] D. B. Johnson, "Mobile host Internetworking using IP loose source routing," *Tech. Rep. CMU-CS-93-128, Carnegie Mellon Univ., Pittsburgh, PA*, Feb 1993.

[5] D. B. Johnson, C. Perkins and J. Arkko, "Mobility Support in IPv6," Internet Draft, IETF, May 2002, (work in progress).

[6] G. Montenegro, "Reverse tunneling for Mobile IP, revised," *RFC 3024*, IETF, Jan 2001.

[7] P. Mockapetris, "Domain names – concepts and facilities," *RFC 1034*, IETF, Nov 1987.

[8] P. Mockapetris, "Domain names – implementation and specification," *RFC 1035,* IETF, Nov 1987.

[9] C. E. Perkins, "IP mobility support," *RFC 2002*, IETF, Oct 1996.

[10] C. E. Perkins, “IP encapsulation within IP,” *RFC 2003*, IETF, Oct 1996.

[11] C. E. Perkins, “Minimal encapsulation within IP,” *RFC 2004*, IETF, Oct 1996.

[12] C. E. Perkins and D. B. Johnson, “Route optimization in Mobile IP,” Internet Draft, IETF, Sep 2001, (work in progress).

[13] A. C. Snoeren and H. Balakrishnan, “An end-to-end approach to host mobility,” *ACM MobiCom 2000, Boston, MA*, Aug 2000.

[14] F. Teraoka, Y. Yokoro, and M. Tokoro, “A network architecture providing host migration transparency,” *Proc. ACM SIGCOMM 91*, pp.209-220, Sep 1991.

[15] F. Teraoka, K. Uehara, H. Sunahara, and J. Murai, “VIP: A protocol providing host mobility,” *Commun. ACM*, Aug 1994.

[16] S.-L. Tsao and J.-C. Liu, “Mobility support for IPv4 and IPv6 Interconnected Networks based on Dual-Stack Model,” Internet Draft, IETF, Feb 2000.