The Current State and Future Developments of IPv6 Multihoming

Alan Ford
School of Electronics and Computer Science
University of Southampton
Southampton, SO17 1BJ, United Kingdom
ajf101@ecs.soton.ac.uk

***Abstract*— Multihoming, the connection of a site to multiple upstream providers, is a common requirement for large network sites, for redundancy and link optimality. IPv4 multihoming is achieved through broadcasting routing information, but this has led to enormous growth in the global routing table, and as such as seen as an inviable solution for IPv6 multihoming. This paper summarises these reasons, and the current support for IPv6 multihoming, and goes on to summarise the various new propositions for providing better multihoming facilities. These include modifications of current standards (host-centric multihoming, and multihoming at site exit routers), as well as new propositions, including the use of SCTP, Hash Based Addresses, Level 3 Shim, and the Host Identity Protocol. This paper evaluates these possibilities against various desirable properties, and sees HIP as being the best, and most comprehensive, solution so far proposed to enabling advanced IPv6 multihoming.**

I. Introduction

MULTIHOMING is the ability of a host or site to access remote destinations via more than one upstream connection, usually from different providers. A typical multihoming scenario is illustrated in Figure 1. Multihoming is a common requirement of many medium-sized networks, including many businesses and ISPs, and can occur for two main reasons. One reason is for link redundancy, allowing a site to retain connectivity when one of the links fails. The other main reason is for optimal use of the links, for example increasing bandwidth, or for quality of service (QoS) factors, such as routing traffic over the topographically closest link to minimise delay, or routing low priority traffic over the cheapest link. In addition, new developments in distributed, pervasive computing may lead to mobile devices being transiently multihomed, and also a site may be temporarily multihomed during a change in upstream provider.

Multihoming has always been a feature of the IPv4 Internet, and this paper begins by discussing these methods. The IPv4 solutions used are not highly scalable, however, nor do they solve all issues that arise when a host is multihomed. Due to changes in the protocol and address allocation methods, the IPv4 solutions cannot be directly used in IPv6, and this paper discusses the current situation, and the various possibilities that have been proposed to apply multihoming in the IPv6 environment.

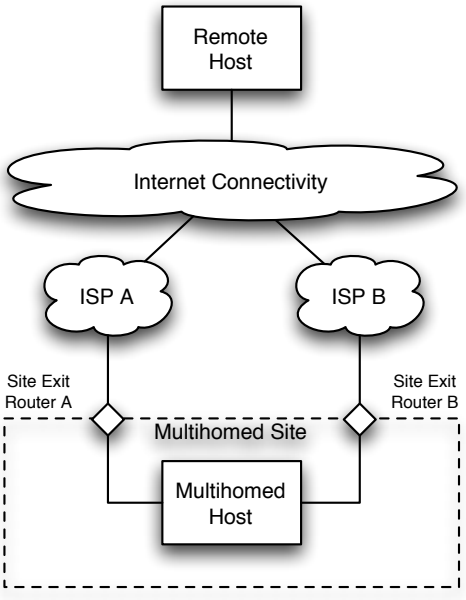

Fig. 1. Example Multihoming Scenario

II. Multihoming in IPv4

The multihoming solutions for a site's IPv4 network will vary based on the size of the site, and the routing agreements it has with its upstream providers. At the highest level, the Default-Free Zone (DFZ) contains those core networks ('tier-1 providers') that have no default route, instead holding a full routing table – having routing details for all network prefixes. These sites are inherently multihomed by having peering agreements for traffic between each other, often at a peering point such as the London Internet Exchange (*LINX*). Each of these providers has a 'Autonomous System' (AS) number [1], which is used for exchanging routing information using BGP [2].

There are two common methods for a medium-sized end-site to be multihomed, using BGP peering with the site's upstream providers, as discussed by Abley [3] and Huston [4]. The first is to use Provider Independent (PI) addresses. These are allocated by a Regional Internet Registry (RIR, such as *RIPE*, for Europe). The end-site then advertises these

addresses and its assigned AS number via BGP to all its upstream providers. This information is propagated to the DFZ allowing all hosts on the Internet to route to this site, following whichever route is optimal.

The alternative method involves using Provider Assigned (PA) addresses, and this is typically used by end-sites too small to obtain PI addresses. Here, a fragment of address space is assigned from one of the site's upstream providers, and it advertises routing to these addresses to all its upstream providers, using BGP. Once again, this routing information is propagated to the DFZ and added to the routing tables.

For smaller end-sites that are unable to undertake BGP peering with their upstream providers, methods based around Network Address Translation (NAT) are sometimes used. For example, a site can use private addressing [5] (such as in 10.0.0.0/8) for its internal hosts, and apply policy routing at the border router, to translate those private addresses to the assigned IP address from the desired upstream provider. A similar scheme can be used if sufficient addresses for all hosts on the network have been allocated from both providers. In such a situation, in order to not break end-to-end communication [6], one provider's public addresses are used on all the internal hosts, and if an alternative upstream provider is chosen, the source address will be translated, in a one-to-one mapping, to an address from the alternative provider's allocation. While this solution provides many of the benefits of multihoming, it does not solve some of the desirable features of multihoming, for example, providing session survivability in the event of a link failure.

III. Evolution of the Problem

The introduction of Classless Inter-Domain Routing [7] (CIDR) in 1994 and 1995 attempted to create an aggregate routing structure for the Internet. In CIDR, addresses are allocated to RIRs in large blocks, who then allocate smaller segments of these blocks to network providers. Customers who get connectivity from these providers then take smaller segments of these address blocks (PA addresses), and these customers are free to split the addressing further, for example to represent their network topology. This leads to a hierarchical routing structure, where routers in the DFZ only need to know routes to the larger address blocks, allowing the routers within those blocks to route traffic appropriately, based on more specific rules. Unfortunately, the elegance of this route aggregation breaks down with multihoming. PI addresses, which are inherently non-aggregatable, and PA addresses that are advertised via the non-aggregated route, both cause additional routes to be added to the DFZ BGP tables. Routing table growth is discussed in detail by Bu [8] and Huston [9], [10]. As their evidence shows, much of the growth has been caused by network multihoming. Bu quotes a figure of 20% – 30% for the proportion of prefixes advertised due to multihoming.

This relentless routing table growth (over 190,000 prefixes are advertised at the time of writing [11]), as illustrated in Figure 2, has caused serious scalability concerns. AS numbers are currently 16-bit, and so far just over 36,000 have been allocated [12] (with around 19,000 in use [11]), and there is

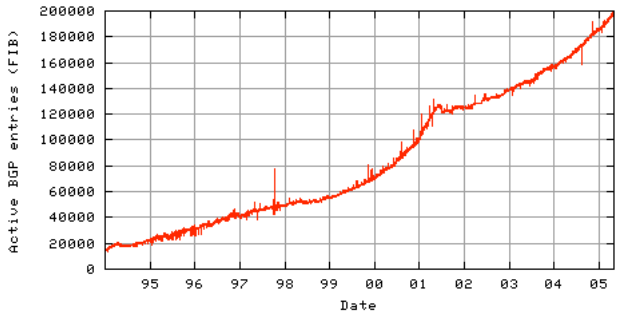

Fig. 2. BGP Entries on AS1221, 1994–2005 [11]

consequently an effort to extend the AS number space to 32-bit [13]. With the design of IPv6, and the introduction of a vastly increased address space (128-bit rather than IPv4's 32-bit), it was felt that the current routing structure of the Internet would not scale to such an increased complexity, and so it was decided to mandate a hierarchical addressing architecture [14]. This is very similar to CIDR, with sites being given small blocks (typically /48s or /64s [15]), aggregated within their provider's larger address allocations. The difference with IPv4 is that all addresses are intended to be assigned in this fashion, with PI addresses almost impossible to obtain. Restrictions on route advertisements means that the BGP-based multihoming solutions from IPv4 cannot be applied to IPv6, and therefore other solutions need to be found.

IV. Current Multihoming Support in IPv6

The IPv6 specification mandates the ability of hosts to have multiple IPv6 addresses [14]. Such multi-addressing can occur for a number of reasons, for example having different addresses for link-local, Unique Local Addresses (ULAs [16]), and globally reachable addresses. In a site that has been multihomed, a host can have multiple global addresses, one assigned for each of the site's upstream providers. This simplest solution is sometimes referred to as 'host multihoming', whereby the host knows its addressing and chooses which source address to use. The alternative is 'site multihoming', where a host is unaware it is multihomed, allowing the site exit routers to handle to multiple routing. This is typically how multihoming is handled in IPv4, and is the problem that has not yet been solved in IPv6. Possible solutions will be discussed in Section VI.

Multi-addressing and host multihoming does, however, go a significant way to providing multihoming support in IPv6. A host is aware of its multiple addresses, and should apply the Default Address Selection algorithms [17] to decide which source and destination addresses to use. It is intended that source address selection is applied by the network layer (if no address is specified), and destination address selection is applied by applications, once a list of addresses is returned from a call such as `getaddrinfo()`. It should be noted that, while address selection is now a standard, many IPv6 implementations today have yet to implement it fully, although simple 'longest matching prefix' rules, as described below, are commonly implemented, as discussed further in Section VII.

The source and destination address selection algorithms are responsible for ordering the list of possible addresses by preference, according to a series of rules. These rules declare a preference: if one possible address matches and another does not, the matching address gets a higher precedence. A host will have a 'policy table', which is similar to a routing table. This matches a prefix (related to an assigned address) with a given precedence, so some prefixes can be preferred above others. It is also possible to 'label' these rules so that they can be matched with equally labelled destination addresses. For source address selection, these comparison rules are as follows:

1) Prefer Same Address (for loopback communication)
2) Prefer Appropriate Scope: Prefer the least specific scope up to the scope of the destination address
3) Prefer Non-Deprecated Addresses
4) Prefer Home Address (for mobility [18], this should be reversible by API)
5) Prefer Addresses on Outgoing Interface(s)
6) Prefer Label Matches
7) Prefer Public Addresses (as opposed to temporary addresses [19], this should be reversible by API)
8) Prefer Longest Matching Prefix

As can be seen, the source address selection algorithm ultimately reaches a stage where the source address is chosen by 'longest matching prefix' – this is preferring a source address that has the most bits (from left to right) in the IPv6 address in common with the destination address. This rule can be superseded, however, if there are other desired rules in the implementation.

For destination address selection, a similar set of rules are employed, to choose which of a destination's possible addresses to use. This relies on first knowing information about a host's possible source addresses.

1) Prefer Usable Destination Addresses (do not use those for which there is no source address that can route to it)
2) Prefer Matching Scope (e.g. prefer a link-local destination if there is also a usable source link-local address)
3) Prefer Non-Deprecated Source Addresses (do not choose a destination address that would require the use of a deprecated source address)
4) Prefer Source Home Address (similar to above, do not require the use of a care-of address)
5) Prefer Label Matches
6) Prefer Higher Precedence (in the policy table)
7) Prefer Native Transport (prefer native IPv6, to 6to4 [20] or IPv6-over-IPv4 tunnelling [21])
8) Prefer Smaller Scope (e.g. prefer link-local addresses, if they are available and valid)
9) Prefer Longest Matching Prefix

These address selection rules provide the basic support for multihoming in IPv6. In a simple multihomed situation, hosts can be allocated multiple addresses, and these rules will choose which one to use for a given connection. This allows limited load-balancing and optimal routing, based on these rules, as well as redundancy, and smooth transitions in renumbering situations [22]. An advanced multihoming system may wish to have many more complex features, however, which the current situation does not provide. The following sections discuss these goals and possible solutions.

V. Goals for IPv6 Multihoming

In RFC3583 [23], Abley et al discuss the requirements for an ideal IPv6 multihoming solution. Firstly, they document the capabilities of IPv4 multihoming, summarised below:

- Link redundancy (in the event of one link failing)
- Load sharing
- Performance improvements and protecting from performance difficulties suffered by providers
- Policy routing (such as Quality of Service, or cost reasons)
- Simplicity
- Transport-layer survivability (transparent re-homing)
- Should not hamper packet filtering (ingress filtering is discussed in detail in RFC3704 [24])

They then add additional requirements that should be met in an IPv6 multihoming solution, as follows:

- Scalability (they key issue with the IPv4 way, discussed earlier)
- Minimal impact on routers
- Be backwards-compatible with legacy IPv6 implementations
- Simple and scalable interaction between the host and the routing system, if required
- Easy for operations staff to manage the system
- Should not require cooperation between a site's multiple transit providers

As can be seen, multi-addressing only solves some of these problems. While it can handle link redundancy and a certain amount of load sharing (depending on the address selection), and it is simple, it does not support some of the crucial multihoming features, in particular session survivability. There is therefore a push to find a comprehensive multihoming solution in IPv6. The following section discusses a variety of multihoming solutions that have been proposed, both from the IETF multi6 working group, and elsewhere.

VI. IPv6 Multihoming Propositions

A. Host-Centric Multihoming and NAROS

Huitema [25] uses the term 'Host-Centric Multihoming' to refer to multihoming using multi-addressing, where it is up to the individual hosts to choose which route to use, by choosing which address to use. This is in contrast to site-multihoming, where it is the site's routers that make this decision. This document proposes a method for hosts to re-establish connections, using an alternative address, after an outage. This is intended to be used as part of a wider multihoming solution.

Address Selection [17], as discussed earlier, is sufficient to handle incoming connections when one of the links has failed. In such a case, the sender will retry all published addresses (assuming that the target host has published all its valid addresses in DNS), until one is found that works.

By default, however, outgoing connections have no knowledge of the state of the links, so the address selection algorithm will be attempted, which may select the failed link, and thus the whole connection will fail. The simplest solution to this problem is reactive, by trying all listed destinations until one is found that works, and Huitema proposes storing this information in a local address cache.

For a proactive solution, Huitema initially proposes using Router Advertisement messages [26] to deprecate prefixes that are currently unavailable. This, however, has a number of problems, in particular where only some routing has failed on a particular provider. An example would be when the provider's upstream connection has failed, but its internal network and other customers are still available.

NAROS (Name, Address, and ROute System) is seen as a possible solution to this problem. It was originally proposed in de Launois' paper [27] as a general host-centric multihoming solution, to allow address (and therefore route) selection based on traffic engineering (such as QoS) policies. NAROS proposes a server which, when queried with a source host and destination address pair, tells the local source which of its source addresses to use, to choose the optimal routing, based on the NAROS server's policies. NAROS servers can be deployed widely and redundantly, for example using anycast [14], [28], since no host-specific state is maintained. NAROS is also able to provide details in varying granularity, and the clients can cache this information for a given lifetime. De Launois concludes that NAROS does not have a significant performance impact, and provides a suitable method of multihomed route discovery.

NAROS would, however, require significant investment in the development and deployment of the NAROS server infrastructure, as well as significant changes to hosts' network implementations (although the design does not prevent legacy hosts from working). In addition, without NAROS, Huitema's address selection proposals appear to be little more than the current address selection methods, with mandated trials of all addresses, and caching of the results. These proposed solutions do not, however, provide mechanisms for some desired multihoming requirements, in particular, for session survivability.

B. RFC3178 – At Site Exit Routers

RFC3178 [29] is an update of Bates' proposal for 'multihoming at site exit routers' [30], which was designed for IPv4. This proposal was originally motivated by a desire to respond to the expanding routing table problem by providing a multihoming solution that did not require new global routes.

For IPv4, Bates' design involved a site receiving two prefix allocations from two upstream ISPs. The site's hosts would use addresses from either allocation, based on which link they were 'near', and the site's internal routing would have local routes for both prefixes. 'Primary links' run from the site's exit routers to each ISP, advertising the primary prefix of that link. In addition to this, 'secondary links' are created, usually in the form of an IP-over-IP tunnel, from the exit router (primary for prefix A) to the ISP router for prefix B,

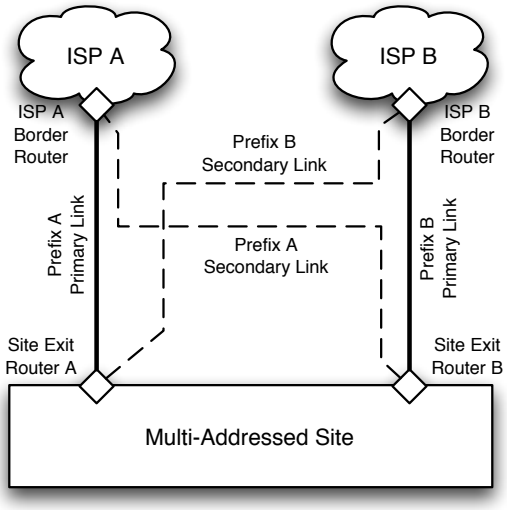

Fig. 3. IPv6 Multihoming at Site Exit Routers

advertising prefix B, with a weak preference. This tunnel runs over a different medium from the primary link for prefix B, providing redundancy for prefix B if the primary link fails. Since it goes to prefix B's ISP, no additional entries are made in the global routing table. The same redundant links are created for prefix A, over the prefix B link. Similarly, internally, hosts with an address from prefix A get a default route with a strong preference for prefix A's primary link, and a weak default route for the secondary link, via the other exit router.

RFC3178 updates this proposal for IPv6. In particular, thanks to IPv6's multi-addressing capabilities, it is possible for each host in the site to be multi-addressed with addresses from all upstream providers, and address selection [17] is used, so there is no need split the site into different address groups. The full network scenario for this multihoming method is illustrated in Figure 3.

It should be noted, however, that this method of multihoming, while not adding load to the global routing table, requires manual configuration and cooperation at the providers and at the site, to set up the tunnel, as well as possibly requiring specific routing to ensure that the tunnels take as different a route as possible as the primary link.

A slight modification of this scheme is proposed by Kim et al [31]. In order to avoid unnecessary overhead and delay through tunnelling, which may involve a non-optimal route, the tunnel is established directly between the exit routers of the multihomed site and the site of the host with which it is communicating. This allows the tunnelled packets to traverse the most optimal path, via the backup ISP. For this proposed solution, a new component, called the Site Exit Router DataBase (SERDB) is proposed, that lists the exit routers for a given site. When a link failure is detected (through ICMPv6 [32] route unavailable messages), the correspondent's site's exit router queries the SERDB, finds the alternative exit router to set up the tunnel to, to continue the communications.

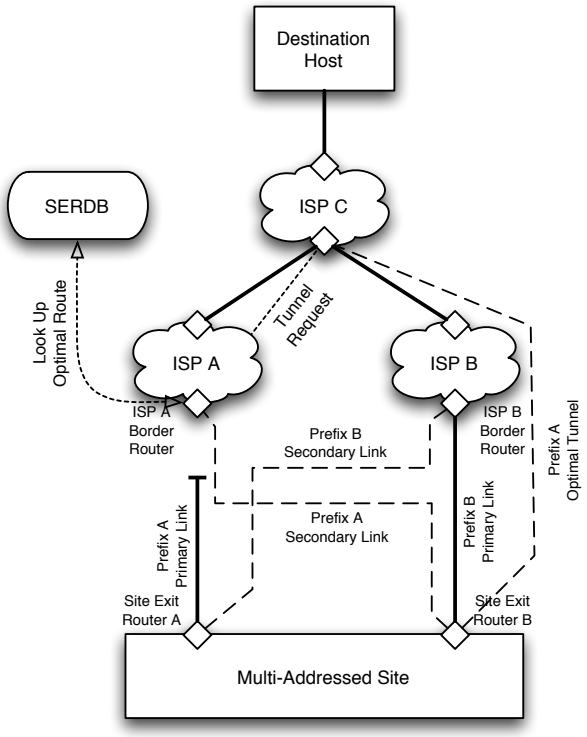

Fig. 4. IPv6 Multihoming at Site Exit Routers using SERDB for Optimal Tunnels

This situation is illustrated in Figure 4. In this example, the primary link for Prefix A has failed, and, upon querying the SERDB, a tunnel is established between the target ISP and the multihomed site, for this prefix, over the most optimal route, via the backup ISP. Little performance difference is reported, compared to the standard RFC3178 setup, and although there is significant work required in creating the SERDB and compatible routing software, it operates transparently to hosts within sites.

C. Prefix Translation

This is a simple method of multihoming, similar to using NAT for small-scale IPv4 multihoming. Hori et al [33] propose a system where a site is allocated multiple prefixes by its providers, and the longest (most specific) prefix is used internally. The site exit router then chooses which upstream provider to use for a particular connection, and changes the prefix of the source address if appropriate, to the prefix of the alternative provider, while keeping the lower 64 bits (the interface ID) the same, to retain end-to-end communication. In their study, they extended this concept, using the Round-Trip Time (RTT) for a connection to choose which upstream connection to use, at the exit router. They found this to attain high performance and provide fault tolerance. It does not, however, provide session survivability, since different address allocations are in use.

This proposal is similar to those discussed earlier, since in all cases the multihomed hosts are reachable via multiple addresses, and it is these different addresses that define which route is chosen. In this case, however, it is the site routers that decide which address to use, rather than the hosts themselves. Although each host is only single-addressed, this does not break end-to-end flow, since even though the address is translated, it remains unique to the host. This solution also removes issues on providing hosts with sufficient information to make a decision on which address to use, leaving it up to the router, which it may be easier to keep updated with policy information, and will not require changes to the host protocol implementations.

D. Use of SCTP

The Stream Control Transmission Protocol [34]–[36] (SCTP) is a new transport-layer protocol, designed to sit on top of IP, as a replacement for TCP. Designed originally for reliable signalling for communications protocols, it has a number of features relating to reliability, error and flow control. Of particular interest is that it also provides support for multihoming. SCTP uses the concept of an ‘association’, rather than a ‘connection’. A SCTP association can be set up between multiple addresses on both the sending and receiving hosts. For example, if each host had two addresses, there are four possible paths between the hosts, as shown in Figure 5.

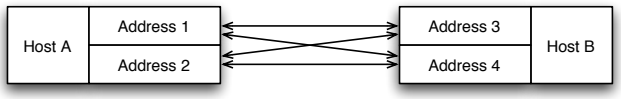

Fig. 5. Possible Paths in an SCTP Session

Initially within an association, one path is considered primary. If a packet retransmit becomes necessary, the sender should choose a different path to use. Similarly, if the endpoint receives duplicate packets, it may indicate that its acknowledgements were not being received, so should also choose a different path to use. SCTP also features ‘heartbeats’ on idle paths, in order to detect if those paths fail.

Ravier [37] has undertaken experimental studies of using SCTP for multihoming. Setting up a test network between two SCTP hosts, he introduced packet loss into the primary link. SCTP responded quickly and transparently, completely the transmission in a little over four seconds, which is slightly more than half the time the same communication took over a single-homed link with the same packet loss. This shows that SCTP is a viable solution for multihoming for link optimisation and redundancy.

SCTP appears to provide a viable multihoming solution for some situations, along with a variety of other features that have use in certain scenarios. SCTP multihoming only works, however, for applications using SCTP as their transport protocol, and this therefore requires a large amount of work by software authors, and may not be suitable for all situations where multihoming would be desirable.

E. Hash Based Addresses (HBA)

The use of Hash-Based Addresses (HBA) is a new proposal by Bagnulo [38], building upon the recently standardised Cryptographically Generated Addresses (CGA) [39]. A CGA is an address where the rightmost 64 bits of an IPv6 address (the host identifier, with the leftmost 64 bits being the subnet prefix) is a hash of a CGA data structure. This contains the host's public key, subnet prefix, and other potential extensions. This allows a particular host's public key to be tied to an address, for security purposes.

HBA takes advantage of the fact that in multihoming, it is likely that a host will know all its possible subnet prefixes (from its different upstream links), and so presents a structure where all these possible prefixes are tied together. HBA takes a similar approach to CGA, but makes the use of public keys optional, and instead uses a CGA-like data structure to store the list of possible subnet prefixes that the host can be in. HBA can co-exist with CGA, by being an extension to the standard CGA parameters.

If the sender decides that the link (and therefore address) used should change (for example, if there has been a link failure), it informs the receiver that it is now communicating from an address in a different subnet. The receiver verifies that the new subnet is in the set of valid possible prefixes given in the original hashed data structure. If it is, existing sessions are allowed to continue to the new address.

It is also proposed that if an HBA is used with a CGA, it is possible to dynamically update the valid address set. This would be useful for multihoming scenarios where the address set is not known beforehand, such as multihoming mobile devices. In such a scenario, if a host wishes to migrate to a previously unlisted address, it informs its peers of a change in the address set using messages signed with its private key, that can be verified by the public key given in the CGA data structure.

F. Level 3 Shim (L3Shim)

The multi6 IETF working group was chartered to research potential IPv6 multihoming solutions, and decide on one to develop as a solution. The architecture chosen is 'Level 3 Shim' [40], [41]. This proposes an extra component to be inserted into the IP layer, to separate the 'identity' and 'locator' features of the protocol (previously both represented by the IP address), as illustrated in Figure 6.

Above the shim layer, is the 'identity' part of the protocol: higher level protocols know to talk to this 'upper layer identifier' (ULID), they do not need to know the location of the host identified by this. The shim provides associations mapping this 'identifier' to the lower-level 'locator', which defines where this host actually is, and therefore how to route traffic to it.

In this proposed architecture, in order to minimise changes required in protocol implementations, both the identity and the locator are IP addresses, and the shim provides mapping between an initial IP address, which defines an identity, and its current location. If a communication needs to change to a different address (re-homes), the application will be unaware,

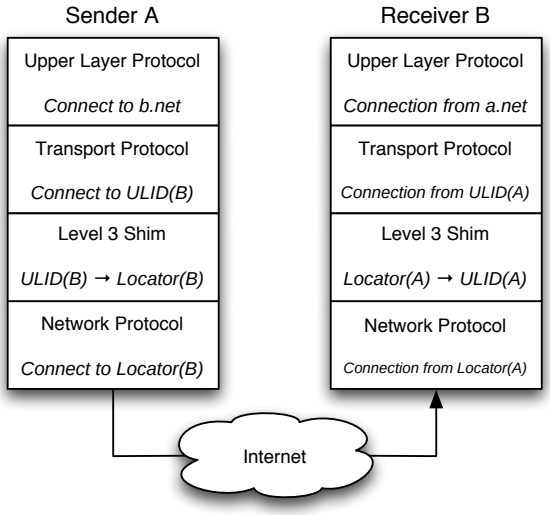

Fig. 6. Use of the Level 3 Shim

and continue to communicate with the original IP address, which the shim will translate to the new locator address. At the receiving end, the shim will also translate source addresses to the standardised ULID, so that if a sender re-homes, the receiver does not notice any changes.

It should be noted that this Level 3 Shim proposition is in its infancy, and many practical issues have yet to be resolved. In particular, as yet there is no concrete design for methods for configuring and selecting the identity/locator mappings, nor for detecting when a re-homing event needs to occur, although initial work here is proposed by Arkko [42]. This tentatively proposes the use of the address selection algorithms [17] discussed earlier, combined with polling messages, to determine the initial addresses to use. It also proposes various possible methods to detect failure during a connection, including intercepting ICMP error messages, or receiving feedback from upper layer protocols.

G. Host Identity Protocol (HIP)

The Host Identity Protocol (HIP) [43], is based around very similar thoughts to the Level 3 Shim, and is in a more advanced state of development. HIP has been designed with consideration to various earlier proposals, and is designed to be a comprehensive solution to provide an identifier/locator split in networking, allowing seamless multihoming and mobility at the protocol level. It is sometimes termed a 'layer 3.5' protocol, sitting between the transport and network layer to translate identities to locators, as illustrated in Figure 7. In the current IP design, the IP address acts as both the endpoint for a socket binding and the location of the host. In HIP, this endpoint is a host identity, which is dynamically bound to the location – the IP address.

It is proposed that each host on a network will have a Host Identity, that is made up of one or more unique Host Identifiers. While such an identifier could be anything that is globally unique, the developers propose [44] that they are the

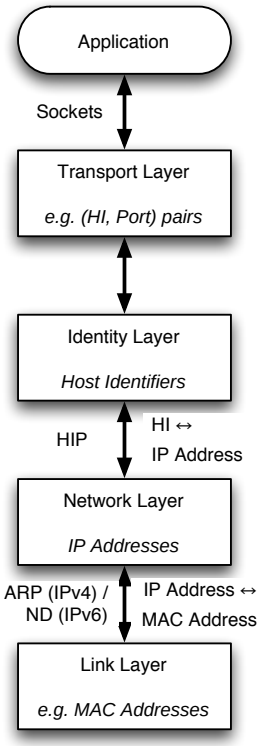

Fig. 7. HIP in the Network Protocol Suite

public keys of public/private key pairs, to allow HIP to also facilitate security and authentication. Each host then has a Host Identity Tag (HIT), which is a 128-bit hash of the public key. Host Identifiers are stored in directory services, such as DNS or LDAP, to allow mappings between (for example) DNS host names and a Host Identifier. It is HITs, however, and not IP addresses, which are used in the network protocols to represent the identities for communication, both for the sender and the receiver.

This decoupling of identity and location provides a mechanism for both mobility and multihoming to be implemented at the protocol level [45], [46]. By providing such a broad-ranged solution, HIP aims to additionally facilitate emerging technologies such as distributed, ubiquitous computing. In terms of multihoming, a Host Identifier can be held by multiple IP addresses (locators) simultaneously (as opposed to mobility, when they are held sequentially).

A HIP session ('association') is activated, in HIP terminology, between an 'initiator' and a 'responder'. Firstly, the initiator queries the directory service to find the known addresses and HIT of the responder. Upon the initiator contacting the responder (possibly using address selection [17] or iterating through possibilities), it receives a 'cookie challenge' [47], to which it has to calculate an answer and return it to the responder. This is a security mechanism to prevent Denial-of-Service attacks through making a large number of connection attempts, by increasing the computational expense of initiating a connection (the responder does not allocate expensive resources until the cookie is returned). Once the HIP session is established, communication continues using IPSec Encapsulating Security Payload (ESP) [48]. Extra HIP packets are sent during the session if some data needs to be changed, such as to inform the initiator of a change in the responder's valid IP addresses (while more designed for mobility, this also has potential for managing dynamic multihoming).

Most work on HIP specification has, however, so far been concentrating on mobility features, and issues such as locator selection, how to handle locator switches for reasons such as QoS (not just on link failure), and the implications of the architecture on transport protocols and security, as well as integration with mobility, have yet to be considered. Despite this, HIP looks a promising endeavour as it seems to solve the major outstanding multihoming issues, especially transparent session survivability, and also provides new protocol features, under one solution.

Even though the HIP specification is incomplete, Henderson et al [49] and Nikander [45] have built HIP prototypes, using Linux (with FreeS/WAN) and NetBSD, respectively. Both papers conclude that HIP is a feasible solution to the problem and tests where hosts have to readdress work correctly, although there is considerable future work, including re-homing decisions and public key exchange, still to be undertaken. Both these papers studied the performance overhead of using HIP, and it was found that there is a notable delay due to cryptographic overhead in the HIP handshake. The time it takes to solve the cookie challenge grows exponentially based on the number of bits involved. For example, a 16 bit puzzle could, in the worst case, cost 2^{16} attempts, which on Henderson's test environment could take over 1 second. Henderson therefore uses smaller cookie challenges, but finds that the overhead from cryptographic key processing still pushes the handshake time to around 1 second. Nikander's test environment has faster machines and sees key-based delays of 300ms, but depending on the complexity, the cookie puzzle could still take anywhere between 300ms and over 2 seconds. Clearly, the extra security features of HIP come at the cost of time delays, but it is worth noting that wide-scale deployment of HIP would be many years away yet, and as such the majority of computers involved in HIP exchanges would probably reach speeds where the cryptographic delay becomes negligible.

VII. Experimentation with Current State

A test network was established within a domestic scenario, using a variety of operating systems (Windows XP (SP1), Linux (kernel 2.6.11), FreeBSD (4.9) and Mac OS X (10.3.9)). The site exit router was on a single IPv4 connection, over which were running two encapsulated IPv6 tunnels. One was a 6to4 block under 2002:/16, and the other was a tunnel of a 'native' allocation under 2001:/16. Router advertisements for both of these blocks were broadcast on the internal network, with equal validity. Both tunnels were configured as default routes on the router. Although the router could differentiate between these tunnels, to the hosts on the network they appeared simply as two different address blocks.

Initially, both tunnels were functioning normally, allowing the normal behaviour of the hosts to be examined. As a

test, traceroutes and SSH connections were attempted from each machine to remote hosts on different IPv6 networks, to see which source addresses were chosen. It was found that Windows, FreeBSD and OS X all implement rudimentary source address selection, as they all chose the 6to4 address for connecting to other hosts under 2002:/16, and chose the 'native' address for 2001:/16 addresses. Linux hosts did not do this, however, and simply chose one of the assigned addresses, and continued to use it for all connections.

The following test was to see how the hosts handled link failure. The 6to4 tunnel was disabled at the router, although the hosts still saw it as an available address. In all operating systems, source addresses were selected as before, but no alternative source addresses were tried when the initial one failed, and therefore connection attempts to other remote 6to4 addresses failed in the test environment. Two AAAA (IPv6 address) records were added to the DNS for one of the multihomed hosts, for both of its addresses from the different blocks. It is currently up to the application to choose which destination address to use for a host. Typically, an application supporting `getaddrinfo()` will receive a list of addresses from DNS (both IPv6 and IPv4), and will iterate through them until it finds one that works. This was the behaviour demonstrated by SSH, which would sometimes find the working address and connect straight away, or sometimes first find the failed address, try it, time out, then move on to trying the next address, which would work.

From these experiments, it can be concluded that current support in major operating systems for even the simplest form of multihoming in IPv6 is incomplete. Limited source address selection, in the form of longest-matching-prefix, is provided, which supports some load sharing, but does not offer failure detection. Destination address selection remains up to applications. While this is possibly adequate, few desirable features are provided, such as caching of address availability information.

VIII. Conclusions

The lack of a standardised solution to multihoming remains a large issue frustrating wider-scale deployment of IPv6, as many large sites rely on multihoming for connection reliability and optimality. The proposed multihoming solutions presented in this paper can be broadly grouped into two categories: those that support connection survivability, and those that do not. Those that do, however, are invariably more complex, and require significant protocol changes, thus breaking the 'simplicity' aim as given in RFC3582 [23]. Therefore, it can be said that no proposed solution yet meets these goals. Having said that, it is difficult to see how such a solution, that is both simple and comprehensive, can be built, working in the new environment that is IPv6. After at least five years of proposals, nothing that solves all goals has yet been suggested.

If session survivability is to be considered an essential requirement of a multihoming solution, and it should be, then changes to the underlying protocol will be unavoidable. Otherwise, the full implementation of address selection would be sufficient, along with some form of failure detection, and a host's multiple addresses can be tried in turn until one that works is found. Protocol changes will not prevent legacy hosts working, as they can still use the current standards, without the extensions.

Host-centric multihoming and prefix translation only serve to improve the current situation for certain circumstances. The use of tunnels at site exit routers, while not in itself requiring protocol changes, either requires manual configuration (which is potentially complex, to ensure varied routing), or some new automatic setup protocol. In addition, it only protects against link failures on the links to the upstream ISP, and not further through the routing system.

SCTP is too high-level a solution for solving all multihoming solutions, so those remaining viable proposals are protocol modifications – HBA, L3Shim, or HIP. It seems strange that L3Shim and HIP are such similar proposals yet are being worked on separately, although HIP is undoubtedly in a more advanced state of development.

HBA seems to provide a fairly elegant solution, using existing standards, with comparably few changes to the protocol structure. HIP does, however, have the added advantage of once-and-for-all providing a solution to the identity/locator split proposition, and therefore transparently enabling mobility, multihoming, and other features, in the protocol. Whether HIP development will prove viable, however, is a lesson for time to tell.

An alternative, pessamistic view would be that, since this elegant solution is so far off, RIRs may start allocating PI address space. There is currently a proposal [50] being considered by *ARIN*, the American RIR, to provide PI IPv6 allocations. Even as a stopgap solution, these addresses could be around for a long time, increasing the global routing table size in exactly the way that was hoped to be avoided. The rapid development of a HIP, or similar, solution is critical to its adoption as a long-term solution.

References

[1] J. Hawkinson and T. Bates, "Guidelines for creation, selection, and registration of an Autonomous System (AS)", IETF RFC 1930, March 1996

[2] Y. Rekhter and T. Li, "A Border Gateway Protocol 4 (BGP-4)", IETF RFC 1771, March 1995

[3] J. Abley., "IPv4 Multihoming Practices and Limitations" (work in progress), IETF Internet-Draft, January 2005

[4] G. Huston, "An Update on Multihoming in IPv6 – Report on IETF Activity", Proceedings of RIPE49, September 2004

[5] Y. Rekhter et al., "Address Allocation for Private Internets", IETF RFC 1918, February 1996

[6] B. Carpenter, "Internet Transparency", IETF RFC 2775, February 2000

[7] V. Fuller et al., "Classless Inter-Domain Routing (CIDR): an Address Assignment and Aggregation Strategy", IETF RFC 1519, September 1993

[8] T. Bu, L. Gao and D. Towsley, "On Routing Table Growth", Proceedings of Global Internet Symposium, 2002

[9] G. Huston, "Commentary on Inter-Domain Routing in the Internet", IETF RFC 3221, December 2001

[10] G. Huston, "Analyzing the Internet's BGP Routing Table", The Internet Protocol Journal, vol. 4 no. 1, March 2001

[11] P. Smith, T. Bates and G. Huston, *CIDR Report*, online at `http://www.cidr-report.org/`, visted March 2005

[12] IANA, *Autonomous System Numbers*, online at `http://www.iana.org/assignments/as-numbers`, visited March 2005

[13] Q. Vohra and E. Chen, "BGP support for four-octet AS number space", IETF Internet-Draft, December 2004

[14] R. Hinden and S. Deering, "Internet Protocol Version 6 (IPv6) Addressing Architecture", IETF RFC 3513, April 2003

[15] IAB and IESG, "IAB/IESG Recommendations on IPv6 Address Allocations to Sites", IETF RFC 3177, September 2001

[16] R. Hinden and B. Haberman, "Unique Local IPv6 Unicast Addresses" (work in progress), IETF Internet-Draft, January 2005

[17] R. Draves, "Default Address Selection for Internet Protocol version 6 (IPv6)", IETF RFC 3484, February 2003

[18] D. Johnson, C. Perkins and J. Arkko, "Mobility Support in IPv6", IETF RFC 3775, June 2004

[19] T. Narten and R. Draves, "Privacy Extensions for Stateless Address Autoconfiguration in IPv6", IETF RFC 3041, January 2001

[20] B. Carpenter and K. Moore, "Connection of IPv6 Domains via IPv4 Clouds", IETF RFC 3056, February 2001

[21] R. Gilligan and E. Nordmark, "Transition Mechanisms for IPv6 Hosts and Routers", IETF RFC 2893, August 2000

[22] F. Baker, E. Lear and R. Droms, "Procedures for Renumbering and IPv6 Network without a Flag Day" (work in progress), IETF Internet-Draft, March 2005

[23] J. Abley, B. Black and V. Gill, "Goals for IPv6 Site Multihoming Architectures", IETF RFC 3582, August 2003

[24] F. Baker and P. Savola, "Ingress Filtering for Multihomed Networks", IETF RFC 3704, March 2004

[25] C. Huitema, R. Draves and M. Bagnulo, "Address Selection in Multihomed Environments" (work in progress), Internet-Draft, October 2004

[26] S. Thomson and T. Narton, "IPv6 Stateless Address Autoconfiguration", IETF RFC 2462, December 1998

[27] C. de Launois, O. Bonaventure and M. Lobelle, "The NAROS Approach for IPv6 Multihoming with Traffic Engineering", Proceedings of QoFIS 2003, October 2003

[28] C. Partridge, T. Mendez and W. Milliken, "Host Anycasting Service", IETF RFC 1546, November 1993

[29] J. Hagino and H. Snyder, "IPv6 Multihoming Support at Site Exit Routers", IETF RFC 3178, October 2001

[30] T. Bates and Y. Rekhter, "Scalable Support for Multi-homed Multi-provider Connectivity", IETF RFC 2260, January 1998

[31] K. Kim, C. Park, T. Kim and S. Kim, "Scalable IPv6 multi-homing scheme based on end-to-end argument", IEEE Communication Letters, vol. 7 no. 7, pp. 340–342, July 2003

[32] A. Conta and S. Deering, "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification", IETF RFC 2463, December 1998

[33] Y. Hori et al, "Design and implementation of an IPv6 gateway allowing effective use of multihome network", 2003 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing, August 2003

[34] R. Stewart et al, "Stream Control Transmission Protocol", IETF RFC 2960, October 2000

[35] L. Ong and J. Yoakum, "An Introduction to the Stream Control Transmission Protocol (SCTP)", IETF RFC 3286, May 2002

[36] R. Stewart and C. Metz, "SCTP: New Transport Protocol for TCP/IP", IEEE Internet Computing, vol. 5 no. 6, November 2001

[37] T. Ravier, R. Brennan and T. Curran, "Experimental Studies of SCTP Multi-Homing", First Joint IEI/IEE Symposium on Telecommunications Systems Research, November 2001

[38] M. Bagnulo, "Hash Based Addresses (HBA)" (work in progress), IETF Internet-Draft, December 2004

[39] T. Aura, "Cryptographically Generated Addresses (CGA)", IETF RFC 3972, March 2005

[40] E. Nordmark and M. Bagnulo, "Multihoming L3 Shim Approach" (work in progress), IETF Internet-Draft, January 2005

[41] G. Huston, "Architectural Commentary on Site Multi-homing using Level 3 Shim" (work in progress), Internet-Draft, February 2005

[42] J. Arkko, "Failure Detection and Locator Selection in Multi6" (work in progress), IETF Internet-Draft, January 2005

[43] R. Moskowitz, P. Nikander, P. Jokela and T. Henderson, "Host Identity Protocol" (work in progress), IETF Internet-Draft, February 2005

[44] R. Moskowitz and P. Nikander, "Host Identity Protocol Architecture" (work in progress), IETF Internet-Draft, January 2004

[45] P. Nikander, J. Ylitalo and J. Wall, "Integrating Security, Mobility and Multi-Homing in a HIP Way", Proceedings of Network and Distributed Systems Security Symposium 2003, Internet Society, February 2003

[46] P. Nikander, J. Arkko and T. Henderson, "End-Host Mobility and Multi-Homing with the Host Identity Protocol" (work in progress), IETF Internet-Draft, February 2005

[47] T. Aura, P. Nikander and J. Leiwo, "DOS-Resistant Auhentication with Client Puzzles", Proceedings of 8th International Workshop on Security Protocols, April 2000

[48] S. Kent and R. Atkinson, "IP Encapsulating Security Payload (ESP)", IETF RFC 2406, November 1998

[49] T. Henderson, J. Ahrenholz and J. Kim, "Experience with the Host Identity Protocol for Secure Host Mobility and Multihoming", Proceedings of Wireless Communications and Networking Conference 2003, IEEE, vol. 3 pp. 2120–2125, March 2003

[50] O. DeLong, "Provider Independent IPv6 Assignments for End-sites", ARIN Policy Proposal 2005-1, online at `http://www.arin.net/policy/proposals/2005_1.html`, February 2005