PM2HW2N 2013 - 8th ACM Intl. Wkshp. Performance Monitoring, Measurement and Evaluation of Heterogeneous Wireless and Wired Networks. Barcelona, Spain. Nov 2013.

Network Layer Soft Handoff for IP Mobility

Ditchaphong Phoomikiattisak
School of Computer Science
University of St Andrews, UK
dp32@st-andrews.ac.uk

Saleem N. Bhatti
School of Computer Science
University of St Andrews, UK
saleem@cs.st-andrews.ac.uk

ABSTRACT

We present an empirical evaluation of network-layer soft handoff for IP mobility. Such functionality is not currently available for Mobile IP. Our new approach, based on the *Identifier Locator Network Protocol (ILNP)*, requires no additional network entities, such as proxies, and it does not require modification of any routing protocols. Only the communicating hosts need to have their end-system protocol stacks updated so it is incrementally deployable. In our performance evaluation, we find that soft handoff minimises packet loss, with the observed packet loss during handoff being no worse than the natural loss of the end-to-end path.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Wireless communication*; C.2.2 [**Computer-Communication Networks**]: Network Protocols

Keywords

Mobility, Soft Handoff, Identifier, Locator, ILNP

1. INTRODUCTION

Mobility is currently enabled below the network layer by lower level network technologies, such as 3G networks or wireless LAN (WLAN / WiFi) networks. Currently, it is challenging to transfer an existing communication session across such network technologies. While work is in progress to allow such transfer of communication sessions at the lower layers (e.g. IEEE 802.21 WG[1]), the 'natural' place for such interworking across technologies is the network layer.

1.1 IP mobility today

The fundamental problem for mobility using IP can be understood by considering the bindings of an IP address within the protocol stack [5], as shown in Table 1. The second column of Table 1 shows that the IP address is today used in state information for the transport layer, but is also assigned to a specific physical interface: the IP address acts as an *identifier* at the transport and physical layer. Effectively, a transport layer communication session is bound 1:1 to a physical interface. The IP address also has significance for routing at the network layer: the IP address acts as a topological *locator*. This semantic overloading means that mobility needs special treatment for IP.

Table 1: Use of names in IP and ILNP.

Protocol layer	IPv4 and IPv6	ILNP (ILNPv6)
Application	FQDN, IP address	FQDN or app.-specific
Transport	IP address	Node Identifier (*NID*)
Network	IP address	Locator (*L64*)
(interface)	IP address	dynamic binding

Mobile IPv4 (MIPv4) [15] uses implicit *indirection* to support mobile nodes. The mobile node has a permanent *home address (HoA)* at its *home network (HN)*, which acts as an identifier, as well as a *care-of-address (CoA)*, which acts as a locator and is assigned from the *foreign network (FN)* into which it roams. From an engineering viewpoint, proxies – a *home agent (HA)* and a *foreign agent (FA)* – connected with IP-in-IP tunnels (between HoA and CoA) are used to give the impression that a mobile node is topologically stable. This creates sub-optimal routing, and end-to-end integrity of the transport protocol session may be lost. The proxies become potential single points of failure and performance bottlenecks, as well as providing an attack vector, e.g. for man-in-the-middle or denial of service attacks.

Mobile IPv6 (MIPv6) [25] initiates communication via the HA as for MIPv4. However, it can then use *redirection* to signal (using *Binding Update (BU)* messages) a topologically correct address from its new location to remote hosts. Architecturally, this still causes problems, as the identity of communication sessions change – end-to-end state invariance is lost at the transport layer. This in turn impacts any end-to-end protocols which must now be engineered to be 'mobility aware' e.g. IPsec.

1.2 Loss during handoff

A major issue that remains is handoff performance: high packet loss could occur when a mobile node moves between networks. Many extensions have been proposed to overcome this problem such as Hierarchical Mobile IPv6 (HMIPv6) [30]

[1] http://www.ieee802.org/21/

PM2HW2N'13, November 3–8, 2013, Barcelona, Spain.

http://dx.doi.org/10.1145/2512840.2512843 ...$15.00.

and Fast Handover for Mobile IPv6 (FMIPv6) [26]. HMIPv6 introduces the Mobility Anchor Point (MAP) to manage mobility of mobile nodes in its local region. So, when a mobile node moves within the same region, the handoff latency and loss are reduced. FMIPv6 reduces the handoff loss by allowing a mobile node to detect that it has moved to another network when it is still connected to its current network. The new CoA (NCoA) can also be determined ahead of the handoff, which can be used immediately after it moves to that subnet. The previous access router also creates a tunnel to provide continuity between use of the previous CoA (PCoA) and NCoA. Any delayed packets arriving at the PCoA would be forwarded to the NCoA.

The third column of Table 1 shows the namespaces in ILNPv6 – an instance of the *Identifier-Locator Network Protocol (ILNP)* [3, 5] engineered as a superset of IPv6 [4, 6, 7]. ILNPv6 supports soft handoff to minimise packet loss. ILNPv6 uses *distinct namespaces* with *dynamic bindings* to implement mobility. The *Node Identifier (NID)* has no topological semantics, and is used at the transport layer. It is a distinct namespace from the topologically-significant *Locator (L)*, which is used at the network layer for routing. Additionally (not visible in Table 1), there are one-to-many dynamic bindings between *NID* and *L64* values; separately there are dynamic bindings between physical interfaces and *L64* values. This means that, in its simplest form, mobility in ILNP is implemented by changing these dynamic bindings between *NID* and *L64* values, and between *L64* values and interfaces. *L64* values for a node can change as the node is mobile without impacting end-to-end state invariance, as the *NID* value does not change once a session is in progress. From an engineering viewpoint, an *NID* value can be derived in the same way as the host-ID for IPv6, and the *L64* value is an IPv6 routing prefix [18].

1.3 Structure of this paper

Our contribution in this paper is the implementation and performance evaluation of a network layer soft handoff mechanism for supporting IP mobility, functionality which is not currently part of the IETF Mobile IP standards. After a brief look at related work in Section 2, we describe ILNP in Section 3. We then describe our implementation and experimental results in Section 4. After a short discussion in Section 5 we conclude in Section 6.

2. RELATED WORK

The Host Identity Protocol (HIP) [20, 21] uses public and private key pairs to separate identity of a host from its IP address. The public key is used as a Host Identifier by higher layer protocols (such as TCP) to represent the host identity, whilst an IP address is used for routing. Hence, HIP requires the deployment and use of strong cryptography, even within protected enclaves. This could impair performance, both of application protocols and of network interface, due to a higher computational burden in per-packet cryptography.

The *Locator Identifier Separation Protocol (LISP)* [16] is a network-based implementation. LISP uses the 'map-and-encap' method for mapping IP addresses into a separate routing schema and encapsulating IP packets sent between LISP routing nodes. This requires additional management and control modules, either introduced into some current network devices or new network devices. Also, the map-and-encap function increases the per-packet protocol overhead and increases the routing complexity of the deployed network. An extension - LISP mobile node (LISP-MN) has also been designed [28], which has a handoff latency of 1.5 RTT (further optimisations pending).

There is also an extension of MIPv6, *Proxy Mobile IPv6 (PMIPv6)* [17]. This approach enhances MIPv6 to be a complete network-based solution. Mobile nodes do not get involved in the mobility management process. A mobile node still has a *HoA* and a *CoA*, but PMIPv6 introduces another entity, a *Mobile Access Gateway (MAG)*, to track movements of mobile nodes on its link and signal a *Proxy Binding Update* message to the mobile node's *Local Mobility Anchor (LMA)* - similar to a HA in MIPv6. The traffic between MAG and LMA uses a bidirectional tunnel. To minimise the handoff latency of PMIPv6, a Fast Handover mechanism is proposed [31]. It applies the concepts of FMIPv6 to improve PMIPv6.

Network-based solutions (such as LISP, MIPv6, HMIPv6, PMIPv6, FMIPv6) require additional network entities. These can often have the advantage that they 'hide' mobility from non-mobile (legacy) nodes, improving backwards compatibility. However, the addition of new network entities adds complexity to the current network landscape: new equipment may be required (increased CAPEX) and there is an overhead for operations (administration overhead for network and systems management, and so an impact on OPEX). Additionally, new network entities, such as proxies, may introduce a single point of failure, become performance bottlenecks and also introduce new points that need to be monitored and protected from security attacks.

Host-based solutions, such as ILNP and HIP, have the potential disadvantage that they require updates to the end-system protocol stack. However, today's modern operating systems, for desktop and mobile devices, push out software updates regularly, so we take the position that deployment of such updates could be managed easily.

3. OVERVIEW OF ILNP

ILNP enables harmonious integration of mobile nodes and mobile networks [2, 27]. ILNP is an end-to-end architecture, so it can be deployed by modification of end-hosts without requiring modification to current core network devices, e.g. router upgrades.

3.1 ILNP architecture

As shown above in Table 1, ILNPv6 (ILNP implemented as a superset of IPv6) has a very different architecture to IP. ILNP provides a cleaner set of namespaces for the protocol stack. Applications can use their own namespace, but default to using FQDNs, consistent with a long-standing IAB Recommendation [13]. Transport-layer protocols should use only a *Node Identifier (NID)*, which has no topological significance. The *NID* always represents a *node* rather than an interface on a node. The network layer uses topologically-significant *Locator (L64)* values only for routing and forwarding. Nodes can choose to use multiple *NID* and *L64* values and use multiple interfaces simultaneously, by adjusting dynamic bindings between them as required.

In summary, for ILNP, *end-to-end protocols bind to NID values*, which are used above the network layer only. *L64* values are, effectively, names of networks (e.g. a network prefix as used today), with dynamic bindings between *NID* values, *L64* values, and between *L64* values and interfaces.

3.2 ILNPv6

ILNP can be implemented as a superset of IPv6, which we call *ILNPv6*. *L64* and *NID* values are encoded into the IPv6 address space – see Figure 1 [6]. The top 64 bits, *L64* , has the same syntax and semantics as a routing prefix in IPv6. So, its value can be determined using existing mechanisms (e.g., IPv6 Router Advertisements). The lower 64 bits, *NID* , has the same syntax as the IPv6 Interface Identifier, but different semantics. A *NID* value names a node, not a specific interface of the node, and is not used for routing in the core network. The *NID* values can be chosen in the same ways as an IPv6 Interface ID is chosen [18]. Therefore, ILNPv6 is backwards-compatible and incrementally deployable with respect to IPv6.

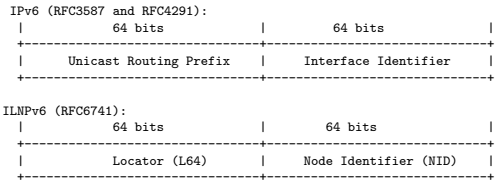

Figure 1: Encoding of NID and L64 values into the IPv6 unicast address bits. An ILNPv6 L64 value has the same syntax and semantics as an IPv6 routing prefix. An ILNPv6 NID value has the same *syntax* as an IPv6 Interface Identifier, but its *semantics* are for a *node* identity, not an *interface* identity.

3.3 Mobility with ILNP

There are two cases for mobility: *host mobility* and *site mobility*; both can be supported by ILNP using, essentially, the same mechanism. In this paper we discuss mainly the former, a scenario in which an individual host (not a whole site) is mobile.

There are two functions to consider in mobility: *rendezvous*, to permit incoming connections to a mobile host; and *handoff* to allow a mobile host to maintain communication sessions that are active during location changes. This paper provides an empirical study on the handoff mechanism of ILNP, and rendezvous will be discussed based on our previous work.

As ILNP supports dynamic bindings between *NID* and *L64* values, handoff is implemented, effectively, by manipulating those bindings. Additionally, one-to-many bindings between *NID* and *L64* values are permitted, which means that a mobile host can 'belong' to, say, two networks simultaneously. This means that *network layer soft handoff* is possible for ILNP, something that is not possible for Mobile IP today. In ILNP, a mobile node learns its new *L64* value from the network it has moved to and starts to use it.

A handoff occurs when an ILNP node changes topological location, from one network to another. In Figure 2, *hard handoff* occurs when a mobile host, X, simply uses a new *L64* value ($L2_X$) discarding its previous one ($L1_X$). This could cause packet loss during the period that the correspondent node, Y, still uses the old *L64* value. Packet loss could occur due to: (a) the packets from Y to X sent with the incorrect destination *L64* values; and (b) packets from X to Y sent with the new source *L64* value which Y does not yet know of. In *soft handoff*, ILNP overcomes this problem by allowing X to have bindings with both $L1_X$ and $L2_X$ when it enters an overlap region between the two networks (e.g. through radio-cell coverage). With soft handoff, the packet loss during handoff is minimised, and may be close to zero. Some packets could still be lost if they were sent from Y to X before X initiated handoff, and reach the old *L64* value after handoff is complete (e.g. due to a high delay path chosen from a multipath forwarding scheme). For both hard handoff and soft handoff, the latency is 1 RTT.

An example of the handoff process is shown in Figure 3; the mobile host H moves from the site network L_1 to the site network L_2. Once H enters the overlap region, at position (2), and detects the new Locator value(s) (as prefixes in IPv6 Router Advertisements), it will: (a) send *Locator Update (LU)* [4] messages (similar to MIPv6 Binding Update messages) to all correspondent nodes, to notify the change in its *L64* value(s) and maintain the existing active sessions; and (b) if required, securely update its relevant DNS entries (e.g. the *L64* record) to allow incoming sessions to be correctly established. At this point, H can discard the old *NID -L64* binding (hard handoff), or maintain the previous *NID /L64* values until either the signal has faded or it recognises that all flows from correspondent nodes are using the new Locator value (soft handoff). Enabling soft handoff is highly desirable for IP mobility support, as it minimises disruption to traffic during handoff.

4. EVALUATION

In order to investigate the performance of our approach, we have created an implementation of a subset of ILNPv6 for testing within a single lab network. The scope of this evaluation is to measure only the *indicative performance* of the network layer handoff of ILNPv6 without any effects from upper layer protocols or from lower-layer wireless protocols. This is not an *absolute* performance evaluation; simulations for scenarios using ILNPv6 with different upper layer protocols (e.g. TCP, UDP and DCCP) and with different wireless mechanisms (e.g. WLAN, 3G, LTE) would be required for *operational performance* evaluations.

4.1 ILNPv6 overlay

Our implementation has been created as a proof-of-concept for mobility support in ILNPv6. It operates as an *overlay network* using UDP/IPv6. This approach was selected for fast development, but a kernel OS implementation is currently in progress. Also, we wished to focus on protocol dynamics rather than on specific engineering details. Use of an overlay is sufficient to demonstrate the operation of ILNPv6 as well as to evaluate the approach using new namespaces. The overlay system is written in C, using UDP/IPv6 on Linux with the standard sockets API. The emulated protocol layers are illustrated in Table 2. UDP/IPv6 multicast is used to emulate link layer collision domaina, effectively, while the actual operation of packet forwarding, based on *NID* and *L64* values, is handled by the ILNPv6 layer.

The application protocol is a reliable packet stream. Each packet in each session has a unique numeric ID allowing the sender to determine if a packet is successfully sent (i.e. by receiving an acknowledgement for the packet). The simple transport protocol (STP) is an unreliable, connectionless protocol. It is a 'dummy' protocol that performs multiplexing of data from the ILNPv6 layer to appropriate applications. The ILNPv6 layer uses a modified IPv6 header as

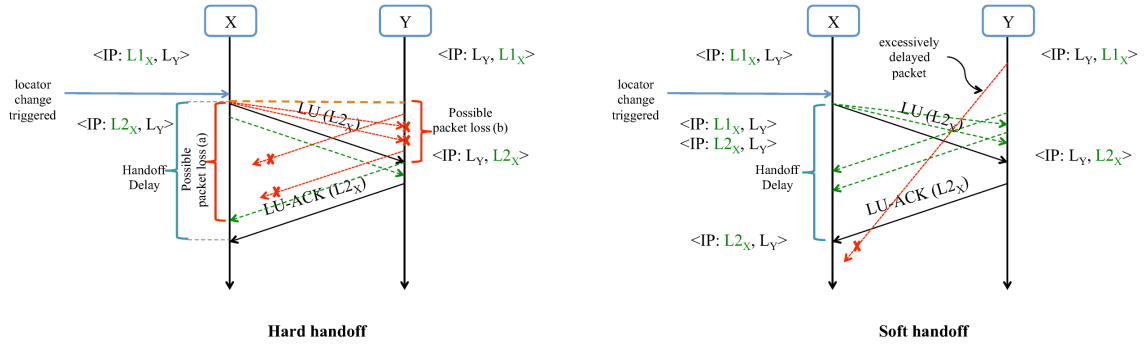

Figure 2: A comparison of hard handoff and soft handoff. In hard handoff (left), mobile node X simply discards its current Locator value ($L1_X$) and uses the new one that has been learned ($L2_X$), e.g. from an IPv6 Router Advertisement as it enters a new radio cell. Packet loss (red/dotted arrows) could occur for X and Y. Once the correspondent node Y knows about the updated L64 value, from the Locator Update (LU) sent by X, subsequent packets will be accepted by X (green/dashed arrows). In soft handoff (right), X uses both $L1_X$ and $L2_X$ simultaneously until it receives the LU Acknowledgement sent by Y, minimising packet loss during the handoff. A packet that is highly delayed before handoff could still be lost, as for Mobile IP.

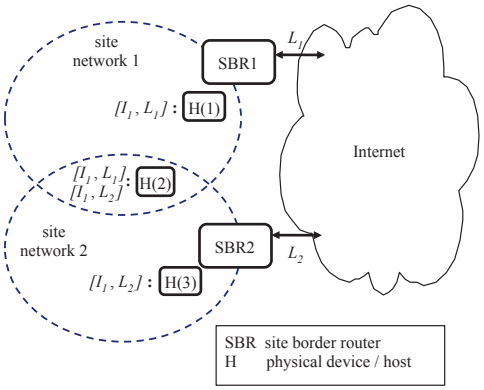

Figure 3: An example scenario of mobility with ILNP (soft handoff). At position (1), host H is in a site network using Locator L_1 and has a binding [I_1, L_1]. H then moves to a site network L_2. When it enters the overlap region, at position (2), it obtains a new Locator value and also has binding [I_1, L_2], yet still maintains binding [I_1, L_1]. When it leaves the overlap region, at position (3), or it detects all correspondent nodes are using the new binding, it discards the old binding.

shown in Figure 1. These ILNPv6 packets are carried in multicast UDP/IPv6 packets.

We use IPv6 multicasting to emulate the different logical networks on the same physical network. Each network has a unique *L64* value which simply maps to an IPv6 multicast group. ILNPv6 hosts appear on different networks (use different *L64* values) by the overlay becoming a member of different multicast groups at the IPv6 layer, as shown in Figure 4. We chose to run the experiment on a wired network instead of wireless network to investigate the actual behaviour of ILNPv6 without unpredictable noise and interference. However, we did emulate several network scenarios e.g. high loss and/or high delay environments.

Table 2: Overlay protocol stack of the prototype.

Protocol layer	Protocol	Comment
Application	Packet transfer	Packets with a numeric ID
Transport	STP	a simple transport protocol
Network	ILNPv6	ILNPv6 Overlay
Link	UDP/IPv6	Unreliable link layer

4.2 Experiment configuration

Based on the topology in Figure 4, we use the application to emulate two kinds of traffic: Voice over IP (VoIP) traffic flows and streamed video flows (non real-time). These are sent from H1 to H2. The VoIP traffic was emulated Skype traffic using a packet size of 300 bytes to generate a 64 kbps flow, based on previous studies [12,14]. The streamed video flow was generated by sending 1400 byte packets (the maximum payload size that we could send for the MTU[2]) every 17 milliseconds. This would generate 658 kbps traffic which is slightly more than the average data rate of YouTube, 632 kbps [32]. Each flow was 65 seconds long.

Both VoIP traffic and video streaming traffic were run over emulations of different network conditions. First, three different delay scenarios: LAN, MAN and WAN. The LAN scenario was the usual network conditions in our test environment (a teaching lab in the School), while the other two were emulated by adding a delay of 10 ms and 100 ms in each direction between H1 and H2. For each delay scenario, packet loss rates of 0%, 5% and 10% were also emulated for each direction of traffic. So, these produce round trip path loss of 0%, 10% and 20%, respectively. All network delay and loss conditions were emulated with the widely-used Linux network emulation software *netem*[3].

[2]After adding the 40 byte ILNPv6 header, 12 byte STP header (our overlay headers), 8 byte UDP header, and 40 byte IPv6 header, with a 1400 byte payload, we have the Ethernet Maximum Transfer Unit (MTU) of 1500 bytes.

[3]`http://www.linuxfoundation.org/collaborate/workgroups/networking/netem/`

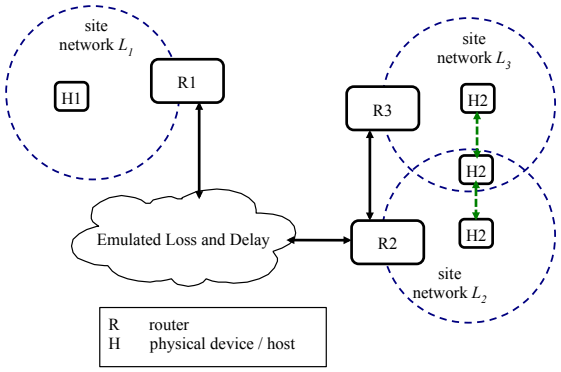

Figure 4: The topology for the experiment. The hosts H1 and H2 reside in different networks, with Locator values of L_1 and L_2, respectively. The green / dashed arrows identify movements of H2 between site networks using *L64* values L_2 and L_3, generating a handoff. The handoffs occur every 5 seconds.

We performed 20 runs for each delay / loss combination listed above. In each run, H2 moves between site network L_2 and site network L_3 every 5 seconds as the flow is in progress. This is emulated by changing the IPv6 multicast group of H2. H1 would be informed about the change of *L64* value of H2 from Locator Update (LU) messages generated by H2. To enable *soft handoff*, an LU acknowledgement (LU-ACK) is also sent by H1 back to H2. If the sender does not receive the LU-ACK in a given timeframe, the LU would be retransmitted. The retransmission timeout for emulated LAN, MAN and WAN networks are set to 5ms, 25ms and 210ms respectively, which is slightly higher than the round-trip time (RTT). Nevertheless, if an LU-ACK has not been received after 20 LU transmissions, the mobile node assumes the handoff has failed.

4.3 Results

Both loss and delay were measured at the application layer of the sender side (H1, in Figure 4). The experienced loss is calculated from the number of sent packets and the number of acknowledged packets. The experienced delay is measured using half of the round-trip time (RTT/2) of each acknowledged packet, as the path was symmetric. The handoff delay is measured at the network level at the sender side from the duration between a LU message being sent and the associated LU-ACK being received as shown in Figure 2. Since each flow was run for 65 seconds and a handoff occurs every 5 seconds, handoffs occurred a maximum of 13 times during a run. Our analysis is based on the timings for 11 of these handoffs only, ignoring the first and the last handoffs, to avoid errors due to small variations in the start and end times of flows between the hosts.

The mean packet loss of each test scenario is shown in Table 4 and Figure 6. The numbers, in the same emulated loss environment, are stable across the different emulated delays, and are close to the emulated values. So, the handoff between networks does not introduce additional packet loss compared to the 'natural' value, even if some LU/LU-ACK messages are lost.

The mean packet delay of each test scenario can be seen in Table 5 and Figure 7. The measured delay is unlikely to be impacted by packet loss because the measured values are close to the emulated values. The video streams produced slightly higher packet delays than the VoIP traffic due to their larger packet size.

The mean handoff delay of each test scenario is presented in Table 6 and Figure 8. The handoff delay is directly proportional to the number of LUs sent, as displayed in Table 7 and Figure 9. These values are directly impacted by the emulated loss, which may cause the LU/LU-ACK handshake to take longer. For 0% emulated loss, the number of sent LUs per handoff is very close to 1 since loss is unlikely to occur. The handoff delay is close to twice the emulated one-way delay (i.e. it is close to the RTT), as expected. In the scenarios of 10% and 20% emulated loss, the handoff duration increases. However, the number of sent LUs per handoff do not change significantly. For 10% emulated loss, around 1-2 LUs from the total 11 handoffs are expected to be lost and require retransmission. Thus, in total, around 12-13 LUs are sent and this is around 1.1-1.2 LUs per handoff. Likewise, for 20% emulated loss, around 2-4 LUs are expected to be lost, resulting in around 13-15 LUs being sent in 11 handoffs, which is around 1.2-1.4 LUs per handoff.

So, we observed no significant gratuitous packet loss when the mobile host moved between networks even when some of the LU/LU-ACK messages were lost. In the higher loss scenarios, there was only a slight observable impact on the handoff duration. The higher delay scenarios did not impact the observed loss rates. So, we believe that ILNPv6 could work well in a range of wireless network scenarios including those with a high loss and/or a high delay.

4.4 Comparison to MIPv6 and its extensions

ILNPv6 uses network-layer soft handoff directly between communicating nodes, which our experiments show to be effective and has good performance. Extensions of MIPv6, such as FMIPv6, attempt to minimise handoff latency to reduce packet loss during the handoff period. While, the MIPv6 mechanism with Binding Updates has been stable for many years, it is a hard handoff mechanism and cannot provide seamless handoff. Many studies have investigated its performance and proposed improvements, such as: using a combination of HMIPv6 and FMIPv6 [24]; customised buffer management [22]; and leveraging link layer information [1]. While such mechanisms have been shown to reduce packet loss during handoff, ILNPv6 still has advantages over such mechanisms:

- *Complexity and Scale.* ILNP does not introduce new entities to the network, and only mobile nodes need to be upgraded. MIPv6 and its extensions require additional entities, leading to a more complex engineering landscape. Complex systems generally scale worse than simple systems.

- *Overhead.* Signalling overhead for handoff with ILNPv6 is minimal. As shown in Figure 2 ILNPv6 requires only LU/LU-ACK exchange for handoff. MIPv6 and its extensions require much more signalling overhead. Compare Figure 2 (black arrows) to Figure 5, the latter showing the handoff overhead of Fast Handover PMIPv6 in Predictive mode.

- *Security.* ILNP introduces no new security risks: it is at least as secure as the current IPv6. MIPv6 and its extensions introduce new network entities that present the possibilities of new attack vectors. For example, an attacker could impact communication by disrupting a proxy rather than the communicating end-systems, with a successful attack impacting all users of that proxy, not just two communicating nodes.

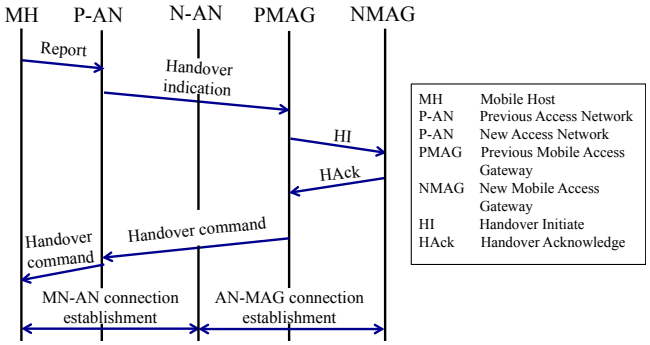

Figure 5: The handoff process of Fast Handover PMIPv6 in Predictive mode. (From RFC5949 [31].)

5. DISCUSSION

We present some discussion and critical analyses relating to the use of ILNP mobility for the global Internet. We make comparisons with Mobile IP, where appropriate and discuss issues of scalability.

5.1 Rendezvous and use of DNS

We have focussed on indicative performance of handoff and reduction of packet loss during handoff. For initiating communications to a mobile host – *rendezvous* – Mobile IP correspondent nodes would use DNS. In Mobile IP, the DNS lookup always resolves to the *Home Address (HoA)* at the *home network (HN)* of the mobile node. For ILNP, DNS would be used with new records for *NID* and *L64* [9] values, but the *L64* value for the mobile node would be updated in the DNS as the node moves. Previously, others have also proposed DNS for use in supporting mobility [23, 29].

Table 3: A record query rates $[1/s]$ (internal to site)

Data set	mean	std dev	max	95%-tile	99%-tile
TTL=1800s	1.31	2.98	176	8	22
TTL=30s	1.58	3.57	168	8	24
TTL=0s	2.36	3.48	68	8	15

This data is taken from [11].

DNS records that hold *L64* values need to have a very low time-to-live (TTL), so that stale *L64* values are not cached. A previous emulation study [19] reports that using a low TTL value of hundreds of seconds should not impact significantly on DNS. Our previous empirical evaluation of an operational DNS deployment [11] shows that values of TTL as low as zero have no significant impact on DNS load. Table 3 (taken from [11]), shows the mean query rate for A records for servers internal to our site: request rates increase when the TTL decreases, as expected, but remains relatively low and still manageable. So, similar TTL values for *L64* records should yield similar DNS load.

The additional overhead of the *L64* update to the DNS is of the same order as the Mobile IP update to the HA when the mobile node moves. Security issues for protecting and validating the MIPv6 Binding Update and the ILNPv6 Locator Update are also similar. ILNP's DNS records (RFC6742 [9]) are already implemented in widely-used DNS software: BIND/*named*[4] and NSD/*Unbound*[5], both of which support DNS security. DNS security is being implemented independently of ILNP and is widely deployed also. As DNS is already deployed worldwide, ILNP does not incur any additional overhead, e.g. managing proxies and tunnels.

5.2 Mobile networks

While this paper focusses on mobile hosts, ILNP supports mobile networks using essentially the same mechanism as described here. It also allows optimisation of mobility management for a whole site/network via ILNPv6 enabled site-border routers [8,10,27]. For Mobile IP, the situation is more complex. In the past, the NEMO IETF WG[6] proposed a separate mechanism for mobile networks. The move is now to produce extensions to Mobile IPv6 for mobile networks, under the NETEXT IETF WG[7]. However, the fundamental constraints of the IP addressing model remain (see Section 1.1), and so use of indirection and redirection are likely to remain, with the associated overhead of proxies and tunnels.

5.3 Backwards compatibility with IPv6

ILNPv6 is an end-host enhancement of IPv6 so could be deployed in the current IPv6 backbone without requiring any changes or upgrades to IPv6 routers. To enable backwards compatibility with IPv6, for example, when an ILNPv6 host communicates with an IPv6 host, a new *ILNPv6 Nonce Option* [7] is introduced. This option is included in the first few initial ILNPv6 packets. When the receiver receives initial packets containing the ILNPv6 Nonce Option, if ILNPv6 is supported, it would respond with ILNPv6 packets and an ILNPv6 session is established. If the receiver does not support ILNPv6, such initial packets would be dropped and an ICMP packet (i.e. 'Parameter Problem') would be sent back. The initiator could re-establish the communication with IPv6 if required. Please refer to [6] for more details.

5.4 Security and privacy

In general, ILNPv6 security is at least as good as IPv6 security. Additionally, to prevent off-path attacks, e.g. forged LU messages, ILNPv6 defines a Nonce Option [7]. If this is insufficient, then the use of IPsec is recommended [2, Sec. 4.4], with *NID* values used as part of the IPsec Security Association, in place of IP addresses. As already mentioned above, DNS security is being deployed independently of ILNPv6, and can be used for updates to *L64* values. Another feature is that ILNPv6 can support both location privacy and identity privacy with mobility [8, 10].

[4]ISC BIND v9.9.3 `http://goo.gl/Nc88p`
[5]NSD v3.2.15 `http://goo.gl/NBfZK`
[6]`http://datatracker.ietf.org/wg/nemo/charter/`
[7]`http://datatracker.ietf.org/wg/netext/`

6. CONCLUSION

We have presented an enhancement of the current Internet architecture that provides mobility support with soft handoff. Our proposal is based on the use of the *Identifier Locator Network Protocol (ILNP)*, a superset of IPv6. We emulated conditions for LAN, MAN and WAN scenarios for mobility. Our empirical evaluation using a testbed emulation shows that the packet loss with soft handoff is minimal and incurs low overhead. The soft-handoff mechanism works directly between communicating hosts and does not require new network entities.

We have made qualitative assessments and comparisons against existing MIPv6 extensions and find that ILNPv6 offers simpler operation, lower overhead and poses no new security risks.

7. REFERENCES

[1] M. Alnas, I. Awan, and R. D. W. Holton. Performance Evaluation of Fast Handover in Mobile IPv6 Based on Link-Layer Information. *J. Syst. Softw.*, 83(10):1644–1650, Oct. 2010.

[2] R. Atkinson, S. Bhatti, and S. Hailes. ILNP: Mobility, Multi-homing, Localised Addressing and Security Through Naming. *Telecommunication Systems*, 42(3):273–291, Dec 2009.

[3] R. Atkinson, S. Bhatti, and S. Hailes. Evolving the Internet Architecture Through Naming. *IEEE JSAC*, 28(8):1319–1325, Oct 2010.

[4] R. Atkinson and S. N. Bhatti. ICMP Locator Update Message for the Identifier-Locator Network Protocol for IPv6 (ILNPv6). RFC 6743, IRTF, Nov 2012.

[5] R. Atkinson and S. N. Bhatti. Identifier-Locator Network Protocol (ILNP) Architectural Description. RFC 6740, IRTF, Nov 2012.

[6] R. Atkinson and S. N. Bhatti. Identifier-Locator Network Protocol (ILNP) Engineering Considerations. RFC 6741, IRTF, Nov 2012.

[7] R. Atkinson and S. N. Bhatti. IPv6 Nonce Destination Option for the Identifier-Locator Network Protocol for IPv6 (ILNPv6). RFC 6744, IRTF, Nov 2012.

[8] R. Atkinson and S. N. Bhatti. Optional Advanced Deployment Scenarios for the Identifier-Locator Network Protocol (ILNP). RFC 6748, IRTF, Nov 2012.

[9] R. Atkinson, S. N. Bhatti, and S. Rose. DNS Resource Records for the Identifier-Locator Network Protocol (ILNP). RFC 6742, IRTF, Nov 2012.

[10] S. Bhatti, R. Atkinson, and J. Klemets. Integrating Challenged Networks. In *Proc. IEEE MILCOM 2011*, Nov 2011.

[11] S. N. Bhatti and R. Atkinson. Reducing DNS Caching. In *Proc. GI2011 – 14th IEEE Global Internet Symposium*, Apr 2011.

[12] D. Bonfiglio, M. Mellia, M. Meo, D. Rossi, and P. Tofanelli. Revealing Skype traffic: when randomness plays with you. In *Proc. SIGCOMM 2007*, pages 37–48, New York, NY, USA, 2007. ACM.

[13] B. Carpenter. Architectural Principles of the Internet. RFC 1958, Internet Architecture Board, Jun 1996.

[14] K. Chen, C. Huang, and C. Huang, P.and Lei. Quantifying Skype user satisfaction. In *Proc. SIGCOMM 2006*, pages 399–410, New York, NY, USA, 2006. ACM.

[15] C. P. (Ed). IP Mobility Support for IPv4, Revised. RFC 5944, IETF, Nov 2010.

[16] D. Farinacci, V. Fuller, D. Meyer, and D. Lewis. The Locator/ID Separation Protocol (LISP). RFC 6830, IETF, Jan 2013.

[17] S. Gundavelli, K. Leung, V. Devarapalli, K. Chowdhury, and B. Patil. Proxy Mobile IPv6. RFC 5213, IETF, Aug 2008.

[18] R. Hinden and S. Deering. IP Version 6 Addressing Architecture. RFC 4291, IETF, Feb 2006.

[19] J. Jung, E. Sit, H. Balakrishnan, and R. Morris. Dns performance and the effectiveness of caching. *IEEE/ACM Trans. Netw.*, 10(5):589–603, Oct. 2002.

[20] R. Moskowitz, P. Nikander, P. Jokela, and T. Henderson. Host Identity Protocol. RFC 5201, IETF, Apr 2008.

[21] P. Nikander, T. H. (Ed), C. Vogt, and J. Arkko. End-host mobility and multihoming with the host identity protocol. RFC 5206, IETF, Apr 2008.

[22] S. Pack and Y. Choi. Performance Analysis of Fast Handover in Mobile IPv6 Networks. In *Personal Wireless Communications*, pages 679–691. Springer Berlin Heidelberg, 2003.

[23] A. Pappas, S. Hailes, and R. Giaffreda. Mobile Host Location Tracking Through DNS. In *LCS2002: 2002 IEEE London Communications Symposium*, Sep 2007.

[24] X. Pérez-Costa, M. Torrent-Moreno, and H. Hartenstein. A performance comparison of Mobile IPv6, Hierarchical Mobile IPv6, fast handovers for Mobile IPv6 and their combination. *SIGMOBILE Mob. Comput. Commun. Rev.*, 7(4):5–19, Oct. 2003.

[25] C. Perkins, D. Johnson, and J. Arkko. Mobility Support in IPv6. RFC 6275, IETF, Jul 2011.

[26] E. R. Koodli. Mobile IPv6 Fast Handovers. RFC 5568, IETF, July 2009.

[27] D. Rehunathan, R. Atkinson, and S. Bhatti. Enabling Mobile Networks Through Secure Naming. In *Proc. IEEE MILCOM 2009*, Oct 2009.

[28] A. Rodriguez Natal, L. Jakab, M. Portoles, V. Ermagan, P. Natarajan, F. Maino, D. Meyer, and A. Cabellos Aparicio. LISP-MN: Mobile Networking Through LISP. *Wireless Personal Communications*, 70(1):253–266, 2013.

[29] A. C. Snoeren and H. Balakrishnan. An end-to-end approach to host mobility. In *Proc. MobiCom 2000*, pages 155–166, 2000.

[30] H. Soliman, C. Castelluccia, K. ElMalki, and L. Bellier. Hierarchical Mobile IPv6 (HMIPv6) Mobility Management. RFC 5380, IETF, Oct 2008.

[31] H. Yokota, K. Chowdhury, R. Koodli, B. Patil, and F. Xia. Fast Handovers for Proxy Mobile IPv6. RFC 5949, IETF, Sep 2010.

[32] M. Zink, K. Suh, Y. Gu, and J. Kurose. Characteristics of YouTube network traffic at a campus network - Measurements, models, and implications. *Comput. Netw.*, 53(4):501–514, Mar. 2009.

Table 4: Mean packet loss from 20 runs

Emulated Delay	Measured Packet Loss* [% ± σ]		
	Emulated loss		
	0%	10%	20%
VoIP (64kbps, 300 byte packets)			
LAN (0 ms)	0.0±0.00	9.9±0.85	19.4±1.20
MAN (10 ms)	0.0±0.01	10.0±0.72	19.1±1.11
WAN (100 ms)	0.0±0.02	9.9±0.88	19.5±1.27
Video Streaming (658kbps, 1400 byte packets)			
LAN (0 ms)	0.0±0.00	9.8±0.62	19.8±1.17
MAN (10 ms)	0.0±0.02	9.7±0.51	19.8±1.16
WAN (100 ms)	0.0±0.01	9.7±0.55	19.9±1.49

* Values are close to emulated values in every case.

Table 5: Mean packet delay from 20 runs

Emulated Delay	Measured Packet Delay* [ms ± σ]		
	Emulated loss		
	0%	10%	20%
VoIP (64kbps, 300 byte packets)			
LAN (0 ms)	1.2±0.18	1.4±0.19	1.4±0.14
MAN (10 ms)	11.6±0.12	11.5±0.12	11.5±0.14
WAN (100 ms)	103.8±0.87	103.5±0.81	103.2±0.75
Video Streaming (658kbps, 1400 byte packets)			
LAN (0 ms)	1.8±0.36	1.9±0.21	1.8±0.12
MAN (10 ms)	12.3±0.36	12.3±0.42	12.3±0.52
WAN (100 ms)	108.1±1.98	108.1±2.44	107.5±2.01

* Values are close to emulated values in every case.

Table 6: Mean handoff delay from 20 runs

Emulated Delay	Measured handoff Delay* [ms ± σ]		
	Emulated loss		
	0%	10%	20%
VoIP (64kbps, 300 byte packets)			
LAN (0 ms)	2.0±0.30	2.8±0.66	3.3±1.44
MAN (10 ms)	22.6±0.46	26.3±2.84	30.2±5.29
WAN (100 ms)	206.2±11.73	240.9±35.32	267.3±45.35
Video Streaming (658kbps, 1400 byte packets)			
LAN (0 ms)	2.3±0.51	3.1±1.07	4.1±1.33
MAN (10 ms)	22.9±0.97	25.5±2.98	27.6±3.91
WAN (100 ms)	205.5±9.30	230.2±30.68	246.4±28.46

* Values are close to the RTT and increase when the loss increases.

Table 7: Mean number of LUs/handoff from 20 runs

Emulated Delay	Sent LU per handoff* [times ± σ]		
	Emulated loss		
	0%	10%	20%
VoIP (64kbps, 300 byte packets)			
LAN (0 ms)	1.0±0.07	1.2±0.14	1.3±0.23
MAN (10 ms)	1.1±0.12	1.3±0.16	1.5±0.26
WAN (100 ms)	1.0±0.07	1.2±0.20	1.3±0.21
Video Streaming (658kbps, 1400 byte packets)			
LAN (0 ms)	1.1±0.10	1.3±0.22	1.5±0.27
MAN (10 ms)	1.2±0.14	1.3±0.18	1.4±0.20
WAN (100 ms)	1.1±0.07	1.2±0.15	1.2±0.14

* Values are close to 1 and slightly increase when the loss increases.

+Error bars for 95% confidence are plotted but may not always be visible.

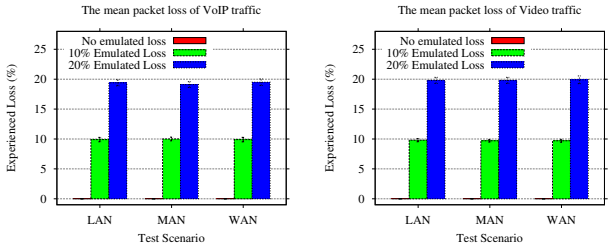

Figure 6: The mean packet loss. Error bars at 95% confidence.+

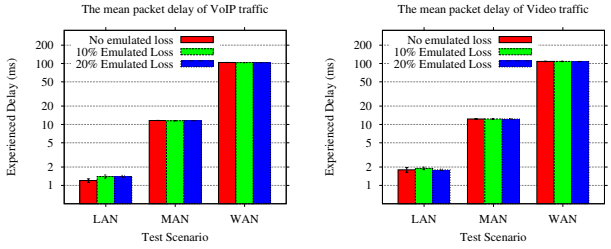

Figure 7: The mean packet delay. Error bars at 95% confidence.+

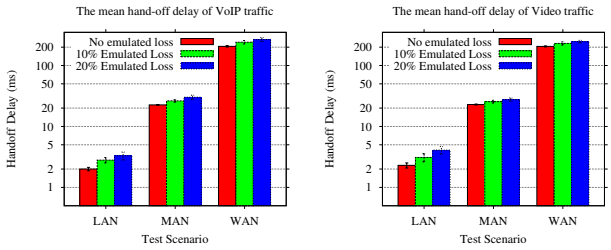

Figure 8: The mean handoff delay. Error bars at 95% confidence.+

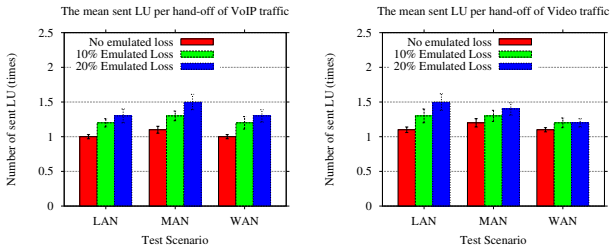

Figure 9: The mean sent LU per handoff. Error bars at 95% confidence.+